SINGLE- AND DUAL-PLANE AUTOMATIC BALANCING OF AN ELASTICALLY-MOUNTED CYLINDRICAL ROTOR WITH CONSIDERATIONS OF COULOMB FRICTION AND GRAVITY

Jeffrey Neal Bolton

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Engineering Mechanics

L. Glenn Kraige, Chairman
Scott L. Hendricks
R. Gordon Kirk
John J. Lesko
Mark A. Stremler

December 1, 2010
Blacksburg, Virginia

Keywords: automatic balancing, rotor balancing, Coulomb friction

SINGLE- AND DUAL-PLANE AUTOMATIC BALANCING OF AN ELASTICALLY-MOUNTED CYLINDRICAL ROTOR WITH CONSIDERATIONS OF COULOMB FRICTION AND GRAVITY

Jeffrey Neal Bolton

ABSTRACT

This work treats dual-plane automatic ball balancing of elastically-mounted cylindrical rotors. The application is primarily to systems with a vertically-oriented single-bearing support, but extension is also made to horizontally-oriented single-bearing support as typically found in a vehicle wheel. The rotor elastic mounting includes three translational degrees of freedom for the body geometric center and three rotational degrees of freedom. Damping is included for each of these six degrees of freedom. The model for the automatic ball balancer consists of up to two arbitrarily-located hollow circumferential races, each of which contains up to two sliding particles. The friction model for the particles includes both viscous and Coulomb friction forces. Of considerable complexity is the logic path for the individual particles being either in motion or stationary relative to the rotor. The exact equations of motion for the overall system are derived via a Newtonian approach. Numerical-integration results show that the balancer performance depends strongly on the friction levels as well as the operating speed of the body. Simulations conducted with a pure static imbalance show that ideal automatic balancing is possible only for vertical-axis rotors that have zero Coulomb friction levels between the balancing particles and the races. Simulations with a horizontal-axis statically-imbalanced rotor show that an automatic balancer can improve performance for certain operating speeds and non-zero Coulomb friction levels in the presence of gravitational forces. Simulations conducted with a pure dynamic imbalance show that there is no inherent mechanism to counteract rotational displacements of the rotor about its geometric center. As a result, the balancing particles exhibit several phenomena described in previous works such as synchronous motion and oscillatory behaviors within their respective races. Simulations for an arbitrarily located imbalance show that rotor performance can be improved using dual-plane balancing techniques for certain operating speeds and Coulomb friction levels. Due to the inherent complexity in eliminating an arbitrarily located mass imbalance, the system is generally unable to reach a perfectly balanced configuration, but performance can be improved for carefully-selected initial conditions.

DEDICATION

I would like to dedicate my dissertation to the loving memory of my first niece, Macey Rebekah Whittaker, who was tragically taken after gracing us with her presence for three brief months. Her life was a joy to all who met her, and she will forever impact those who had the privilege of knowing her. She is, and will always be, my little "beauty pot." I love you and I look forward to seeing you again someday.

ACKNOWLEDGEMENTS

I would first like to thank my advisor, mentor, and friend Dr. Glenn Kraige for his significant contribution in helping me complete my academic journey. Without his wisdom, advice, help, and gentle prodding, I would never have gotten this far at Virginia Tech. I would also like to thank the members of my committee, Dr. Scott Hendricks, Dr. Gordon Kirk, Dr. Jack Lesko, and Dr. Mark Stremler for their willingness to serve on my committee and help me complete my dissertation.

I would next like to thank my wife Aimee for her patience and support of me for the last five years. She has stood by me as I tried to balance course work, teaching loads, research demands, and family responsibilities. I would also like to thank my parents, David and Sandy Bolton, for their significant contributions to not only my education, but my life. They devoted themselves to my upbringing and gave me the tools necessary to succeed not only as a student, but as a person, friend, husband, and father. Additionally, I would like to think my adopted parents, Barry and Vicki Blizzard, for supporting me in my efforts at Virginia Tech. Even though Vicki is no longer with us, I know that she is proud of me and that she continues to encourage me from her eternal home in Heaven. I would also like to thank my Creator and Savior, Jesus Christ. Without His constant moving and guiding in the background of the everyday events and trials of life, I would never have found myself at Virginia Tech in the situation I am in. He has never let me down, and while I often question His methods, I never doubt His wisdom and power in achieving His goals. He has been the one constant in my journey from high school to completion of my Ph.D. and I am completely indebted to Him for allowing me this opportunity.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

O	Nominal position of the geometric center of the cylinder
P	Displaced position of the geometric center of the cylinder
G	Location of the mass center of the cylinder/imbalance system
$\mathbf{r}_{P/O}$	Position vector which locates P with respect to O
$\mathbf{r}_{G/P}$	Position vector which locates G with respect to P
$\{\mathbf{O}\}$	Inertial coordinate frame centered at O
$\{\mathbf{P}_1\}$	Coordinate frame centered at P having a rotation θ_1 about the inertial X-axis
$\{\mathbf{P}_2\}$	Coordinate frame centered at P having a rotation θ_1 about the inertial X-axis followed by rotation θ_2 about the intermediate y_1-axis
$\{\mathbf{P}\}$	Coordinate frame centered at P, fixed to the cylinder, having a rotation θ_1 about the inertial X-axis, followed by rotation θ_2 about the intermediate y_1-axis, and then a rotation θ_3 about the body-fixed z-axis
θ_1	Angle of rotation about the inertial X-axis
θ_2	Angle of rotation about the intermediate y_1-axis
θ_3	Angle of rotation about the body-fixed z-axis
$[\mathbf{R}_1]$	Direction cosine matrix for an x-axis rotation
$[\mathbf{R}_2]$	Direction cosine matrix for a y-axis rotation
$[\mathbf{R}_3]$	Direction cosine matrix for a z-axis rotation
M	Mass of the cylinder
r_i	Inner radius of the cylinder
r_o	Outer radius of the cylinder
d	Width of the cylinder
m	Mass of the cylinder imbalance
$\mathbf{r}_m$	Position vector which locates the cylinder imbalance with respect to the geometric center of the cylinder P

r Radial distance of the mass imbalance from the longitudinal axis of symmetry of the cylinder

β_m Angle which locates the cylinder imbalance, measured counterclockwise positive from the x-axis in the body-fixed coordinate frame

d_m Internal offset from the geometric center of the cylinder to the mass imbalance

m_j Mass of a balancing particle

$\mathbf{r}_j$ Position vector which locates balancing particle j with respect to the geometric center of the cylinder P

β_j Angle which locates balancing particle j in its respective race, measured from the body-fixed x-axis

d_j Internal offset from the geometric center plane of the cylinder to the plane of the race in the positive or negative z-direction

k_X Spring constant for resistance to displacement in the inertial X-direction

k_Y Spring constant for resistance to displacement in the inertial Y-direction

k_Z Spring constant for resistance to displacement in the inertial Z-direction

k_{θ_1} Torsional spring constant for resistance to rotation about the inertial X-axis

k_{θ_2} Torsional spring constant for resistance to rotation about the intermediate y_2-axis

k_{θ_3} Torsional spring constant for resistance to rotation about the body-fixed z-axis

c_X Damping coefficient for resistance to velocity in the inertial X-direction

c_Y Damping coefficient for resistance to velocity in the inertial Y-direction

c_Z Damping coefficient for resistance to velocity in the inertial Z-direction

c_{θ_1} Torsional damping coefficient for resistance to angular velocity about the inertial X-axis

c_{θ_2} Torsional damping coefficient for resistance to angular velocity about the intermediate y_1-axis

c_{θ_3} Torsional damping coefficient for resistance to angular velocity about the body-fixed z-axis

μ_s Coefficient of static friction between a balancing mass and the cylinder

μ_k Coefficient of kinetic friction between a balancing mass and the cylinder

μ_c Viscous drag coefficient between the balancing mass and the fluid in the race

F Generic force applied to a body

M Generic moment applied to a body

[ ] First level grouping notation in equation formatting and present only for terms requiring two levels of grouping; default notation for matrices

( ) Second level grouping notation in equation formatting and default grouping notation for all trigonometric functions and terms requiring only one level of grouping

⟦ ⟧ Third level grouping notation in equation formatting

{ } Fourth level grouping notation in equation formatting and default notation for vectors

⟨ ⟩ Fifth level grouping notation in equation formatting

1 Introduction

1.1 Background

Automatic balancing is a technique employed to continuously counteract unwanted vibrations resulting from imbalances found in rotating machinery. Imbalances can create hazardous operating conditions by triggering translational and rotational vibrations which in turn cause large stresses [1]. In other cases, imbalances can be factors in poor performance, high noise levels, excitation of adjoining systems, reduced bearing life, and reduced human comfort. An imbalance can arise through imperfections in the manufacturing process or whenever there is a physical change in equipment resulting from wear, damage, modification, or the installation of new parts. Because it is often impractical to take a machine out of service to realign and balance its components, especially if balancing needs to be done often, engineers have long searched for ways to automatically balance equipment through the use of simple devices that allow for mass redistribution to compensate for any unknown imbalance. Applications of automatic balancers (ABs) include optical disk drives, jet engines, internal-combustion engines, washing machines, food blenders, grinding wheels, gyroscopes, and vehicle wheels.

There are two types of ABs, active and passive [2]. Active devices utilize computers and sensors which continuously read the vibrations in a piece of equipment, apply control laws to counteract these vibrations, and either trigger the mechanical manipulation of balancing masses [3-4] or apply lateral forces by using actuators to eliminate vibrations [5]. By controlling the amount, orientation, and speed of the balancing mass, or the size, location, and application speed of the actuator force, the machine can be balanced very effectively, but in a complex and often expensive fashion.

A passive form of an AB usually consists of freely moving masses, such as a ball (automatic ball balancer or ABB), a ring, or a pendulum, attached to and concentric with the primary axis of rotation for the equipment. When operated above the critical speed for the equipment, the balancers have a tendency to move to positions that counteract the inherent imbalance of the system and do so without the use of computers and control laws. From this point, the abbreviation AB will be used in reference to passive balancers only.

1.2 Literature Review

The use of ABs was first given prominence by Thearle [6] in 1932. Thearle modeled a long rotor held in place on one end by a pivot, permitting rotation only, and on the other end by an isotropic support offering stiffness and damping in two directions. Thearle developed the equations of motion assuming that the transverse displacement of the rotor was small enough that the end of the rotor exhibited planar motion only. To balance the rotor, a device containing two steel balls moving in a circular race was fitted to the support end of the rotor. A clutch was released, and the balls, acting under a "centrifugal-force field" were displaced relative to the race until they reached an equilibrium state which counteracted any imbalance in the rotor. A clutch was then engaged which would fix the balls relative to the rotor. Thearle did not leave the device attached to the rotor to offer continual balancing, but rather used the device to determine the necessary location and size of a balancing weight that could be rigidly attached to the rotor. The device could then be moved to the other end of the rotor and the process repeated to offer two-plane balancing. Thearle's description of the necessary driving forces which make the balancer possible neglects any gravitational effects and offers little consideration of damping effects on the balls in the race. He does mention a rolling resistance applied to the balls from the race, which indicates an assumption that the balls roll without slipping. Thearle comments that "the coefficient of rolling resistance should be made as small as possible by careful hardening and finishing of the race and by the use of high-grade steel balls." Thearle was also able to validate his design through experimentation and made a significant contribution with his novel approach at balancing.

Thearle [7] expanded his work in 1950 to discuss a new type of ABB which used a heavy liquid to balance a rotating device such as an extractor. This type of device, named a Leblanc balancer for its 1916 inventor, typically consists of a vertically oriented rotor, mounted in an elastic support, surrounded by a cylindrical chamber partially filled with a heavy liquid. Leblanc proposed that if the rotor had an imbalance and was deflecting from the spin axis, the fluid surrounding the rotor would be displaced to the opposite side of the chamber and hence balance the system. Thearle concluded that this simple balancer was not able to completely counteract the imbalance because if it did eliminate the rotor deflection, then the fluid would be uniformly distributed around the rotor meaning it has eliminated the very effect (displaced fluid)

that allowed it to balance in the first place. While this simple version does not adequately offer balancing features necessary for high-speed devices, Thearle did offer some improvements. Thearle suggested attaching a flanged annular ring to the outside of an extractor that featured internal vanes to hold the extractor in place relative to the ring. In this way, the fluid was kept separate from the deflections of the extractor and was allowed to flow freely in the outer ring. Thearle also proposed another improvement which used a second annular ring to allow for more precise balancing effects. Thearle concluded that while the modified Leblanc balancer was able to reduce the amplitude of vibrations by 90%, it could not be completely effective at balancing a rotor.

Thearle [8] next examined a ring balancer which consisted of mounting two rings on the shaft of a rotor such that their inner surface was allowed to roll along the shaft surface. The rings were mounted at different locations along the shaft, which would introduce a small couple about the rotor mount that Thearle neglected in his analysis. The rings were enclosed in an oil-filled casing to help damp out any unwanted oscillations of the rings and bring them up to rotor speed smoothly. Thearle concluded that while the rings do a reasonable job of balancing the rotor at speeds above the critical speed (resonant frequency), the existence of friction between the ring/shaft interface was detrimental to their operation. Efforts to eliminate friction at the interface through the use of rubber supports only seemed to make the situation worse by creating a tendency for the rings to "whip viciously at its natural frequency whenever the rotational speed is above this natural frequency." This whipping would persist during operation rendering the ring balancer ineffective.

Thearle [8] then examined a pendulum balancer in which two disks were attached to the rotor shaft and several pendulums were then attached between the disks with pins. The pendulum pivot points were located at a distance larger than any expected eccentricity in the rotor and motion of the system was assumed to be planar in nature. Thearle found that the pendulum approach was only about 50% effective at eliminating vibrations and was costly to build.

At this point, Thearle [8] returned to his 1932 concept and formulated a new ABB which utilized six balls instead of two. The use of more balls increased the capacity and sensitivity of the balancer. In this model, the balancing device featured two possible ball locations. A lower cavity held the balls equally distributed around the rotor to eliminate any imbalance while the

machine was at rest or operating at very low speeds. Once the speed was increased, the balls were free to travel up a small incline to an upper cavity where they would then redistribute themselves around the perimeter and thus balance the rotor. Thearle's model included the effect of gravity, but did not include the existence of friction between the balls and the inner surface of the balancer. Additionally, the model assumed that motion takes place in a single plane. Thearle concluded that the balancer was able to remove almost 100 percent of the oscillations for the system. He further remarked that the rotor must be above critical speed and that the one drawback of the device is that it cannot be placed in the same plane as the imbalance due to its extra bulk. This, he comments, will create an additional couple that stresses the mount.

Shortly after Thearle's work was published, Ernst [9] published his version of an ABB in 1951 which greatly resembled Thearle's 1932 work except Ernst used three balancing balls instead of two. Ernst mounted his device on a grinding wheel and performed experiments to determine the amount of balancing achieved by the device. Tests showed that his device did eliminate much of the vibration due to machine imbalance and was able to balance the rotor better than previous methods hand-balancing methods. While Ernst did not perform a theoretical study of his design, he did comment on several concerns with the balancer. He stated that it was imperative that the track containing the balls be concentric with the rotor axis if a balanced state was to be achieved. He also stated that a low coefficient of rolling friction was required between the balls and the raceway. If the rolling friction was not low, then it would counteract the effect of a "centrifugal-force" which seeks to move the ball to its balanced position. However, given certain positions of the balls in relation to the center of rotation for the unbalanced rotor, if the eccentricity of the rotor is not large enough, a residual imbalance of considerable size can remain since the friction force will force the ball to remain in its place by overcoming the "centrifugal force". Ernst thought the reason he was able to balance the wheel so precisely, as evidenced by his experiments, was because, "In reality, there exists a dynamic condition with the ball being acted upon by several sets of forces which *may* combine to produce a resultant motion in a desired direction" (italics added). Ernst offered the argument that the weight of the ball might produce an oscillatory effect that helped position the ball in a balanced state. He also said, "Other dynamic factors that may act during the balancing operation to produce slight oscillatory motion of the balls in their raceway, and consequently aid in their perfect positioning, include small departures from constant angular velocity of the revolving system due to slight departures

from a perfectly circular orbit." He further stated that friction effects were negligible because it is necessary to have viscous damping act on the balls to prevent them from moving after they first reach their balance positions. Ernst did not reinforce his claims with a mathematical framework and seemed to rely on common sense arguments and experimental observations.

The next major publication on ABs came in 1964 when Alexander [10] published his model of a long rotor utilizing three sets of counterweights located in the forward, center, and aft sections of a cylindrical body. The body was confined to pitch and displace in only one plane and was assumed to be free in space subject to only forces caused by the dynamic imbalance of the body. Alexander said, "The restriction of the motion of the assembly was made to simplify the form of the final equations of motion of the system to a point where they could be solved by reasonable methods since the general motion would be too involved to be solved with the analog equipment which was available for this problem." As long as the ratio of roll-to-pitch moments of inertia was kept small, these simplifications would not detract much from his approach. Alexander derived his equations of motion via Lagrange's approach and ran two phases of simulations for his model. In the first phase, only one set of counterweights was examined, and the effects of viscous damping on the counterweights and Coulomb friction were included. In the second phase of simulations, three sets of counterweights were utilized and only viscous damping was applied. The interaction forces of viscous damping and Coulomb friction were treated as generalized forces not derivable from a potential function. Alexander found that the interaction forces played a critical role in balancing the rotor. If the forces were too small, the counterweights had too much freedom and took too long to move to their correct balancing positions. On the other hand, if the forces were too large, the counterweights took too long to reach their correct position because their relative motion was strongly resisted. Alexander also found that the number of counterweights in a given race does not affect its balancing properties. The main thing was that the race contained sufficient mass to counteract the system imbalance.

Cade [11] in 1965 was the next contributor to ABs and served mainly to highlight areas where ABBs could be utilized including automotive tires, grinding wheels, spin dryers, food blenders, and gyroscopes. Cade remarked that for an ABB to work effectively, the rotor must be operated above its critical speed and be closely balanced prior to ABB installation. The thought here is that an ABB cannot account for massive imbalances since it typically utilizes small counterweights.

In 1975, Sharp [12] published his model of an ABB consisting of a rigid rotor that spins with constant angular velocity in a horizontal plane and is supported by an isotropic bearing consisting of linear springs and dampers. Like Thearle [6], Sharp utilized an ABB consisting of a circular track featuring two balls that were assumed to roll without slip on the inner surface of the track and were subjected to viscous damping forces. Sharp developed the equations of motion for this five degree-of-freedom system via Newtonian mechanics, linearized the resulting equations, and examined the stability of the equilibrium positions of the two balls. Sharp studied the stability of the balls in terms of six parameter values: ratio of ball mass to rotor mass, ratio of ball radius to track radius, ratio of rotor eccentricity to track radius, bearing stiffness, bearing damping, and ball damping. Sharp concluded that the two-ball balancer was stable for a wide range of these parameters and that any frictional resistance from Coulomb effects would hinder the ability of the balls to reach their ideal positions and should be minimized. Sharp circumvented the Coulomb friction effect in his analysis by requiring the balls to roll without slip in the track resulting in a known kinematic constraint on the relative motion of each ball.

In 1977 Hedaya and Sharp [13] published a model of an ABB that consisted of a long rotor supported on either end by isotropic bearings that were constrained to move in horizontal planes and consisted of linear springs and dampers. The rotor was driven with a constant angular velocity and featured two circular tracks spaced on either side of the shaft center which contained two spherical balls that rolled without slipping on the inner surface of the track. Since the supports for the shaft moved in their own respective planes, the shaft and circular tracks were free to rotate from their initial vertical orientation. The effect of gravity was considered and the balls were subjected to viscous damping forces which opposed their relative velocity in the circular track. Similar to [12], Coulomb friction effects were partially circumvented through kinematic constraints. The equations of motion for the eight degree-of-freedom system were developed via Newtonian mechanics, linearized, and examined for stability information. The stability analysis indicated that the system would self-balance for a wide range of parameter values similar to those reported in [12] and this was confirmed by experimentation. The authors stressed the importance of operating the rotor “substantially” above the critical speed, the necessity of having sufficient damping in the supports, and the extreme complexity in the effects of ball damping on the stability of the system.

Bövik and Högfors [14] performed an analysis of an ABB in 1986. The first model mentioned featured an unbalanced rotor constrained by an anisotropic bearing modeled with linear springs and dampers and forced to rotate at constant angular velocity in a horizontal plane. A circular groove was fitted to the rotor concentric with its spin axis and an arbitrary number of particles were constrained to move within this groove. The particles were subjected to a linear viscous damping force, while gravitational effects and Coulomb friction were ignored. The author's second model featured an unbalanced rotor allowed to exhibit non-planar motions. In this model, one point on the rotor was fixed, but the center of mass was allowed to displace in three dimensions. The ABB consisted of a smooth cylinder concentric with the primary spin axis of the rotor. An arbitrary number of balancing particles were introduced into the cylinder and allowed to move along the inner surface of the cylinder meaning they were free to change angular positions relative to the rotor and longitudinal positions within the short depth of the cylinder. The rotor was subjected to viscous damping moments and the particles were subjected to linear damping forces. Gravitational and Coulomb effects were ignored. Next, the authors performed a multiple-scales perturbation analysis of the two models to discuss stability properties and autobalancing properties of the ABBs. First, the planar rotor utilizing one balancing particle in a circular groove was examined and Bövik found that the equilibrium points of the system were globally stable and the system would autobalance above the critical speed. The planar rotor was then studied with two balancing particles and it was again determined that globally stable equilibrium points did exist and the system would autobalance for certain values of the spin frequency. Finally, the non-planar rotor equipped with two balancing particles was studied. After linearizing the governing equations, a range of operating speeds where local stability existed was discovered and it was shown that the system would autobalance.

Later in 1986, a paper on an automatic pendulum balancer was published by Kubo and Jinouchi [15]. The authors modeled a planar unbalanced disk mounted on a flexible isotropic shaft that was given a constant angular speed. A general number of pendulums were attached on the disk at some radial distance from the primary spin-axis of the rotor. The attachments featured leaf springs and damping. Gravitational effects were ignored. To simplify the analysis, the authors assumed that motion of the rotor was restricted to a single direction and the number of pendulums was set at four. The method of averages was applied to the resulting equations. It was discovered that the pendulums could only reduce the vibration amplitude by 30% at

supercritical speeds. It was also noted that complete balancing was possible but only if the pendulums were attached to the center of the shaft, a practice not readily achieved.

In 1988 Tadeusz [16] examined the stability of an ABB by probing the effects of motion resistance for the balancing masses, eccentricity of the race containing the masses, and vibration of the supporting structure foundation. Tadeusz' first model consisted of an unbalanced rotor constrained to move in a horizontal plane and supported so that motion occurred in a single direction under the action of a linear spring and damper. The balancing masses in the ABB consisted of one or two balls subjected to rolling resistance but no gravitational or viscous damping forces. Additionally, the author stated that sliding frictional effects would be ignored since they would result in very high position errors for the balancing bodies. Tadeusz found that rolling resistance kept the balancing bodies from reaching their optimum balancing position and this effect was directly related to the coefficient of rolling resistance. Tadeusz next examined the effect of race eccentricity by allowing the circular race containing the balancing balls to be displaced from the geometric center of the rotor by a small amount. This was shown to adversely affect ABB performance and the errors introduced were shown to be directly related to the offset of the race from the shaft. Finally, Tadeusz examined the effect of foundation vibrations by mounting the rotor on an anisotropic pivot constrained with linear springs and dampers and subjected to harmonic motion in two directions. In this model, the balls were subjected to a viscous medium that helped damp their motion. In general, vibration of the foundation resulted in oscillatory motion of the balancing bodies between solution extremes. However, it was discovered that when the frequency of foundation vibrations matched the spin frequency of the rotor, the balancing masses would assume steady positions that were slightly shifted from their optimal balancing positions. Further, if the support was isotropic and the vibration amplitudes in both directions were equivalent and 90° out of phase, then the balancing masses were able to assume steady positions in the immediate vicinity of the optimal positions. In any case, the vibration of the foundation did affect the final solution, but Tadeusz theorized that this extra energy might allow the balancing masses to overcome some of the negative effects of rolling resistance to improve balancing properties.

Lee [17] published an ABB model in 1996 which featured a vertical plane rotor held by an isotropic support of linear springs and dampers. The rotor was operated at a constant angular speed and featured a circular groove containing an arbitrary number of balancing balls and a low

viscosity damping fluid. Coulomb friction forces and impact between the balancing balls were excluded and the equations of motion were developed via Lagrange's approach. The equations were simplified by ignoring the inertia terms related to the balancing balls, assuming all of the balls had an equivalent mass, and ignoring gravitational forces. Further simplification was given by ignoring the equation of motion related to the angular displacement of the rotor about its spin axis. The stability of the system equilibrium points for static and dynamic imbalances were analyzed with Floquet theory and computed using numerical integration. It was discovered that the ABB was able to balance the system above the critical speed, but the author commented that if an ABB is intended for use in a complicated system where it is not known a priori how much imbalance is present, it would be difficult to select an appropriate set of design characteristics for the ABB to be effective.

Rajalingham [18] first published his work on ABBs in 1998 by examining an unbalanced, flexible, vertically-mounted rotor equipped with an ABB. The model consisted of a rotor constrained to move in a horizontal plane and fitted with a concentric circular track partially filled with oil and containing a single balancing mass. The rotor support was considered isotropic, and to simplify the analysis, damping was ignored for the support and the balancing mass. Gravitational effects were also neglected. Further, it was assumed that the ABB track was smooth eliminating frictional effects. The equations of motion for the rotor and balancing ball were developed and examined to determine equilibrium points and corresponding stability properties. Two parameters were developed to simplify the analysis and consisted of a mass ratio (mass of the balancing ball compared to the mass of the rotor) and the fractional unbalance modification value (mass ratio multiplied by the ratio of ABB track radius to the unbalance eccentricity of the rotor). The equations of motion were linearized and perturbed to examine stability of the equilibrium points. It was found that the presence of a balancing mass would reduce the system critical speed by a small amount that was directly related to the mass ratio parameter. Further, at subcritical speeds, the balancing mass could actually worsen the rotor performance. At supercritical speeds, the ABB could eliminate residual vibrations, but if the mass ratio and ABB radius were not chosen carefully, the ball could not find a stable equilibrium position and unwanted vibrations persisted. Plots were developed which showed stable and unstable regions in the two parameter planes as functions of the operating speed for the rotor.

Additionally, the authors pointed out that at supercritical speeds, a band of instability permeated the stable operating region in both parameter planes.

Later in 1998, Rajalingham [19] expanded his previous work [18] by examining the importance of rolling friction on the balancing properties of an ABB. The model featured a three degree-of-freedom unbalanced rotor that could translate in a horizontal plane and rotate at a constant angular velocity. A balancing ball was constrained to roll without slip in a circular track concentric with the spin axis of the rotor. The equations of motion were developed and linearized to examine equilibrium positions and stability properties using a perturbation technique. Stability plots of the balanced equilibrium positions showed a band of instability that decreased in size as the operating speed increased. Additionally, this unstable band could change size depending on the relationship between the mass ratio of the balancing ball and the rotor, the rolling radius of the balancing ball, and the eccentricity of the unbalanced rotor. Similar to previous results, at speeds below the critical speed of the rotor, the majority of system parameter combinations resulted in unstable operation of the rotor. At speeds above the critical speed, there was again a small band of instability for certain parameter choices (mass ratio, rolling radius, etc.) that remained regardless of how fast the rotor was operated. The boundary lines between stable and unstable operation could shift in the parameter space depending on the chosen values for system parameters, but the overall system behavior was consistent. The ability of the balancing ball to roll on the ABB surface did cause a favorable reduction in the overall vibration suppression properties of the ABB. This effect also served to lower the unstable super critical speed range bringing it closer to the critical speed for the rotor. This situation could exacerbate the vibration of the rotor if the mass ratio between the balancing ball and the rotor were not chosen carefully. Overall, the balancer was seen to improve rotor performance.

Rajalingham [20] published again in 2006 on the use of ABBs in balancing vertically oriented cantilever rotors. The model featured a long, vertically oriented cantilever beam with an unbalanced disk centered at its free end. An ABB was fitted to the disk which featured several balancing balls constrained to move in a circular track that was concentric with the primary axis of the beam. The center of the disk was constrained to displace in a horizontal plane but was allowed to rotate through a 1-2-3 Euler angle set. The disk was not subjected to gravitational forces. The resulting angular velocity of the body-fixed coordinate frame was simplified under the assumption that the disk would experience no torsional vibrations and the operating speed

would be constant. The balancing balls were seen to apply a force to the edge of the ABB which would affect the rotor's motion, but the moments produced by the balancing balls about the disk center were ignored. The flex of the cantilever beam was modeled according to Euler-Bernoulli beam theory with one fixed end and one free end. Viscous damping in the form of air drag was included for the rotor but it was not assumed to make a moment about the disk center. Further, the balancing balls were subjected to only a viscous drag force proportional to the velocity of the ball relative to the track, and they were not allowed to collide with one another. The equations of motion were developed and nondimensionalized. The equations were then linearized and the stability of equilibrium positions for the case of two balancing balls was developed using Routh-Hurwitz criteria. Plots showed a very thin region of stable operation that only existed above the critical speed of the rotor and only for certain values of the mass ratio between the balancing balls and the rotor. When the system parameter values were chosen within this stable region, the rotor was seen to achieve balanced operation. Additional plots using equivalent system parameters, but altered initial conditions of the balancing balls, yielded unstable behavior that did not vanish with time. Overall, initial angular positions for the balancing masses were seen to play a crucial role in determining the long-term behavior of the rotor and ABB effectiveness. Further, simulations of the full nonlinear equations showed that unstable speeds from the linear analysis would actually behave in a stabled balanced configuration illustrating the unreliability of the linear results. The authors commented that the inherent complexity of the system requires that individual parameters be studied using simplified models to determine which factors have the largest influence on the behavior of the rotor and ABB.

Chung [21] published his first work on ABBs in 1999. The model featured an unbalanced disk located on a flexible shaft supported by two bearings. The disk was constrained to have only planar motions and the shaft was assumed to have a linear stiffness and damping. An ABB containing an arbitrary number of balancing balls was fitted to the outside of the disk, concentric with the primary axis of rotation. The ABB contained fluid which provided a viscous damping effect on the balls but no Coulomb friction forces were included. Gravitational forces, impacts between balancing balls, and radii of the balancing balls were ignored. The equations of motion were developed by using Lagrange's approach with the rotor assumed to operate at a constant angular speed. The equations of motion were rendered non-autonomous and put in state form to examine requirements for the equilibrium positions of the balancing balls. The nonlinear

equations yielded both balanced and unbalanced performance equilibrium positions of the balls based on the values for the ABB radius and operating speed. Next, a perturbation technique was used to examine the linear variational equations in the neighborhood of the equilibrium positions. Stability of the equilibrium positions was done using only two balancing balls. It was discovered that a stable balanced operation was only possible if the mass of the balancing balls could effectively counter the eccentricity presented by the imbalance in the disk. In this situation, the two balls would settle to symmetric, diametrically opposed positions opposite the imbalance. Routh-Hurwitz criteria was used to develop stability plots for the various system parameters (damping for the disk, damping for the balls, mass ratio a ball to the disk, ratio of ABB radius to disk eccentricity, and ratio of spin speed to critical speed). Plots showed that for the viscous damping factor of the disk, there was a stable unbalanced region of operation below the critical speed, and a stable balanced region of operation beginning approximately 1.5 times the critical speed of the rotor. The system was unstable in the region directly at and above the critical speed. The authors pointed out that from a theoretical standpoint, the ABB cannot balance the rotor in the absence of damping because energy dissipation is essential to ABB performance. They did comment that in some experiments the ABB would balance the system even when viscous fluid was not used in the ABB. They cite that "it is believed this phenomenon occurs due to the friction between the race and balls, because the friction is also a dissipation mechanism of energy." Plots of the eccentricity of the rotor and the mass ratio for the balancing ball and rotor exhibited similar behavior having a stable unbalanced region of operation below the critical speed with a stable balanced region of operation at an operating speed approximately 1.5 times the critical speed. It was observed that if the ratio of disk eccentricity to ABB radius was too large, there would be a stable unbalanced region of operation regardless of how far above the critical speed the rotor was operated. It is interesting to note that the stability plot for the mass ratio showed a tiny sliver of stable balanced operation just above the critical speed. The region was very small and would be difficult in practice to reach since it is bounded on each side by large unstable regions of operation. The authors pointed out that considerable care should be taken when designing the ABB since balancing ball mass, ABB radius, and damping factors greatly influence the performance of the system. Numerical simulations were conducted on the nonlinear equations of motion in an attempt to verify the stability analysis. The authors chose an implicit time integration method known as the generalized-α method. This was done to allow the

use of larger time steps for shorter computation times as opposed to Runge-Kutta routines which require small time steps to guarantee accuracy. The numerical studies mostly confirmed the stable/unstable regions of operation expected by the linear analysis.

Hwang and Chung [22] published work with ABBs in 1999 and studied the stability of an ABB featuring double races. The model featured a rotor mounted on a flexible shaft (Jeffcott rotor) that was given a constant angular velocity. The ABB consisted of two concentric grooves of differing radii mounted in the same plane as the rotor and each carrying an arbitrary number of balancing balls. Gravitational forces, as well as friction between the balls and the grooves, were neglected. The balls were subjected to a viscous drag force and the displacement of the rotor shaft was characterized by linear stiffness and damping properties. The equations of motion were developed via Lagrange's approach and linearized to examine equilibrium points for the balancing balls and stability properties of the ABB. Simulations utilizing one ball in either race concluded that while the ABB can completely balance the system in certain cases, different combinations of system parameters render the balancing effect null even at speeds above the critical speed. The authors concluded that simply putting an ABB on a rotor that operates at high speeds does not ensure that the device will eliminate unwanted vibrations.

Kim and Chung [23] published work on an ABB for optical disk drives in 2002. The model featured a rectangular feeding deck supported at each corner by an isotropic rubber suspension having linear stiffness and damping properties. An unbalanced motor spindle and ABB were attached to the feeding deck and offset from the geometric center of the feeding deck but on the centerline for the feeding deck in the length direction. The ABB consisted of a circular groove concentric with the motor, which was filled with a viscous fluid containing an arbitrary number of balancing balls modeled as particles. Gravitational and frictional effects were neglected in the analysis. The feeding deck and attached motor spindle were allowed to displace in a horizontal plane and rotate about a vertical axis offering three degrees of freedom. The equations of motion were developed via Lagrange's approach and were simplified under the assumption that the translational and rotational displacements of the feeding deck were small. The linearized equations of motion were derived by using a perturbation approach in the neighborhood of equilibrium solutions, and the equilibrium positions of the balancing balls were obtained. To investigate stability of the equilibrium positions, the authors simplified the number of balancing balls to two and used Floquet theory. It was discovered that as the stiffness of the

rubber suspension was increased, the operating speed of the spindle must be increased and the stable balanced region of operation was reduced. The authors also confirmed that the ABB could not balance the spindle when the operating speed was less than the translational natural frequency of the system.

Next, the authors investigated the effect of suspension damping on stability and found that for certain values of damping, an unstable region appeared in the midst of the stable region above the supposedly stable translational natural frequency of the rotor. For their model, the rotational natural frequency was higher than the translational natural frequency, and a plot of stability for damping showed that the ABB had no balancing effect below the translational natural frequency. Between the two frequencies, stable operation was possible, but not in a region close to the translational natural frequency. A small unstable region existed around the rotational natural frequency, but would vanish if the damping was large enough. This unstable region would grow as the stiffness and operating speed of the rotor were increased. The authors suggested that both rotational and translational natural frequencies have a large effect on the performance of the spindle and ABB and the suspension for the disk drive should be designed to have sufficient damping to counteract any parametric instability from rotational motions. Numerical simulations of the nonlinear equations of motion were able to verify the analysis of ABB performance.

In 2003, Chung and Jang [24] published a model of a Stodola-Green rotor with an attached ABB. The model consisted of a rotor on a flexible shaft that was attached to a wall. The mass of the shaft was neglected when compared to the mass of the rotor. The geometric center of the rotor was assumed to have only planar motions, but the rotor itself was allowed to exhibit rigid-body rotations with a 3-1-2 set of Euler angles. A single-plane ABB was mounted concentric with the rotor. The ABB contained a viscous fluid and an arbitrary number of balancing balls modeled as particles. Gravitational and frictional effects were neglected in the analysis. The equations of motion were developed by using Lagrange's approach. In the potential energy analysis, torsional and longitudinal shaft deflections were neglected and only strain energy from bending deflections was considered. The shaft was assumed to be isotropic and prismatic. Rayleigh's dissipation function included both translational and rotational damping for the rotor as well as viscous damping for the balancing balls. The equations were simplified by assuming the radial displacement of the rotor geometric center, the "1" and "2"

Euler angle rotations for the rotor, and the rotor eccentricity were all small. A perturbation technique was used to obtain linearized equations of motion in the neighborhood of equilibrium solutions as well as equations for the balanced equilibrium positions of the balls. To simplify the analysis, only two balls were considered in the stability study and only the equilibrium positions corresponding to balanced operation were explored. A Routh-Hurwitz criterion was utilized to examine the stability of the ABB in relationship to several system parameters. The parameters included ratios of spin speed to critical speed, ball mass to rotor mass, disk eccentricity to ABB radius, the ball damping parameter, and translational and rotational damping parameters. The rotor had a rotational natural frequency which was higher than the translational natural frequency. It was observed for nearly every parameter study that the rotational natural frequency had no bearing on the stability performance of the ABB. The authors discovered that the mass of the balancing balls was sufficient to counteract the imbalance in the rotor. Additionally, the authors found that damping for the balls and translational damping for the rotor were required if the ABB were to balance the rotor. Damping for the rotational degrees of freedom of the rotor was not seen to prohibit balancing provided the operating speed was sufficient. Numerical simulations of the nonlinear equations of motion with two balancing balls were done to verify the stability analysis. The studies showed that the rotor could have stable unbalanced performance, unstable performance, and stable balanced performance for various combinations of system parameters as predicted by the stability analysis. For the stable balanced operation, the ABB was able to eliminate both translational and rotational vibrations of the rotor.

Chung [25] published again in 2005 to examine the effect of gravity and angular velocity on ABB performance. The model featured a Jeffcott rotor with concentric ABB attached at the midpoint of a flexible shaft mounted on two bearings. The rotor was oriented vertically in one case and horizontally in the second case to examine the influence of gravity on ABB performance. The ABB consisted of a circular track filled with a viscous fluid and an arbitrary number of balancing balls. The mass of the shaft was neglected as compared to the mass of the rotor and the balancing masses were modeled as particles. The rotor was assumed to exhibit planar motions only but have variable angular velocity and angular acceleration. The equations of motion were developed using Lagrange's approach. The shaft was assumed to have a linear stiffness and damping. Frictional and non-planar effects were not considered for this analysis. To simplify the analysis, the system was limited to two balancing masses. To examine the effect

of gravity, only the horizontally oriented rotor was utilized because it was thought that gravity acting perpendicular to the plane of the vertical rotor would have no bearing on the ABB performance. For the analysis of the horizontally oriented rotor, the angular velocity of the rotor was assumed constant and given a value above the critical speed. Analysis showed that the rotor with ABB would come to a steady unbalanced mode of operation from the resulting static deflection of the rotor caused by gravity. This motion featured small oscillations of the balancing masses that were considerably improved over the rotor vibrations in the absence of an ABB since it eliminated whirling motions. Next, the authors explored the effect of three different angular velocity profiles for the horizontally and vertically mounted rotors. The authors simulated a step, ramp, and spline profile for each rotor and found that the horizontally mounted rotor had residual vibrations for all three profiles while the vertically mounted rotor had none. Again, this was due to the effect of gravity on the system and the assumption of planar motions for the rotor. The step angular velocity profile created the largest transient behavior while the splined angular velocity profile, as expected, had the smallest.

Kim, Lee, and Chung [26] published again in 2005 to explore a three-dimensional model of an ABB on an optical disk drive. The model featured a rectangular feeding deck supported at each corner by an isotropic rubber washer offering linear stiffness and damping in three directions. An unbalanced motor spindle with attached ABB was mounted to the feeding deck and offset by a short distance from the geometric center of the feeding deck along the centerline of the feeding deck. The ABB contained a circular track filled with a viscous fluid and an arbitrary number of balancing balls. Gravitational effects and Coulomb friction effects were ignored. The feeding deck was assumed to be a thin rectangular plate that was perfectly balanced. Additionally, the mass center of the feeding deck and the mass center of the rotor system were located in the same plane because most commercial optical disk drives are thin. The feeding deck was allowed to have three translational and three rotational degrees of freedom through a 3-1-2 Euler angle set. The equations of motion were developed via Lagrange's approach. It is interesting to note that the inertia matrix for the rotor system was assumed to be principal indicating that the slight imbalance of the rotor was not taken into account when computing the products of inertia for the system. The equations were simplified assuming the rigid body motions of the feeding deck were small. The perturbation method was used to obtain linearized equations of motion in the neighborhood of balanced equilibrium positions. The

equilibrium positions associated with unbalanced performance were not considered in the study. For numerical simplicity, the number of balancing balls was fixed at two even though commercial disk drives typically feature up to ten balancing balls. The authors found that there were essentially two classes of equations for the system consisting of in-plane motions for the feeding deck and rotor and out-of-plane motions. These were decoupled from one another in the neighborhood of equilibrium positions and could be considered separately for stability purposes. The in-plane motions were of primary concern since the system was assumed to have little out of plane motion. Floquet theory was used to examine the stability of the system. Several system parameters were studied including non-dimensional damping factors for the suspension washers and ABB fluid, the ratio of rotor eccentricity to ABB radius, the ratio of balancing ball mass to rotor mass, and the ratio of operating speed to natural frequency. Plots of the damping factors against the non-dimensional operating speed parameter showed that the system was unstable if the operating speed was below the first critical speed of the rotor and also unstable in a small region just above the first critical speed. In addition, a small region of instability existed around the second natural frequency of the rotor but disappeared if the damping was large enough. Plots showing the effect of rotor eccentricity and balancing ball mass showed that balancing was only possible above the first natural frequency and had an upper limit for stability based on how unbalanced the disk was, and a lower limit based on the chosen mass of the balancing balls. Essentially, if the disk was too unbalanced, the ABB could not correct its performance. It is interesting to note that if the damping on the rotor was too small, regions of instability just before and just after the second natural frequency would appear in the plots showing stability compared to balancing ball mass. A plot of the out-of-plane equation of stability for balancing ball mass showed a small region of instability around the out-of-plane natural frequency for the rotor in an area very similar to that found for the in-plane motions. This meant that the in-plane motions dominated the stability characteristics for the system. Numerical simulations of the nonlinear equations of motion verified the results from the stability analysis and showed that the ABB could eliminate unwanted vibrations for certain values of the system parameters but not in all cases. The authors concluded that stability analysis of the linearized in-plane equations was sufficient for determining balancing characteristics of the rotor. They did comment that if the mass centers of the feeding deck and motor spindle were in different planes, then the model would probably need to be modified to account for more three-dimensional effects. The authors

suggested that two planes be used when considering a more general case, but they did not go into a thorough analysis of this. Next, the authors showed that even if the ABB did not have enough mass to balance the rotor, its presence greatly reduced vibrations of the system when compared to a similar rotor without an ABB provided the operating speed was above the first natural frequency for the system. Finally, the authors showed that a two-ball ABB could effectively simulate the ten-ball ABB which most commercial optical drives are fitted with since the balancing performance and settling times for the balls were similar.

In 2000, Sperling [27] extended earlier work [28-30] to study self-synchronization effects in ABBs. The model consisted of an unbalanced rotor mounted on two isotropic bearings which consisted of linear springs and dampers. Each support resisted motion in two directions. The mass center of the rotor was assumed to have planar motions and the rotor was allowed to rotate about two axes and was operated at a constant angular speed. Two ABBs were attached to the rotor on either side of the imbalance and each consisted of two point masses traveling in a viscous medium in a circular track concentric with the axis of primary rotation for the rotor. Gravitational and frictional effects were neglected. The equations of motion were developed via Lagrange's approach and were linearized to examine the existence of any equilibrium positions for the balancing masses by using the method of averaging. It was assumed the rotor was operating above the critical speed to ensure at least the possibility of effective balancing by the ABBs. To determine the stability of equilibrium positions, small perturbations of the phase angles were studied. Sperling found that in order for stability to be possible, the rotor must be operated above the critical speed and that if the polar moments of inertia are too large, then the rotor will have trouble balancing and might not balance at all. Computer simulations were conducted to verify the results of the numerically integrated dimensionless forms of the original equations of motion. The initial conditions were such that all state variables were zero or at rest. It was found that the rotor would experience a transient vibration during run-up. After this transient, the rotor would settle into a balanced condition for periods of time followed again by transient vibrations. In order for the masses to synchronize with the rotor and counteract the imbalances, the viscosity of the ABB fluid must be large enough to slow the balancing masses to match the critical speed, but not so large that they prevented the masses from reaching their equilibrium positions in a reasonable amount of time. Sperling commented that “the closer the polar inertia is to the transverse one, the greater the sensitivity of the compensating motion with

respect to the coefficients of the viscosity of the balancers and to the initial conditions." This means the application of ABBs is more restricted than an active force-type balancer, but nonetheless is useful for certain applications.

Ryzhik and Sperling [31] furthered their work in 2002 by studying two-plane balancing of rotors. The model consisted or a rigid rotor mounted on two isotropic supports having linear springs and dampers that resisted motion in two directions. The rotor was given a static and dynamic imbalance through two point masses of differing size situated at different eccentricities along the rotor axis. Two ABBs were fitted to the rotor, one on either side of the mass center and inherent imbalance, which consisted of circular tracks concentric with the rotor axis but having different radii. Two balls were placed in each track and allowed to move freely in a viscous medium. Motion of the mass center was assumed to be planar, and the rotor was allowed to rotate about two axes passing through the mass center. The balls were assumed to roll without slipping on the edge of the ABB. Gravitational effects, rolling resistance, and Coulomb friction were ignored. The equations of motion were developed using Lagrange's approach and then linearized. To simplify the analysis of equilibrium positions and their stability, the ABBs were assumed physically identical and placed at equivalent locations from the rotor center of mass. Additionally, the inertial parameters of the balancing masses were neglected in comparison with the rotor mass, while the influence of the support forces was waived. The method of averaging was used to determine equilibrium position requirements for the balancing masses and analyzed with Sylvester's criteria. One overall stability condition was developed which stated that the polar moment of inertia squared must be greater than the product of the two transverse moments of inertia for the rotor. It was found that if the rotor operated above the highest critical speed, then balancing could occur. Increasing the support damping was seen to lessen the necessary time for balancing to occur. If the viscous medium provided low damping, the balls were seen to oscillate around their optimal positions. If the viscous medium provided high damping, the balls would no longer oscillate but take too long to move to their optimal positions. Certain values of rotor speed illustrated the "Sommerfeld" effect which created large rotor vibrations and caused the balancing masses to rotate with an angular velocity equal to the second critical speed of the rotor. The Sommerfeld effect occurred as the rotor approached a critical speed. The balancing masses were seen to move with a speed close to the natural frequency of the rotor even after the rotor has accelerated through the critical speed and was

operating at a higher speed. The border rotational speed between the two critical speeds of the rotor where Sommerfeld-type motions existed was found to depend strongly on the support damping for the rotor and weakly on the viscous damping for the balls. If the balls were made unequal to one another, then Sommerfeld effects occurred at considerably higher rotation speeds than if the balancing balls were equal. Later work [32] by the authors confirmed these findings and also showed that near critical speeds, the difference in rotational speeds of the rotor and the balls would become markedly increased while in regions beyond the critical speed, the balls would synchronize with the rotor speed and move with constant phases to provide some balancing and vibration reduction.

Ryzhik and Sperling [33] published again in 2004 to study partial compensation of unbalanced rotors having both one- and two-plane ABBs in which they greatly expanded on some previous findings [34]. The model featured a vertically mounted rotor with two point mass imbalances located at different locations along the rotor axis. The rotor was supported by two isotropic supports consisting of linear springs and dampers. Each support could resist motion in two directions. Two ABBs were mounted at different locations along the rotor axis. Each ABB contained two balls free to move in a viscous medium. Gravity and all frictional effects were neglected. The balls were assumed to roll without slipping on the ABB track. The equations of motion were developed via Lagrange's approach and the existence of equilibrium positions was studied with the method of averaging. It was again discovered that complete balancing could only take place if the operating speed of the rotor was beyond the second critical speed of the rotor. Partial balancing was shown to occur in a frequency range just after the first critical speed in rotors when the polar moment of inertia was greater than the transverse moment of inertia. Studies of one ABB and two balancing balls showed that while complete balancing is impossible, proper location and sizing of the ABB could be quite effective in eliminating unwanted vibrations. This result was also found when two ABBs were used, if the two planes were too close to one another, then different types of motion could occur that led to unwanted motions. Simulations confirmed these results and showed that if rotor speed was constant, the balls would eventually find their stable positions. Accelerating the rotor drastically affected the behavior of the balancing masses. Again, Sommerfeld motions were observed when passing through the critical speeds and led to large rotor vibrations. Additionally, low ball damping created oscillations of the balls while high damping increased the balancing time. It should be

noted, that partial balancing is still better than no balancing and can lead to significantly reduced vibration levels at certain operating speeds.

Sperling et al. [35] published again in 2004 and examined single-plane balancing of a rotor with an ABB. The model featured an unbalanced rigid rotor mounted on two isotropic bearings consisting of linear springs and dampers. An arbitrary number of point masses of differing size, radial and angular position were located at separate transverse sections of the rotor. A single plane ABB was fitted to the rotor and contained an arbitrary number of balancing masses that could have unique mass and eccentricity characteristics. While only one plane was utilized for study, the general equations were developed with an arbitrary number of planes. The balls were subjected to a viscous medium in their ABB and assumed to roll without slip on the surface of the ABB. Gravity and Coulomb friction effects were ignored. Additionally, the motion of the rotor was characterized by two translational and two rotational degrees of freedom. The equations of motion were developed via Lagrange's approach and linearized. The method of averaging was employed to determine general conditions for existence and stability of the equilibrium positions for the balls. The authors studied the ability of a single ball balancer and found that such an ABB was not able to guarantee balanced performance since the mass must be perfectly sized and be placed in the same plane as the imbalance. A single ball was able to reduce vibrations in certain conditions, but only eliminate vibrations for a few conditions. Next, a two-ball balancer was studied and it was discovered that three types of equilibrium positions were possible. The balls could settle to distinct locations symmetric about the imbalance line and eliminate all vibrations, they could settle to identical positions or diametrically opposed positions in which case, vibrations were not guaranteed to vanish. In order to eliminate any vibrations, the combined eccentricity of the balls must be greater than the eccentricity of the imbalance. Next, plots were developed showing the existence of stable equilibrium positions for various combinations of rotor parameters. First, the authors studied "long" rotors where the transverse moment of inertia is greater than the polar moment of inertia. Then, "spherical" rotors, where the transverse moment of inertia equals the polar moment of inertia, were studied. Finally, "disk" rotors, where the transverse moment of inertia is less than the polar moment of inertia, were studied. Each rotor was simulated with an ABB that was mounted in the geometric center of the shaft and at some transverse location away from the geometric center of the shaft. The plots indicated that the existence and stability of the equilibrium positions has a highly

complex relationship with system parameters and stability cannot be guaranteed simply by providing sufficient balancing masses and high operating speeds. Numerical simulations of "long" and "disk" rotors, having one and two balancing balls in a single plane, were completed and agreed quite well with the theoretical analysis of equilibrium behavior. It was seen that in the areas surrounding critical speeds, the balls exhibit a Sommerfeld effect and caused large vibrations of the rotor. Further, there were areas in between the two critical speeds of a "long" rotor where stability was possible and vibrations were minimized. Due to the complex nature of the equations, the authors concluded that it is essential to pick correct values of the system parameters (support stiffness and damping, ball damping, ball mass, etc.) in order to ensure good operating conditions.

Ryzhik and Sperling [36] published again in 2004 and examined non-synchronous motions near critical speeds in a rotor equipped with a single-plane ABB. The model consisted of a vertically oriented rotor held by two isotropic supports consisting of linear springs and dampers. The rotor was both statically and dynamically unbalanced and the motion of the rotor was characterized by two translational and two rotational degrees of freedom. A single-plane ABB was fitted to the rotor and contained an arbitrary number of balancing balls subjected to a viscous medium and assumed to roll without slip on the ABB circumference. The equations of motion were developed by using Lagrange's equations and linearized. It was further assumed that the primary imbalance plane of the rotor did not coincide with the plane of the ABB and the rotor was driven at a constant speed. Simulations and experimental data both showed that near critical speeds, the balancing balls move together at a different speed than that of the rotor, and have the same phase. The method of averaging was used to determine the requirements on equilibrium positions and numerical simulations were conducted to determine the border range of spin speeds for the non-synchronous motions of the balls for a disk-type rotor. The authors first examined the effect of a slowly increasing driving moment on the rotor. Analytical studies showed that the rotor velocity should follow closely with the nominal speed but the simulations showed a lagging effect where the rotor fell behind the nominal speed. Further, when passing through the critical speed, there was a sharp difference in the simulation and the analytical predictions of the ball motions as they exhibited non-synchronous motions at a speed close to the rotor system eigenvalue. In order to counteract these regions of unwanted large motions, the authors stated that it is necessary to decrease the mass and/or eccentricity of the balls and

increase the damping parameter for the viscous medium. It was also observed that the angular positions assumed by the balls were highly dependent on their damping and the balls would not always assume stable positions after passing through the Sommerfeld area. If the damping on the balls was too high, they would occupy unstable regions for long periods of time. If the Sommerfeld area is large, there were two resonant frequencies which provided maximum vibration amplitudes. The authors commented that minimizing these areas to provide smooth operating conditions would be extremely complicated and they were unclear of how to achieve it. They did comment that ball masses and eccentricities should be kept very small in order to minimize any unbalanced "centrifugal" forces. An experimental apparatus consisting of a rotor and ABB was used to help validate the theoretical predictions. The authors commented again on the extreme importance of selecting good operating parameters for the rotor to ensure smooth operation and long term stability of the angular positions for the balancing masses.

Ryzhik and Sperling [37] returned to the model presented in [35] to consider the effect of support anisotropy on single-plane ABB performance. The rotor was mounted on two orthotropic elastically-damped supports and the equations of motion were developed via Lagrange's approach and linearized. General conditions for the existence and stability of equilibrium positions for the balls were developed by using the method of averaging. It was found that anisotropic supports yielded forward whirl and backward whirl components for the orbital motion centers of the ABBs. To eliminate the forward whirl motions for the rotor, the forward whirl portions of the ABBs must vanish in at least two planes. The authors used a single-plane ABB with two balancing balls which did not eliminate the forward whirl for the entire rotor, but did eliminate it in the plane of the ABB. The backward whirl components for the ABB were shown to only disappear if the proportion of the harmonical influence coefficients for both transverse directions of the balancing balls and the rotor imbalance with respect to the path center of the balancing balls were identical. A rotor supported by two symmetric anisotropic bearings having a pure static imbalance was analyzed. It was found that a single-plane ABB could eliminate all vibrations for this case in regions above the critical speeds. Next, purely tilting vibrations were analyzed for a rotor having a fixed-point at some location along its shaft. It was found that for long rotors, operating regions exist where forward whirl, and simultaneously backward whirl, could be eliminated for the rotor as a whole. For disk-type rotors, a region of stable compensation exists that was only noticeable if the support had

sufficient anisotropy that offered a large enough range between the two natural frequencies in the transverse directions. Simulations of both isotropic and anisotropic supports were conducted. For a pure statically-imbalanced rotor, Sommerfeld motions occurred for both support types near critical speeds. However, the region of Sommerfeld effects was larger for the anisotropic rotor since there were two critical speeds. Additionally, there was a double-rotational-frequency component for support vibrations in the anisotropic case. Next, a long rotor was subjected to purely tilting vibrations by fixing one end of the shaft in a rigid support. The simulation showed that if the remaining support was isotropic or anisotropic, complete balancing was achieved with a single-plane ABB. A disk-type rotor was then fixed at one end and given either an isotropic or anisotropic support at the other end. Simulations showed that the anisotropic support offered radically different behavior than the isotropic case. The isotropic case featured only forward whirl motions while the anisotropic case featured backward whirl components as well. Next, a long rotor having both static and dynamic imbalances, two unique isotropic or anisotropic supports, and a two-plane ABB was considered. Simulations showed that the anisotropic supports contained wider areas of critical speeds which offered noticeable double-frequency components in the vibrations. Beyond the critical speed the vibration characteristics of both support types was similar and the ABBs were able to completely balance the rotor. Results were similar when conducted with a disk-type rotor. The only difference being that a single-plane ABB was utilized which eliminated vibrations in the plane of the device, but not for the rotor as a whole. Overall, anisotropic supports were shown to not hinder ABB effectiveness but did offer some performance differences for the rotor.

In 2001, Wettergren [38] published his first work on ABBs by examining a stiff shaft mounted in two anisotropic bearings featuring linear springs and dampers. The ABB was attached to the shaft and featured a groove partially filled with oil that contained two balls. The oil provided a viscous drag force on the balls, but gravitational and Coulomb friction effects were neglected. Motion was assumed to occur in a single plane and the balls were assumed to roll without slip in the groove but were allowed to collide with one another. The contact forces between balancing balls were obtained with Hertz contact theory. The equations of motion for the system were developed by using Newtonian mechanics and numerically integrated with a Runge-Kutta-Fehlberg scheme. Simulations showed that the oil damping reduced stability regions for the balls. If the balls were caught in an unstable region, they were seen to

continuously rotate in the groove of the ABB. The balls were able to reduce vibrations in certain regions above the first critical speed of the rotor and below the second critical speed.

Wettergren [39] published again in 2002 and continued work on his previous model [38]. Simulations with one and two balls were done and it was discovered that adding a second ball to the ABB tended to increase the size of the instability regions around the rotor critical speed. Two balls performed better than one above the critical speed since the size of the rotor imbalance did not need to be known a priori. The two balls were observed to reach symmetric positions opposite the imbalance. Wettergren also observed that high bearing damping served to suppress instabilities in the system and adding more balls to the system increased the amount of time needed for the balls to reach a stable position.

Kang and Chao [40] published work on an ABB in 2001 by modeling an optical disk drive. The rotor shaft was treated as a rigid body and the attached rotor was constrained to move in a horizontal plane. The rotor was mounted to a stator supported by an anisotropic support of linear springs and dampers that each resisted displacement in a single direction. The ABB consisted of a circular track, slightly offset from the geometric center of the rotor, containing two balls modeled as perfect spheres and assumed to be point masses. Gravitational effects were ignored. The balls were subjected to an aerodynamic drag force from fluid in the track and were assumed to roll without slipping on the track from the action of rolling friction at their contact point. It was further assumed that the ball material was much stiffer than the track material which meant that the balls would deform the track slightly at their leading edge. This deformation induced a "rolling resistance moment" about the ball center from the action of the contact force and rolling resistance. The equations of motion for the four degree-of-freedom system were developed via Newtonian mechanics and, for simplicity, the support was assumed to be isotropic. The equilibrium points were examined by using asymptotic analysis via the method of multiple scales. A total of 24 equilibrium points were determined, but only three of the solutions differ from a physical standpoint. The balls were able to settle at distinct angular positions (Type-I), identical angular positions (Type-II and -III), or opposite angular positions (Type-IV). Stability of the equilibrium points was examined by constructing stability diagrams with respect to the operating speed ratio and the sizing ratio (mass of a balancing ball compared to the mass of the system) and the Routh-Hurwitz criteria. Type-I equilibrium points were possible provided three conditions were met: the system must be operated above the support

frequency, the combined mass of the balancing balls must be greater than the rotor imbalance, and the total mass of the balancing balls must be limited by the sizing ratio. Type-I solutions covered a fairly large area of the parameter space and afforded excellent balancing properties. Type-II solutions offered slim possibility of effective balancing since the balls settled at the same point and must be sized perfectly to eliminate the effects of rotor imbalance. Type-III solutions were seen to have a stable parameter space very similar to that of Type-I, but their balancing effectiveness was vastly inferior compared to Type-I solutions. Type-IV solutions did not have any stable range in the tested parameter space.

Huang and Chao [41] next published a paper in 2002 that continued to examine the application of ABBs to optical disk drives. The model utilized was identical to [40] and was used to examine the effects of runway eccentricity, rolling resistance, and viscous drag forces. The model was simplified by assuming a constant angular speed for the rotor, an isotropic support, and one balancing ball with no rolling resistance, viscous drag force, or relative motion when in steady state. It was discovered that runway eccentricity must be minimized for effective balancing to take place. It was also discovered that for a given value of rolling resistance, there were continuous finite ranges of solutions where the balls could settle and thus maintain a residual vibration in the system. Thus, rolling resistance must be minimized to achieve good balance. To examine the effects of the viscous drag force, it was assumed that the balls were perfectly smooth and the fluid in the track was uniformly distributed and had no relative motion with the track. Results indicated that as the viscosity of the fluid in the track was increased, the settling times for the ball were reduced. The authors commented that for effective balancing, runway eccentricity must be reduced, rolling resistance must be minimized, and fluid viscosity must be selected to provide a smooth and fast transient response for the balls. Experiments were also conducted which agreed well with the theoretical predictions from the analytical model.

Chao and Huang [42] continued work on the optical disk drive in 2003. This model featured a vertically mounted rotor supported by three linear spring and damper pairs that each resisted displacement in a single direction. The support also featured two torsional spring and damper pairs which resisted angular displacements of the rotor shaft about a single axis. Two of the linear spring and damper pairs along with one of the torsional spring and damper pairs were positioned a finite distance up the shaft from the bottom of the rotor. The remaining spring and damper pairs were located at the bottom of the shaft and were capable of resisting displacements

along the axis of primary rotation. Thus, the rotor was allowed to displace in three directions and tilt through a pseudo 3-1-3 set of Euler angles. The third angle in the rotation was the constant spin speed for the rotor. The ABB was mounted a short distance below the rotor and featured a perfectly circular concentric track containing an arbitrary number of balls considered to be perfect spheres and modeled as point masses. The balls were assumed to remain in contact with the outer edge of the track and were not subjected to rolling friction or impacts. The balls were subjected to a viscous drag force, and gravitational forces were included. The equations of motion were developed via Lagrange's approach and simplified under the assumption that the eccentricity of the rotor and the motion variables corresponding to tilting of the rotor shaft were small. Only quadratic terms were retained in the equations of motion which were numerically integrated with 4^{th} and 5^{th} order Runge-Kutta methods. The ABB was seen to dramatically decrease lateral vibrations in the unbalanced rotor but offer little improvement in the vertical direction. It should be noted that the vertical direction was approximately twice as stiff as the lateral directions. Additionally, the authors found that the tilting angle of the rotor was most improved when the ABB was mounted a short distance below the plane of the rotor. Mounting the ABB in the same plane as the rotor did little to improve tilting behavior while mounting above the rotor worsened the tilting behavior. It was discovered that there was a range of positions below the rotor for which the ABB offered significant improvements in tilting.

Sung and Chao [43] examined the effects of nonlinear damping washers on ABBs mounted to optical disk drives in 2005. Again, the authors built on their previous work [40-41] but changed the support features of the horizontal-plane rotor to nonlinear springs having linear and cubic terms. The corresponding dampers were assumed to be linear. Again, the two balancing balls were modeled as perfect spheres made of a material much stiffer than the ABB track. The balls were subjected to viscous drag forces and a rolling resistance caused by the deformation of the track at the leading edge of the balls. To simply the model, the supports were assumed isotropic and the rotor was given a constant angular speed. The equations of motion were developed via Newtonian mechanics, and the method of multiple scales was used to approximate steady-state solutions for the ball positions. Similar to [40], three physically different types of stable solutions were discovered and their stability was investigated in the presence of nonlinear springs with either a positive (hard) or negative (soft) cubic term. Type-I positions were basically unaltered by the cubic term while Type-II and Type-III positions were

enlarged with a positive cubic term and reduced with a negative cubic term. Since Type-II and Type-III positions detracted from the ability of the rotor to settle into the preferred stability of Type-I positions, soft washers were seen to provide better stability performance and would be preferred. The authors repeated previous comments [41] regarding the ability of the balls to settle into finite angular regions about the Type-I equilibrium positions of the balls given the presence of rolling resistance. These ranges were seen to increase as the friction coefficient or operating speed of the rotor increased.

Chao [44] published later in 2005 specifically on the effects of rolling friction in an optical disk drive ABB. The model for the rotor closely resembled that in [40-41, 43] with the rotor constrained to move in a horizontal plane and mounted on an isotropic support of linear springs and dampers. The ABB featured a circular race slightly offset from rotor and contained two balls assumed to be perfect spheres which maintained contact with the outer edge of the race. The effect of ball slipping was ignored since gyroscopic effects from the balls were assumed small. Additionally, the effect of micro-sliding on rolling friction and the effects of gravity were ignored. The ball material was assumed to be much harder than the race so that slight deformations occurred at the leading edge of the ball. Finally, the rolling friction of the ball was assumed to be primarily created by hysteresis material loss of the race material. Secondary causes of rolling friction included small levels of geometry inconformity between the ball and race caused by poor manufacturing, etc. The rolling friction model of the ball was derived by examining a circular-form expression for the contact pressure between a rigid ball and an elastic race. The race was assumed to undergo an "elastic hysteresis loop" of deformation that resulted in a loss of energy and created a "rolling friction moment" about the center of the ball. It was concluded that this moment depended on the normal force of the ball to the four-thirds power. The equations of motion were developed via Newtonian mechanics and approximate steady-state solutions were obtained by using the method of multiple scales. Similar to previous work, three types of equilibrium points were discovered, but only the optimum Type-I positions (distinct angular positions for each ball) were discussed in light of stability. It was found that smaller rolling friction moments would give rise to larger stability regions which would help in the overall design of ABB parameters. Similar to [43], the authors noted that the presence of rolling friction increased the attractive stability regions about the optimum Type-I positions and these regions increased as the parametric factor of race material

was increased. The parametric factor of race material was dependent on Young's modulus and Poisson's ratio for the race material, as well as the rolling friction coefficient which was based on the size of the contact area between the ball and deformed race. A simple experiment was performed to calculate the rolling friction coefficient where a rectangular plate was positioned on top of four balls that rolled along a fixed base plate. Hanging weights were attached to the rectangular plate and an accelerometer was used to capture the accelerations of the system to determine the friction forces acting at the ball and base plate interface. The balls were assumed to roll without slipping and parametric factors for copper, steel, and aluminum were developed. It was found that steel had the smallest parametric factor and offered better balancing performance by forcing the balls to settle into smaller regions around the optimum Type-I balance positions. Experiments with an ABB utilizing steel, copper, and aluminum materials closely resembled the behaviors predicted by the analytical model. Balls were seen to reach final positions very close to the regions predicted and the ABB utilizing steel had the best performance characteristics.

Chao [45] published again in 2005 and studied the effects of torsional motions on optical disk drives with ABBs. The model featured a rectangular stator held in place by four corner supports consisting of linear spring and damper pairs. Each spring damper pair was identical and allowed to resist displacements in two directions. The rotor was mounted to the stator and an ABB containing an arbitrary number of balls in a circular race was mounted to the rotor and slightly off centered due to a prescribed race eccentricity. The balls were assumed to be perfect spheres and remained in contact with the outer edge of the race. Further, the balls were assumed to roll without slip and subjected to a viscous drag force and rolling resistance moment from deformations of the contact point between the ball and the race. Additionally, the balls experienced a frictional force from surface roughness acting at the deformation point of the race that was assumed kinetic in nature. As the unbalanced rotor spun, it allowed the stator to exhibit reciprocating torsional motions. All motion was assumed to take place in a horizontal plane, and gravitational effects were neglected. Additionally, the forward shift of the contact point between the balancing ball and the ABB track due to race deformation was assumed small to simplify the force expressions for the rolling resistance moment. The equations of motion were developed via Lagrange's approach and the method of multiple scales was utilized to study the asymptotic behavior of the system. To simplify analysis, only two balls were utilized. Results showed that

there were three modal frequencies and four mathematical types of equilibrium positions that could occur, but only three physically different positions. Type-I positions were the desired positions and featured distinct angular positions for each ball and the elimination of all residual vibrations. Type-II and -III positions featured identical angular positions of the balls and had three subsets of solutions that depended on operating the rotor near one of the three modal frequencies. Type-II and -III positions only eliminated vibrations if the product of the combined mass of each ball and ABB radius was identical to the product of the rotor imbalance and its eccentricity. This is hard to do in practice and these solutions type usually only lower vibration levels. Type-IV positions featured the balancing balls settling to opposite sides of the ABB, affording no vibration reduction. Stability of each type of solution was examined with Routh-Hurwitz criteria while neglecting the eccentricity of the ABB. Stability diagrams illustrating ball mass were plotted for each type of position over a range of frequencies. The remaining system parameters were kept constant. It was seen that for Type-I solutions to be completely stable, the rotor must be operated at a frequency higher than the first modal frequency. Additionally, the maximum counter-balance which the balls can apply must be greater than the inherent imbalance in the rotor. Type-II solutions were seen to have stable regions below the first modal frequency and between the second and third modal frequencies. It should be noted that the first two frequencies were almost identical for the study and plots mainly illustrated the presence of two resonant frequencies. Type-III solutions were only stable below the first modal frequency. Additionally, the unstable region surrounding each modal frequency widened as the ball mass increased. Numerical simulations were done with Runge-Kutta routines and served to validate the stability analysis presented.

Chao [46] next published in 2007 and examined nonplanar modeling of an optical disk drive with ABB. The model closely resembled that in [42] featuring a vertically-mounted rotor operated at a constant speed which was allowed to translate in three directions and tilt in two directions. Each of these motions was resisted by a linear spring and damper pair. The ABB was attached just below the rotor and the rotor/shaft assembly was assumed rigid. The race containing the balancing balls was a perfect circle and the balls themselves were perfect spheres modeled as point masses. The balls were constrained to remain in contact with the outer edge of the race at all times and were subjected to a viscous drag force while rolling friction was ignored. A 3-2-3 Euler angle set was utilized to determine the orientation of the rotor where the last “3”

represents the constant angular speed of the rotor along its primary axis of rotation. The equations of motion were developed via Lagrange's approach with gravitational potential energy being included. Numerical simulations were conducted with the 4^{th} and 5^{th} order Runge-Kutta routines with zero, one, and two balancing balls in the ABB. Results showed that the one- and two-ball simulations were both able to effectively eliminate lateral vibrations of the rotor in a finite time, but neither was able to eliminate residual tilting angles of the rotor shaft since the plane of the ABB was below the plane of the rotor. The residual torsional angle of the rotor shaft was also seen to decay to zero in the presence of both one- and two-ball ABBs in a finite amount of time. Experiments were conducted to test the analytical model and results showed that typically, systems with an ABB do better than systems without. However, there were certain frequencies where the residual vibrations of an ABB system were greater than those without an ABB. Further, the vibration levels observed in the tilting directions of the rotor shaft closely resembled those found in the numerical studies. Finally, some experimental results showed that the performance of the ABB was drastically affected by self-resonances along each degree of freedom for the system and could create very large vibrations at certain operating speeds.

Chao [47] published in 2008 on a low-torque ball re-positioning scheme for optical disk drives equipped with an ABB. Essentially, this work utilized a previously developed model [40-41, 43] but added an active controller to aid the ABB performance. Since it was previously established that when operated above the critical speed, the balancing balls could settle into small finite ranges surrounding the optimum equilibrium positions, and thereby leave the system unbalanced, sensors were installed that would determine the vibration levels of the rotor and then lower the operating speed to near critical values. The logic behind this was that the equilibrium ranges for the balls were very small near the critical speed of the rotor and by driving the rotor to the critical speed, the balls would be forced to follow suit and be drawn closer to the optimum balancing positions. This was determined with a sliding-mode ball observer control algorithm and was verified in principle by both numerical and experimental simulations. Since this study represents a hybrid combination of both passive ABBs with active controllers, it is not in the same vein as the previous work, but nonetheless, offers important insight into ABB operation.

In 2004, Olsson [48] studied a planar unbalanced rotor with attached ABB. His work focused primarily on the motion of the balancing balls and gave little description of the forces acting on the rotor itself. The ABB was assumed to have a general radius that could be modified

to include defects in manufacturing, and contained only two balls subjected to a viscous drag force. Gravitational and Coulomb friction effects were neglected. The overriding principle guiding this research was that the balls were moving slowly relative to the rotor such that the steady-state solution for the motion of the rotor could be applied and the stability of the equilibrium points for the system readily extracted. Olsson studied the effects of a perfectly concentric track as well as eccentric and elliptic forms. As previously established, the concentric track allowed for perfect balancing while the eccentric and elliptic did not. Next, Olsson studied the effect of an axial extension of the rotor. This effectively served to make the rotor long in nature and necessitated use of two ABBs. This need was verified numerically and served to reinforce previous work.

In 2005, van de Wouw [49] published an article on ABBs that examined the influence of dry friction on the equilibrium positions of the balancing balls. The model consisted of an unbalanced rotor constrained to move in a horizontal plane and supported by an isotropic support of linear springs and dampers that each resisted displacement in one direction. The rotor was driven at a constant angular velocity. The ABB consisted of a circular track, concentric with the geometric center of the rotor, which contained two identical balls. The balls were subjected to a viscous drag force and a Coulomb friction force while gravitational effects and collisions between the balls were ignored. The equations of motion were formulated through a Lagrangian approach where Lagrange multipliers were used to model the generalized forces from the Coulomb friction. The equilibrium points of the system were computed. Van de Wouw discovered that the presence of Coulomb friction dramatically increases the size of the equilibrium set for the balancing balls and the coefficient of friction must be kept very small if the ABB is to perform well. Even with small values of the coefficient of friction, the ABB was seen to have a large deterioration in balancing properties. The angular velocity of the system was also seen to be very important and was able to dramatically increase or decrease the size of the equilibrium set for the two balls. Coulomb friction also served to make the performance of the unbalanced rotor with ABB worse than without an ABB. The author performed experiments to determine the static friction level in a single-ball ABB that was constrained to move in a horizontal plane and discovered that, in practice, the equilibrium positions of the balls are typically not spread evenly over the entire equilibrium space as predicted by the theory. The author commented that the start-up profile of the device can affect the final position of the balls

and that no definitive statements could be made about the attractivity and stability of the equilibrium points based on the numerical calculations or experiments. It should be noted that while the author did utilize Coulomb friction effects with logic decisions in his algorithm, the frictional force modeled was assumed to be either kinetic in nature or nonexistent. The complicated behavior of a static friction force was not considered. Van de Wouw did publish [50-52] on the stability and attractivity of equilibrium points in nonlinear systems which feature friction and collision through an analysis of Lyapunov stability, but conclusions were not placed in lieu of an ABB.

In 2005, Meraz [53] published his work on an ABB which featured an unbalanced rotor with flexible shaft having two balancing systems located at either end of the shaft. The rotor operated with a constant angular speed, and the deflection of the shaft center was assumed to be planar. The ABBs featured a circular groove containing a damping fluid and a total of two balancing balls. The ends of the shaft were subjected to damping and allowed to displace requiring the use of a 2-1-3 Euler angle set where the third angle corresponded to the spin axis of the rotor. Gravity and Coulomb friction was neglected for the system, and so were any torsional or longitudinal deflections of the shaft. The equations of motion for the six degree-of-freedom system were developed via Lagrange's equations and linearized to examine stability properties of equilibrium points with Routh-Hurwitz criteria. Meraz discovered that the balancing balls did not assume a position which ensured complete balancing of the rotor and he cited "rubbing of balancing bodies against the sides of drums" along with shape irregularities or improper axial weight distribution of the balancing bodies as possible causes.

Yang et al. [54] performed their work on ABBs in 2005 by considering an unbalanced rotor operating in a horizontal plane on an isotropic support of linear springs and dampers. An ABB comprised of an annular structure with a groove containing balls was mounted concentric with the rotor. The balls in the ABB were subjected to a tangential friction force and a viscous drag force. The friction force was denoted as the driving force for the balls. So long as this force was the appropriate size, the balls would be able to seek out equilibrium positions for certain operating speeds. The equations of motion were developed via Newtonian mechanics and numerically integrated. Simulations utilizing only viscous drag forces showed that the balls would seek equilibrium positions and perfectly balance the rotor above the critical speed. Simulations with both viscous and Coulomb friction forces showed that the friction served to

increase the stopping region of the balls around the optimum equilibrium positions and thus detract from the balancing performance of the ABB. Additionally, the time required for the balls to settle to an equilibrium position was greatly reduced. The balls were seen to stop at randomly distributed places within the stopping region surrounding the equilibrium positions based on their given initial conditions. It was observed that initial positions of the balls become important in whether or not they are able to reach their equilibrium positions and for certain conditions, the balls were able to balance quite effectively. Simulations involving friction forces only showed that the coefficient of kinetic friction was very important in determining where the balls came to rest: too low of a value did not stop the ball when it got close to an equilibrium point while too large of a value stopped it too soon. In some cases, if the rotational speed were not fast enough, the balls would settle to positions that only marginally improved the balancing characteristics of the rotor. Without the viscous forces, the balls were seen to always fall into an oscillatory steady state where they continue moving in a direction opposite to the rotation of the system. Sometimes one ball would continue traveling in circles around the groove while the second ball oscillated about its equilibrium point. It was also observed that certain system conditions would allow one ball to continue increasing its velocity as it traveled around the ABB groove. The author commented that if impacts had been taken into account in the model, this type of motion would have to change when the two balls collided. Yang also commented that by utilizing severe disturbances and running the motor near critical speeds, it is possible relocate the balls to better balancing positions. One final comment from his analysis indicated that the most important design parameter appeared to be the mass of the balancing balls. Systems were seen to perform better utilizing small balancing masses instead of large ones. Experiments were conducted that utilized different types of oil to simulate different frictional values in the ABB. The data supported the finding that friction tends to reduce the ability of a system to balance perfectly. The experiments were also able to illustrate the oscillatory steady state mentioned previously.

Horvath et al. [55] published their work in 2005 on an AB utilizing pendulum balancers. The model consisted of an unbalanced, thin disk constrained to move in a horizontal plane and was supported by an isotropic support consisting of linear springs and dampers. Two arbitrary point-mass pendulums were attached to the geometric center of the disk with frictionless pivots which were subjected to torsional damping. Gravitational effects were neglected in the analysis.

The equations of motion for the system were derived via Lagrange's approach and the equilibrium points were computed. The equations were linearized to examine the stability of the equilibrium points with the Hartman-Grobman theorem. Analytically, it was demonstrated that the two pendulum balancers could eliminate all unwanted vibrations above the critical speed. Below the critical speed, the pendulums did find a stable equilibrium point, but the system continued to exhibit whirling motions about the primary spin axis. Experimental studies confirmed this finding and it was also discovered that if the pendulums were not identical, they were still able to eliminate most of the vibrations, but not all. The author commented that it is difficult to create identical pendulums because of manufacturing inconsistencies. He further commented that it is not possible to mount identical pendulums in the same plane to perfectly balance the rotor. Experimentally, pendulums were observed to acquire the same equilibrium positions and have the same balancing effect on a rotor given numerous startups. An ABB was also developed experimentally and observed to behave rather differently. The balls in the ABB acquired numerous equilibrium positions upon startup and were shown to improve rotor performance, but not as consistently as the pendulums. The authors did not explore these inconsistencies with the ABB but did comment that deformations of the contact point for the ball and ABB as well as frictional forces were likely the cause. The authors commented that given the better distribution of forces acting on a pendulum balancer, as compared with a ball balancer, the pendulum balancer would have better sensitivity and perform better than an ABB.

Horvath [56] continued his work with pendulum balancers in 2008. He cited that pendulum balancer eliminate deficiencies found in ABBs such as "contacting between the balls as the balance state is sought by the system, which leads to impulsive excitation and transient disturbances." Horvath also cited the typical use of liquids in ABBs to apply damping forces to the balls and how many applications such as sensitive electronics need to be run clean and dry, without the possibility of fluid leakage. Horvath last cited the large influence of stiction effects caused by friction on the balls ability to consistently balance the device in question. The model utilized was identical to that in [55] featuring a thin, unbalanced disk operating in a horizontal plane without the influence of gravity and supported by an isotropic support consisting of linear springs and dampers. The main difference in this work was that the pendulums could be attached with frictionless pivots to positions offset from the geometric center of the disk. It was observed that as the pendulums moved further from the geometric center of the disk, the

vibration levels worsened. The author concluded that pendulum balancers are quite sensitive to pendulum placement and care should be taken to precisely align the pendulum pivot with the center of the disk or rotor. Experimental studies confirmed this trend in the pendulum balancer by closely resembling the numerical investigations. Horvath next commented on the forces which act on both a pendulum balancer and a ball balancer. He noted that the resisting force for a pendulum balancer is concentrated at the support bearing which typically lies close to the geometric center of the shaft. A ball balancer features resisting forces which are concentrated at the outer edge of the balancer and involve frictional effects. This difference lowers the resisting torque in a pendulum balancer and serves to improve its balancing performance. Experiments, similar to those in [55] were performed and showed the variance in equilibrium positions of the balls given different startups. The authors commented that "this effect can be traced to the deformation of the contact point of the ball and channel surface due to the high normal forces generated by the centripetal acceleration and the resulting dry friction." The deformation of the contact surface was said to create a pocket which confined the ball and altered the directions of the constraining forces on the ball and opposes its ability to reach the optimum equilibrium position.

In 2006, Green and Champneys [57] published on ABBs and performed a bifurcation analysis of the equilibrium positions. Their model featured an unbalanced rigid rotor with its geometric center attached to a rigid shaft. The rotor was mounted on an anisotropic support of linear springs and dampers which resisted motion in a single direction. The rotor was driven at a constant angular speed by an external moment applied to its geometric center. Motion was assumed to take place in a vertical plane and gravitational effects were included. An ABB was fitted to the rotor which contained an arbitrary number of balancing balls modeled as point masses. The balls were subjected to viscous drag forces resulting from contact with oil in the ABB race. Coulomb friction effects were ignored in the analysis and the balls were not allowed to collide. The equations of motion were developed by using Lagrange's approach. In the potential energy analysis, the slight change in height of the rotor mass center, caused by the imbalance mass, was neglected. The equations were nondimensionalized and the support was changed to be isotropic. Additionally, gravitation effects were neglected by requiring the rotor to rotate at high speeds or be placed in a horizontal plane. The authors utilized only two balancing masses to simplify calculations. Three types of steady-state solutions were discovered. State-1

positions were balanced and featured distinct ball positions that eliminated vibrations of the rotor center. The other solutions, States 2 and 3, were unbalanced and featured finite residual vibrations of the rotor. State-2 positions had the balancing balls at the same angular position, while State-3 positions featured balls at opposite angular positions within the ABB. State-1 positions were possible only if the mass of the balls was large enough to cope with the rotor imbalance. Numerical bifurcation diagrams were developed through an ad hoc Routh-Hurwitz method with a scatter-gun approach to search for boundaries of stability and instability. The bifurcation diagrams examined the stability of the equilibrium solutions by examining the ratio of a balancing mass to the rotor mass, the ratio of operating speed to critical speed, the ratio of rotor eccentricity to ABB radius, the viscous damping factor for the rotor, and a viscous damping factor for the balls. The authors found that for certain values of the system parameters, no stable states were discovered. Additionally, the State-3 positions were always unstable and could undergo Hopf bifurcations at certain operating speeds. Numerous plots were created which illustrated a very complex behavior of the equilibrium points in parameter spaces. It was found that State-1 positions were always stable if the operating speed was 2.5 times the critical speed of the rotor. If the operating speed was between the critical speed and 2 times the critical speed, the details of system behavior were seen to be quite varied. Next, the authors examined the limit cycle behaviors of the system. They discovered that having system parameters that gave rise to stable equilibrium positions was not enough to ensure vibration reduction or elimination through the ABB. Additionally, the operator must specify appropriate initial conditions to ensure the balls will be attracted to a balanced state and not an intermediate limit cycle or non-steady state. The authors commented that the scope of parameter values explored in the analysis was very brief, but more in-depth studies would only be advantageous if a specific rotor and ABB were being studied. Numerical simulations were performed which showed the system settling into limit cycles from certain initial conditions. Simulations also showed the ability of the ABB to balance the rotor for certain operating conditions and the presence of residual vibrations given poor initial condition choices. In several instances, the ABB was seen to worsen the performance of the rotor. In other instances, the system exhibited large oscillatory transient behavior before settling into a balanced equilibrium state. The authors noted that increasing the damping of both the rotor and the balls appeared to have a “highly desirable effect.” It was also observed that if the operating speed was 1.9 times the critical speed, as long as the balls were

launched from directly opposite positions within the ABB, the system always settled into a balanced State-1 position. Finally, the authors commented that even if the ABB was able to balance the rotor, the initial transient motions of the system can be quite large.

Green and Champneys [58] published again in 2006 with the goal of analyzing the transient motions found in ABBs on unbalanced rotors. The model was identical to [57] and featured an isotropic support with two balancing masses. The equations of motion were written in a nondimensionalized form, linearized about the equilibrium points, and converted to state-space form. Examination of the eigenvalues associated with the coefficient matrix of the state-space equations, showed that the rotor responded more at high frequencies while the balancing masses responded more at low frequencies. Additionally, the eigenvalue problem for the system was observed to be ill-conditioned since many of the eigenvalues have large real components making them sensitive to perturbations. Further, the coefficient matrix was highly non-normal and small perturbations can lead to large changes in the possible eigenvalues. The authors next examined the pseudospectra of the eigenvalues by subjecting the system to both structured and random perturbations. Contour plots of the eigenvalues were developed and the eigenvalues most sensitive to perturbations were seen to move from a negative real part to a positive real part with increasing perturbation levels. It was found that random perturbations would most likely show up in changes in the motion of the rotor itself through excitation of the natural frequencies. However, if perturbations were applied to only the stiffness matrix, the effect would most likely be visible in a change in the motion of the balancing balls. Numerical simulations of the nonlinear equations were run to validate the pseudospectrum analysis. The system exhibited large transient motions that decayed according to the rates predicted by the linearized system. Additionally, for low values of damping, the trajectory decayed "according to the most sensitive eigenvalue," as predicted by the pseudospectrum analysis, "and not necessarily the right most eigenvalue." The authors found the pseudospectrum analysis of a first order system to give better accuracy on transient behaviors than an analysis with second order systems. Changing the damping of the system also led to different behaviors in the transient motion. Increasing the internal damping led to a highly oscillatory exponential decay while increasing the external damping led to a sharp drop in the initial response followed by an asymptotic decay. Increasing the damping internally and externally did result in a balanced steady state, but increasing the external damping too much was seen to alter the attractive nature of the balanced steady states.

The authors' final comment was that "One must balance the want of decreasing the length of transient response by increasing the external damping ratio with the threat of unwanted instabilities." Much of the work in [57-58] was also repeated and examined by Green, et al, [59] later in 2006.

Also in 2006, Rodrigues and Champneys [60] published on two-plane balancing of an unbalanced rigid rotor with a misaligned shaft. The model featured a balanced rotor shaft that held two point mass imbalances at two distinct locations along the shaft. The size of each mass was different, and they were given unique angular and radial positions about the primary axis of the rotor. The ends of the rotor were mounted in isotropic supports consisting of three pairs of linear springs and dampers. Each spring and damper pair was able to resist motion in three directions since the rotor shaft was allowed to displace in three directions and rotate in three directions through a 1-2-3 set of Euler angles. Two ABBs were attached to the shaft outboard of the imbalances and mass center of the shaft. Each ABB consisted of a circular track containing an arbitrary number of balancing balls and was oriented normal to the rotor axis. The radii of the two ABBs were different, and the balancing balls were modeled as point masses that were not assumed to be equivalent. The balls were subjected to viscous drag forces from fluid in the ABB, but were not subjected to Coulomb friction or gravitational forces. The equations of motion were developed via Lagrange's approach. For simulation purposes, the rotor was operated at constant speed with the center of mass exhibiting only planar motion. The imbalance masses were assumed identical in size, radial distance from the rotor axis, and lateral displacement along the rotor shaft on either side of the center of mass. Further, the balancing balls were assumed identical and each ABB had the same radius and fluid damping characteristics. Numerical simulations of the nonlinear equations were conducted and the ball damping parameter was studied. For an underdamped case, the balancing balls completely destabilized and made the rotor vibrate worse than when it only possessed the imbalance masses. With high damping, oscillations were prevented but the balls took a long time to settle into a balanced position. Ideally, critical damping was sought where the balancing balls will eliminate any residual vibrations in the shortest amount of time.

Rodrigues and Champneys [61] published again on two-plane balancing in 2008 by expanding their work in [60]. The model consisted or a rigid rotor supported at either end by a compliant linear bearing. The bearings consisted of a single spring and damper which resist

displacement in two directions. The bearings were not assumed to be identical. The geometric center of the rotor was confined to move in a horizontal plane because axial motions were ignored, but the rotor was allowed to rotate in essentially three directions through a 3-2-1 Euler angle set. The first angle in the Euler sequence represents a constant angular speed for the rotor along its primary axis. The mass center of the rotor was considered to be slightly offset from its geometric center and left the system unbalanced. The inertia matrix for the rotor was assumed to be diagonal lacking product of inertia terms. Two ABBs were mounted to the rotor on either side of the mass center and oriented normal to the rotor axis. Each ABB featured two identical balls that rolled along a circular race filled with viscous fluid which provided damping. Coulomb friction and gravitational effects were neglected in the analysis. The radii for the ABBs were identical. The equations of motion were formulated via Lagrange's approach and nondimensionalized. It should be noted that cubic terms were neglected in the kinetic and potential energy analysis and the rotor was assumed to be "long," meaning the transverse moment of inertia was greater than the polar moment of inertia. Steady state solutions were computed, and at most, only one physically unique balanced state was observed. The other three types of equilibrium solutions featured balls coincident with one another in either one or both of the ABBS which left residual vibrations in the system. Complete balancing was only possible if the balls had enough mass to counteract the imbalance in the rotor. Numerical simulations with the nonlinear equations of motion were conducted for both static and dynamic imbalances in the rotor. Bifurcation diagrams similar to those in [57] were developed. Since the rotor was allowed more angular freedom in this model, the steady-state positions had smaller regions of stability. No stability was possible below the critical speed, and the equilibrium points exhibited very complex stability behavior for parameter surveys of the rotor damping, shaft eccentricity, and ball damping. Additionally, Hopf and pitchfork bifurcations were observed for the tilting motions of the rotor at several locations in the balancing mass parameter space leading to areas of destabilized steady-states surrounded by areas of stability. For the dynamic imbalance, bifurcations diagrams showed that high rotation speeds coupled with high support damping drastically reduced the size of any stability region. Next, numerical simulations were conducted to verify the behavior of equilibrium points from the bifurcation analysis. In some instances, the ABB was seen to balance the rotor but had initial transients with amplitudes much higher than the unbalanced rotor operating in the absence of the ABB. Other parameter values led to

severely degraded balancing characteristics of the rotor in the presence of an ABB where vibration amplitudes increased, in some cases, by a factor of ten over the non ABB rotor. These simulations served to illustrate the highly nonlinear equations and extremely coupled effects between numerous parameters, and the difficulty in trying to achieve consistently balanced rotor performance. The authors commented that the out of plane tilting motions for the rotor create many new bifurcation points in the various parameter spaces the stability relationship between the rotor and system parameters is highly complex.

Rodrigues and Champneys [62] continued their work on two-plane balancing with ABBs later in 2008 by considering device asymmetries and the effect of the rotor run-up on the balancing characteristics of an unbalanced rotor. The model was identical to that found in [61] featuring a rigid rotor supported on either end by two linear compliant bearings and driven at a constant speed. Two ABBs each containing two identical balls subjected to viscous damping were mounted normal to the rotor axis on either side of the rotor center of mass. The equations of motion were developed via Lagrange's approach and transformed into complex variable space to better examine equilibrium positions and stability properties. For balancing to occur, the angular speeds of the balls were required to synchronize with the rotational speed of the rotor and the balls must orient themselves to cancel out any imbalance in the system. Effects such as isotropic bearings, dry friction between the race and the balls, and gravity were not included in the analysis. Additionally, no geometric defects were modeled. Bifurcation diagrams of static and dynamic imbalances were created and the relationship between eccentricity and spin speed was explored. It was found that for a disk type rotor, where the polar moment of inertia is greater than the transverse moment of inertia, possessing a static imbalance, there was no stable region of balanced operation. The authors reported that "influence of the gyroscopic term is such that the eigenfrequency corresponding to the conical whirl is always greater than the rotor speed." Therefore, the rotor will not be balanced with respect to conical motions. Further, since the conical mode has no associated critical speed, disk rotors need only be balanced with respect to static imbalances and require only a single-plane ABB. For a long rotor, having either a static or dynamic imbalance, stable balanced regions do exist primarily above the second critical speed of the rotor. Changing the mass of the balls relative to one another or spacing the ABB planes unequally about the geometric center of the rotor did little to affect the overall balancing characteristics of the ABB but did increase the complexity of the associated bifurcation

diagrams. It should be noted that changing the mass of the balls relative to one another was shown to sometimes hinder the performance of the ABB since the balls were not allowed to settle to symmetric positions about the imbalance. Next, the effect of the rotor run-up was explored. It was found that as the rotor speed was slowly increased, the balls would synchronize with it. While passing through the critical speeds, the balls would become stalled, travelling at the eigenfrequency of the rotor, and then resynchronize with the rotor after a short period. This is similar to the Sommerfeld effect in which a machine has difficulty passing through a critical speed because its driving motor is weak. It was also seen that if the damping value of the ABB was too small, the balls might never resynchronize with the rotor. As the balls resynchronized with the rotor, there was a period of increased vibrations in the rotor caused by a desynchronizing of the balls with one another. In an effort to eliminate this effect, it was commented that the race could be partitioned or the balls held stationary relative to the rotor until supercritical speeds were reached.

Rodrigues and Champneys [63] continued work on the two-plane balancing with ABBs later in 2008 by considering the effects of support asymmetries. The model used was identical to that in [61] featuring a rigid rotor supported on either end by two linear compliant bearings and driven at a constant speed. The bearings were assumed to have different stiffness and damping characteristics. Two ABBs featuring a circular race and each containing two identical balls subjected to viscous damping were mounted normal to the rotor axis on either side of the rotor center of mass. The equations of motion were developed by using Lagrange's approach, written in complex notation and linearized. The effects of geometric nonlinearities based on the tilting motions of the rotor were ignored along with gravity, variable spin speeds, and dry friction between the balls and the races. Bifurcation diagrams of the equilibrium points were considered in relation to balancing mass and rotor eccentricity. The authors discovered that support asymmetry and anisotropy did little to alter the stable regions for the equilibrium positions so long as the rotor was operated above the critical speed and its inherent imbalance was not extreme.

Also in 2008, Green and Champneys [64] returned to their work on planar rotors by considering an ABB carrying multiple balls. They returned to a model of a planar rotor mounted on an isotropic support of linear springs and dampers that each resisted motion in a single direction. The rotor was operated at a constant speed and a partitioned ABB carrying three or

four identical balls traveling in a race was fitted to the rotor. The balls were modeled as particles and forced to remain in contact with the outer edge of the race. The balls were subjected to viscous damping from contact with fluid in the race. Dry friction effects and gravity were not considered. The equations of motion were developed via Lagrange's approach and nondimensionalized. The steady state solutions were computed and stability criteria were developed. Three types of equilibrium positions were found but only State-1 positions were balanced. State-2 and State-3 positions still afforded slight residual vibrations of the rotor. Bifurcation diagrams of three and four balls were developed in relation to a parameter space consisting of the viscous damping factor of the support, damping factor for the balls in the race, ratio of rotor eccentricity to ABB radius, and ratio of balancing mass to rotor mass. The bifurcation diagrams showed very small stability regions that allowed for balanced rotor performance. This result was contrary to that in [57] and suggests that using more balancing masses appears to have a detrimental effect on the ability for the rotor motion to stabilize into a balanced state. If the viscous damping factor was too high, the rotor could not achieve complete balancing even though the theoretical model permits it. Bifurcation diagrams showed that the small stability region around the critical speed in a two-ball ABB disappears for the three-ball ABB and resurfaces when four balls are used. Overall, it was found that increasing to four balancing balls further deteriorated the performance and stability of the rotor. Numerical simulations showed a three-ball ABB, operating under parameters that would ensure State-1 balanced stability for the balls, was able to eliminate rotor vibrations, but did so with large chaotic motions for the first part of the simulation. These chaotic motions could be detrimental to the rotor performance and result in large stresses. Further, operating the rotor above the critical speed did not guarantee stability and the rotor often illustrated large chaotic motions that never settled to a steady state. The authors comment on the need for a partitioned race since the balls tended to slip relative to the rotor during speed increases through resonances. By partitioning the race, all equilibrium positions that feature several balls coincident with one another would be eliminated, and collisions between the balls would be prohibited. Conversely, for certain parameter ranges or imbalance locations, fixed partitions would not allow the balancing masses to effectively eliminate rotor vibrations. To allow for collisions of the balls with their partitions, a simple impact law was used that multiplied the relative angular velocity of a ball by the coefficient of restitution for the partition with a corresponding direction reversal. It

was found that for certain ranges of operating speed, the balls would continuously impact the partitions which served to create large oscillatory vibrations of the rotor. For high enough operating speeds, the balls were seen to impact the partition only once and then settle into a stable balanced state. Experiments were conducted to verify the numerical results and showed that a three-ball partitioned balancer could be effective for certain operating conditions. Numerical bifurcation diagrams were created to compare with experimental results. The nonlinear equations were numerically integrated with an event-driven algorithm to check for impact of the balls with the partitions. The results differed from the experimental conclusions and the authors commented that "the balls may be subject to additional stiction forces. . . [that] are not incorporated in our current model." There was partial agreement of some findings with the experimental results, but the authors noted that it would be worth further study to examine the combined effects of viscous damping and stiction effects in the future.

In 2006, Lu [65] published his first article on ABBs by examining an optical disk drive. His model consisted of an unbalanced disk operated with a constant angular speed, constrained to move in a horizontal plane, and supported by an isotropic foundation consisting of linear springs and dampers that could each be displaced in two directions. An ABB was fitted to the disk which contained a single ball, treated as a particle, which traveled in a circular groove and was subjected to viscous drag forces. Coulomb friction and gravitational effects were ignored for this work. The equations of motion were developed via Lagrange's approach and nondimensionalized. The equilibrium positions were computed and it was found that perfect balancing was only possible if the ball had the same eccentricity as the disk and was located directly opposite the center of mass of the disk. Next, the equations of motion were linearized and Routh-Hurwitz criteria was used to examine the stability of the equilibrium positions. It was found that stability was only possible with operating speeds greater than the critical speed of the disk. Further, no stable equilibrium points existed above the critical speed when there was no support damping.

Lu [66] continued his work in 2008 by extending his analysis of the system in [65] This model featured a three-ball ABB. Lu again ignored Coulomb friction effects between the balls and the ABB, and cited the need for viscous only effects because "experimental results conducted with optical disk drives agree well with the analytical predictions based on the viscous damping model," and that the neglect of Coulomb friction effects "greatly simplifies the analysis

work." The rotor was operated with a constant angular speed in a horizontal plane and was attached to an isotropic support of springs and dampers that could displace in two directions. The equations of motion for the five-degree-of-freedom system were formulated via Lagrange's approach and the equilibrium positions for the balls were computed. The stability of the equilibrium points was first analyzed with a linearized system and Routh-Hurwitz criteria. One eigenvalue was seen to always have a zero real part which made stability analysis of a linearized system impossible. The author resorted to examining steady-state behaviors subjected to disturbances. The author compared the time integration of the equations of motion for the three-ball ABB with the previous two-ball ABB and found that the three-ball ABB had larger regions of stability and seemed to be more robust at balancing.

Lu et al. [67] published again in 2009 and examined a two-ball ABB. The model again consisted of an unbalanced rotor constrained to move in a horizontal plane with a constant angular velocity. The rotor was held by an isotropic support of linear springs and dampers which could each be displaced in two directions. The ABB consisted of a circular track concentric with the geometric center of the rotor, containing two balls subjected to viscous drag forces only. Coulomb friction and gravitational effects were again ignored. The equations of motion were formulated via Lagrange's approach. The equilibrium positions were determined, and the equations of motion linearized to utilize Routh-Hurwitz criteria to examine the stability of the equilibrium positions. A total of six equilibrium positions were computed with four of them always being unstable. Additionally, there was at most one stable equilibrium position for any given rotational speed. Experiments served to validate the theory by showing the performance of an unbalanced rotor improving dramatically with an attached ABB in regions above the critical speed.

In 2007 Majewski [68] modeled an ABB attached to a rotor on a long flexible shaft attached to pinned supports. The author gave little information on the characteristics of the supports and the forces which acted on the balls in the ABB, but it is certain that gravitational effects were ignored along with any Coulomb friction effects. The balls were subjected to a viscous damping force, and the shaft which held the ABB and the rotor was allowed to displace and twist. The equations of motion for the six-degree-of-freedom system were developed via Lagrange's approach and simulated numerically to discuss performance of the ABB. It was observed that when the ABB was placed in the plane of the rotor, it would completely balance

the system. If the ABB was not in the plane of the rotor, complete balance could not be achieved but vibration amplitudes were decreased. The simulations utilized only one plane for balancing, but the author did mention that two planes would be beneficial.

In 2007, Filimonikhina and Filimonikhin [69] published their work on the general theory of ABs. No specific type of balancer (pendulum, ball, etc.) was illustrated, and the model consisted of a freely spinning body with attached AB. Damping was ignored, and the authors examined the effect of both static and dynamic imbalances on the body. Energy methods were used to determine requirements for the axial moment of inertia of the body if the AB was to effectively balance the system. The authors discovered that both prolate and oblate bodies can be balanced effectively for static imbalances given certain characteristics of the moment of inertia about the primary axis of operation. Further, for dynamic imbalances, it was discovered that prolate bodies are able to effectively balance the system given certain characteristics of the primary moment of inertia. Conditions for oblate bodies in dynamic balancing were not given in the paper possibly because two planes are necessary for dynamic balancing of a rigid body which places the two planes relatively close together in an oblate body.

Cheng et al. [70] published their work on an AB in 2007. Unlike previous attempts utilizing balls, pendulums, etc., the authors utilized a speed-dependent vibration absorber (SDVA) to counteract imbalances on an optical disk drive. The SDVA consisted of a central hub that connected to the operating shaft of the disk drive, and four ribs which connected the central hub to a thin outer rim. By altering the geometric properties of the SDVA, its natural frequency could be altered by the spin hardening and spin softening effects found in rotating flexible bodies. This frequency alteration was then optimized with shape optimization techniques and finite element simulations. The finite element simulations produced four natural modes for the SDVA, and the natural frequency corresponding to the radial deflection mode shape was chosen as the mode to optimize to counter imbalances in the disk drive. A finite element simulation was run to determine the performance of the SDVA on a disk drive and it was discovered that the SDVA could effectively suppress vibrations from imbalances for a wide range of operating speeds. Two SDVA prototypes were developed for experimental purposes and were shown to effectively reduce the radial vibrations of the disk drive. This version of an AB is considered to have a simple mechanism and simple fabrication procedures since it can be produced with an injection mold machine.

Ehyaei and Moghaddam [71] published an ABB paper in 2009 that featured a flexible rotating shaft (Stodola-Green rotor model) equipped with an arbitrary number of ABB devices. A general number of mass imbalances were distributed along the length of the shaft and concentrated masses (representing bearings) were placed at the ends of the shaft. The ends of the shaft were supported by linear isotropic elastic supports that included damping. Each support was allowed to have different characteristics from the other. The shaft was assumed to obey Euler-Bernoulli beam theory and its internal damping was neglected. The centroid of each ABB was assumed to exhibit only planar motion, but each ABB was allowed to rotate through a 3-1-2 Euler angle set about its centroid and this motion was resisted by translational and rotational damping effects. Each ABB was equipped with a general number of balls that traveled in a circular groove and were subjected to viscous drag forces with Coulomb friction and gravitational effects ignored. The equations of motion were formulated using Lagrange's approach. The torsional and longitudinal deflections of the shaft were neglected in the potential energy analysis and the balls were treated as particles for the kinetic energy analysis. The equations of motion were linearized and the stability of the corresponding equilibrium points was examined with Routh-Hurwitz criteria. To simplify their analysis, the authors limited their model to one imbalance mass and two ABB devices, each containing two balls. The authors concluded that perfect balancing is only possible when the imbalances lie in the same planes as the ABBs, but balancing is improved when the ABB devices lie close to the planes containing the mass imbalances. Further, the authors concluded that balancing was only possible in regions above the critical speed and the viscous damping coefficient of the ABB fluid is especially important for determining the balancing characteristics of the ABB devices.

Liu and Ishida [72] published their ABB work in 2009. Their model featured a vertically oriented rotor attached to the center of an elastic shaft having linear stiffness and damping characteristics. The ABB consisted of two balls mounted in a circular ring that surrounded the rotor. The balls were subjected to a viscous drag force within the ring and gravitational and Coulomb friction effects were ignored. Additionally, the balls were assumed to always remain in contact with the ring and the rotational energy of the balls was neglected. The outer edge of the ABB was allowed to contact a second ring supported in two directions by linear springs and dampers. It was assumed that the spring stiffnesses and damping coefficients for this support were identical in the two planar directions, and the supporting springs were preloaded to provide

discontinuous load characteristics. Motion of the system was constrained to a horizontal plane and the rotor had a constant angular speed. The equations of motion for the four degree-of-freedom system were developed via Lagrange's approach and linearized to examine stability properties of equilibrium positions of the balls with Routh-Hurwitz criteria. The authors first examined the stability of an ABB alone and found that complete balancing could only be achieved above the critical speed of the rotor. Additionally, they discovered large-amplitude rotor vibrations occurring in a small range beyond the critical speed. When the outer ring containing discontinuous springs was used, the large-amplitude vibrations were shown to be greatly reduced in the neighborhood of the critical speed and verified with experimental results.

Also in 2009, DeSmidt [73] published his work on an ABB that featured an unbalanced rotor mounted on a flexible shaft supported at either end by linear isotropic elastic supports that included damping. The supports were assumed to have equivalent characteristics. The shaft was driven with a constant angular speed, considered to obey Euler-Bernoulli beam theory, and included internal damping. The rotor and attached ABB were allowed to displace due to shaft flexure. The ABB device consisted of a circular groove containing two balancing balls and was placed in a plane separate from the rotor. The balls were subjected to a viscous drag force but no Coulomb friction effects or gravitational forces. The equations of motion were formulated via Lagrange's approach with the rotational energy of the shaft being neglected. Numerical investigations on the localized equilibrium positions of the balls showed that the initial angular position of the balls relative to the ABB did not matter. Investigation of the steady-state response of the rotor showed that the localized equilibrium positions of the balls were highly dependent on the operating speed of the shaft and its proximity to the lateral rotor natural frequencies. The balls were seen to increase vibrations below the critical speed of the rotor, and to reach symmetric positions which cancel the effects of the imbalance when operated above the critical speed. As the speed was increased, the two balls moved closer together until they reached a merged (saturated) position where they lay on top of one another. This behavior was observed for each bending mode of the rotor. DeSmidt also found that within certain stable operating ranges between critical speeds of the rotor, the ABB could fully suppress any lateral vibrations. In regions surrounding the critical speeds, the ABB could reduce vibrations, but not eliminate them. This was due to the offset between the ABB plane and the rotor plane. Additionally, it was shown that decreasing the viscous forces acting on the balls served to

decrease the ranges of stable equilibrium speeds but have no effect on the speeds at which transition of the balls from a balanced position to a merged position occurred. It was also shown that the most effective location for the ABB was in the plane of the imbalance. Time-domain simulations showed, contradictory to the localized equilibrium analysis, that the initial angular positions of the balls strongly affected their ability to reach a global equilibrium position in the ABB. Further simulations did show that the ABB could correct imbalances as long as the rotor was operated above critical speed.

1.3 Present Contributions

As outlined above, previous and current research attempts utilizing ABBs typically feature a number of simplifications aimed at reducing the numerical complexity of simulations and tractability of analytical solutions. Due to the inherent complexity, strong coupling, and high nonlinearity of the rigid-body dynamics and multi-body interactions featured in ABBs, such simplifications could possibly lead to inaccurate predictions on system behavior. Researchers tend to assume only planar motions in their models, and those that offer three-dimensional motions usually require that the geometric center of the rotor displace in a planar fashion. Further, most authors neglect the effect of gravity on the balancing masses in the AB by either assuming rotor spin speed is high enough to generate large radial forces that overwhelm the effects of gravity, or that the rotor itself lies in a horizontal plane with gravity acting perpendicular to the plane of motion. Additionally, most research attempts utilize only a single balancing plane (due to an assumption of planar motions) which requires the location of the imbalance be known a priori if the ABB is to balance the system. In practice, this is often not the case, and researchers cite the need for two or more balancing planes if the rotor is to have satisfactory performance.

A major shortcoming in nearly every research effort to date is the assumption that balancing particles are subjected to only viscous damping forces while Coulomb friction forces are neglected. This assumption is made in an effort to eliminate the intense logic decisions required for numerical solutions and to allow tractable forms for equilibrium point equations and simple stability analyses for balanced ABB performance. Those research attempts that do include dry friction typically assume a rolling friction model that does not allow for slipping of

the balancing particle relative to the ABB surface. Again, this simplification is made to more easily describe the motion of the balancing particles, and such a simplification is an idealization at best and not always descriptive of the actual motions in ABs. The one or two models which include Coulomb friction are greatly simplified by assuming only a single balancing particle that has only planar motions. Further, these papers neglect the variable nature of the Coulomb friction force when the balancing particle comes to rest relative to the rotor.

It is the goal of this work to address some of the above shortcomings and offer a more realistic model of true ABB performance, specifically as it pertains to Coulomb friction effects. To that end, a system comprised of a six-degree-of-freedom elastically-mounted cylindrical rotor featuring dual-plane ABBs is modeled. The cylindrical-rotor elastic mounting includes three translational degrees of freedom for the body geometric center and three rotational degrees of freedom. Linear damping is included for each of these six degrees of freedom. The model for the automatic balancer consists of up to two arbitrarily-located hollow circular races, each of which contains up to two balancing masses modeled as particles. The friction model for the particles includes both viscous and Coulomb forces. Of considerable complexity is the logic path for the individual particles being either in motion or stationary relative to the rotating body. The exact equations of motion for the overall system are derived via a Newtonian approach, and these equations are integrated by a custom four-cycle Runge-Kutta numerical integration scheme.

1.4 Dissertation Overview

Chapter 1 has offered a rather exhaustive study on previous research attempts in the realm of automatic balancers and identified the state-of-the art of ABs. In Chapter 2, the equations of motion for the proposed system will be developed utilizing rigid-body kinematics and kinetics. Chapter 3 will offer a discussion of the various cases of motion that can be exhibited by the system of four balancing particles and cylindrical rotor, as well as a treatment of the necessary programming logic and numerical integration strategy required for simulations. Chapter 4 will give the reader a brief overview of the governing theory which makes ABs practical for some applications. Chapter 5 will discuss the results of a rather exhaustive study on the effect of the coefficients of static and kinetic friction between the particles and the race. Chapter 6 will give a summary of the results obtained from the numerical studies and offer conclusions and insights into any observed phenomena. Additionally, a brief discussion regarding steps for future work will be included in this final chapter.

2 Derivation of System Equations of Motion

The problem of interest involves a six degree-of-freedom cylindrical rotor of mass M that possesses a mass imbalance, represented by a particle of additional mass m at some arbitrary location in its occupied spatial region. An imbalance can arise from imprecise manufacturing processes or from damage that has resulted in a local change in the shape of the body. The existence of such an imbalance places the center of mass for the rotor at some distance from the primary axis of rotation and will result in unwanted and possibly detrimental translational and rotational vibrations of the body. In an effort to counteract the effects of the imbalance, two small races are fitted to the inner radius of the rotor, each carrying two balancing masses which are modeled as particles. The races lie in different transverse planes that are offset from the mid-plane of the rotor to provide a two-plane balancing effect to oppose any possible static and dynamic imbalance caused by the point mass m. This situation is illustrated in Figure 2.1. In the figure, the red particles represent balancing particles m_j and the blue particle represents the effective mass imbalance m of the rotor.

The rotor is allowed to displace from an inertial point O and allowed to rotate to new orientations. Its displacement from O is resisted by three linear springs and three dampers, with one spring-damper pair being parallel to each of the inertial X, Y, and Z-axes. It is assumed that each spring and damper remains parallel to its respective axis. Additionally, rotation around additional coordinate axes is resisted by three torsional springs and three torsional dampers. The torsional springs and dampers resist rotation about intermediate coordinate system axes as the rotor turns from its inertial orientation to its final orientation due to the effects of the imbalance. It is assumed that the torsional springs and dampers in the intermediate coordinate frames do not have components directed along the primary axis of rotation for the rotor. The displacement and orientation of the rotor is illustrated in Figure 2.2.

2.1 Kinematics

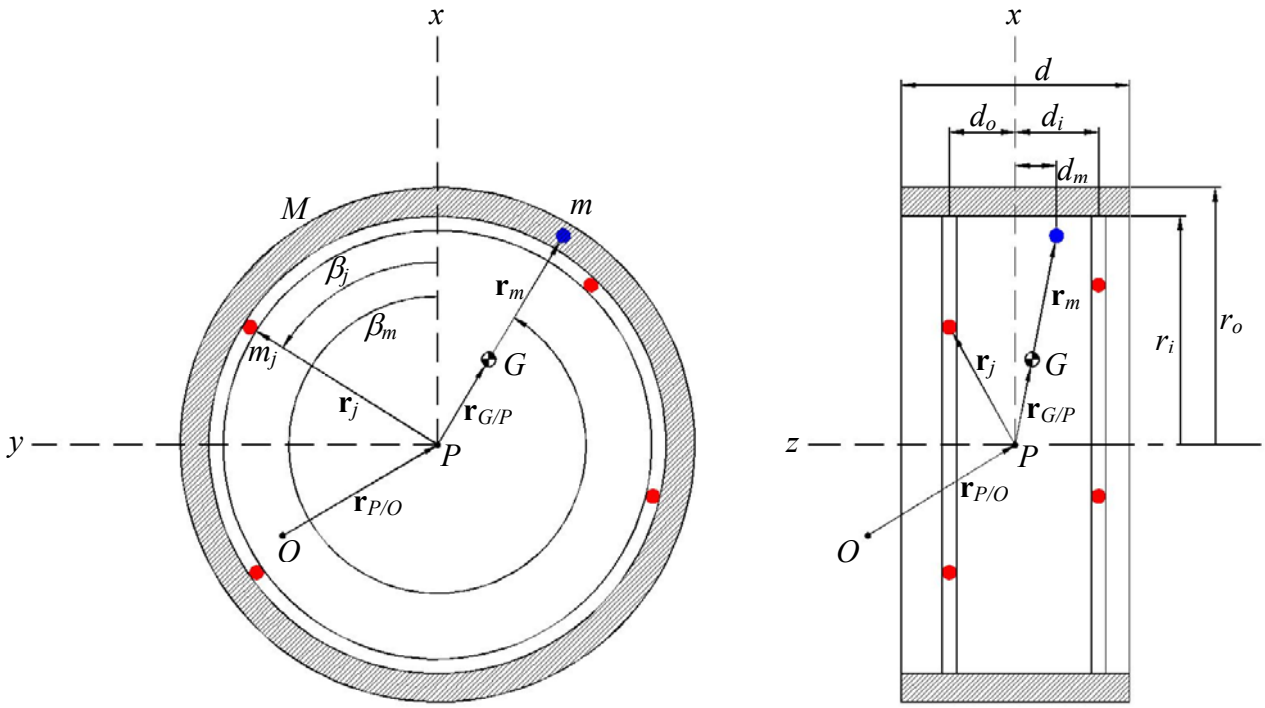

Figure 2.1: Longitudinal and Transverse Cross Sections of the Cylindrical Rotor

2.1.1 Coordinate Transformation Matrices

The rotor has a primary axis of rotation about its longitudinal direction and will achieve rotations about intermediate axes caused by the presence of an off-centered imbalance. The size of rotations about the intermediate axes will be smaller than the rotation about the primary axis so it will be paramount to select an appropriate coordinate system and Euler angle representation for the displacement and change in orientation of the rotor. For reasons that will become apparent, a 1-2-3 Euler angle set is selected. The derivation and rationale behind this decision follows.

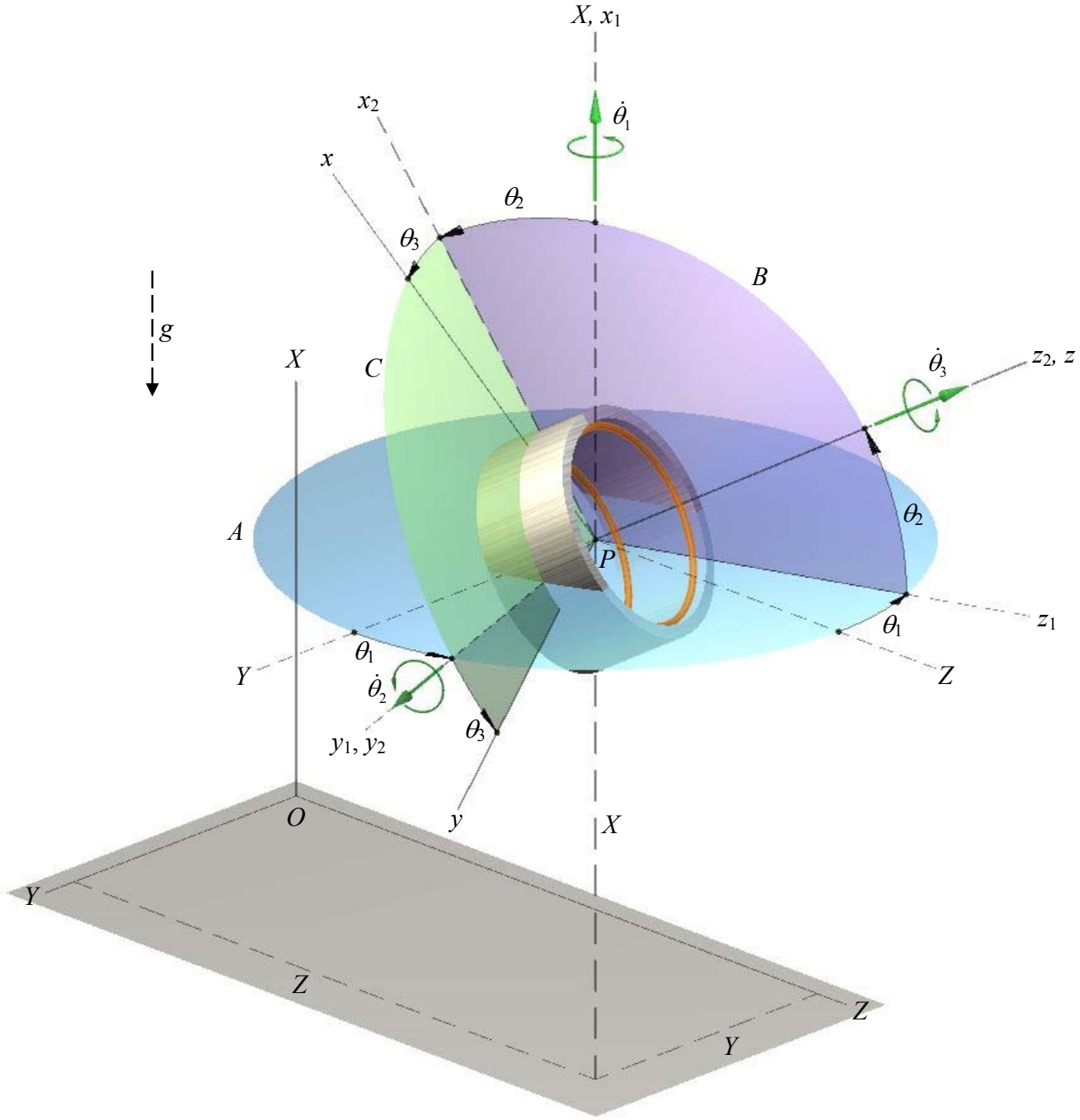

Figure 2.2: Displaced Configuration of the Cylindrical Rotor

Because the rotor is subjected to numerous forces and moments occurring about different axes of interest within the body, it is important to develop the coordinate transformations that relate quantities in the various frames of reference. The {**O**} frame consisting of the *X*-*Y*-*Z* axes is taken as the inertial frame of reference for the rotor. The rotor is first subjected to a rotation of θ_1 about the inertial *X*-axis which results in intermediate axes y_1 and z_1 which lie in plane *A*.

This new coordinate frame designated $\{\mathbf{P}_1\}$ and has its origin at the geometric center of the rotor, *P*. The $\{\mathbf{P}_1\}$ coordinate frame is related to the inertial $\{\mathbf{O}\}$ coordinate frame by the following transformation:

$$\{\mathbf{P}_1\} = [\mathbf{R}_1]\{\mathbf{O}\} \tag{2.1}$$

where $[\mathbf{R}_1]$ is shorthand notation for a direction cosine matrix resulting from a rotation about an *x*-axis, and has the following form:

$$[\mathbf{R}_1] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & \sin\theta_1 \\ 0 & -\sin\theta_1 & \cos\theta_1 \end{bmatrix} \tag{2.2}$$

It should be noted that curly braces enclosing a single bold-type character will be used to signify a vector of orthogonal coordinate frame directions while a single bold-type character will be used to represent a vector within a coordinate frame. Additionally, square brackets enclosing a single bold-type character will denote a square matrix. Thus, we may take a vector written in the inertial $\{\mathbf{O}\}$ frame and pre-multiply by the corresponding direction cosine matrix $[\mathbf{R}_1]$ to characterize the vector in the succeeding $\{\mathbf{P}_1\}$ frame. It is also noted that the direction cosine matrix $[\mathbf{R}_1]$ is an orthogonal matrix because its rows and columns represent orthogonal unit vectors.

Additionally, it is often helpful to transform vectors in the intermediate $\{\mathbf{P}_1\}$ frame to the inertial $\{\mathbf{O}\}$ frame. This is accomplished by pre-multiplying both sides of Eq. (2.1) with the inverse of $[\mathbf{R}_1]$, and recognizing that the inverse of an orthogonal matrix is equal to its transpose. We obtain the following transformation:

$$\{\mathbf{O}\} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & -\sin\theta_1 \\ 0 & \sin\theta_1 & \cos\theta_1 \end{bmatrix} \{\mathbf{P}_1\} \tag{2.3}$$

So, we may take a vector written in the $\{\mathbf{P}_1\}$ frame and pre-multiply by the transpose of the corresponding direction cosine matrix $[\mathbf{R}_1]$ to characterize the vector in the inertial $\{\mathbf{O}\}$ frame.

Following the rotation about the inertial *X*-axis, the rotor is subjected to a second rotation of θ_2 about the intermediate y_1-axis in the $\{\mathbf{P}_1\}$ frame. This rotation results in a second set of

intermediate axes x_2 and z_2 which lie in plane *B*. This new coordinate frame is designated $\{\mathbf{P}_2\}$ and is related to the intermediate $\{\mathbf{P}_1\}$ coordinate frame by the following transformation:

$$\{\mathbf{P}_2\} = [\mathbf{R}_2]\{\mathbf{P}_1\} \tag{2.4}$$

where $[\mathbf{R}_2]$ is shorthand notation for a direction cosine matrix resulting from a rotation about a *y*-axis, and has the following form:

$$[\mathbf{R}_2] = \begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix} \tag{2.5}$$

Thus, we may take a vector written in the intermediate $\{\mathbf{P}_1\}$ frame and pre-multiply by the corresponding direction cosine matrix $[\mathbf{R}_2]$ to characterize the vector in the succeeding $\{\mathbf{P}_2\}$ frame. Additionally, we can relate the intermediate $\{\mathbf{P}_2\}$ coordinate frame to the inertial $\{\mathbf{O}\}$ frame by substituting for $\{\mathbf{P}_1\}$ from Eq. (2.1). This is shown in the following transformation:

$$\{\mathbf{P}_2\} = [\mathbf{R}_2]\{\mathbf{P}_1\} = [\mathbf{R}_2][\mathbf{R}_1]\{\mathbf{O}\} = \begin{bmatrix} \cos\theta_2 & \sin\theta_1\sin\theta_2 & -\cos\theta_1\sin\theta_2 \\ 0 & \cos\theta_1 & \sin\theta_1 \\ \sin\theta_2 & -\sin\theta_1\cos\theta_2 & \cos\theta_1\cos\theta_2 \end{bmatrix}\{\mathbf{O}\} \tag{2.6}$$

Now, we may take a vector written in the inertial $\{\mathbf{O}\}$ frame and characterize it in the second intermediate $\{\mathbf{P}_2\}$ frame by pre-multiplication of the previous matrix. Conversely, we may take a vector in the $\{\mathbf{P}_2\}$ frame and characterize it in the inertial frame by pre-multiplying the vector by the transpose of the previous matrix. We obtain the following transformation

$$\{\mathbf{O}\} = \begin{bmatrix} \cos\theta_2 & 0 & \sin\theta_2 \\ \sin\theta_1\sin\theta_2 & \cos\theta_1 & -\cos\theta_2\sin\theta_1 \\ -\cos\theta_1\sin\theta_2 & \sin\theta_1 & \cos\theta_1\cos\theta_2 \end{bmatrix}\{\mathbf{P}_2\} \tag{2.7}$$

Following the rotation about the intermediate y_1-axis, the rotor is subjected to a third and final rotation of θ_3 about the intermediate z_2-axis in the $\{\mathbf{P}_2\}$ frame. This rotation is about the primary operating axis for the rotor and results in the body-fixed axes *x* and *y* which lie in plane *C*. This final coordinate frame is designated $\{\mathbf{P}\}$, and is related to the intermediate $\{\mathbf{P}_2\}$ coordinate frame by the following transformation:

$$\{\mathbf{P}\} = [\mathbf{R}_3]\{\mathbf{P}_2\} \tag{2.8}$$

where $[\mathbf{R}_3]$ is shorthand notation for a direction cosine matrix resulting from a rotation about a *z*-axis, and has the following form:

$$[\mathbf{R}_3] = \begin{bmatrix} \cos\theta_3 & \sin\theta_3 & 0 \\ -\sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

Thus, we may take a vector written in the intermediate $\{\mathbf{P}_2\}$ frame and pre-multiply by the corresponding direction cosine matrix $[\mathbf{R}_3]$ to characterize the vector in the body-fixed $\{\mathbf{P}\}$ frame. Additionally, we can relate the body-fixed $\{\mathbf{P}\}$ coordinate frame to the inertial $\{\mathbf{O}\}$ frame by substituting for $\{\mathbf{P}_2\}$ from Eq. (2.6). This is shown in the following transformation:

$$\{\mathbf{P}\} = [\mathbf{R}_3]\{\mathbf{P}_2\} = [\mathbf{R}_3][\mathbf{R}_2][\mathbf{R}_1]\{\mathbf{O}\} \rightarrow$$

$$\{\mathbf{P}\} = \begin{bmatrix} \cos\theta_2\cos\theta_3 & \sin\theta_1\sin\theta_2\cos\theta_3 + \cos\theta_1 s\theta_3 & \sin\theta_1\sin\theta_3 - \cos\theta_1\sin\theta_2\cos\theta_3 \\ -\cos\theta_2\sin\theta_3 & \cos\theta_1\cos\theta_3 - \sin\theta_1\sin\theta_2\sin\theta_3 & \cos\theta_1\sin\theta_2\sin\theta_3 + \sin\theta_1\cos\theta_3 \\ \sin\theta_2 & -\sin\theta_1\cos\theta_2 & \cos\theta_1\cos\theta_2 \end{bmatrix}\{\mathbf{O}\} \tag{2.10}$$

Now, we may take a vector written in the inertial $\{\mathbf{O}\}$ frame and characterize it in the body-fixed $\{\mathbf{P}\}$ frame by pre-multiplication of the previous matrix. Conversely, we may take a vector in the $\{\mathbf{P}\}$ frame and characterize it in the inertial frame by pre-multiplying the vector by the transpose of this matrix which results in the following transformation:

$$\{\mathbf{O}\} = \begin{bmatrix} \cos\theta_2\cos\theta_3 & -\cos\theta_2\sin\theta_3 & \sin\theta_2 \\ \sin\theta_1\sin\theta_2\cos\theta_3 + \cos\theta_1\sin\theta_3 & \cos\theta_1\cos\theta_3 - \sin\theta_1\sin\theta_2\sin\theta_3 & -\sin\theta_1\cos\theta_2 \\ \sin\theta_1\sin\theta_3 - \cos\theta_1\sin\theta_2\cos\theta_3 & \cos\theta_1\sin\theta_2\sin\theta_3 + \sin\theta_1\cos\theta_3 & \cos\theta_1\cos\theta_2 \end{bmatrix}\{\mathbf{P}\} \tag{2.11}$$

At this point, we have vector transformations to and from the inertial frame $\{\mathbf{O}\}$ to each successive frame $\{\mathbf{P}_1\}$, $\{\mathbf{P}_2\}$, and $\{\mathbf{P}\}$ and vice-versa. We also have transformations from $\{\mathbf{P}_1\}$ to $\{\mathbf{P}_2\}$ and $\{\mathbf{P}_2\}$ to $\{\mathbf{P}\}$. Because rigid-body dynamics is typically done in either an inertial reference frame or with respect to a body-fixed coordinate system, we need to define the transformation from $\{\mathbf{P}_1\}$ to $\{\mathbf{P}\}$ which is accomplished by substituting for $\{\mathbf{P}_2\}$ in Eq. (2.8) with Eq. (2.4) as shown below

$$\{\mathbf{P}\} = [\mathbf{R}_3]\{\mathbf{P}_2\} = [\mathbf{R}_3][\mathbf{R}_2]\{\mathbf{P}_1\} = \begin{bmatrix} \cos\theta_2\cos\theta_3 & \sin\theta_3 & -\sin\theta_2\cos\theta_3 \\ -\cos\theta_2\sin\theta_3 & \cos\theta_3 & \sin\theta_2\sin\theta_3 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix}\{\mathbf{P}_1\} \qquad (2.12)$$

At this point we have transformations from each coordinate frame to the inertial frame $\{\mathbf{O}\}$ and to the body-fixed frame $\{\mathbf{P}\}$. Now, we are ready to examine the angular velocity of the rotor.

2.1.2 Angular Velocity of the Body-Fixed Frame

Before we can begin writing vectors in the various coordinate frames, we will need to establish a convention for representing the associated unit vectors of each frame. The convention utilized is as follows:

$$\{\mathbf{O}\} \rightarrow \mathbf{I}, \mathbf{J}, \mathbf{K}$$
$$\{\mathbf{P}_1\} \rightarrow \mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1$$
$$\{\mathbf{P}_2\} \rightarrow \mathbf{i}_2, \mathbf{j}_2, \mathbf{k}_2$$
$$\{\mathbf{P}\} \rightarrow \mathbf{i}, \mathbf{j}, \mathbf{k}$$

where the "*i*" unit vectors are oriented along the corresponding *x*-axes, the "*j*" unit vectors are oriented along the corresponding *y*-axes, and the "*k"* unit vectors are oriented along the corresponding *z*-axes. Because capital letters were chosen for the inertial reference axes, capital letters are used for the inertial unit vectors. Similarly, because lowercase subscripted letters were chosen for the two intermediate frames, lowercase subscripted letters are used for the intermediate sets of unit vectors. Finally, because only lowercase letters were chosen for the body-fixed axes, only lowercase letters are used for the body-frame unit vectors.
We can write the angular velocity expression of the rotor as follows

$$\boldsymbol{\omega} = \dot{\theta}_1\mathbf{I} + \dot{\theta}_2\mathbf{j}_1 + \dot{\theta}_3\mathbf{k}_2 = \dot{\theta}_1\mathbf{i}_1 + \dot{\theta}_2\mathbf{j}_2 + \dot{\theta}_3\mathbf{k} \qquad (2.13)$$

where we have recognized that the inertial *X*-axis is identical with the intermediate x_1-axis; the intermediate y_1-axis is identical with the intermediate y_2-axis; and, the intermediate z_2-axis is identical with the body-fixed *z*-axis.

It is desired to express the angular velocity of the rotor in a common set of unit vectors and to select the body-fixed coordinate frame as the frame of reference. We may transform the $\mathbf{i}_1$

and $\mathbf{j}_2$ into body-fixed vectors with the transformations in Eqs. (2.12) and (2.8) respectively. Carrying out this operation and collecting terms results in the following expression for the angular velocity of the rotor as written in the body-fixed coordinate frame.

$$\begin{aligned}
&\boldsymbol{\omega} = \dot{\theta}_1\mathbf{i}_1 + \dot{\theta}_2\mathbf{j}_2 + \dot{\theta}_3\mathbf{k} \rightarrow \\
&\boldsymbol{\omega} = \dot{\theta}_1\left(\cos\theta_2\cos\theta_3\mathbf{i} - \cos\theta_2\sin\theta_3\mathbf{j} + \sin\theta_2\mathbf{k}\right) + \dot{\theta}_2\left(\sin\theta_3\mathbf{i} + \cos\theta_3\mathbf{j}\right) + \dot{\theta}_3\mathbf{k} \rightarrow \\
&\boldsymbol{\omega} = \left(\dot{\theta}_1\cos\theta_2\cos\theta_3 + \dot{\theta}_2\sin\theta_3\right)\mathbf{i} + \left(\dot{\theta}_2\cos\theta_3 - \dot{\theta}_1\cos\theta_2\sin\theta_3\right)\mathbf{j} + \left(\dot{\theta}_1\sin\theta_2 + \dot{\theta}_3\right)\mathbf{k}
\end{aligned} \tag{2.14}$$

The scalar expressions in front of the unit vectors represent the components of the angular velocity of the rotor in the body-fixed coordinate frame and may be written in matrix notation for ease of manipulation. This results in the following

$$\begin{Bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{Bmatrix} = \begin{bmatrix} \cos\theta_2\cos\theta_3 & \sin\theta_3 & 0 \\ -\cos\theta_2\sin\theta_3 & \cos\theta_3 & 0 \\ \sin\theta_2 & 0 & 1 \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix} \tag{2.15}$$

where ω_x, ω_y, and ω_z represent the components of angular velocity about the body-fixed axes. Because we will be interested in the time history of the various angles, this nonorthogonal matrix must be inverted. The inverse of this expression is found to be

$$\begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix} = \begin{bmatrix} \dfrac{\cos\theta_3}{\cos\theta_2} & -\dfrac{\sin\theta_3}{\cos\theta_2} & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ -\dfrac{\sin\theta_2\cos\theta_3}{\cos\theta_2} & \dfrac{\sin\theta_2\sin\theta_3}{\cos\theta_2} & 1 \end{bmatrix} \begin{Bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{Bmatrix} \tag{2.16}$$

It is clear from this expression that the equations of motion will experience a singularity whenever the angle θ_2 approaches $\pi/2$. The model presented will be quite stiff in regard to rotation about the y_1 or y_2-axis which means θ_2 will always be small and never approach the singularity. This constraint on the size of θ_2 is the primary reason a 1-2-3 Euler angle set was chosen.

2.1.3 Angular Acceleration of the Body-Fixed Frame

Because we have written the angular velocity of the rotor in the body-fixed coordinate frame, we may take a time derivative of this expression to obtain the angular acceleration of the rotor in the body-fixed coordinate frame.

$$\begin{aligned}
&\boldsymbol{\alpha} = \dot{\boldsymbol{\omega}} = \frac{d\boldsymbol{\omega}}{dt} \rightarrow \\
&\boldsymbol{\alpha} = \left(\ddot{\theta}_1 \cos\theta_2 \cos\theta_3 - \dot{\theta}_1\dot{\theta}_2 \sin\theta_2 \cos\theta_3 - \dot{\theta}_1\dot{\theta}_3 \cos\theta_2 \sin\theta_3 + \ddot{\theta}_2 \sin\theta_3 + \dot{\theta}_2\dot{\theta}_3 \cos\theta_3\right)\mathbf{i} \\
&+\left(\dot{\theta}_1\dot{\theta}_2 \sin\theta_2 \sin\theta_3 - \ddot{\theta}_1 \cos\theta_2 \sin\theta_3 - \dot{\theta}_1\dot{\theta}_3 \cos\theta_2 \cos\theta_3 + \ddot{\theta}_2 c\theta_3 - \dot{\theta}_2\dot{\theta}_3 \sin\theta_3\right)\mathbf{j} + \\
&\left(\ddot{\theta}_1 \sin\theta_2 + \dot{\theta}_1\dot{\theta}_2 \cos\theta_2 + \ddot{\theta}_3\right)\mathbf{k}
\end{aligned} \tag{2.17}$$

2.1.4 Position Vectors

The geometric center of the rotor P is located relative to the nominal position of the rotor center O by the following vector, as written in the inertial $\{\mathbf{O}\}$ frame

$$\mathbf{r}_{P/O} = X\mathbf{I} + Y\mathbf{J} + Z\mathbf{K} \tag{2.18}$$

The mass imbalance m is located relative to the geometric center of the rotor P by the following vector, as written in the body frame $\{\mathbf{P}\}$

$$\mathbf{r}_m = r\cos\beta_m\mathbf{i} + r\sin\beta_m\mathbf{i} + d_m\mathbf{k} \tag{2.19}$$

The locations of the balancing particles m_j are given by the following vector expressions, as viewed from the body-fixed frame

$$\mathbf{r}_j = r_j\cos\beta_j\mathbf{i} + r_j\sin\beta_j\mathbf{i} + d_j\mathbf{k} \quad \text{for} \quad j = 1{:}4 \tag{2.20}$$

The location of the center of mass of the rotor and imbalance in the body-fixed frame is

$$\mathbf{r}_{G/P} = \frac{\sum m_k\mathbf{r}_k}{\sum m_k} = \frac{m\mathbf{r}_m}{m+M} = \frac{m}{m+M}\left(r\cos\beta_m\mathbf{i} + r\sin\beta_m\mathbf{i} + d_m\mathbf{k}\right) \tag{2.21}$$

where the balancing particles have not been included because they will be dealt with individually due to the presence of frictional forces acting at the interface between the balancing particles and the rotor surface. Now, the location of the center of mass can be found by combining Eqs. (2.18) and (2.21). The position vector to the center of mass is

$$\mathbf{r}_{G/O} = \mathbf{r}_{P/O} + \mathbf{r}_{G/P} = X\mathbf{I} + Y\mathbf{J} + Z\mathbf{K} + \frac{m}{m+M}\left(r\cos\beta_m\mathbf{i} + r\sin\beta_m\mathbf{i} + d_m\mathbf{k}\right) \tag{2.22}$$

where we have characterized the vector in two different coordinate frames. This vector can be written entirely in the inertial frame by multiplying the last term in Eq. (2.22) by the matrix in Eq. (2.11). Doing so gives

$$\begin{aligned}\mathbf{r}_{G/O} = &\left(X + \frac{m}{M+m}\left[r\cos\theta_2\cos\left(\beta_m+\theta_3\right) + d_m\sin\theta_2\right]\right)\mathbf{I} + \\ &\left(Y + \frac{m}{M+m}\left[r\sin\theta_1\sin\theta_2\cos\left(\beta_m+\theta_3\right) + r\cos\theta_1\sin\left(\beta_m+\theta_3\right) - d_m\cos\theta_1\cos\theta_2\right]\right)\mathbf{J} + \\ &\left(Z + \frac{m}{M+m}\left[r\sin\theta_1\sin\left(\beta_m+\theta_3\right) + d_m\cos\theta_1\cos\theta_2 - r\cos\theta_1\sin\theta_2\cos\left(\beta_m+\theta_3\right)\right]\right)\mathbf{K}\end{aligned} \tag{2.23}$$

2.1.5 Acceleration Vectors

It is now possible to obtain the acceleration of points *P* and *G* in the inertial frame of reference. The acceleration of *P* is obtained by differentiating the vector in Eq. (2.18) twice with respect to time and recognizing that the time derivatives of the inertial unit vectors are zero. Thus, we obtain

$$\mathbf{a}_{P/O} = \ddot{X}\mathbf{I} + \ddot{Y}\mathbf{J} + \ddot{Z}\mathbf{K} \tag{2.24}$$

Similarly, the acceleration of the center of mass is found by differentiating the vector in Eq. (2.23) twice with respect to time. Doing so gives

$$\mathbf{a}_{G/O} = \left\langle \ddot{X} + \frac{m}{M+m}\left\{ d_m\left(\ddot{\theta}_2\cos\theta_2 - \dot{\theta}_2{}^2\sin\theta_2\right) + r\left[\sin\left(\beta_m+\theta_3\right)\left(2\dot{\theta}_2\dot{\theta}_3\sin\theta_2 - \ddot{\theta}_3\cos\theta_2\right) - \right.\right.\right.$$
$$\left.\left.\left.\cos\left(\beta_m+\theta_3\right)\left(\ddot{\theta}_2\sin\theta_2 + \cos\theta_2\left[\dot{\theta}_2{}^2 + \dot{\theta}_3{}^2\right]\right)\right]\right\}\right\rangle \mathbf{I} + \left\langle \ddot{Y} + \frac{m}{M+m}\left\{ d_m\left[\cos\theta_1\left(2\dot{\theta}_1\dot{\theta}_2\sin\theta_2 - \right.\right.\right.\right.$$
$$\left.\left.\ddot{\theta}_1\cos\theta_2\right) + \sin\theta_1\left(\ddot{\theta}_2\sin\theta_2 + \cos\theta_2\left[\dot{\theta}_1{}^2 + \dot{\theta}_2{}^2\right]\right)\right] + r\left[\cos\left(\beta_m+\theta_3\right)\left(\ddot{\theta}_3\cos\theta_1 + \right.\right.$$
$$\cos\theta_2\left[\ddot{\theta}_2\sin\theta_1 + 2\dot{\theta}_1\dot{\theta}_2\cos\theta_1\right] - \sin\theta_2\left[\sin\theta_1\left(\dot{\theta}_1{}^2 + \dot{\theta}_2{}^2 + \dot{\theta}_3{}^2\right) - \ddot{\theta}_1\cos\theta_1\right] - 2\dot{\theta}_1\dot{\theta}_3\sin\theta_1\Big) -$$
$$\sin\left(\beta_m+\theta_3\right)\left(\ddot{\theta}_1\sin\theta_1 + \sin\theta_2\left[\ddot{\theta}_3\sin\theta_1 + 2\dot{\theta}_1\dot{\theta}_3\cos\theta_1\right] + 2\dot{\theta}_2\dot{\theta}_3\sin\theta_1\cos\theta_2 + \right.$$
$$\left.\left.\left.\cos\theta_1\left[\dot{\theta}_1{}^2 + \dot{\theta}_3{}^2\right]\right)\right]\right\}\right\rangle \mathbf{J} + \left\langle \ddot{Z} + \frac{m}{M+m}\left\{ d_m\left[\sin\theta_2\left(2\dot{\theta}_1\dot{\theta}_2\sin\theta_1 - \ddot{\theta}_2\cos\theta_1\right) - \right.\right.\right.$$
$$\cos\theta_2\left(\ddot{\theta}_1\sin\theta_1 + \cos\theta_1\left[\dot{\theta}_1{}^2 + \dot{\theta}_2{}^2\right]\right)\Big] + r\Big[\cos\left(\beta_m+\theta_3\right)\left(\ddot{\theta}_3\sin\theta_1 + \cos\theta_2\left[2\dot{\theta}_1\dot{\theta}_2\sin\theta_1 - \right.\right.$$
$$\left.\ddot{\theta}_2\cos\theta_1\right] + \sin\theta_2\left[\ddot{\theta}_1\sin\theta_1 + \cos\theta_1\left(\dot{\theta}_1{}^2 + \dot{\theta}_2{}^2 + \dot{\theta}_3{}^2\right) + 2\dot{\theta}_1\dot{\theta}_3\cos\theta_1\right]\Big) + \sin\left(\beta_m+\theta_3\right)\left(\ddot{\theta}_1\cos\theta_1 + \right.$$
$$\left.\left.\left.\left.\ddot{\theta}_3\sin\theta_2\cos\theta_1 + 2\dot{\theta}_2\dot{\theta}_3\cos\theta_1\cos\theta_2 - \sin\theta_1\left[\dot{\theta}_1{}^2 + \dot{\theta}_3{}^2 + 2\dot{\theta}_1\dot{\theta}_3\sin\theta_2\right]\right)\right]\right\}\right\rangle \mathbf{K} \tag{2.25}$$

It will also be of interest to write the acceleration of the geometric center of the rotor in the body-fixed frame. This is done by multiplying the vector in Eq. (2.24) by the matrix in Eq. (2.10). Doing so results in

$$\mathbf{a}_{P/O} = \Big[\cos\theta_3\left(\ddot{X}\cos\theta_2 + \sin\theta_2\left[\ddot{Y}\sin\theta_1 - \ddot{Z}\cos\theta_1\right]\right) + \sin\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right)\Big]\mathbf{i} +$$
$$\Big[\cos\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right) - \sin\theta_3\left(\ddot{X}\cos\theta_2 + \sin\theta_2\left[\ddot{Y}\sin\theta_1 - \ddot{Z}\cos\theta_1\right]\right)\Big]\mathbf{j} + \tag{2.26}$$
$$\Big[\ddot{X}\sin\theta_2 + \cos\theta_2\left(\ddot{Z}\cos\theta_1 - \ddot{Y}\sin\theta_1\right)\Big]\mathbf{k}$$

2.1.6 Inertia Matrix

Due to the lack of symmetry in the model caused by the mass imbalance, the inertia matrix for the rotor and imbalance will be fully populated. Because the angular velocity and angular acceleration of the rotor are written in the body-fixed coordinate frame, the inertia matrix will also be written in this frame. It should be noted that the balancing particles are not included in this calculation.

$$I_{xx} = \frac{1}{4}M\left(r_o{}^2 + r_i^2\right) + \frac{1}{12}Md^2 + m\left(\left[r\sin\beta_m\right]^2 + d_m{}^2\right) \tag{2.27}$$

$$I_{yy} = \frac{1}{4}M\left(r_o{}^2 + r_i^2\right) + \frac{1}{12}Md^2 + m\left(\left[r\cos\beta_m\right]^2 + d_m{}^2\right) \tag{2.28}$$

$$I_{zz} = \frac{1}{2}M\left(r_o{}^2 + r_i^2\right) + mr^2 \tag{2.29}$$

$$I_{xy} = I_{yx} = mr^2\cos\beta_m\sin\beta_m \tag{2.30}$$

$$I_{xz} = I_{zx} = mr\cos\beta_m d_m \tag{2.31}$$

$$I_{yz} = I_{zy} = mr\sin\beta_m d_m \tag{2.32}$$

The elements above are combined to form the inertia matrix, which has the following form:

$$[\mathbf{I}] = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \tag{2.33}$$

It will also be of interest to compute the inertia matrix of the entire system including the balancing particles to determine the how well the balancing particles are able to transform the body-fixed axes into principal axes. The individual elements of this matrix follow, where the subscript *P* is used to denote the interest in principal axes.

$$I_{Pxx} = \frac{1}{4}M\left(r_o{}^2 + r_i^2\right) + \frac{1}{12}Md^2 + m\left(\left[r\sin\beta_m\right]^2 + d_m{}^2\right) + \sum_{j=1}^{4} m_j\left(\left[r_j\sin\beta_j\right]^2 + d_j^2\right) \tag{2.34}$$

$$I_{Pyy} = \frac{1}{4}M\left(r_o{}^2 + r_i^2\right) + \frac{1}{12}Md^2 + m\left(\left[r\cos\beta_m\right]^2 + d_m{}^2\right) + \sum_{j=1}^{4} m_j\left(\left[r_j\cos\beta_j\right]^2 + d_j^2\right) \tag{2.35}$$

$$I_{Pzz} = \frac{1}{2}M\left(r_o{}^2 + r_i^2\right) + mr^2 + \sum_{j=1}^{4} m_j r_j{}^2 \tag{2.36}$$

$$I_{Pxy} = I_{Pyx} = mr^2\cos\beta_m\sin\beta_m + \sum_{j=1}^{4} m_j r_j{}^2\cos\beta_j\sin\beta_j \tag{2.37}$$

$$I_{Pxz} = I_{Pzx} = mr\cos\beta_m d_m + \sum_{j=1}^{4} m_j r_j \cos\beta_j d_j \tag{2.38}$$

$$I_{Pyz} = I_{Pzy} = mr\sin\beta_m d_m + \sum_{j=1}^{4} m_j r_j \sin\beta_j d_j \tag{2.39}$$

These expressions are similarly combined to form the inertia matrix for principal axis investigation.

$$\left[\mathbf{I}_P\right] = \begin{bmatrix} I_{P_{xx}} & -I_{P_{xy}} & -I_{P_{xz}} \\ -I_{P_{yx}} & I_{P_{yy}} & -I_{P_{yz}} \\ -I_{P_{zx}} & -I_{P_{zy}} & I_{P_{zz}} \end{bmatrix} \tag{2.40}$$

2.2 Kinetics

At this point, we will turn our focus to the various forces which act on the rotor in the displaced configuration shown in Figure 2.2. The reader is asked to consult Figure 2.3 for a close-up view of the rotor showing the various forces which act on the rotor from the combination of springs, dampers, and interaction forces between the rotor and the balancing particles.

2.2.1 Linear Spring Forces

The three linear springs resist displacement of the geometric center of the rotor in the inertial coordinate frame $\{\mathbf{O}\}$. It is assumed that each spring is only stretched in a direction parallel to one of the inertial frame axes. The resultant force of the three linear springs may be written as

$$\mathbf{F}_{k_{XYZ}} = \mathbf{F}_{k_X} + \mathbf{F}_{k_Y} + \mathbf{F}_{k_Z} = -k_X X\mathbf{I} - k_Y Y\mathbf{J} - k_Z Z\mathbf{K} \tag{2.41}$$

where the subscript k on the force $\mathbf{F}$ denotes a linear spring stiffness, and the subscripted XYZ denotes the axes of stretch in an inertial frame.

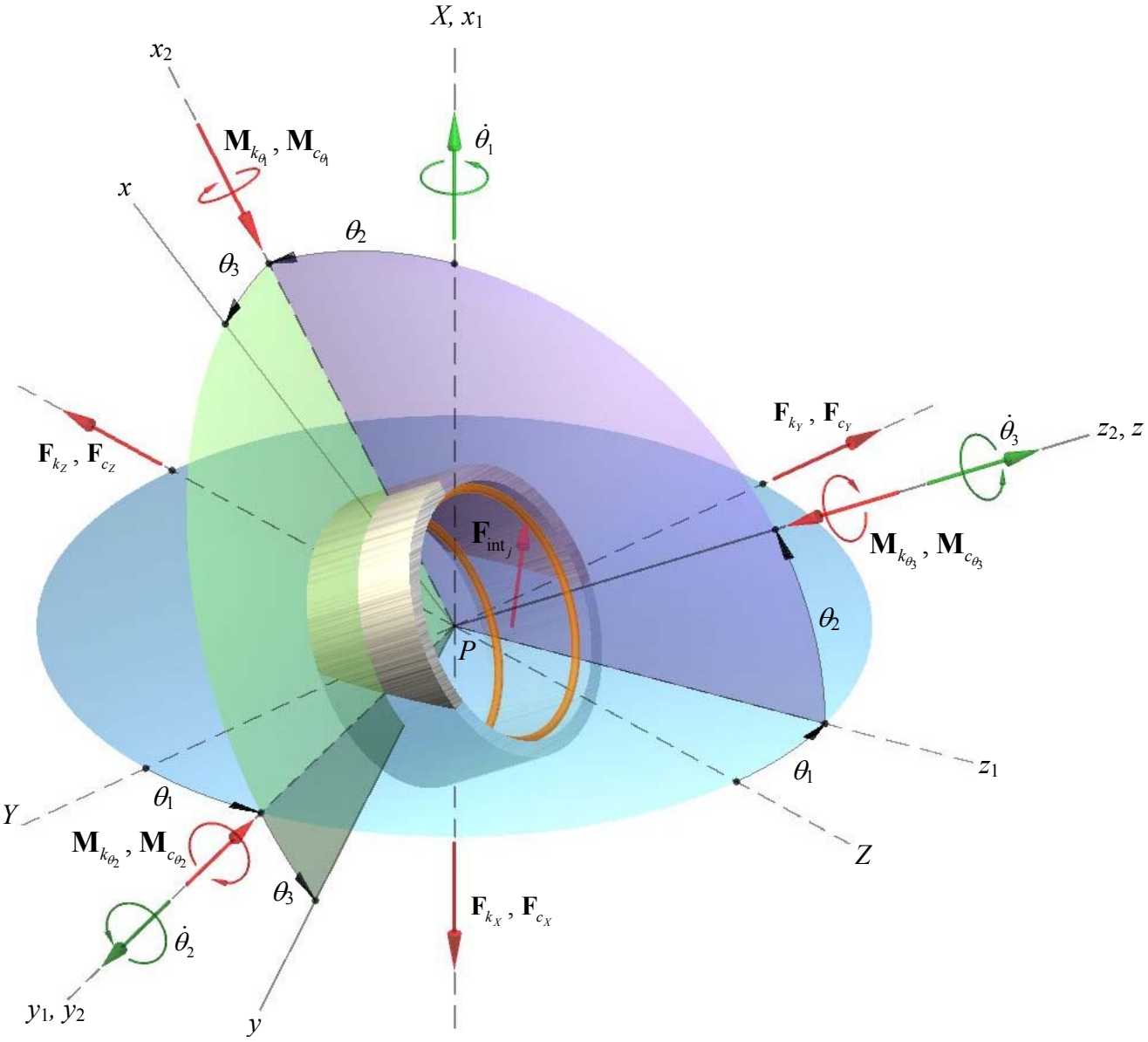

Figure 2.3: Free-Body Diagram of the Cylindrical Rotor

2.2.2 Linear Damper Forces

In addition to the three linear springs, there are three linear dampers which resist the velocity of the geometric center of the rotor in the inertial frame. As with the linear springs, it is assumed that each damper is only moved in a direction parallel to one of the inertial frame axes. The resultant force of the three linear dampers may be written as

$$\mathbf{F}_{c_{XYZ}} = \mathbf{F}_{c_X} + \mathbf{F}_{c_Y} + \mathbf{F}_{c_Z} = -c_X \dot{X}\mathbf{I} - c_Y \dot{Y}\mathbf{J} - c_Z \dot{Z}\mathbf{K} \tag{2.42}$$

where the subscript *c* on the force **F** denotes a linear damping coefficient, and the subscripted *XYZ* denotes the axes of movement in an inertial frame.

2.2.3 Torsional Spring Moments

The three torsional springs each resist angular displacement about a specific axis of rotation. It is assumed that the spring which resists rotation about the inertial *X*-axis for the first rotation of θ_1 lies on the intermediate x_2-axis instead of the primary axis of rotation. This is done so that the moment produced by this spring will not have a component that lies along the body-fixed *z*-axis, and provided the second rotation of θ_2 is small, is a reasonable assumption. We may write the three torsional spring moments as follows

$$\mathbf{M}_{k_{\theta_1}} = -k_{\theta_1}\theta_1\mathbf{i}_2 = -k_{\theta_1}\theta_1\left(\cos\theta_3\mathbf{i} - \sin\theta_3\mathbf{j}\right) \tag{2.43}$$

$$\mathbf{M}_{k_{\theta_2}} = -k_{\theta_2}\theta_2\mathbf{j}_2 = -k_{\theta_2}\theta_2\left(\sin\theta_3\mathbf{i} + \cos\theta_3\mathbf{j}\right) \tag{2.44}$$

$$\mathbf{M}_{k_{\theta_3}} = -k_{\theta_3}\theta_3\mathbf{k} \tag{2.45}$$

2.2.4 Torsional Damper Moments

The three torsional dampers each resist angular velocity about a specific axis of rotation. As before, it is assumed that the damper which resists rotation about the inertial *X*-axis for the angular velocity $\dot{\theta}_1$ lies on the intermediate x_2-axis instead of the primary axis of rotation. We may write the three torsional damper moments as follows

$$\mathbf{M}_{c_{\theta_1}} = -c_{\theta_1}\dot{\theta}_1\mathbf{i}_2 = -c_{\theta_1}\dot{\theta}_1\left(\cos\theta_3\mathbf{i} - \sin\theta_3\mathbf{j}\right) \tag{2.46}$$

$$\mathbf{M}_{c_{\theta_2}} = -c_{\theta_2}\dot{\theta}_2\mathbf{j}_2 = -c_{\theta_2}\dot{\theta}_2\left(\sin\theta_3\mathbf{i} + \cos\theta_3\mathbf{j}\right) \tag{2.47}$$

$$\mathbf{M}_{c_{\theta_3}} = -c_{\theta_3}\dot{\theta}_3\mathbf{k} \tag{2.48}$$

2.2.5 Forces Acting on a Balancing Particle

In additional to the springs and dampers which resist the linear and angular displacement of the rotor, there are forces of interaction that develop at the interface between the balancing particles and the rotor surface. For simplicity, only one such force, $\mathbf{F}_{\text{int}_j}$, is shown on Figure 2.3. To better characterize this force, it is helpful to examine the motion of a single balancing particle as seen in the body-fixed frame {**P**}. This is illustrated in Figure 2.4.

It should be noted that the gravitational force acting on each balancing particle has not been shown because the figure focuses on the forces and accelerations that are easily represented in the body-fixed coordinate frame {**P**}. The force of gravity is easily represented in the inertial coordinate system because it acts in the negative *X*-direction. We may represent this in the body-fixed coordinate frame as

$$\mathbf{F}_{g_j} = -m_j g\mathbf{I} = -m_j g\left(\cos\theta_2\cos\theta_3\mathbf{i} + \cos\theta_2\sin\theta_3\mathbf{j} + \sin\theta_2\mathbf{k}\right) \tag{2.49}$$

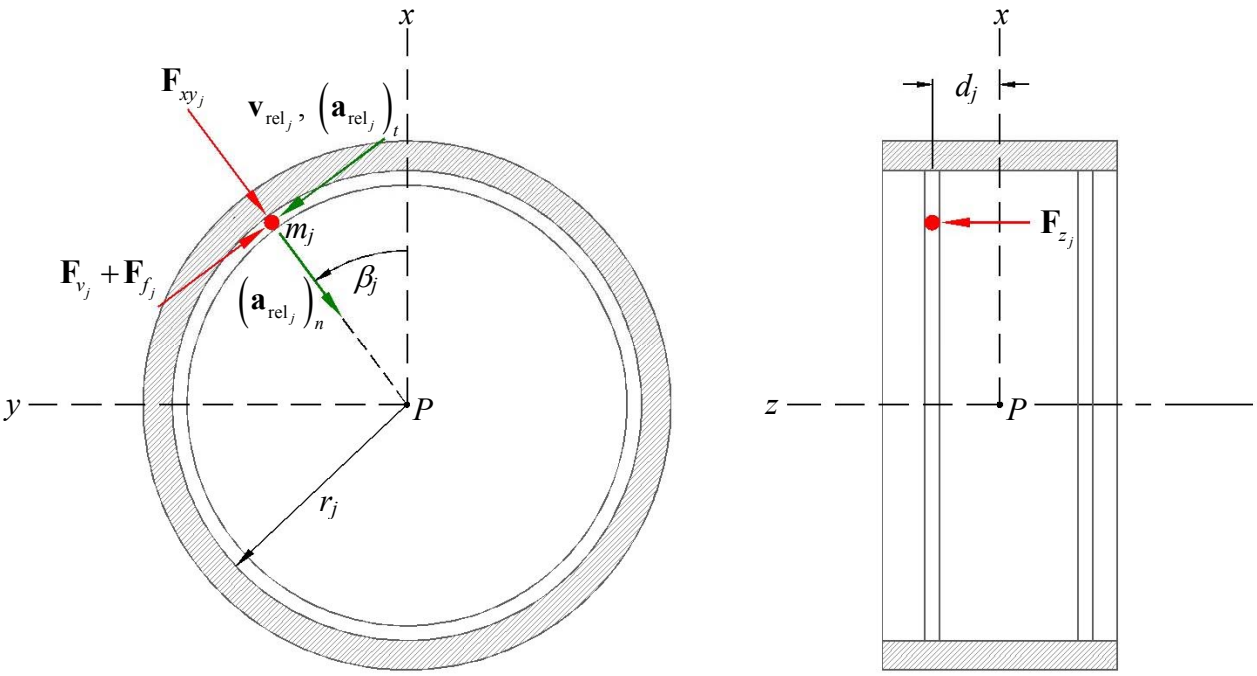

Figure 2.4: Free-Body Diagram of a Balancing Particle

It should also be noted that because the motion of each balancing particle is most easily described with a frame of reference attached to the rotor, the forces and accelerations which each particle experiences will be expressed in the body-fixed frame {**P**}.

As illustrated above, each balancing particle will experience a radial normal force $\mathbf{F}_{xy}$ resulting from contact with the inner surface of the rotor, and a longitudinal normal force $\mathbf{F}_z$ resulting from contact with the sidewall of the race. In addition, there will be a tangential force $\mathbf{F}_v + \mathbf{F}_f$ composed of a viscous drag force from fluid/mass interaction in the race and a Coulomb frictional force caused by the motion of the particle relative to the rotor surface. The viscous drag force acting on the particle is always present and assumed to be proportional to the relative velocity of the balancing particle in the race and has the form

$$F_{v_j} = -\mu_c r_j \dot{\beta}_j \tag{2.50}$$

where μ_c represents a viscous drag coefficient for the fluid/mass interaction in the race. The resultant force which acts on each balancing particle may be written as

$$\begin{aligned}\mathbf{F}_j = &\left(\left[F_{f_j} + \mu_c r_j \dot{\beta}_j\right]\sin\beta_j - F_{xy_j}\cos\beta_j - m_j g\cos\theta_2\cos\theta_3\right)\mathbf{i} + \\ &\left(m_j g\cos\theta_2\sin\theta_3 - \left[F_{f_j} + \mu_c r_j \dot{\beta}_j\right]\cos\beta_j - F_{xy_j}\sin\beta_j\right)\mathbf{j} + \left(F_{z_j} - m_j g\sin\theta_2\right)\mathbf{k}\end{aligned} \tag{2.51}$$

2.2.6 Interaction Forces

The rotor will experience a force equal and opposite to the forces shown in Figure 2.4, which act on the balancing particle. Thus, we may write the interaction force on the rotor as

$$\begin{aligned}\mathbf{F}_{\text{int}_j} = -\mathbf{F}_{f_j} - \mathbf{F}_{v_j} - \mathbf{F}_{xy_j} - \mathbf{F}_{z_j} = &\left(F_{xy_j}\cos\beta_j - \left[F_{f_j} + \mu_c r_j \dot{\beta}_j\right]\sin\beta_j\right)\mathbf{i} + \\ &\left(\left[F_{f_j} + \mu_c r_j \dot{\beta}_j\right]\cos\beta_j + F_{xy_j}\sin\beta_j\right)\mathbf{j} - F_{z_j}\mathbf{k}\end{aligned} \tag{2.52}$$

To maintain consistency among unit vectors, this vector must be written in the inertial frame $\{\mathbf{O}\}$ which is accomplished with Eq. (2.11). This results in

$$\begin{aligned}
\mathbf{F}_{\text{int}_j} &= \Big(F_{xy_j}\cos\left(\beta_j+\theta_3\right)\cos\theta_2-\left[F_{f_j}+\mu_c r_j\dot{\beta}_j\right]\sin\left(\beta_j+\theta_3\right)\cos\theta_2-F_{z_j}\sin\theta_2\Big)\mathbf{I}+\\
&\Big[\cos\theta_1\Big(F_{xy_j}\sin\left(\beta_j+\theta_3\right)+\cos\left(\beta_j+\theta_3\right)\left[F_{f_j}+\mu_c r_j\dot{\beta}_j\right]\Big)+F_{z_j}\sin\theta_1\cos\theta_2+\\
&\sin\theta_1\sin\theta_2\Big(F_{xy_j}\cos\left(\beta_j+\theta_3\right)-\sin\left(\beta_j+\theta_3\right)\left[F_{f_j}+\mu_c r_j\dot{\beta}_j\right]\Big)\Big]\mathbf{J}+\\
&\Big[\sin\theta_1\Big(F_{xy_j}\sin\left(\beta_j+\theta_3\right)+\cos\left(\beta_j+\theta_3\right)\left[F_{f_j}+\mu_c r_i\dot{\beta}_j\right]\Big)+\\
&\cos\theta_1\sin\theta_2\Big(\sin\left(\beta_j+\theta_3\right)\left[F_{f_j}+\mu_c r_i\dot{\beta}_j\right]-F_{xy_j}\cos\left(\beta_j+\theta_3\right)\Big)-F_{z_j}\cos\theta_1\cos\theta_2\Big]\mathbf{K}
\end{aligned} \tag{2.53}$$

This interaction force will also create a moment about the geometric center of the rotor *P*. Because the moments are most easily represented in the body-fixed frame, the moment made by the interaction force about point *P* is found by taking a cross product of the position vector in Eq. (2.20) with the force vector in Eq. (2.52). This results in

$$\begin{aligned}
\mathbf{M}_{\text{int}_j} = \mathbf{r}_j\times\mathbf{F}_{\text{int}_j} &= \Big(-d_j\cos\beta_j\left[F_{f_j}+\mu_c r_j\dot{\beta}_j\right]-\sin\beta_j\left[d_j F_{xy_j}+r_j F_{z_j}\right]\Big)\mathbf{i}+\\
&\Big(\cos\beta_j\left[d_j F_{xy_j}+r_j F_{z_j}\right]-d_j\sin\beta_j\left[F_{f_j}+\mu_c r_j\dot{\beta}_j\right]\Big)\mathbf{j}+r_j\Big(F_{f_j}+\mu_c r_i\dot{\beta}_j\Big)\mathbf{k}
\end{aligned} \tag{2.54}$$

2.2.7 Acceleration of a Balancing Particle

Because the balancing particles move relative to the rotor surface, and the geometric center of the rotor is not fixed, it is essential to write an inertial expression for the acceleration of a balancing particle. This is done with the well-known [74] five-term acceleration expression for rotating frames

$$\mathbf{a}_j = \mathbf{a}_{P/O}+\boldsymbol{\alpha}\times\mathbf{r}_j+\boldsymbol{\omega}\times\left(\boldsymbol{\omega}\times\mathbf{r}_j\right)+2\boldsymbol{\omega}\times\mathbf{v}_{\text{rel}_j}+\mathbf{a}_{\text{rel}_j} \tag{2.55}$$

where the terms $\mathbf{v}_{\text{rel}_j}$ and $\mathbf{a}_{\text{rel}_j}$ represent the velocity and acceleration of the balancing particle measured relative to the coordinate frame fixed to the rotor, respectively. From Figure 2.4, it is seen that

$$\mathbf{v}_{\text{rel}_j} = -r_j\dot{\beta}_j\sin\beta_j\mathbf{i}+r_j\dot{\beta}_j\cos\beta_j\mathbf{j} \tag{2.56}$$

The relative acceleration of the balancing particle can be thought of in two components, a tangential component arising from a change in the angular velocity of the particle relative to the rotor, and a normal component caused by having a relative speed while traveling in a curve along the surface of the rotor. The relative acceleration can be written as

$$\mathbf{a}_{\mathrm{rel}_j} = \left(\mathbf{a}_{\mathrm{rel}_j}\right)_t + \left(\mathbf{a}_{\mathrm{rel}_j}\right)_n = r_j\ddot{\beta}_j\left(-\sin\beta_j\mathbf{i} + \cos\beta_j\mathbf{j}\right) + r_j\dot{\beta}_j^{\,2}\left(-\cos\beta_j\mathbf{i} - \sin\beta_j\mathbf{j}\right) \tag{2.57}$$

Substitution of Eqs. (2.14), (2.17), (2.20), (2.26), (2.56), and (2.57) yields the following expression for the inertial acceleration of a balancing particle characterized in the body-fixed $\{\mathbf{P}\}$ frame.

$$\begin{aligned}
\mathbf{a}_j = &\Big\{\cos\theta_3\left(\ddot{X}\cos\theta_2 + \sin\theta_2\left[\ddot{Y}\sin\theta_1 - \ddot{Z}\cos\theta_1\right]\right) + \sin\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right) - \\
&r_j\ddot{\beta}_j\sin\beta_j - r_j\dot{\beta}_j^{\,2}\cos\beta_j + d_j\Big(\cos\theta_3\left[\ddot{\theta}_2 + \dot{\theta}_1^{\,2}\sin\theta_2\cos\theta_2\right] + \\
&\sin\theta_3\left[2\dot{\theta}_1\dot{\theta}_2\sin\theta_2 - \ddot{\theta}_1\cos\theta_2\right]\Big) - \frac{r_j}{4}\Big[\Big[\dot{\theta}_2^{\,2}\cos\theta_3\cos\left(\beta_j + \theta_3\right) + \dot{\theta}_3\cos\beta_j\left(2\dot{\beta}_j + \dot{\theta}_3\right) + \\
&4\sin\beta_j\left(\ddot{\theta}_3 + \ddot{\theta}_1\sin\theta_2\right) + \dot{\theta}_1^{\,2}\left(-2\cos\left(2\theta_2\right)\cos\theta_3\cos\left(\beta_j + \theta_3\right) - \cos\left(\beta_j + 2\theta_3\right) + \right. \\
&\left.3\cos\beta_j\right) - 8\dot{\theta}_1\left(\dot{\theta}_2\sin\theta_3\cos\theta_2\cos\left(\beta_j + \theta_3\right) - \sin\theta_2\cos\beta_j\left[\dot{\beta}_j + \dot{\theta}_3\right]\right)\Big]\Big\}\mathbf{i} + \\
&\Big\{\cos\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right) - \sin\theta_3\left(\ddot{X}\cos\theta_2 + \sin\theta_2\left[\ddot{Y}\sin\theta_1 - \ddot{Z}\cos\theta_1\right]\right) + r_j\ddot{\beta}_j\cos\beta_j - \\
&r_j\dot{\beta}_j^{\,2}\sin\beta_j - d_j\Big(\cos\theta_3\left[\ddot{\theta}_1\cos\theta_2 - 2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\right] + \sin\theta_3\left[\ddot{\theta}_2 + \dot{\theta}_1^{\,2}\sin\theta_2\cos\theta_2\right]\Big) - \\
&r_j\Big[\Big[\dot{\theta}_3\left(2\dot{\beta}_j + \dot{\theta}_3\right)\sin\beta_j - \left(\ddot{\theta}_3 + \ddot{\theta}_1\sin\theta_2\right)\cos\beta_j - \dot{\theta}_2^{\,2}\sin\theta_3\cos\left(\beta_j + \theta_3\right) + \\
&\dot{\theta}_1^{\,2}\left(\sin^2\theta_2\sin\beta_j + \cos^2\theta_2\cos\theta_3\sin\left(\beta_j + \theta_3\right)\right) + 2\dot{\theta}_1\left(\sin\theta_2\sin\beta_j\left[\dot{\beta}_j + \dot{\theta}_3\right] - \right. \\
&\left.\dot{\theta}_2\cos\theta_2\cos\theta_3\cos\left(\beta_j + \theta_3\right)\right)\Big]\Big\}\mathbf{j} + \Big\{\ddot{X}\sin\theta_2 + \cos\theta_2\left(\ddot{Z}\cos\theta_1 - \ddot{Y}\sin\theta_1\right) - \\
&-d_j\left(\dot{\theta}_2^{\,2} + \dot{\theta}_1^{\,2}\cos^2\theta_2\right) + r_j\Big[\cos\left(\beta_j + \theta_3\right)\left(\dot{\theta}_1\cos\theta_2\left[2\dot{\beta}_j + 2\dot{\theta}_3 + \dot{\theta}_1\sin\theta_2\right] - \ddot{\theta}_2\right) + \\
&\sin\left(\beta_j + \theta_3\right)\left(2\dot{\theta}_2\left[\dot{\beta}_j + \dot{\theta}_3\right] + \ddot{\theta}_1\cos\theta_2\right)\Big]\Big\}\mathbf{k}
\end{aligned} \tag{2.58}$$

2.2.8 Angular Momentum

At this point, we are ready to write the angular momentum vector for the rotor. The angular momentum is found by multiplying the inertia matrix in Eq. (2.33) with the angular velocity of the rotor as seen in the body-fixed frame in Eq. (2.14). This vector may be written as

$$\begin{aligned}&\mathbf{H} = [\mathbf{I}]\boldsymbol{\omega} \rightarrow \\ &\mathbf{H} = \left(I_{xx}\omega_x - I_{xy}\omega_y - I_{xz}\omega_z\right)\mathbf{i} + \left(-I_{yx}\omega_y + I_{yy}\omega_y - I_{yz}\omega_z\right)\mathbf{j} + \left(-I_{zx}\omega_x - I_{zy}\omega_y + I_{zz}\omega_z\right)\mathbf{k}\end{aligned} \tag{2.59}$$

Because both the mass-moment-of-inertia matrix and angular velocity of the rotor are written about a set of axes with origin at *P*, the angular momentum vector will be about the geometric center of the rotor, *P*. Substituting for the various values in Eq. (2.59) yields the following expression for the angular momentum of the cylinder about the geometric center *P*

$$\begin{aligned}\mathbf{H}_P = &\left[\dot{\theta}_1\left(\cos\theta_2\left[I_{xx}\cos\theta_3 + I_{xy}\sin\theta_3\right] - I_{xz}\sin\theta_2\right) + \dot{\theta}_2\left(I_{xx}\sin\theta_3 - I_{xy}\cos\theta_3\right) - I_{xz}\dot{\theta}_3\right]\mathbf{i} + \\ &\left[\dot{\theta}_2\left(I_{yy}\cos\theta_3 - I_{xy}\sin\theta_3\right) - \dot{\theta}_1\left(\cos\theta_2\left[I_{xy}\cos\theta_3 + I_{yy}\sin\theta_3\right] + I_{yz}\sin\theta_2\right) - I_{yz}\dot{\theta}_3\right]\mathbf{j} + \\ &\left[\dot{\theta}_1\left(\cos\theta_2\left[I_{yz}\sin\theta_3 - I_{xz}\cos\theta_3\right] + I_{zz}\sin\theta_2\right) - \dot{\theta}_2\left(I_{xz}\sin\theta_3 + I_{yz}\cos\theta_3\right) + I_{zz}\dot{\theta}_3\right]\mathbf{k}\end{aligned} \tag{2.60}$$

It will also be of interest to compute the time rate of change of the angular momentum vector for use in our kinetics equations. It is important to remember that the angular momentum of the rotor is written in a rotating coordinate frame, so its time derivative will involve an extra cross-product term resulting from a change in rotation of the unit vectors. The time derivative of **H** is thus

$$\dot{\mathbf{H}} = \frac{d}{dt}\left([\mathbf{I}]\boldsymbol{\omega}\right) = [\dot{\mathbf{I}}]\boldsymbol{\omega} + [\mathbf{I}]\dot{\boldsymbol{\omega}} + \boldsymbol{\omega}\times\left([\mathbf{I}]\boldsymbol{\omega}\right) = [\mathbf{I}]\boldsymbol{\alpha} + \boldsymbol{\omega}\times\left([\mathbf{I}]\boldsymbol{\omega}\right) \tag{2.61}$$

where the inertia matrix for the rotor/imbalance is constant in time, and the time derivative of the angular velocity is the angular acceleration of the rotor as defined in Eq. (2.17). We may substitute for the above values and simplify to obtain the following vector:

$$\dot{\mathbf{H}}_P = \Big\langle -\dot{\theta}_1 \Big\{ \dot{\theta}_2 \cos\theta_3 \Big(\sin\theta_2 \big[I_{xx} + I_{yy} - I_{zz} \big] + 2\cos\theta_2 \big[I_{xz} \cos\theta_3 - I_{yz} \sin\theta_3 \big] \Big) + \dot{\theta}_3 \Big[\cos\theta_2 \Big(\sin\theta_3 \big[I_{xx} - I_{yy} + I_{zz} \big] - 2I_{xy} \cos\theta_3 \Big) - 2I_{yz} \sin\theta_2 \Big] \Big\} + I_{yz} \dot{\theta}_3^2 + \dot{\theta}_1^2 \Big[\sin\theta_3 \cos^2\theta_2 \Big(I_{xz} \cos\theta_3 - I_{yz} \sin\theta_3 \Big) + \sin\theta_2 \cos\theta_2 \Big(I_{xy} \cos\theta_3 + \big[I_{yy} - I_{zz} \big] \sin\theta_3 \Big) + I_{yz} \sin^2\theta_2 \Big] + \ddot{\theta}_1 \Big(\cos\theta_2 \big[I_{xx} \cos\theta_3 + I_{xy} \sin\theta_3 \big] - I_{xz} \sin\theta_2 \Big) + \ddot{\theta}_2 \Big(I_{xx} \sin\theta_3 - I_{xy} \cos\theta_3 \Big) - I_{xz} \ddot{\theta}_3 + \dot{\theta}_2 \dot{\theta}_3 \Big(\cos\theta_3 \big[I_{xx} - I_{yy} + I_{zz} \big] + 2I_{xy} \sin\theta_3 \Big) - \dot{\theta}_2^2 \Big(I_{yz} \cos^2\theta_3 + I_{xz} \cos\theta_3 \sin\theta_3 \Big) \Big\rangle \mathbf{i} +$$

$$\Big\langle \dot{\theta}_1 \Big\{ \dot{\theta}_3 \Big[\cos\theta_2 \Big(\cos\theta_3 \big[I_{xx} - I_{yy} - I_{zz} \big] + 2I_{xy} \sin\theta_3 \Big) - 2I_{xz} \sin\theta_2 \Big] + \dot{\theta}_2 \sin\theta_3 \Big(\sin\theta_2 \big[I_{xx} + I_{yy} - I_{zz} \big] + 2\cos\theta_2 \big[I_{xz} \cos\theta_3 - I_{yz} \sin\theta_3 \big] \Big) \Big\} - I_{xz} \dot{\theta}_3^2 + \dot{\theta}_1^2 \Big[\cos\theta_3 \cos^2\theta_2 \Big(I_{xz} \cos\theta_3 - I_{yz} \sin\theta_3 \Big) + \sin\theta_2 \cos\theta_2 \Big(I_{xy} \sin\theta_3 + \big[I_{xx} - I_{zz} \big] \cos\theta_3 \Big) - I_{xz} \sin^2\theta_2 \Big] - \ddot{\theta}_1 \Big(\cos\theta_2 \big[I_{xy} \cos\theta_3 + I_{yy} \sin\theta_3 \big] + I_{yz} \sin\theta_2 \Big) + \ddot{\theta}_2 \Big(I_{yy} \cos\theta_3 - I_{xy} \sin\theta_3 \Big) - I_{yz} \ddot{\theta}_3 + \dot{\theta}_2 \dot{\theta}_3 \Big(\sin\theta_3 \big[I_{xx} - I_{yy} - I_{zz} \big] - 2I_{xy} \cos\theta_3 \Big) + \dot{\theta}_2^2 \Big(I_{xz} \sin^2\theta_3 + I_{yz} \cos\theta_3 \sin\theta_3 \Big) \Big\rangle \mathbf{j} +$$

$$\Big\langle \ddot{\theta}_1 \Big(I_{zz} \sin\theta_2 + \cos\theta_2 \big[I_{yz} \sin\theta_3 - I_{xz} \cos\theta_3 \big] \Big) - \ddot{\theta}_2 \Big(I_{yz} \cos\theta_3 + I_{xz} \sin\theta_3 \Big) + I_{zz} \ddot{\theta}_3 + \dot{\theta}_1 \dot{\theta}_2 \Big[2\sin\theta_2 \Big(I_{xz} \cos\theta_3 - I_{yz} \sin\theta_3 \Big) + \cos\theta_2 \Big(I_{zz} + \cos(2\theta_3) \big[I_{yy} - I_{xx} \big] - 2I_{xy} \sin(2\theta_3) \Big) \Big] + \dot{\theta}_2^2 \Big(\sin\theta_3 \cos\theta_3 \big[I_{yy} - I_{xx} \big] + I_{xy} \cos(2\theta_3) \Big) - \frac{\dot{\theta}_1^2 \cos\theta_2}{2} \Big[2\sin\theta_2 \Big(I_{yz} \cos\theta_3 + I_{xz} \sin\theta_3 \Big) + \cos\theta_2 \Big(2I_{xy} \cos(2\theta_3) + \sin(2\theta_3) \big[I_{yy} - I_{xx} \big] \Big) \Big] \Big\rangle \mathbf{k} \tag{2.62}$$

2.2.9 Equations of Motion

At this point, all pertinent force and moment vectors have been written in their appropriate coordinate frames, and Newton's second law can be applied to the rotor and each individual balancing particle to obtain the equations of motion for the system of bodies. The governing equations needed are [74]

$$\Sigma \mathbf{F} = m\mathbf{a} \tag{2.63}$$

$$\Sigma \mathbf{M}_P = \dot{\mathbf{H}}_P + \mathbf{r}_{G/P} \times m\mathbf{a}_P \tag{2.64}$$

For the rotor, Eq. (2.63) will be written in the inertial frame $\{\mathbf{O}\}$ while Eq. (2.64) will be written in the body-fixed frame $\{\mathbf{P}\}$. For each individual balancing particle, Eq. (2.63) will be

written in the body-fixed frame {**P**} and the use of Eq. (2.64) is unnecessary because the forces acting on the particle are concurrent. Writing Eq. (2.63) for the rotor results in

$$\mathbf{F}_{k_X} + \mathbf{F}_{k_Y} + \mathbf{F}_{k_Z} + \mathbf{F}_{c_X} + \mathbf{F}_{c_Y} + \mathbf{F}_{c_Z} + \sum_j \mathbf{F}_{\text{int}_j} = \left(M + m\right)\mathbf{a}_{G/O} \tag{2.65}$$

Writing Eq. (2.64) for the rotor results in

$$\mathbf{M}_{k_{\theta_1}} + \mathbf{M}_{k_{\theta_2}} + \mathbf{M}_{k_{\theta_3}} + \mathbf{M}_{c_{\theta_1}} + \mathbf{M}_{c_{\theta_2}} + \mathbf{M}_{c_{\theta_3}} + \sum_j \mathbf{M}_{\text{int}_j} = \dot{\mathbf{H}}_P + \mathbf{r}_{G/P} \times \left(M + m\right)\mathbf{a}_{P/O} \tag{2.66}$$

Finally, writing Eq. (2.63) for an individual balancing particle results in

$$\mathbf{F}_j = m_j \mathbf{a}_j \tag{2.67}$$

2.3 Coulomb Friction

To this point, nothing has been said regarding the nature of the Coulomb friction force $\mathbf{F}_{f_j}$. Because Coulomb friction is present between the balancing particles and the race, it is possible for this friction force to be such that it prohibits any relative motion of the particle within the race. At other times, it might occur that this friction force is not sufficient to hold the particle in place relative to the rotor and motion occurs in the body-fixed frame. Thus, there must be two different representations of the friction force based on the motion properties of the balancing particle under consideration.

2.3.1 Kinetic Friction Force

If the particle is moving with a relative velocity, i.e. $\dot{\beta}_j \neq 0$, then the friction force is a kinetic friction force, and has the form

$$F_{f_j} = \mu_k N_j \operatorname{sgn}\left(\dot{\beta}_j\right) = \mu_k \sqrt{F_{xy_j}^2 + F_{z_j}^2} \operatorname{sgn}\left(\dot{\beta}_j\right) \simeq \mu_k F_{xy_j} \operatorname{sgn}\left(\dot{\beta}_j\right) \tag{2.68}$$

where it has been assumed the normal force acting on the particle is dominated by the radial force F_{xy}, and the longitudinal force F_z is negligible for this calculation. This is a reasonable

assumption based on the nature of the rotor's motion given certain constraints for rotation about the body-fixed axes. The signum function is used to ensure proper orientation of the kinetic friction force because it always opposes the relative motion of the particle.

2.3.2 Static Friction Force

If the particle is not moving relative to the rotor, i.e. $\dot{\beta}_j = 0$, then the friction force is a static friction force whose magnitude is unknown, but is instead limited by the maximum allowable static friction force which has the form

$$F_{\max} = \mu_s N_j = \mu_s \sqrt{F_{xy_j}^2 + F_{z_j}^2} \simeq \mu_s F_{xy_j} \tag{2.69}$$

where again, it is assumed that the normal force acting on the particle is composed primarily of the radial force F_{xy}. Thus, the static friction force has the requirement that

$$F_{f_j} \leq F_{\max} \tag{2.70}$$

Whenever the friction force is less than or equal to the maximum allowable static friction force, the requirement is also that there be no relative acceleration of the particle with respect to the rotor meaning $\ddot{\beta}_j = 0$ as well. Because of the possibility for relative motion to exist or not exist between a balancing particle and the rotor, the governing kinetic Eqs. (2.65), (2.66), and (2.67) can take on several different forms depending on the motion properties of the individual balancing particles and the resulting form for the Coulomb friction force given by Eq. (2.68) or (2.70).

3 Numerical Integration

With the proper equations of motion written for both the rotor and the individual balancing particles, we are ready to develop the numerical integration schema and logic necessary to simulate the motion of the system. Referring back to the vector Eqs. (2.65), (2.66), and (2.67) there are a total of 18 scalar equations, six for the rotor, and three each for the four balancing particles. The equations are all highly nonlinear so an appropriate numerical integration routine must be adopted. For this work, a four-cycle Runge-Kutta routine will be employed through the MatLab software package. The logic and implementation of said routine is detailed below.

3.1 Coulomb Friction Effect on Equations of Motion

Because each balancing particle has the ability to stop relative to the rotor due to stiction effects from Coulomb friction forces, it is imperative that a check be made at every integration step to determine the motion properties of each balancing particle. Should a particle become stationary relative to the rotor, the friction force acting on it will change from the kinetic frictional force value and alter the equations of motion for the rotor.

3.1.1 Cases of Relative Motion for the Balancing Particles

Because there are four particles which move independently from one another, there are 16 different situations that can occur as summarized in Table 1. The cases are formulated based on whether or not a balancing particle is in motion relative to the rotor, and are assigned a code number to quickly illustrate this property. A check mark (✓) will be used to signify relative motion of a balancing particle at the current time step and a circle (○) will be used to indicate that a particle has become stationary relative to the rotor.

Table 3.1: Relative Motion Cases for the Balancing Particles

Case	Code	Relative Motion of Balancing Particle			
		1	2	3	4
1	0	○	○	○	○
2	1	✓	○	○	○
3	2	○	✓	○	○
4	3	○	○	✓	○
5	4	○	○	○	✓
6	12	✓	✓	○	○
7	13	✓	○	✓	○
8	14	✓	○	○	✓
9	23	○	✓	✓	○
10	24	○	✓	○	✓
11	34	○	○	✓	✓
12	123	✓	✓	✓	○
13	124	✓	✓	○	✓
14	134	✓	○	✓	✓
15	234	○	✓	✓	✓
16	1234	✓	✓	✓	✓

It should be apparent that the code given to each case identifies the particles in motion relative to the rotor. To illustrate, code 1 means only particle "one" is moving relative to the rotor, while code 24 means both particles "two" and "four" are moving relative to the rotor. The number and location of the various balancing particles is case dependent and will be explained in detail in Chapter 5. Table 3.1 represents the number of situations that occurs in a general simulation utilizing all four balancing particles in a dual-plane configuration. Specific simulations might only utilize a subset of these sixteen motion cases.

3.1.2 Equations for a Balancing Particle with Relative Motion

Because there are 16 different cases of relative motion for the balancing particles, there will be 16 different sets of equations of motion for the rotor. In order to correctly solve for all of the system motion variables and forces, a check must be performed at the beginning of every integration time step to determine which, if any, balancing particles have come to rest relative to the rotor, and which set of equations is to be implemented. To this end, we return to the equations of motion for a single balancing particle as illustrated in Eq. (2.67).

Assuming particle *j* has relative motion along the rotor surface, we substitute the Coulomb friction force given by Eq. (2.68) into Eq. (2.67) and simplify by subtracting the $m_j\mathbf{a}_j$ term from both sides of the equation. This results in the following three scalar equations

$$
\begin{aligned}
&F_{xy_j}\left(\mu_k \operatorname{sgn}\dot{\beta}_j \sin\beta_j - \cos\beta_j\right) + \frac{m_j}{4}\Big\{4\cos\theta_3\Big[\sin\theta_2\left(\ddot{Z}\cos\theta_1 - \ddot{Y}\sin\theta_1\right) - \cos\theta_2\left(g + \ddot{X}\right)\Big] - \\
&4\sin\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right) - 4d_j\Big[\cos\theta_3\left(\dot{\theta}_1^2\cos\theta_2\sin\theta_2 + \ddot{\theta}_2\right) + \sin\theta_3\left(2\dot{\theta}_1\dot{\theta}_2\sin\theta_2 - \right. \\
&\left.\ddot{\theta}_1\cos\theta_2\right)\Big] + r_j\Big[4\dot{\beta}_j^{\,2}\cos\beta_j + 8\dot{\beta}_j\dot{\theta}_3\cos\beta_j + \dot{\theta}_1^2\left(3\cos\beta_j - 2\cos\left(2\theta_2\right)\cos\theta_3\cos\left(\beta_j + \theta_3\right)\right. \\
&\left.-\cos\left(\beta_j + 2\theta_3\right)\right) + 4\dot{\theta}_2^{\,2}\cos\theta_3\cos\left(\beta_j + \theta_3\right) + 4\dot{\theta}_3^{\,2}\cos\beta_j\Big]\Big\} + r_j\Big\{m_j\Big[\sin\beta_j\left(\ddot{\beta}_j + \ddot{\theta}_3 + \right. \\
&\left.\ddot{\theta}_1\sin\theta_2\right) + 2\dot{\theta}_1\left(\cos\beta_j\sin\theta_2\left[\dot{\beta}_j + \dot{\theta}_3\right] - \dot{\theta}_2\cos\theta_2\sin\theta_3\cos\left(\beta_j + \theta_3\right)\right)\Big] + \mu_c\dot{\beta}_j\sin\beta_j\Big\} = 0
\end{aligned}
\tag{3.1}
$$

$$
\begin{aligned}
&m_j\Big\{\cos\theta_2\sin\theta_3\left(g + \ddot{X}\right) - \ddot{Z}\left(\sin\theta_1\cos\theta_3 + \cos\theta_1\sin\theta_2\sin\theta_3\right) + \ddot{Y}\left(\sin\theta_1\sin\theta_2\sin\theta_3 - \right. \\
&\left.\cos\theta_1\cos\theta_3\right) + d_j\Big[\cos\theta_3\left(\ddot{\theta}_1\cos\theta_2 - 2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\right) + \sin\theta_3\left(\dot{\theta}_1^2\sin\theta_2\cos\theta_2 + \ddot{\theta}_2\right)\Big] - \\
&r_j\ddot{\beta}_j\cos\beta_j - r_j\left(2\dot{\theta}_1\dot{\theta}_2\cos\theta_2\cos\theta_3\cos\left(\beta_j + \theta_3\right) + \ddot{\theta}_3\cos\beta_j\right)\Big\} - \\
&F_{xy_j}\left(\sin\beta_j + \mu_k\cos\beta_j\operatorname{sgn}\dot{\beta}_j\right) + r_j\Big\{m_j\Big[\dot{\theta}_1^2\cos^2\theta_2\cos\theta_3\sin\left(\beta_j + \theta_3\right) + \\
&\sin\beta_j\left(\dot{\beta}_j + \dot{\theta}_3 + \dot{\theta}_1\sin\theta_2\right)^2 - \ddot{\theta}_1\sin\theta_2\cos\beta_j - \dot{\theta}_2^{\,2}\sin\theta_3\cos\left(\beta_j + \theta_3\right)\Big] - \mu_c\dot{\beta}_j\cos\beta_j\Big\} = 0
\end{aligned}
\tag{3.2}
$$

$$
\begin{aligned}
&F_{z_j} + m_j\Big\{\ddot{Y}\sin\theta_1\cos\theta_2 - \sin\theta_2\left(g + \ddot{X}\right) - \ddot{Z}\cos\theta_1\cos\theta_2 + d_j\left(\dot{\theta}_1^2\cos^2\theta_2 + \dot{\theta}_2^{\,2}\right) - \\
&r_j\Big[\cos\left(\beta_j + \theta_3\right)\Big(\dot{\theta}_1\cos\theta_2\left[2\dot{\beta}_j + 2\dot{\theta}_3 + \dot{\theta}_1\sin\theta_2\right] - \ddot{\theta}_2\Big) + \sin\left(\beta_j + \theta_3\right)\Big(\ddot{\theta}_1\cos\theta_2 + \\
&2\dot{\theta}_2\left[\dot{\beta}_j + \dot{\theta}_3\right]\Big)\Big]\Big\} = 0
\end{aligned}
\tag{3.3}
$$

Each balancing particle has only one degree of freedom, so we can use two of the equations to solve for the interaction forces F_{xy_j} and F_{z_j} in terms of the kinematic variables of the rotor and the balancing particle. Solving Eqs. (3.1) and (3.3) yields

$$F_{xy_j} = \frac{1}{4\left(\cos\beta_j - \mu_k \operatorname{sgn}\dot{\beta}_j \sin\beta_j\right)}\Big\{4\mu_c r_j \dot{\beta}_j \sin\beta_j - m_j\Big[\Big[2d_j\Big(\cos\theta_3\Big[2\ddot{\theta}_2 + \dot{\theta}_1^2\sin\left(2\theta_2\right)\Big] -$$
$$2\sin\theta_3\Big[\ddot{\theta}_1\cos\theta_2 - 2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\Big]\Big) + 4g\cos\theta_2\cos\theta_3 - r_j\Big(4\sin\beta_j\Big[\ddot{\beta}_j + \ddot{\theta}_3 + \ddot{\theta}_1\sin\theta_2\Big] +$$
$$4\dot{\theta}_2^{\,2}\cos\theta_3\cos\left(\beta_j + \theta_3\right) + 4\dot{\theta}_3\Big[2\dot{\beta}_j + \dot{\theta}_3\Big]\cos\beta_j + \dot{\theta}_1^2\Big[-2\cos\left(2\theta_2\right)\cos\theta_3\cos\left(\beta_j + \theta_3\right) - \qquad (3.4)$$
$$\cos\left(\beta_j + 2\theta_3\right) + 3\cos\beta_j\Big] + 8\dot{\theta}_1\Big[\sin\theta_2\cos\beta_j\left(\dot{\beta}_j + \dot{\theta}_3\right) - \dot{\theta}_2\sin\theta_3\cos\theta_2\cos\left(\beta_j + \theta_3\right)\Big]\Big) -$$
$$4r_j\cos\beta_j\dot{\beta}_j^{\,2} + 4\cos\theta_3\left(\ddot{X}\cos\theta_2 + \sin\theta_2\left[\ddot{Y}\sin\theta_1 - \ddot{Z}\cos\theta_1\right]\right) + 4\sin\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right)\Big]\Big\}$$

$$F_{z_j} = m_j\Big\{\sin\theta_2\left(g + \ddot{X}\right) - d_j\left(\dot{\theta}_1^2\cos^2\theta_2 + \dot{\theta}_2^{\,2}\right) + r_j\Big[\cos\left(\beta_j + \theta_3\right)\Big(\dot{\theta}_1\cos\theta_2\Big[2\dot{\beta}_j + 2\dot{\theta}_3 +$$
$$\dot{\theta}_1\sin\theta_2\Big] - \ddot{\theta}_2\Big) + \sin\left(\beta_j + \theta_3\right)\Big(2\dot{\theta}_2\Big[\dot{\beta}_j + \dot{\theta}_3\Big] + \ddot{\theta}_1\cos\theta_2\Big)\Big] - \ddot{Y}\sin\theta_1\cos\theta_2 + \ddot{Z}\cos\theta_1\cos\theta_2\Big\} \qquad (3.5)$$

Now, we substitute Eq. (3.4) into Eq. (3.2) to develop a single equation of motion for the balancing particle without reference to the unknown forces of interaction. Doing this yields the following equation:

$$m_j\Big\{\cos\theta_2\sin\theta_3\left(g + \ddot{X}\right) - \ddot{Z}\left(\sin\theta_1\cos\theta_3 + \cos\theta_1\sin\theta_2\sin\theta_3\right) + \ddot{Y}\left(\sin\theta_1\sin\theta_2\sin\theta_3 -$$
$$\cos\theta_1\cos\theta_3\right) + d_j\Big[\Big[\cos\theta_3\left(\ddot{\theta}_1\cos\theta_2 - 2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\right) + \sin\theta_3\left(\dot{\theta}_1^2\sin\theta_2\cos\theta_2 + \ddot{\theta}_2\right)\Big] -$$
$$r_j\ddot{\beta}_j\cos\beta_j - r_j\left(2\dot{\theta}_1\dot{\theta}_2\cos\theta_2\cos\theta_3\cos\left(\beta_j + \theta_3\right) + \ddot{\theta}_3\cos\beta_j\right)\Big\} -$$
$$\frac{\sin\beta_j + \mu_k\cos\beta_j\operatorname{sgn}\dot{\beta}_j}{4\left(\cos\beta_j - \mu_k\operatorname{sgn}\dot{\beta}_j\sin\beta_j\right)}\Big\{4\mu_c r_j\dot{\beta}_j\sin\beta_j - m_j\Big[\Big[2d_j\Big(\cos\theta_3\Big[2\ddot{\theta}_2 + \dot{\theta}_1^2\sin\left(2\theta_2\right)\Big] -$$
$$2\sin\theta_3\Big[\ddot{\theta}_1\cos\theta_2 - 2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\Big]\Big) + 4g\cos\theta_2\cos\theta_3 - r_j\Big(4\sin\beta_j\Big[\ddot{\beta}_j + \ddot{\theta}_3 + \ddot{\theta}_1\sin\theta_2\Big] +$$
$$4\dot{\theta}_2^{\,2}\cos\theta_3\cos\left(\beta_j + \theta_3\right) + 4\dot{\theta}_3\Big[2\dot{\beta}_j + \dot{\theta}_3\Big]\cos\beta_j + \dot{\theta}_1^2\Big[-2\cos\left(2\theta_2\right)\cos\theta_3\cos\left(\beta_j + \theta_3\right) -$$
$$\cos\left(\beta_j + 2\theta_3\right) + 3\cos\beta_j\Big] + 8\dot{\theta}_1\Big[\sin\theta_2\cos\beta_j\left(\dot{\beta}_j + \dot{\theta}_3\right) - \dot{\theta}_2\sin\theta_3\cos\theta_2\cos\left(\beta_j + \theta_3\right)\Big]\Big) -$$
$$4r_j\cos\beta_j\dot{\beta}_j^{\,2} + 4\cos\theta_3\left(\ddot{X}\cos\theta_2 + \sin\theta_2\left[\ddot{Y}\sin\theta_1 - \ddot{Z}\cos\theta_1\right]\right) +$$
$$4\sin\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right)\Big]\Big\} + r_j\Big\{m_j\Big[\Big[\dot{\theta}_1^2\cos^2\theta_2\cos\theta_3\sin\left(\beta_j + \theta_3\right) +$$
$$\sin\beta_j\left(\dot{\beta}_j + \dot{\theta}_3 + \dot{\theta}_1\sin\theta_2\right)^2 - \ddot{\theta}_1\sin\theta_2\cos\beta_j - \dot{\theta}_2^{\,2}\sin\theta_3\cos\left(\beta_j + \theta_3\right)\Big] - \mu_c\dot{\beta}_j\cos\beta_j\Big\} = 0 \qquad (3.6)$$

This same process is repeated for every particle that has motion relative to the rotor. Next, we will examine the situation where a particle has come to rest relative to the rotor.

3.1.3 Equations for a Balancing Particle without Relative Motion

As previously noted, it is possible for a balancing particle to come to rest relative to the rotor due to the presence of Coulomb friction. In this event, the motion variables corresponding to the angular velocity and angular acceleration of the particle within its race become zero, and an unknown friction force is developed at the particle/rotor interface. This force will affect the motion variables for the rotor and must be determined via a static friction force that is limited by the maximum allowable static friction force given by Eq. (2.69). In order to account for this force, we return to Eq. (2.67) and set the angular velocity ($\dot{\beta}_j$) and angular acceleration ($\ddot{\beta}_j$) of the balancing particle equal to zero and enforce a stiction effect. Now, we can simplify by subtracting the $m_j\mathbf{a}_j$ term from both sides of the equation and obtain the following three scalar equations.

$$
\begin{aligned}
&F_{f_j}\sin\beta_j - F_{xy_j}\cos\beta_j + m_j\Big[\Big[\cos\theta_3\left(\sin\theta_2\left[\ddot{Z}\cos\theta_1 - \ddot{Y}\sin\theta_1\right] - \cos\theta_2\left[g + \ddot{X}\right]\right) - \\
&d_j\left(\cos\theta_3\left[\ddot{\theta}_2 + \dot{\theta}_1^2\sin\theta_2\cos\theta_2\right] + \sin\theta_3\left[2\dot{\theta}_1\dot{\theta}_2\sin\theta_2 - \ddot{\theta}_1\cos\theta_2\right]\right) + \\
&\frac{r_j}{4}\Big(\dot{\theta}_1^2\left[-2\cos(2\theta_2)\cos\theta_3\cos\left(\beta_j+\theta_3\right) - \cos\left(\beta_j+2\theta_3\right) + 3\cos\beta_j\right] + \\
&4\left[\dot{\theta}_2^{\,2}\cos\theta_3\cos\left(\beta_j+\theta_3\right) + \dot{\theta}_3^{\,2}\cos\beta_j\right]\Big) - \sin\theta_3\left(\ddot{Y}\cos\theta_1 + \ddot{Z}\sin\theta_1\right)\Big] + \\
&m_j r_j\left(\sin\beta_j\left[\ddot{\theta}_3 + \ddot{\theta}_1\sin\theta_2\right] + 2\dot{\theta}_1\left[\dot{\theta}_3\sin\theta_2\cos\beta_j - \dot{\theta}_2\sin\theta_3\cos\theta_2\cos\left(\beta_j+\theta_3\right)\right]\right) = 0
\end{aligned}
\tag{3.7}
$$

$$
\begin{aligned}
&m_j\Big[\Big[d_j\left(\cos\theta_3\left[\ddot{\theta}_1\cos\theta_2 - 2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\right] + \sin\theta_3\left[\ddot{\theta}_2 + \dot{\theta}_1^2\sin\theta_2\cos\theta_2\right]\right) + \\
&\sin\theta_3\cos\theta_2\left(g + \ddot{X}\right) - r_j\left(\ddot{\theta}_3\cos\beta_j + 2\dot{\theta}_1\dot{\theta}_2\cos\theta_2\cos\theta_3\cos\left(\beta_j+\theta_3\right)\right) + \\
&\ddot{Y}\left(\sin\theta_1\sin\theta_2\sin\theta_3 - \cos\theta_1\cos\theta_3\right) - \ddot{Z}\left(\sin\theta_1\cos\theta_3 + \cos\theta_1\sin\theta_2\sin\theta_3\right)\Big]\Big] \\
&-F_{f_j}\cos\beta_j - F_{xy_j}\sin\beta_j + m_j r_j\Big(\sin\beta_j\left[\dot{\theta}_3 + \dot{\theta}_1\sin\theta_2\right]^2 - \\
&\ddot{\theta}_1\sin\theta_2\cos\beta_j + \dot{\theta}_1^2\cos^2\theta_2\cos\theta_3\sin\left(\beta_j+\theta_3\right) - \dot{\theta}_2^{\,2}\sin\theta_3\cos\left(\beta_j+\theta_3\right)\Big) = 0
\end{aligned}
\tag{3.8}
$$

$$
\begin{aligned}
&F_{z_j} + m_j\Big\{d_j\left(\dot{\theta}_2^{\,2} + \dot{\theta}_1^2\cos^2\theta_2\right) - \sin\theta_2\left(g + \ddot{X}\right) - r_j\Big[\Big[\sin\left(\beta_j+\theta_3\right)\left(\ddot{\theta}_1\cos\theta_2 + 2\dot{\theta}_2\dot{\theta}_3\right) + \\
&\cos\left(\beta_j+\theta_3\right)\left(\dot{\theta}_1\cos\theta_2\left[2\dot{\theta}_3 + \dot{\theta}_1\sin\theta_2\right] - \ddot{\theta}_2\right)\Big]\Big] + \ddot{Y}\sin\theta_1\cos\theta_2 - \ddot{Z}\cos\theta_1\cos\theta_2\Big\} = 0
\end{aligned}
\tag{3.9}
$$

These three equations allow us to solve for the three components of the interaction force. This results in the following

$$\begin{aligned}F_{xy_j} = \frac{m_j}{4}\Big\{&-4\cos\left(\beta_j+\theta_3\right)\left(\cos\theta_2\left[g+\ddot{X}\right]+\sin\theta_2\left[\ddot{Y}\sin\theta_1-\ddot{Z}\cos\theta_1\right]\right)+\\ &2d_j\left(2\sin\left(\beta_j+\theta_3\right)\left[\ddot{\theta}_1\cos\theta_2-2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\right]-\cos\left(\beta_j+\theta_3\right)\left[2\ddot{\theta}_2+\dot{\theta}_1^{\,2}\sin\left(2\theta_2\right)\right]\right)+\\ &r_j\Big[\Big[4\left(\dot{\theta}_2^{\,2}\cos^2\left(\beta_j+\theta_3\right)+\dot{\theta}_3^{\,2}\right)-\dot{\theta}_1\left(\dot{\theta}_1\left[2\cos^2\theta_2\cos\left(2\beta_j+2\theta_3\right)+\left(2\theta_2\right)-3\right]+\\ &4\dot{\theta}_2\cos\theta_2\sin\left(2\beta_j+2\theta_3\right)-8\dot{\theta}_3\sin\theta_2\right)\Big]-4\sin\left(\beta_j+\theta_3\right)\left(\ddot{Y}\cos\theta_1+\ddot{Z}\sin\theta_1\right)\Big\}\end{aligned} \tag{3.10}$$

$$\begin{aligned}F_{f_j} = \frac{m_j}{2}\Big[\Big[&2\cos\theta_2\sin\left(\beta_j+\theta_3\right)\left(g+\ddot{X}\right)+d_j\left(\sin\left(\beta_j+\theta_3\right)\left[2\ddot{\theta}_2+\dot{\theta}_1^{\,2}\sin\left(2\theta_2\right)\right]+\right.\\ &2\cos\left(\beta_j+\theta_3\right)\left[\ddot{\theta}_1\cos\theta_2-2\dot{\theta}_1\dot{\theta}_2\sin\theta_2\right]\Big)+r_j\Big(\sin\left(2\beta_j+2\theta_3\right)\left[\dot{\theta}_1^{\,2}\cos^2\theta_2-\dot{\theta}_2^{\,2}\right]-\\ &2\left[\dot{\theta}_1\dot{\theta}_2\cos\theta_2\cos^2\left(\beta_j+\theta_3\right)+\ddot{\theta}_3+\ddot{\theta}_1\sin\theta_2\right]\Big)+2\ddot{Y}\Big(\sin\theta_1\sin\theta_2\sin\left(\beta_j+\theta_3\right)-\\ &\left.\cos\theta_1\cos\left(\beta_j+\theta_3\right)\right)-2\ddot{Z}\left(\sin\theta_1\cos\left(\beta_j+\theta_3\right)+\sin\theta_2\cos\theta_1\sin\left(\beta_j+\theta_3\right)\right)\Big]\end{aligned} \tag{3.11}$$

$$\begin{aligned}F_{z_j} = m_j\Big\{&\sin\theta_2\left(g+\ddot{X}\right)-d_j\left(\dot{\theta}_2^{\,2}+\dot{\theta}_1\cos^2\theta_2\right)+r_j\Big[\Big[\cos\left(\beta_j+\theta_3\right)\Big(\dot{\theta}_1\cos\theta_2\left[2\dot{\theta}_3+\dot{\theta}_1\right]-\\ &\ddot{\theta}_2\Big)+\sin\left(\beta_j+\theta_3\right)\left(\ddot{\theta}_1\cos\theta_2+2\dot{\theta}_2\dot{\theta}_3\right)\Big]\Big]-\ddot{Y}\sin\theta_1\cos\theta_2+\ddot{Z}\cos\theta_1\cos\theta_2\end{aligned} \tag{3.12}$$

This same process is repeated for every particle that has come to rest relative to the rotor.

3.2 Numerical Integration Strategy

Now that the equations governing a particular balancing particle either in motion or non-motion relative to the rotor have been developed, we are ready to discuss the solution of the system equations of motion.

3.2.1 Overview of the Four-Cycle Runge-Kutta Numerical Integration Scheme

It should be apparent from the forces involved and the complex motion properties of the system that the equations of motion for the rotor and balancing particles are extremely nonlinear and highly coupled. As such, we will numerically solve the equations of motion via a four-cycle

Runge-Kutta (RK4) routine. The RK4 method utilizes a fourth-order Taylor series to compute the values of the system motion variables at successive time steps with the following equation

$$y_{k+1} = y_k + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4 \tag{3.13}$$

where the k_1, k_2, k_3, and k_4 are given by the following equations

$$\begin{aligned} k_1 &= f\left(t_k, y_k\right) \\ k_2 &= f\left(t_k + \frac{h}{2}, y_k + \frac{hk_1}{2}\right) \\ k_3 &= f\left(t_k + \frac{h}{2}, y_k + \frac{hk_2}{2}\right) \\ k_3 &= f\left(t_k + h, y_k + hk_3\right) \end{aligned} \tag{3.14}$$

The y_k in the above equations represents the state vector for the system and consists of the following quantities

$$y_k = \left(X, Y, Z, \theta_1, \theta_2, \theta_3, \beta_j, \dot{X}, \dot{Y}, \dot{Z}, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\beta}_j\right) \tag{3.15}$$

where all values are taken at time t_k. The h in the above equations represents the time interval chosen between successive computations. The k vectors represent the time derivatives of the state vector at different locations within the current time step. Thus, a typical k vector has the following form

$$k_i = \left(\dot{X}, \dot{Y}, \dot{Z}, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\beta}_j, \ddot{X}, \ddot{Y}, \ddot{Z}, \ddot{\theta}_1, \ddot{\theta}_2, \ddot{\theta}_3, \ddot{\beta}_j\right) \tag{3.16}$$

The f in the above equations represents the functions that govern the dynamics of the system and are obtained from Eqs. (2.65), (2.66), and (2.67). Typically, to solve for the k vectors, the system is put into state-space form where the second-order time derivatives of the motion variables are solved for in terms of the remaining terms. To simplify calculations for this process, we will circumvent the traditional state-space method and utilize a matrix method consisting of the following equation

$$\left[\mathbf{M}\left(t, y\right)\right]\left\{\mathbf{y}''\right\} = \left\{\mathbf{f}\left(t, y\right)\right\} \tag{3.17}$$

The $\{\mathbf{y}''\}$ represents a vector of all the accelerations for the system at the current time step, i.e., the second half of the k_i vector. The [$\mathbf{M}(t, y)$] represents a coefficient matrix multiplying the accelerations, and $\{\mathbf{f}(t, y)\}$ represents all remaining terms in the equations of motion. This method eliminates the need to decouple the acceleration terms in the equations of motion and allows a direct solution for the accelerations of the system at the current time step. We can compute the accelerations through the following equation

$$\{\mathbf{y}''\} = \left[\mathbf{M}\left(t, y\right)\right]^{-1} \left\{\mathbf{f}\left(t, y\right)\right\} \tag{3.18}$$

Since we are solving an initial value problem and we already know the velocities of the system at the current time step, we use Eq. (3.18) to compute the accelerations at the current time step and append the accelerations to the velocities to form the necessary k_i vectors for the Runge-Kutta scheme.

3.2.2 Relative Motion Check & Logic Decisions

At every time step, a check must be made of all four balancing particles to determine whether or not they have become stationary relative to the rotor, $\dot{\beta}_j = 0$. Since the solution will be done numerically, the likelihood of a balancing particle having an angular velocity in its race that is exactly zero is highly unlikely. To account for this, the relative angular velocity of the balancing particle will be compared with a tolerance value, *tol,* where $tol << 1$. If the absolute value of the relative angular velocity of the particle is greater than or equal to the tolerance, $\left|\dot{\beta}_j\right| \geq tol$, then the particle possesses relative motion with respect to the rotor, the friction force is kinetic in nature, and it is assumed that the particle remains in motion relative to the rotor for the duration of the current time step. If, however, the absolute value of the relative angular velocity of the particle is less than the tolerance value, $\left|\dot{\beta}_j\right| < tol$, then the particle is assumed to have stopped relative to the rotor. When this occurs, the friction force becomes static in nature and the relative angular acceleration of the particle goes to zero. It is imperative that this assumption of static friction be validated because it is possible that the particle is merely changing direction relative to the rotor and is passing through a "zero" without actually stopping. To check this assumption, Eqs. (3.10) and (3.11) must be solved and substituted into Eqs. (2.69)

and (2.70). If the static friction force is less than or equal to the maximum allowable static friction force, then the assumption of stoppage is valid and the particle is assumed to remain stopped for the duration of the current time step. If the static friction force is greater than the maximum allowable static friction force, then the particle is merely passing through a "zero" as it changes direction and the friction force is still kinetic in nature. When this occurs, the current time step must be computed again with the relative motion equations for that balancing particle before continuing in the integration process. Depending on which particles become stationary or remain in relative motion, a different set of equations must be solved. This is because the equations governing the interaction forces between the particles and the rotor have different forms based on the existence of relative motion for the balancing particles. There are a total of 16 different motion types that can occur as outlined in Table 3.1 and each motion case has its own assumptions and set of equations of motion that must be numerically integrated for the current time step. The logic behind the numerical integration procedure for each motion case is outlined in Table A.1 of Appendix A. For a partial listing of the code necessary for solving the equations of motion for each case as well as the corresponding interaction forces, the reader is encouraged to examine Appendices B-D.

In all, there are sixteen motion cases that can occur at the beginning of a time step. Based on the returned values for the accelerations of the system and corresponding interaction forces along with checks of the maximum allowable static friction forces at the beginning of the time step, i.e. computation of the k_1 vector in the Runge-Kutta scheme, the program will either continue with the current time step and original assumptions, or will choose a different set of assumptions and compute the four k_i vectors for the current time step starting from the beginning of the time step. This results in a total of 81 different logic paths through each current time step and places considerable complexity on the programming involved. It is for this reason that the majority of research conducted in automatic balancers ignores the effects of Coulomb friction, and those that include it utilize only one or two balancing particles with significant simplifications in the governing equations of motion.

4 Automatic Balancing Principles

Before any numerical simulations are discussed, it will be helpful to illustrate the guiding principles which make automatic balancers effective in reducing unwanted vibrations in rotating machinery. A simple two-dimensional model of a rotor and support structure will be utilized to draw conclusions regarding the effectiveness of an AB.

4.1 Operation Below the Rotor Critical Speed

Figure 4.1 shows a typical AB setup for an unbalanced rotor. The rotor consists of a uniform disk of mass M supported by linear springs and dampers in the inertial x- and y-directions. The gray box surrounding the rotor is for illustrative purposes only to give attachment to the springs and dampers that are required to stretch in directions parallel to the inertial coordinate-frame axes. For the purposes of establishing a general theory for why ABs work, we will assume that the stiffness of the springs are identical. Further, we will assume that the viscous damping coefficients are identical and very small. This is done so that the damped natural frequency differs only slightly from the undamped natural frequency of the supports. The rotor is assumed to operate at a constant speed Ω and possesses an imbalance mass m that is located some distance e from the geometric center P of the rotor. The presence of this imbalance, or defect, will create a translational displacement of the geometric center P from the balanced position O and will cause the geometric center to move in a circle about point O.

When the rotor operates at a speed below the natural frequency of the support, the rotor and the mass imbalance will be in phase with one another. This means that the geometric center P and the mass imbalance m will be on the same side of the balanced position O. In an effort to balance the rotor a particle of mass m_j is introduced at some radial distance r_j from the geometric center. In the figure, this radial distance is shown to be at the periphery of the rotor, but it could in fact be placed at any physically meaningful radial distance from the geometric center P. To balance the rotor, the balancing particle must travel to the opposite side of the rotor from the imbalance mass m and occupy the position denoted m_j'. Further, the particle must be properly sized, meaning it is able to counteract the effect of the imbalance. Formally, this is written as

$$m_j r_j = me \tag{4.1}$$

These two criteria will ensure that the center of mass for the rotor lies at the geometric center of the rotor P, which will then align with the balanced position O.

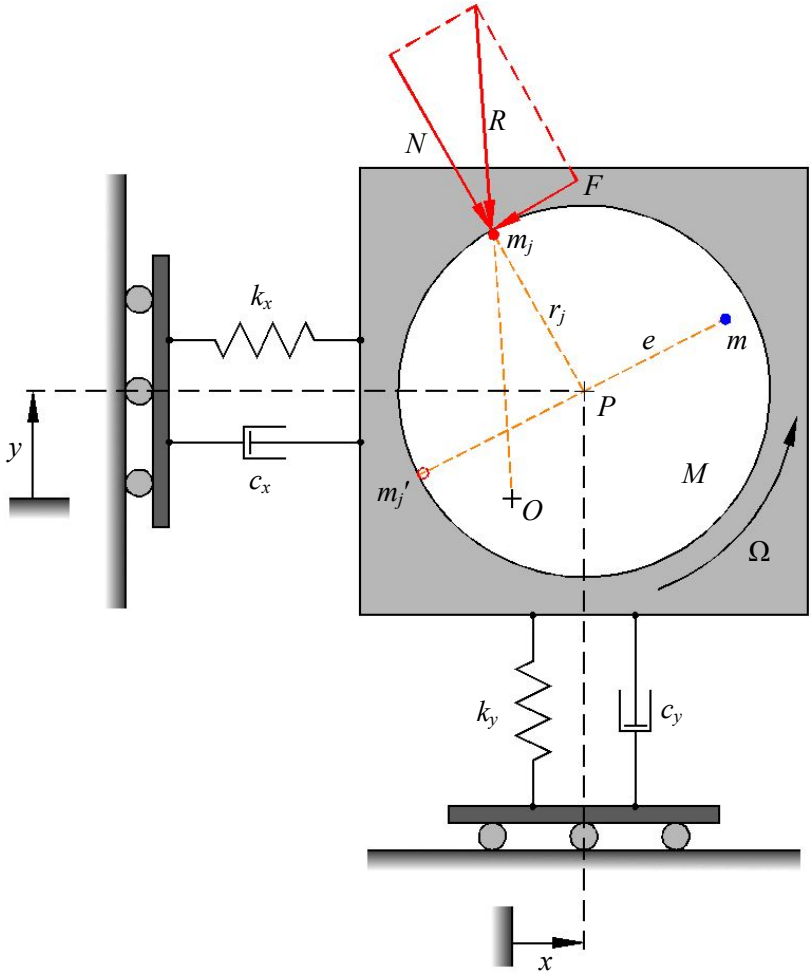

Figure 4.1: Automatic Balancer Operation Below Critical Speed

The balancing particle will have a net acceleration towards the center of rotation, which is the balanced position O, which will create a resultant force R which acts on the particle. Assuming the walls of the ABB are smooth, the resultant force R can be broken down into two components: a normal force N which is perpendicular to the circular track of the AB and is directed toward the geometric center P, and a tangentially directed force F parallel to the AB track. Since the walls of the AB are smooth, and there is no externally applied force being input to the balancing mass, it is not possible to create the force F that is consistent with the net acceleration of the particle. As a result, the balancing mass will be forced to move in a direction

opposite to that of F and will shift to the side of the rotor containing the imbalance m. This movement will create a degradation in performance since the rotor will now feature a 'heavier' spot than previously.

4.2 Operation Above the Rotor Critical Speed

If the situation is now reversed and the rotor is operated above its critical speed, an interesting phenomenon occurs. Figure 4.2 shows the AB performance above critical speed. Because we are now operating above the critical speed the geometric center P and the imbalance m become out of phase with one another. This means they are on opposite sides of the balanced position O. We again introduce a balancing particle m_j at some radial distance r_j from the geometric center. As before, the balancing particle will have a net acceleration directed towards the center of rotation O which can be broken down into a normal component perpendicular to the ABB track, and a tangential component parallel to the AB track. Since the track is smooth and there is no additional force applied to the balancing particle, we are unable to produce the force F consistent with the net acceleration of the particle, and the particle will travel in a direction opposite to F until it stops in a position directly opposite the imbalance mass from the geometric center P. If the particle is sized appropriately using the previously established criteria, then the rotor will, after a short transient period of its motion, settle into a balanced operation.

In summary, an AB has the capacity to balance a rotor provided it operates above the critical speed, or natural frequency, of the support, and if the balancing mass or masses are sized appropriately. In practical applications, a plurality of balancing masses is used to eliminate the necessity of using a perfectly sized mass located at the correct radial distance from the primary axis of operation, or spin-axis, for the rotor. This feature implies that the balancing masses for the rotor must be able to position the center of mass for the combination of rotor, imbalance mass, and balancing masses on the primary axis of rotation of the rotor. This is written formerly as

$$m\mathbf{r}_m + \sum_j m_j \mathbf{r}_j = \mathbf{0} \tag{4.2}$$

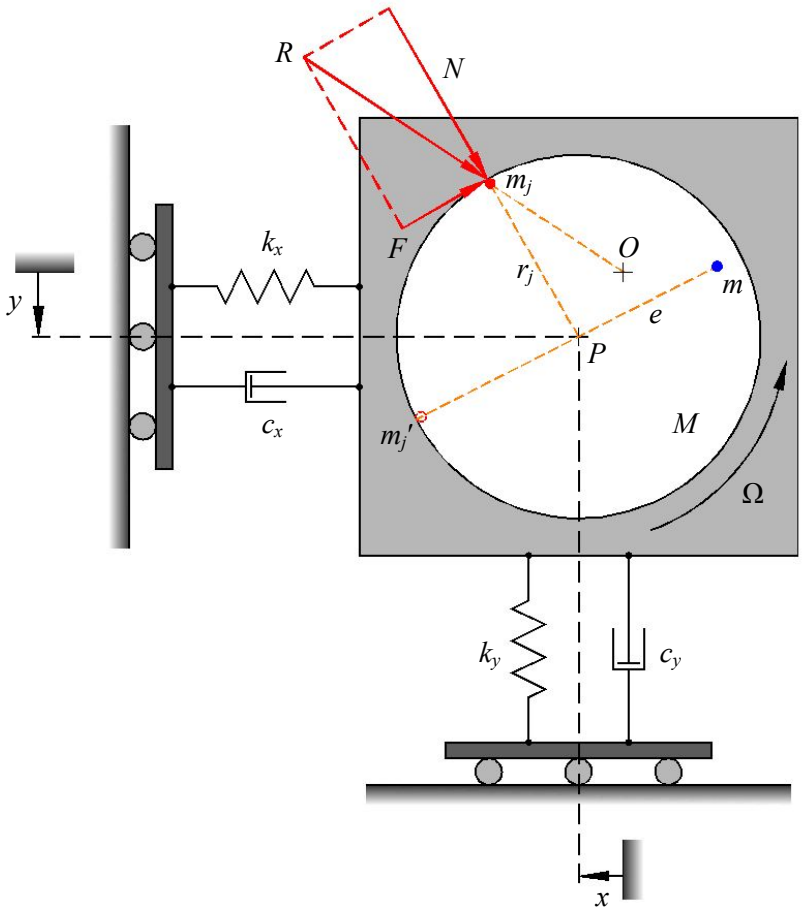

Figure 4.2: Automatic Balancer Operation Above Critical Speed

The above argument has only addressed the suppression of translational vibrations perpendicular to the primary axis of rotation for the rotor and said nothing of whirling motions or wobble effects due to the presence of products of inertia about a set of body-fixed coordinate axes for the rotor. There is currently no established theory regarding the suppression of these motions in the case of an arbitrary imbalance which creates both static and dynamic imbalance effects in the rotor. It is clear that if these motions are to be eliminated, the AB must be able to eliminate all products of inertia. This is written formally as

$$\begin{aligned} I_{xy} &= mr_{m_x} r_{m_y} + \sum_j m_j r_{j_x} r_{j_y} = 0 \\ I_{xz} &= mr_{m_x} r_{m_z} + \sum_j m_j r_{j_x} r_{j_z} = 0 \\ I_{yz} &= mr_{m_y} r_{m_z} + \sum_j m_j r_{j_y} r_{j_z} = 0 \end{aligned} \tag{4.3}$$

Of primary concern will be the elimination of the product of inertia in the plane perpendicular to the primary axis of rotation for the rotor. There is a grand total of six requirements that must be met if the AB is to balance the rotor perfectly: three center-of-mass coordinates and three products of inertia must go to zero as measured in a body-fixed coordinate system fixed at the geometric center of the rotor.

5 Numerical Results

Now that the various sets of governing equations of motion have been developed, we are ready to explore the effects of varying such quantities as Coulomb friction levels on the performance of the rotor and ABB. Individual parameter variations will be done for a range of operating speeds.

5.1 Overview of Numerical Simulations

To examine the effects of Coulomb friction, numerous simulations will be examined. First, simulations for a statically imbalanced rotor will be examined in Section 5.2. The statically imbalanced rotor features only translational displacements of the geometric center P in the plane of the rotor and will be considered in a vertical-axis orientation where gravitational effects are neglected (as is commonly found in the literature) and also a horizontal-axis orientation where gravitational effects are included. This work will use the terms vertical-axis and vertically-oriented interchangeably to indicate that the plane of the rotor is perpendicular to the line of action for the gravitational field. Similarly, the terms horizontal-axis and horizontally-oriented will refer to a rotor that lies parallel to the line of action for the gravitational field. The balancing performance of a one- and two-particle balancer will be explored for a range of operating speeds that lie above and below the critical speed as defined by the support characteristics.

Next, we will examine a dynamically imbalanced rotor in Section 5.3. The dynamically imbalanced rotor features only angular displacements about the intermediate coordinate axes with no translational displacement of P. Owing to the fact that this is a three-dimensional situation, dual-plane balancing performance will be examined using one particle per race and also two particles per race. The same support characteristics will be explored utilizing operating speeds both above and below the critical speed(s) of the support, but only vertically-oriented rotors lacking gravitational effects will be considered in this section.

Finally, in Section 5.4, we will explore the balancing performance of a dual-plane balancer with two particles per race fitted to a rotor having both a static and dynamic imbalance. We will consider only vertically-oriented rotors to minimize the number of variables in this more

general scenario. As before, a range of operating speeds above and below the critical speed(s) of the rotor will be examined.

5.2 Single-Plane Balancing of a Statically Imbalanced Rotor

The first statically-imbalanced rotor we will examine is shown in Figure 5.2.1. The rotor possesses an imbalance mass, shown in blue, located at its mid-plane on the inner radius of the cylindrical rotor at an angular position of 0° relative to the body-fixed coordinate axes. We will first introduce a single balancing particle m_1, shown in red, having identical mass with the imbalance, and located in the same plane as the imbalance at the same radial distance from P. To balance the rotor, the particle must travel to the 180° position, shown by the dark red particle labeled m_1', relative to the body-fixed coordinate system. The calculation for this is shown below.

$$\overline{x} = 0 = mr_i + m_1 r_1 \cos\beta_1 = mr_i\left(1+\cos\beta_1\right) \rightarrow \beta_1 = \pm\pi \tag{5.1}$$

When we switch to the case with two balancing particles m_1 and m_2, we again require that their mass, radial distance from P and axial location are identical with the imbalance mass. The situation is illustrated in Figure 5.2.2 with the balancing particles shown in red. To satisfy the requirement that the center of mass for the rotor system must lie at P for balanced performance, we determine that the two particles should end up symmetric about the body-fixed x-axis and have an angle of 120° between them as shown by the dark red particles m_1' and m_2'. The calculation for this is shown below.

$$\left.\begin{aligned} \overline{x} = 0 = mr_i + m_1 r_1 \cos\beta_1 + m_2 r_2 \cos\beta_2 = mr_i\left(1+\cos\beta_1+\cos\beta_2\right) \\ \overline{y} = 0 = m_1 r_1 \sin\beta_1 + m_2 r_2 \sin\beta_2 = mr_i\left(\sin\beta_1+\sin\beta_2\right) \end{aligned}\right\} \rightarrow \begin{cases} \beta_1 = \pm\dfrac{2\pi}{3} \\ \beta_2 = \mp\dfrac{2\pi}{3} \end{cases} \tag{5.2}$$

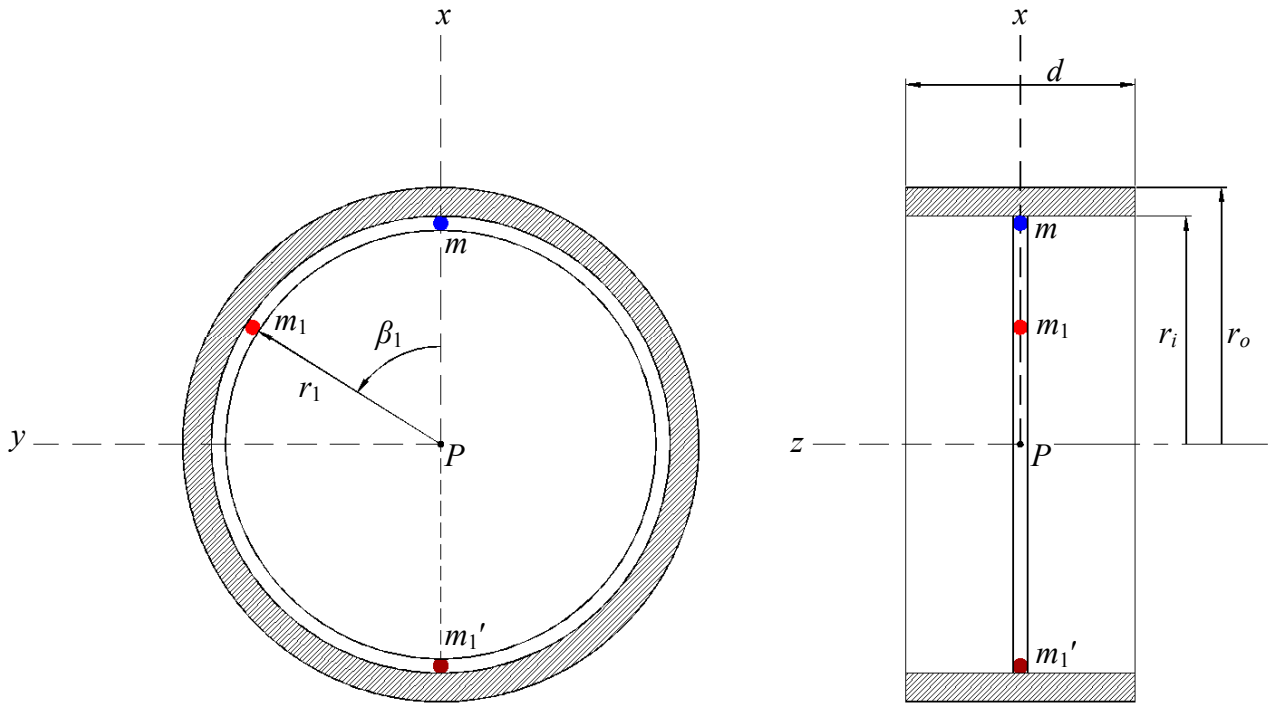

Figure 5.2.1: Single-Plane Balancing of a Statically-Imbalanced Rotor Utilizing a Single Balancing Particle

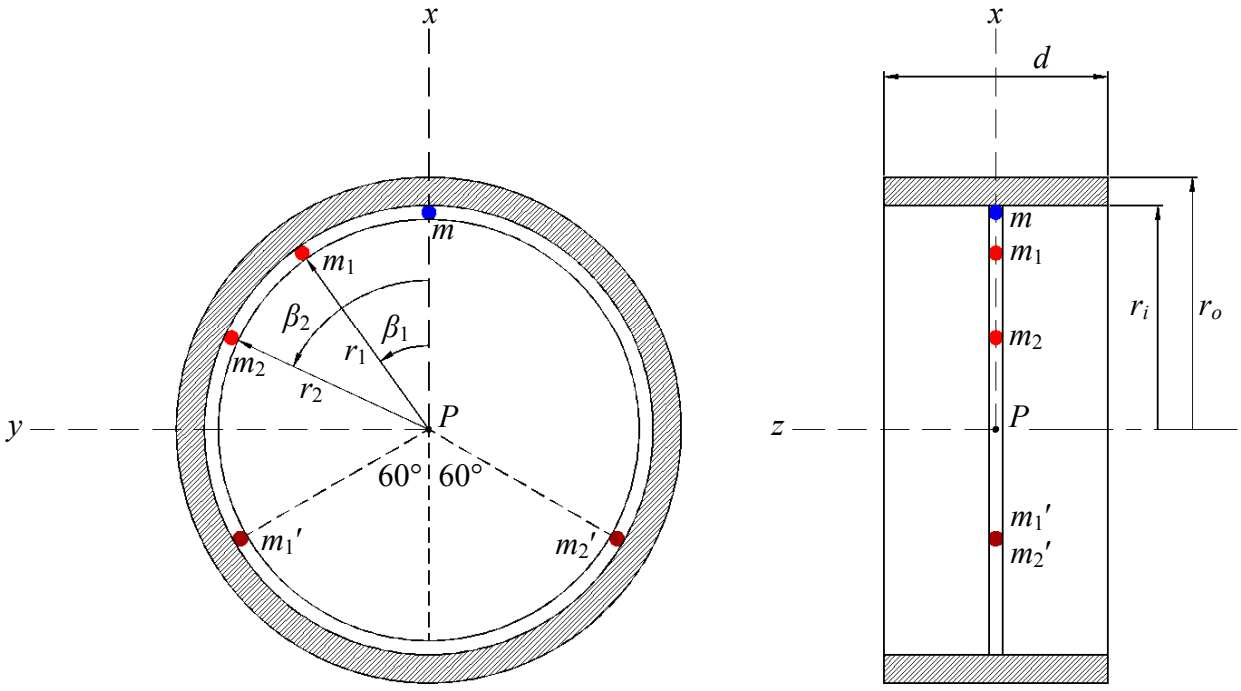

Figure 5.2.2: Single-Plane Balancing of a Statically-Imbalanced Rotor Utilizing Two Balancing Particles

Table 5.2.1 lists the selected system properties for cases involving a statically-imbalanced rotor having a single balancing particle or two balancing particles.

Table 5.2.1: Selected Values for System Properties (Statically-Imbalanced Rotor)

Property	Value	Property	Value
Mass of the rotor, M	1.242 slugs	Spring stiffness along X-axis, k_X	1000 lb/ft
Mass of the imbalance, m	0.001941 slugs	Spring stiffness along Y-axis, k_Y	1000 lb/ft
Mass of a balancing particle, m_j	0.001941 slugs	Spring stiffness along Z-axis, k_Z	100,000 lb/ft
Acceleration of gravity, g	32.2 ft/sec^2	Torsional stiffness along X-axis, k_{θ_1}	5000 lb-ft/rad
Inner radius of rotor, r_i	8 in.	Torsional stiffness along y_2-axis, k_{θ_2}	5000 lb-ft/rad
Outer radius of rotor, r_o	9 in.	Torsional stiffness along z-axis, k_{θ_3}	0
Radial distance of imbalance, r	8 in.	Damping coefficient along X-axis, c_X	49.3 lb-sec/ft †
Radial distance of balancing particle, r_j	8 in.	Damping coefficient along Y-axis, c_Y	49.3 lb-sec/ft †
Depth of the rotor, d	8 in.	Damping coefficient along Z-axis, c_Z	500 lb-sec/ft
Internal offset of the imbalance from rotor mid-plane, d_m	0 in.	Damping coefficient along X-axis, c_{θ_1}	59.3 lb-ft-sec/rad †
Internal offset of the race containing the balancing particles, $d_{1,2}$	0 in.	Damping coefficient along y_2-axis, c_{θ_2}	59.3 lb-ft-sec/rad †
Angular position of the imbalance from the x-axis, β_m	0°	Damping coefficient along z-axis, c_{θ_3}	0
		Viscous drag coefficient for the balancing particles in the race, μ_c	0.005 lb-sec/ft

† Damping values were selected to provide a viscous damping factor of 0.7.

It should be noted that the majority of the system's physical properties were selected to closely approximate a passenger vehicle wheel. The chosen dimensions for the rotor depth, and inner and outer radii make the polar moment of inertia for the disk about its spin axis greater than a transverse moment of inertia which classifies the rotor as a disk-type or short rotor.

Care must be taken when choosing an operating speed for the rotor to ensure that a resonant frequency is not selected. Based on the chosen values for the system properties, Table 5.2.2 summarizes the undamped natural frequencies resulting from the above properties. Please note that the table lists characteristics based solely on the wheel and imbalance mass with no consideration of any balancing particles. Table 5.2.3 lists the selected initial conditions for the simulations involving either one balancing particle or two balancing particles. For cases involving only one balancing particle, only the initial angular positions and velocities with a subscript 1 are utilized. Table 5.2.4 lists the range of Coulomb friction levels that will be explored for all numerical simulations. In each case, the value of the coefficient of kinetic

friction has been taken to be 75% of the value of the static friction. The rationale for this stems from the fact that kinetic friction values tend to be slightly lower than static friction values for most contacting surfaces.

Table 5.2.2: Undamped Natural Frequencies (Statically-Imbalanced Rotor)

Linear Spring Natural Frequencies	Torsional Spring Natural Frequencies
$\omega_{n_x} \simeq \sqrt{\frac{k_X}{M+m}} = 28.4$ rad/sec	$\omega_{n_{\theta_1}} \simeq \sqrt{\frac{k_{\theta_1}}{I_{xx}}} = 118.1$ rad/sec
$\omega_{n_y} \simeq \sqrt{\frac{k_Y}{M+m}} = 28.4$ rad/sec	$\omega_{n_{\theta_2}} \simeq \sqrt{\frac{k_{\theta_2}}{I_{yy}}} = 117.9$ rad/sec
$\omega_{n_z} \simeq \sqrt{\frac{k_Z}{M+m}} = 284$ rad/sec	$\omega_{n_{\theta_3}} \simeq \sqrt{\frac{k_{\theta_3}}{I_{zz}}} = 0$

Table 5.2.3: Initial Conditions with One or Two Balancing Particles (Statically-Imbalanced Rotor)

Rotor Position	Rotor Velocity	Rotor Orientation	Rotor Angular Velocity	Balancing Mass Angular Position	Balancing Mass Angular Velocity
$X_0 = 0$	$\dot{X}_0 = 0$	$\theta_{1_0} = 0$	$\dot{\theta}_{1_0} = 0$	$\beta_{1_0} = 30°$	$\dot{\beta}_{1_0} = 0.15$ rad/sec
$Y_0 = 0$	$\dot{Y}_0 = 0$	$\theta_{2_0} = 0$	$\dot{\theta}_{2_0} = 0$	$\beta_{2_0} = 272°$	$\dot{\beta}_{2_0} = 0.15$ rad/sec
$Z_0 = 0$	$\dot{Z}_0 = 0$	$\theta_{3_0} = 0$	$\dot{\theta}_{3_0} \left(= \omega_{3_0}\right)$ varies		

Table 5.2.4: Coulomb Friction Levels of Interest for Numerical Simulations

Coefficient of Static Friction, μ_s	Coefficient of Kinetic Friction, μ_k
0	0
0.001	0.00075
0.1	0.075

The friction values chosen give a good range of interface characteristics between the balancing particles and the race/rotor interface. The zero value was chosen to allow comparison with the majority of the literature which neglects dry friction effects. The lowest value of 0.001 approximates rolling resistance values commonly found in practice between steel surfaces, while the upper value of 0.1 approximates a sliding friction mode on the order of lubricated steel on steel [75].

Table 5.2.5 lists the range of operating speeds that will be explored for the selected support stiffnesses in the statically-imbalanced rotor. The operating speeds were chosen based

on the natural frequencies of the rotor listed in Table 5.2.2 and were staggered below and above each translational natural frequency with the exception of the highest natural frequency along the *Z*-axis. Rotational natural frequencies have no effect for a pure static imbalance since there is no wobble.

Table 5.2.5: Operating Speeds for Statically-Imbalanced Rotor

Initial *z*-axis Angular Velocity, rad/sec
15
50
200

5.2.1 Vertically-Oriented Rotor with One Balancing Particle

We will begin by examining a vertically-oriented rotor having a single balancing particle. We will examine the performance of the AB for a range of Coulomb friction levels as outlined in Table 5.2.4 and a range of operating speeds as outlined in Table 5.2.5. Figures 5.2.1.2.1 through 5.2.1.2.9, grouped at the end of section 5.2.1, show the behavior of the balancing particle and the displacement of the geometric center *P*. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. The a-part of the figure shows the behavior of the balancing particle while the b-part shows the magnitude of the displacement of the geometric center *P*, in inches.

Table 5.2.1.1 summarizes the behavior of the balancing particle at the various operating speeds and Coulomb friction levels by listing the final angular position of the particle relative to the body-fixed coordinate axes as well as the ultimate stopping time of the balancing particle. For cases that have no Coulomb friction where the particle never comes to rest relative to the cylinder, the ultimate stopping time is determined as the time when the particle reaches and stays within 1° of its final approximate angular position. Table 5.2.1.2 summarizes the behavior of the geometric center *P* for the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of *P* and the steady-state amplitude of the displacement of *P*. Table 5.2.1.3 shows the maximum and final amplitudes of the displacement of *P* in the absence of a balancer to establish a baseline dataset for evaluating the performance of the rotor and automatic balancer together, and Table 5.2.1.4 compares these baseline values to the values

obtained in Table 5.2.1.2 by examining the difference in the maximum and steady-state amplitudes of *P*.

Table 5.2.1.1: Behavior of the Balancing Particle (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

Initial *z*-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees	Ultimate Stopping Time, sec
15	0	-90.82	154.4†
	0.001	31.13	0.2947
	0.1	30.04	0.0589
50	0	-180.00	5.121†
	0.001	-151.49	3.2725†
	0.1	29.98	0.2355
200	0	180.00	3.7243†
	0.001	30.07	0.3027
	0.1	30.74	0.4889

† Indicates that $\pm$ 1° tolerance was used to determine the particle stop time.

Table 5.2.1.2: Behavior of the Geometric Center (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

Initial *z*-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of *P*, (10^{-2}) inches	Steady-State Displacement of *P*, (10^{-2}) inches
15	0	0.6744	0.4693
	0.001	0.6529	0.6518
	0.1	0.6539	0.6535
50	0	3.580	0.0000
	0.001	3.579	0.5882
	0.1	3.580	2.307
200	0	9.758	0.0000
	0.001	9.762	2.408
	0.1	9.762	2.404

Table 5.2.1.3: Rotor Performance in the Absence of an Automatic Balancer (Vertically-Oriented Statically Imbalanced Rotor)

Initial *z*-axis Angular Velocity, rad/sec	Maximum Displacement of *P*, (10^{-2}) inches	Steady-State Displacement of *P*, (10^{-2}) inches
15	0.3383	0.3379
50	1.853	1.194
200	5.052	1.248

Table 5.2.1.4: Comparison of Rotor Performance With and Without an Automatic Balancer (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Displacement of P, (10^{-2}) inches	Difference in Steady-State Displacement of P, (10^{-2}) inches
15	0	+0.3360	+0.1313
	0.001	+0.3146	+0.3138
	0.1	+0.3155	+0.3155
50	0	+1.727	-1.194
	0.001	+1.727	-0.6058
	0.1	+1.728	+1.113
200	0	+4.706	-1.248
	0.001	+4.710	+1.159
	0.1	+4.710	+1.155

A positive (+) sign in the above table indicates an increase (worsening) in the absolute displacement of P. A negative (-) sign indicates a decrease (improvement) in the absolute displacement of P.

5.2.1.1 Summary of Findings

For the cases where $\omega_3 = 15$ rad/sec, the balancer is unable to make any improvements to the rotor's performance. The steady-state amplitude of displacement for the rotor geometric center is increased for all friction levels and is largest when the coefficient of static friction is at its highest level of $\mu_s = 0.1$. The steady-state amplitude effectively doubles for this situation increasing by 93.4% from $0.3379(10^{-2})$ inches to $0.6535(10^{-2})$ inches. The maximum displacement of the geometric center increases for all cases, and is effectively doubled in all cases. The maximum displacement occurs when the Coulomb friction level is zero and increases by 199.35% from $0.3383(10^{-2})$ inches to $0.6744(10^{-2})$ inches. This makes sense because we have added a second 1-oz. imbalance to the rotor that has an initial position close to the existing 1-oz. imbalance.

We note that as the Coulomb friction level is increased, the particles come to rest relative to the rotor in shorter time periods. For the case of zero Coulomb friction, the particle moves for the duration of the simulation but reaches within 1° of its steady-state position in 154.4 seconds. As the coefficient of static friction is increased only slightly to $\mu_s = 0.001$, the particle is seen to come to stop within only 0.2947 seconds. This is a 99.98% reduction in stoppage time. Once the Coulomb friction level reaches 0.1, the particle stops within 0.0589 seconds which is an

80.01% reduction in stoppage time as compared with the intermediate Coulomb friction value of 0.001. This drastic reduction in stoppage time keeps the particle close to its initial angular position, which is on the heavy side of the rotor, and contributes to the performance degradation.

For the cases where $\omega_3 = 50$ rad/sec, the balancer is able to make improvements to the rotor's performance when the Coulomb friction level is kept low. For the case of $\mu_s = 0$, the particle reduces the steady-state amplitude of the rotor geometric center to zero, and accomplishes this in a relatively short time period of 6 seconds. For the case where $\mu_s = 0.001$, the balancer improves performance, but is unable to eliminate all residual displacement of P. The steady-state amplitude is reduced by 50.74% from $1.194(10^{-2})$ inches to $0.5882(10^{-2})$ inches. This is certainly an improvement, but the existence of a very small Coulomb friction level prohibits the balancer from working perfectly. We see that performance is worsened once the Coulomb friction level increases to $\mu_s = 0.1$, with the steady-state amplitude increasing by 93.22% from $1.194(10^{-2})$ inches to $2.307(10^{-2})$ inches. This happens because the particle is forced to come to rest relative to the rotor in a short period of time before it is able to reach its balanced position opposite the imbalance. A similar doubling in the maximum displacement of P was also observed for each simulation at this operating speed.

We also note that the stoppage times for these three simulations are much different than for the case of $\omega_3 = 15$ rad/sec. When the Coulomb friction level is set to zero, the particle reaches within 1° of its balanced position in only 5.121 seconds. This is 30 times shorter than the case when we operated below the translational natural frequency. Interestingly, when the Coulomb friction level was increased to 0.001 and 0.1, the stoppage time values were higher than in the case when the operating speed was 15 rad/sec. This seems counterintuitive since the higher operating speed will generate a higher centripetal acceleration, resulting in an increase in the normal force and a corresponding increase in the frictional force acting on the particle. It should be noted that the system being studied is highly nonlinear and there is more at work than apparent from a simple surface explanation. It is also worthy of mention that the particle did not actually stop during the $\omega_3 = 50$ rad/sec simulation when the friction level was set to $\mu_s = 0.001$.

For the cases where $\omega_3 = 200$ rad/sec, the balancer was only able to improve performance when the friction level was set to zero. For this case, the particle eliminated all vibration and reached its balanced position in 3.7243 seconds. Further, it reduced the steady-state amplitude of the geometric center to levels below the rotor performance in the absence of a balancer within

0.5 seconds, and effectively eliminated all vibrations within 3 seconds. When the Coulomb friction level was increased to 0.001 and 0.1, the particle was seen to come to rest relative to the rotor fairly quickly, precluding the possibility of reaching the balanced position. Again, only a modest increase in the Coulomb friction level renders the balancer completely ineffective at reducing unwanted vibrations. As before, the increase in steady-state amplitude is approximately doubled seeing its maximum effect when $\mu_s = 0.001$ increasing by 92.95% from $1.248(10^{-2})$ inches to $2.408(10^{-2})$ inches.

The stoppage times for the particle when $\mu_s = 0$ and $\mu_s = 0.001$ are, as expected, shorter than for the case when $\omega_3 = 200$ rad/sec. The case of zero Coulomb friction sees a reduction from 5.121 seconds to 3.7243 seconds to reach within 1° of the balanced position, and the case of rolling resistance type Coulomb friction sees a reduction from 3.2725 seconds to 0.3027 seconds. The interesting point comes when the Coulomb friction level is set the highest at 0.1. Here, we find an increase with the stoppage time going from 0.2355 seconds to 0.4889 seconds, an increase of 107.6%. Again, the system being studied is highly nonlinear, and it is not easy to state exactly how the particle is being affected at higher operating speeds.

5.2.1.2 Plots of Rotor Performance and Balancing Particle Behavior

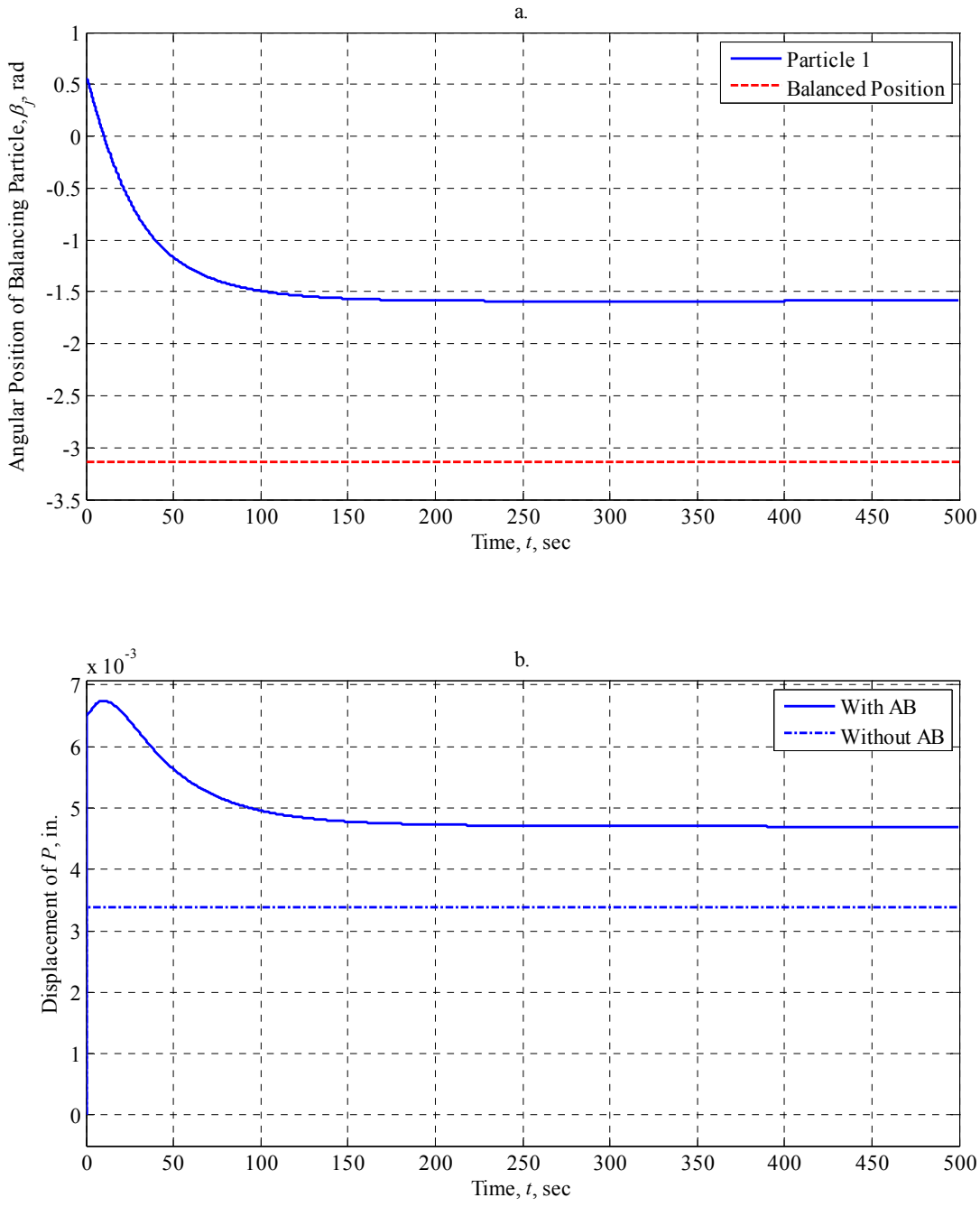

Figure 5.2.1.2.1: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

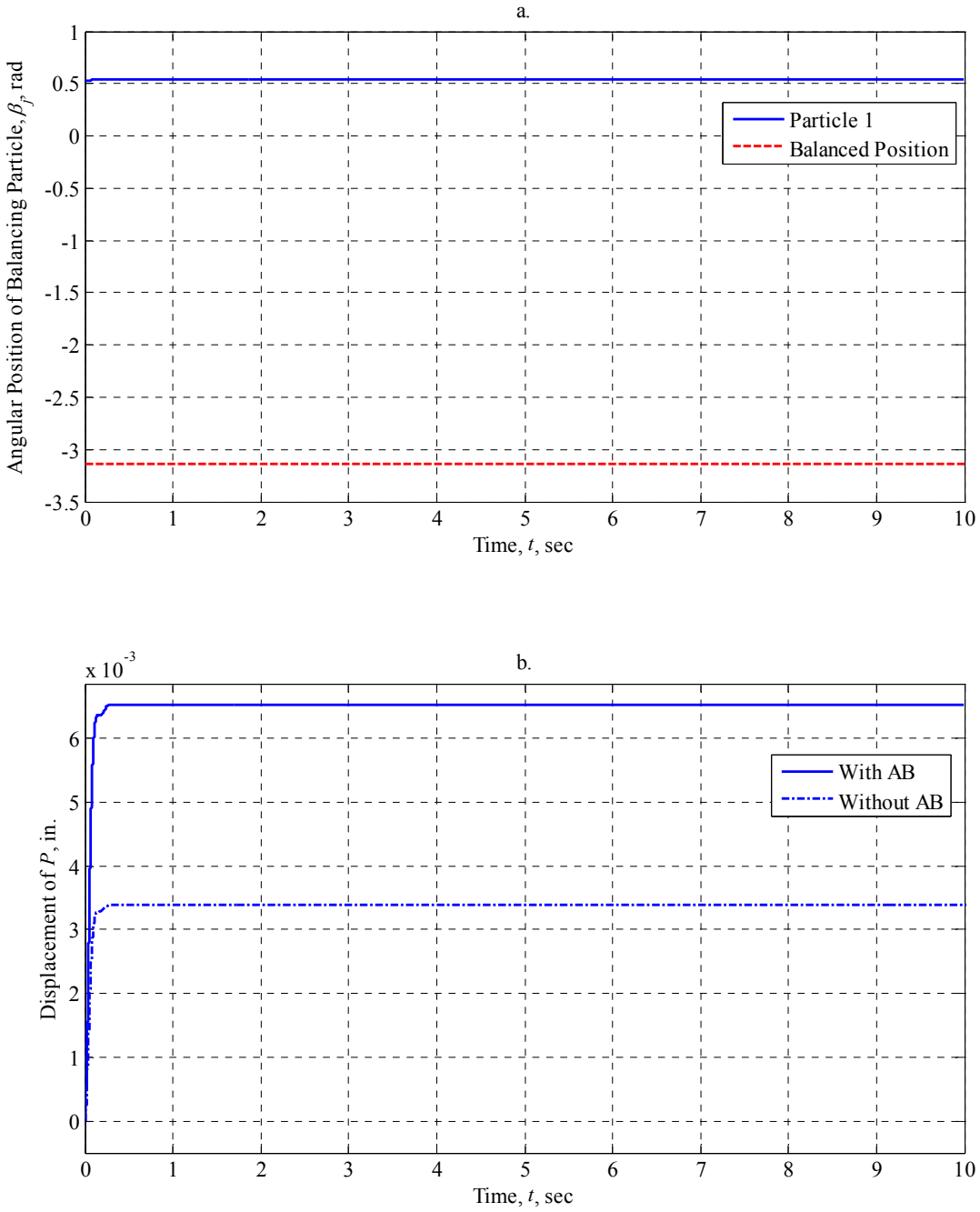

Figure 5.2.1.2.2: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

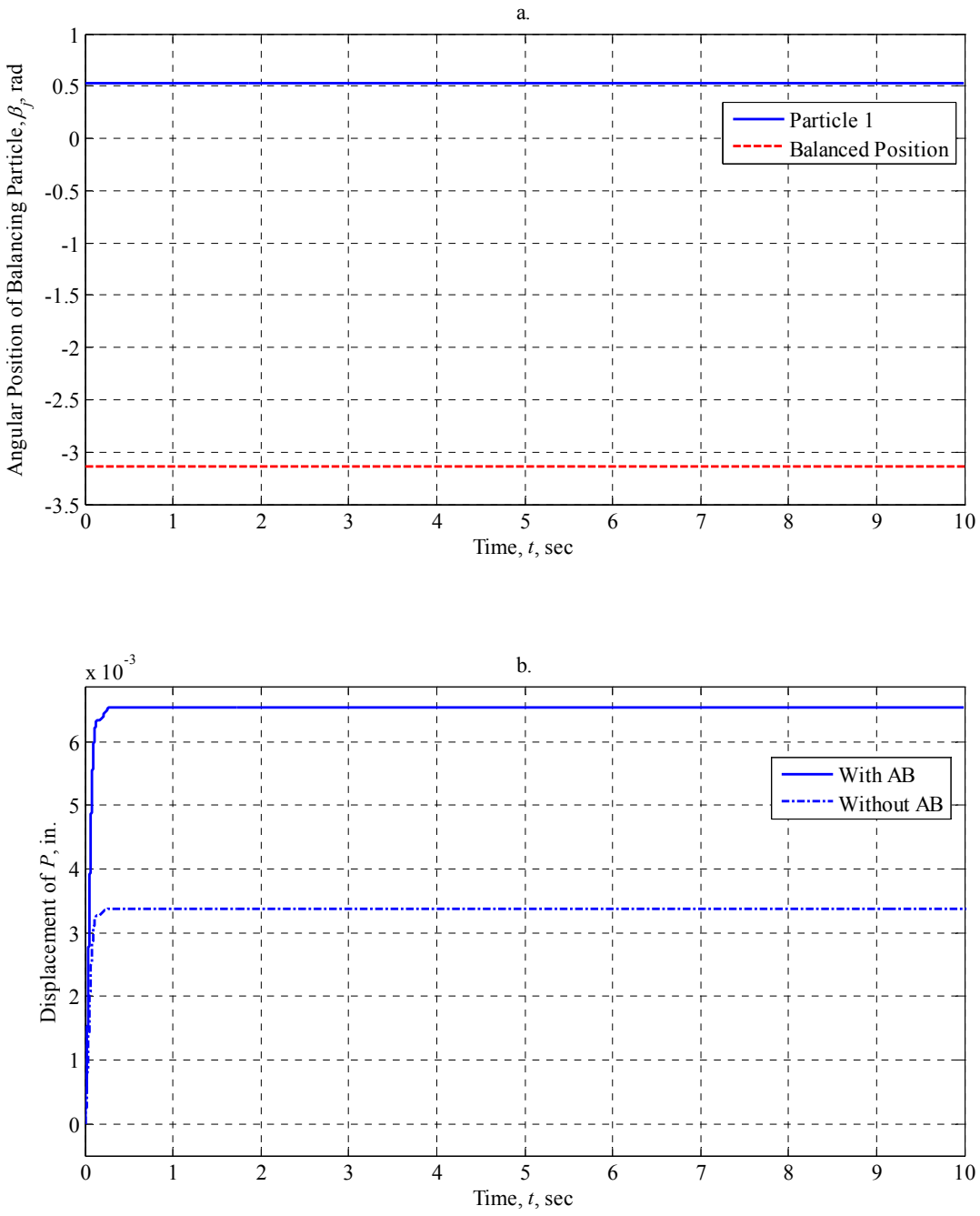

Figure 5.2.1.2.3: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

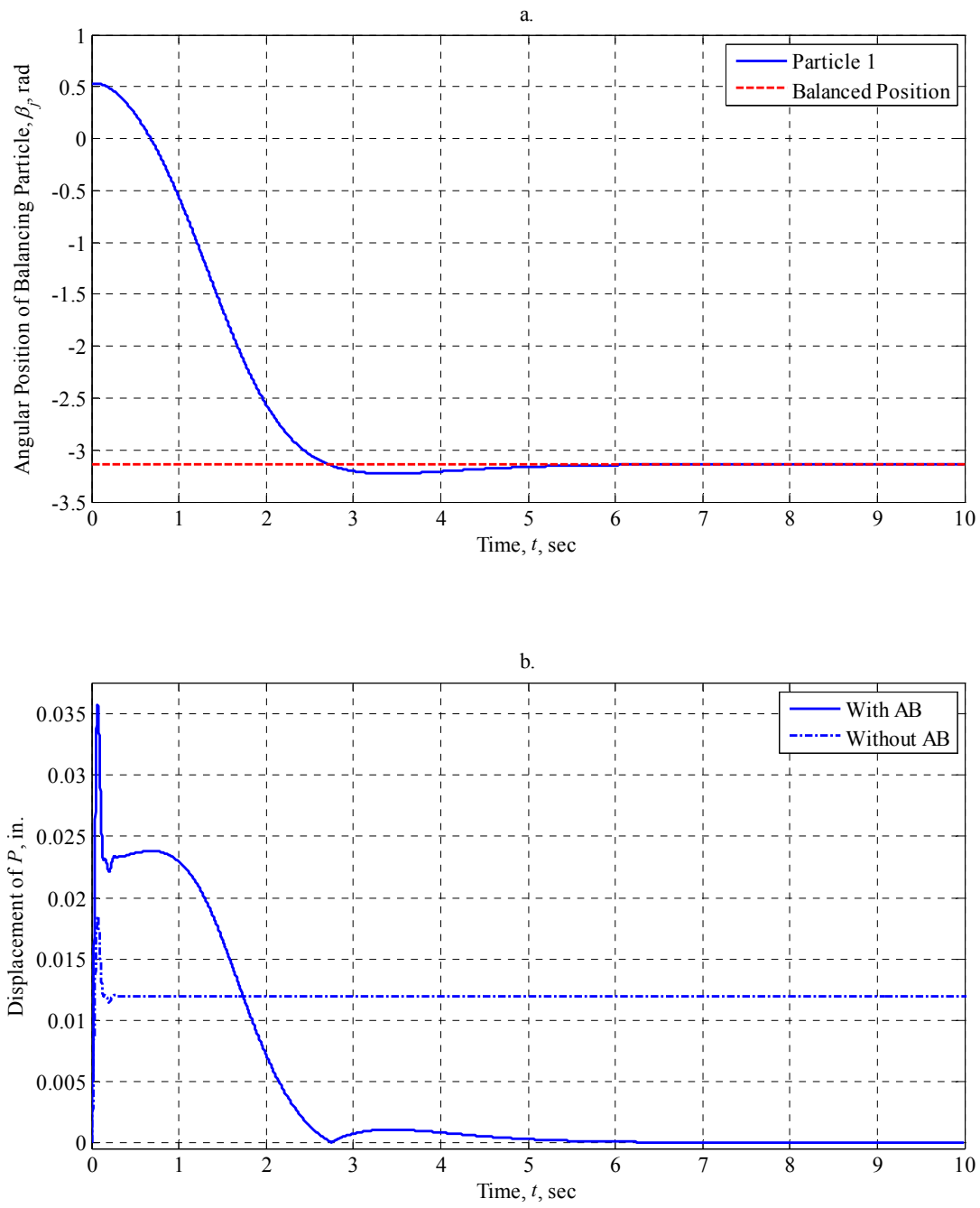

Figure 5.2.1.2.4: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

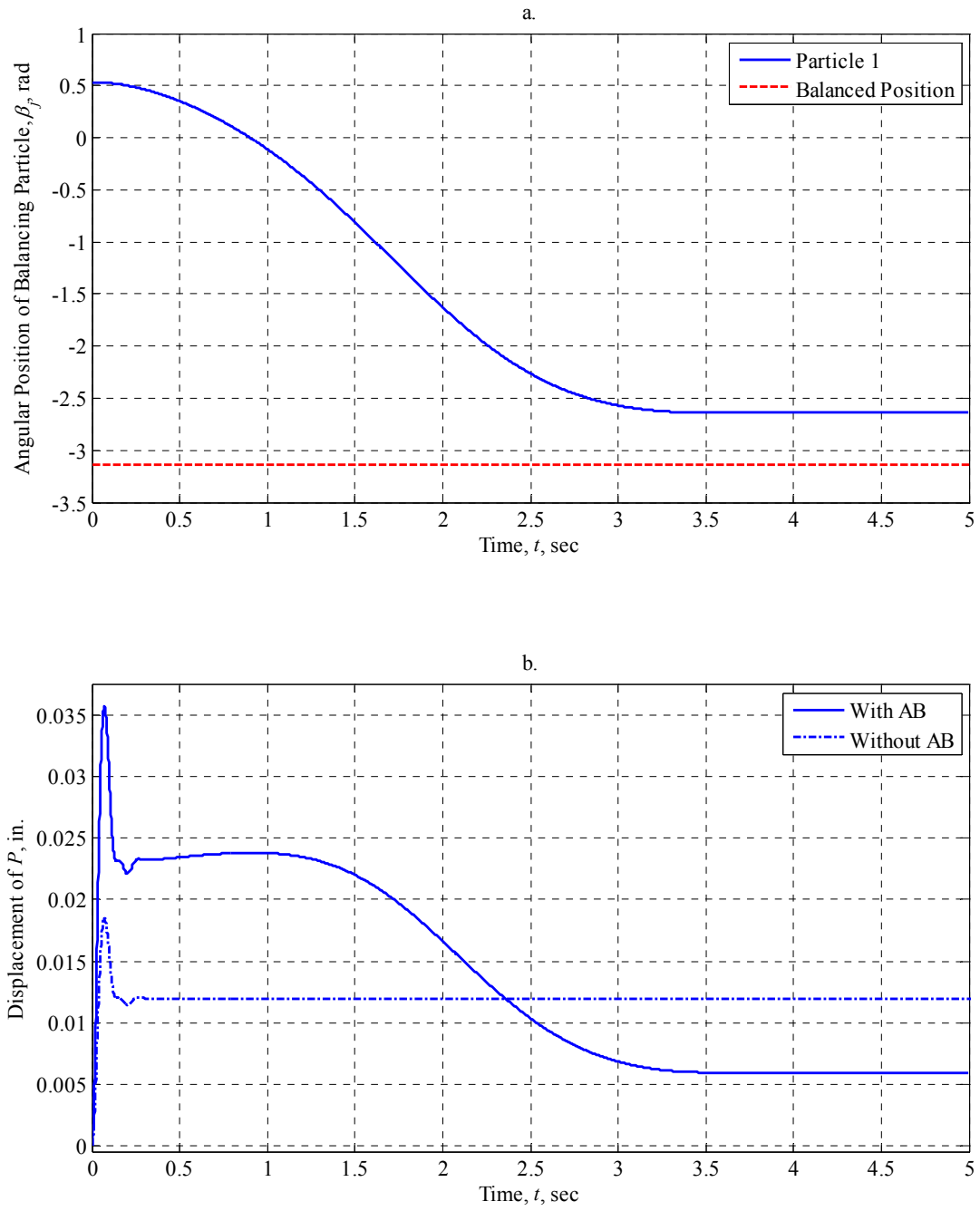

Figure 5.2.1.2.5: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

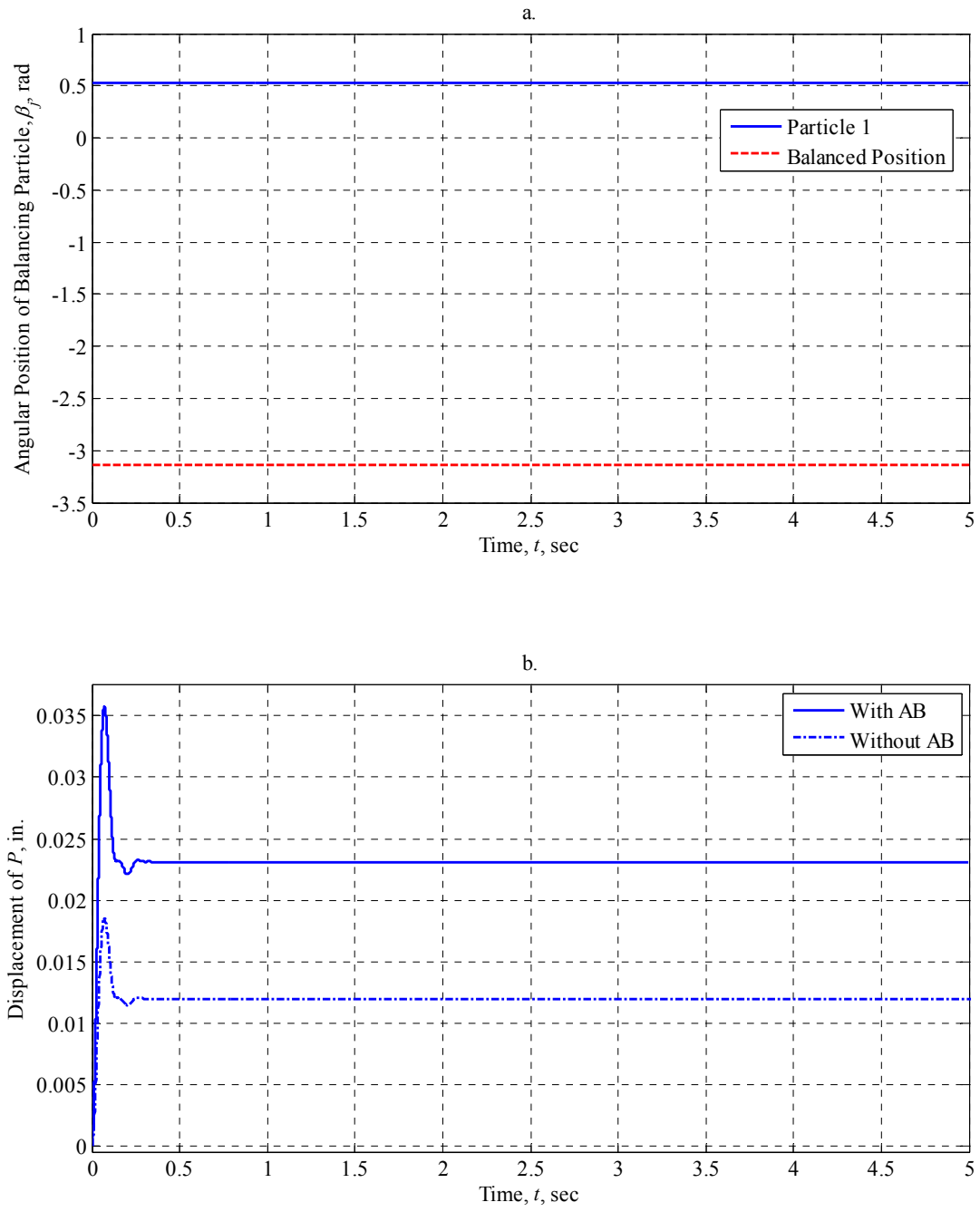

Figure 5.2.1.2.6: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

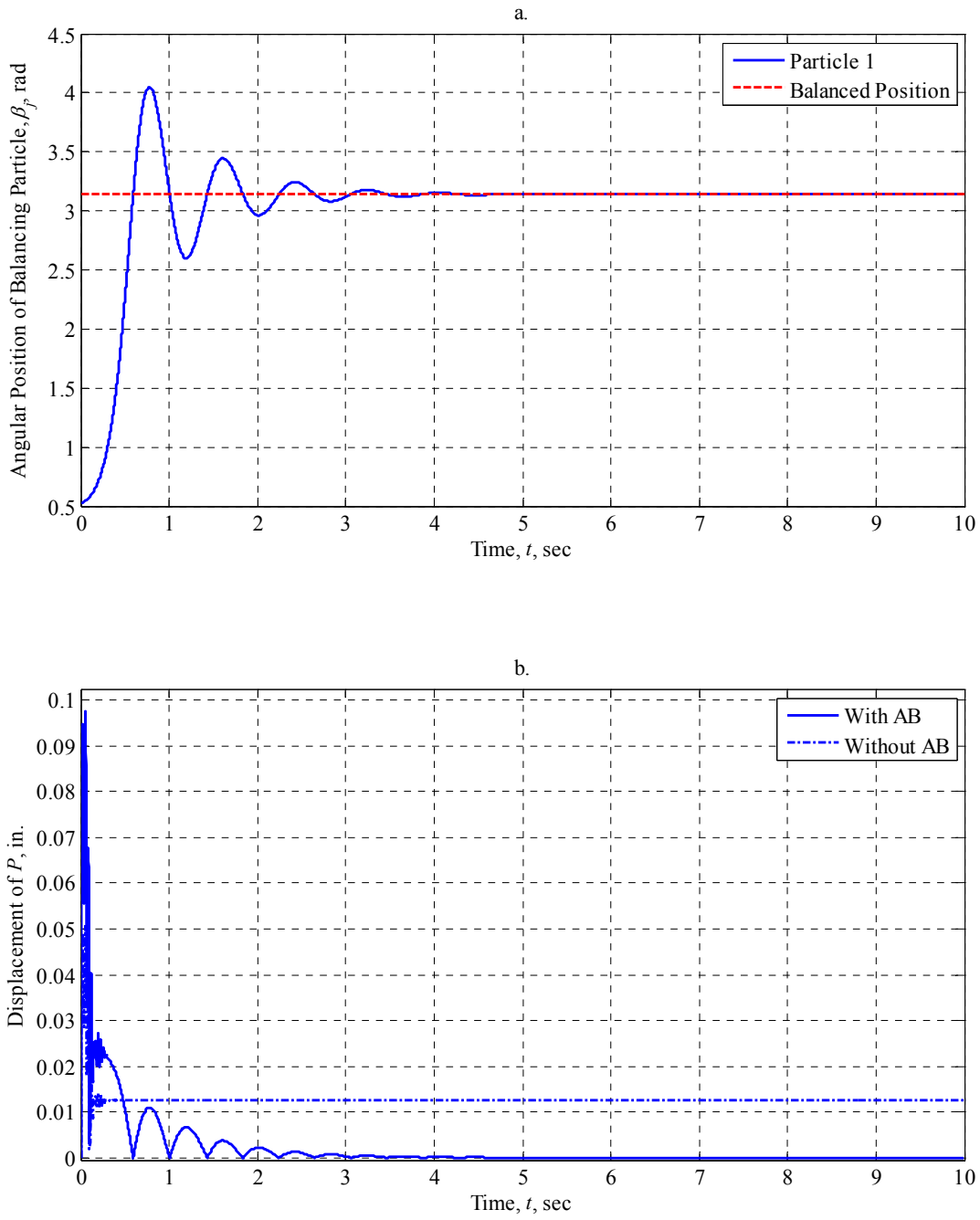

Figure 5.2.1.2.7: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

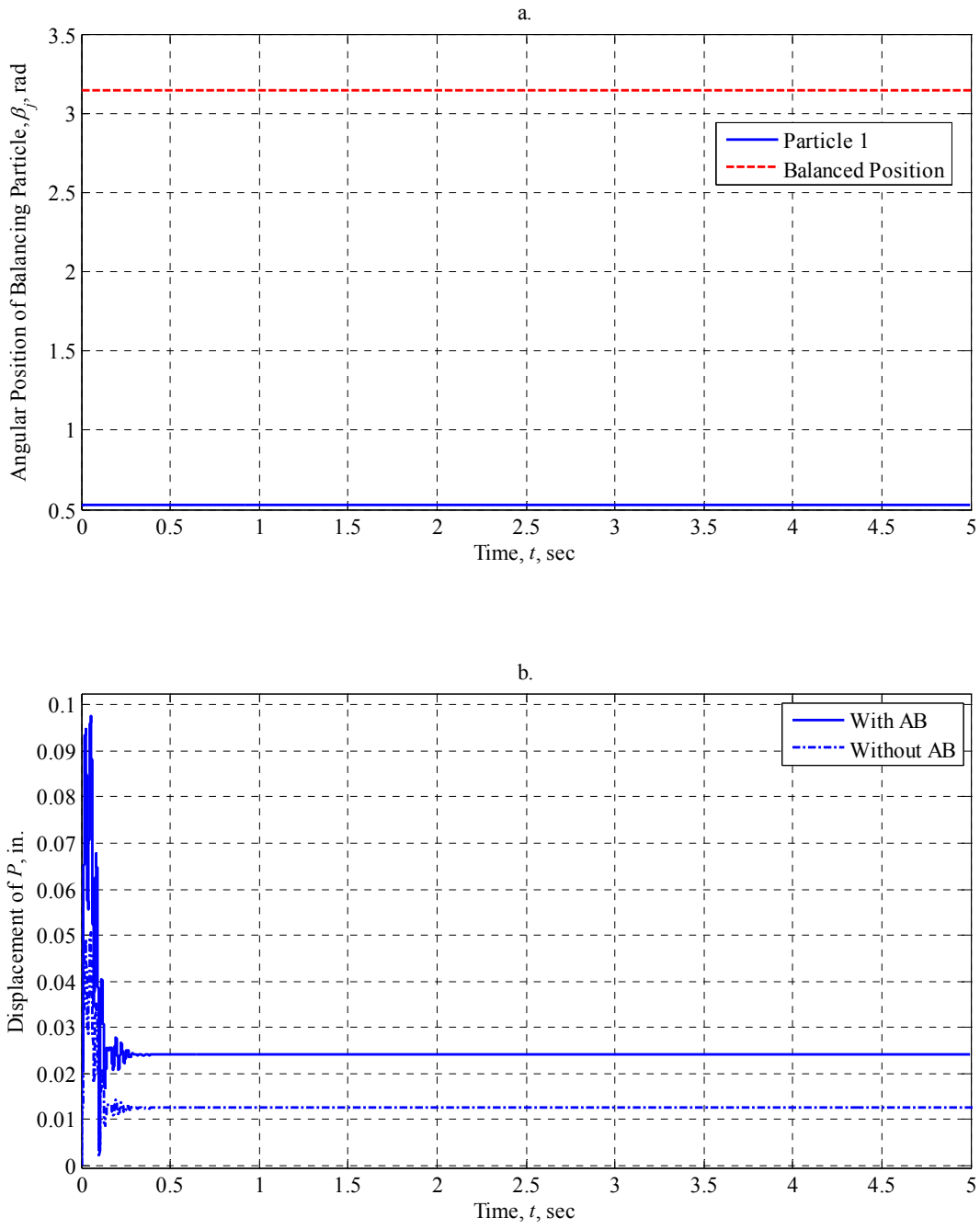

Figure 5.2.1.2.8: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

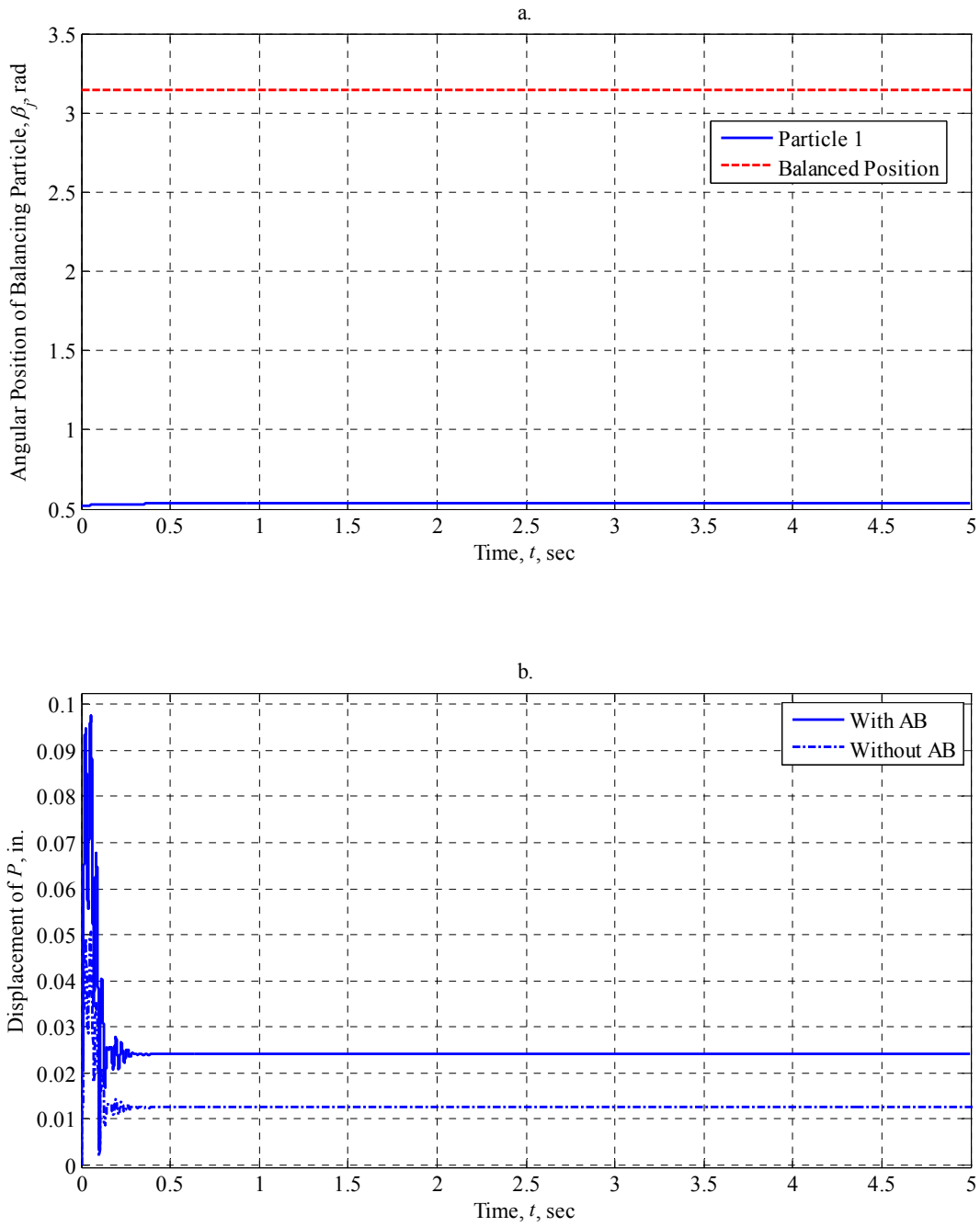

Figure 5.2.1.2.9: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

5.2.2 Vertically-Oriented Rotor with Two Balancing Particles

We now turn our attention to the case of balancing a vertically-oriented rotor using two balancing particles. We will again determine the performance of the AB for the range of Coulomb friction levels in Table 5.2.4 and the range of operating speeds in Table 5.2.5. The performance of the AB will again be evaluated on whether or not it is able to eliminate or reduce rotor vibration as compared to the unbalanced performance alone. We will also explore the final resting position of the balancing particle and its motion relative to the rotor. Figures 5.2.2.2.1 through 5.2.2.2.9, grouped at the end of section 5.2.2, show the behavior of the balancing particles and the displacement of the geometric center *P*. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. The a-part of the figure shows the behavior of the balancing particle, while the b-part shows the magnitude of the displacement of the geometric center *P*, in inches.

Table 5.2.2.1 summarizes the behavior of the balancing particles at the selected operating speeds and Coulomb friction levels by listing their final angular positions and ultimate stopping times. As before, if a particle never comes to rest relative to the cylinder during the time interval considered, the ultimate stopping time is determined as the time when the particle reaches and stays within 1° of its final approximate angular position. Table 5.2.2.2 summarizes the behavior of the geometric center *P* at the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of *P* and the steady-state amplitude of the displacement of *P*. Table 5.2.1.3 is repeated here as Table 5.2.2.3 since it illustrates the maximum and steady-state amplitudes of the displacement of *P* in the absence of a balancer previously established as a baseline dataset for evaluating the performance of the rotor and automatic balancer together. Table 5.2.2.4 compares these baseline values to the values obtained in Table 5.2.2.2 by examining the difference in the maximum and steady-state amplitudes of *P*.

Table 5.2.2.1: Behavior of the Balancing Particles (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees		Ultimate Stopping Time, sec	
		β_1	β_2	β_1	β_2
15	0	— ‡	— ‡	— ‡	— ‡
	0.001	-40.33	273.63	199.489	0.698
	0.1	29.79	272.04	2.956	0.212
50	0	-120.00	120.00	11.857†	13.180†
	0.001	30.46	229.58	0.623	43.475
	0.1	29.99	271.97	0.2119	0.3676
200	0	120.00	240.00	3.3358†	2.2905†
	0.001	64.72	204.43	0.3165	1.2112
	0.1	58.39	258.85	18.6214	8.1378

† Indicates that $\pm$ 1° tolerance was used to determine the particle stop time.

‡ Indicates that no steady-state value could be determined.

Table 5.2.2.2: Behavior of the Geometric Center (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0	1.002	— ‡
	0.001	0.8247	0.8103
	0.1	0.6664	0.6382
50	0	3.637	0.0000
	0.001	3.637	1.4766
	0.1	3.642	2.332
200	0	9.682	0.0000
	0.001	9.755	0.8867
	0.1	9.926	1.664

‡ Indicates that no steady-state value could be determined.

Table 5.2.2.3: Rotor Performance in the Absence of an Automatic Balancer (Vertically-Oriented Statically-Imbalanced Rotor)

Initial z-axis Angular Velocity, rad/sec	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0.3383	0.3379
50	1.853	1.194
200	5.052	1.248

Table 5.2.2.4: Comparison of Rotor Performance With and Without an Automatic Balancer (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Displacement of P, (10^{-2}) inches	Difference in Steady-State Displacement of P, (10^{-2}) inches
15	0	+0.6637	— ‡
	0.001	+0.4863	+0.4723
	0.1	+0.3280	+0.3003
50	0	+1.784	-1.194
	0.001	+1.785	+0.2826
	0.1	+1.789	+1.138
200	0	+4.630	-1.248
	0.001	+4.703	-0.3617
	0.1	+4.874	+0.4155

‡ Indicates that no steady-state value could be determined.

A positive (+) sign in the above table indicates an increase (worsening) in the absolute displacement of P. A negative (-) sign indicates a decrease (improvement) in the absolute displacement of P.

5.2.2.1 Summary of Findings

For the cases where ω_3 = 15 rad/sec, we see a totally different behavior than for a single balancing particle when the Coulomb friction level is set to zero. For this case, the two particles never reach a steady-state position but instead travel around the race together. This creates an oscillatory effect for the displacement of the geometric center and increases its vibration levels for the majority of the simulation as seen by Figure 5.2.2.2.1. There are periodic points, occurring approximately every 180 seconds, where the displacement decreases below the levels found in the absence of a balancer. It is impossible to pinpoint an exact time period because the distance between successive peaks on the displacement of P increases as the simulation proceeds indicating a change in frequency. This is not unexpected since viscous damping is present and running a simulation for long periods of time will introduce a reduction in the driving frequency of the rotor about the body-fixed z-axis. It is interesting to note that the maximum amplitude of displacement occurs when the Coulomb friction level is zero, showing a change of 196.2% from 0.3383(10^{-2}) inches to 1.002(10^{-2}) inches. Since the particles never come to rest, they are able to move to the heavy side of the rotor during their circuit creating a three-mass imbalance system that effectively triples the displacement of the geometric center.

Once the Coulomb friction level is increased to 0.001, the particles are seen to come to rest, albeit in very different amounts of time, and the rotor performance is worsened. The steady-state amplitude for the geometric center increases 139.8% from $0.3379(10^{-2})$ inches to $0.8103(10^{-2})$ inches. There is a similar increase when the Coulomb friction level is increased to 0.1, but it is not as large since the particle stop faster and are unable to travel as far around the periphery to a position closer to the imbalance.

When the Coulomb friction level is 0.001, Particle 1 is seen to stop after 199.5 seconds while Particle 2 stops after only 0.698 seconds. It is difficult to say exactly what accounts for this drastic difference in stoppage times, but it is again noted that the interplay of forces and the high nonlinearity in the system are difficult to grasp. There is a similar disparity for a Coulomb friction level of 0.1 where Particle 1 stops after 2.956 seconds and Particle 2 stops after only 0.212 seconds.

For the cases where $\omega_3 = 50$ rad/sec, the two particles are able to perfectly balance the rotor at zero Coulomb friction levels, but worsen its performance when the Coulomb friction level is slightly increased. At $\mu_s = 0$, Particle 1 gets within 1° of its steady state position within 11.857 seconds, while Particle 2 takes a similar 13.180 seconds. All displacement of the geometric center vanishes after approximately 14 seconds, or shortly after Particle 2 reaches its balanced position. Once the Coulomb friction level is increased to 0.001 and 0.1, the particles are unable to reach their balanced positions and worsen the rotor performance increasing the steady-state amplitude of P by 23.66% from $1.194(10^{-2})$ inches to $1.4766(10^{-2})$ inches for $\mu_s = 0.001$, and 95.28% from $1.194(10^{-2})$ inches to $2.332(10^{-2})$ inches for $\mu_s = 0.1$. In all cases, the maximum displacement of P is similar and has an increase of approximately 96.4% from $1.853(10^{-2})$ inches to an average value of $3.639(10^{-2})$ inches.

When the Coulomb friction level is set to $\mu_s = 0.001$, Particle 1 is seen to stop in only 0.623 seconds, but Particle 2 takes 43.475 seconds to come to a complete stop. Comparing that stop time with the upper portion of Figure 5.2.2.2.5, we see that Particle 2 has effectively stopped after approximately 1.25 seconds. The difference between the two numbers is likely attributed to slight numerical errors in computations within MatLab. The simulation was run at several different time step values to check consistency, and the absolute stopping time of 43.475 seconds was consistently found. Once the Coulomb friction level is increased to $\mu_s = 0.1$, both

particles are seen to stop quickly with 0.2119 seconds for Particle 1 and 0.3676 seconds for Particle 2.

For the cases where $\omega_3 = 200$ rad/sec, interestingly, the performance of the rotor is improved for both the zero Coulomb friction level and the rolling-resistance Coulomb friction level of 0.001. For $\mu_s = 0$, the steady-state amplitude of displacement for the geometric center is eliminated after approximately 5 seconds. Each particle is seen to dramatically overshoot its balanced position, only to quickly reverse course and reach equilibrium in a few seconds. This shows the robustness of the balancer and its ability to effectively balance the rotor in the absence of Coulomb friction. At this level, particles 1 and 2 are seen to reach within 1° of their balanced position after 3.3358 seconds and 2.2905 seconds, respectively.

Once the Coulomb friction level is increased to $\mu_s = 0.001$, the particles are stopped short of their balanced positions, but have moved close enough to reduce the steady-state amplitude of displacement for the geometric center by 29.0% from $1.248(10^{-2})$ inches to $0.8867(10^{-2})$ inches. It is possible that this increase in balancing performance from 50 rad/sec to 200 rad/sec allows more energy to be transferred to the balancing particles allowing them to get closer to their correct positions within the race. Once the Coulomb friction level is increased to $\mu_s = 0.1$, the particles stop well short of their balanced positions and worsen performance by increasing the steady-state amplitude of displacement by 33.3% from $1.248(10^{-2})$ inches to $1.664(10^{-2})$ inches. As seen at other operating speeds, the maximum displacement of the geometric approximately increases by a factor of 2 going from $5.052(10^{-2})$ inches to an average value of $9.788(10^{-2})$ inches, an increase of 93.7%.

It is interesting to note the stoppage times at the non-zero Coulomb friction levels. At $\mu_s = 0.001$, particles 1 and 2 come to rest after 0.3165 and 1.2112 seconds, respectively. At $\mu_s = 0.1$, particles 1 and 2 come to rest after 18.6214 and 8.1378 seconds, respectively. This is a similar phenomenon to what was observed for the case of a single particle, where for the high operating speed of 200 rad/sec, the stoppage time for the particle increased from 0.3027 to 0.4889 seconds even though the Coulomb friction increased by a factor of 100. The change for the system of two particles is much more pronounced showing an increase of 60-fold for Particle 1 and 7-fold for Particle 2, but the system is much more complicated having interplay between two balancing particles and the rotor. Again, with the complexity of the system involved, it is difficult to be absolutely certain why the stoppage times increase for the two particles.

5.2.2.2 Plots of Rotor Performance and Balancing Particle Behavior

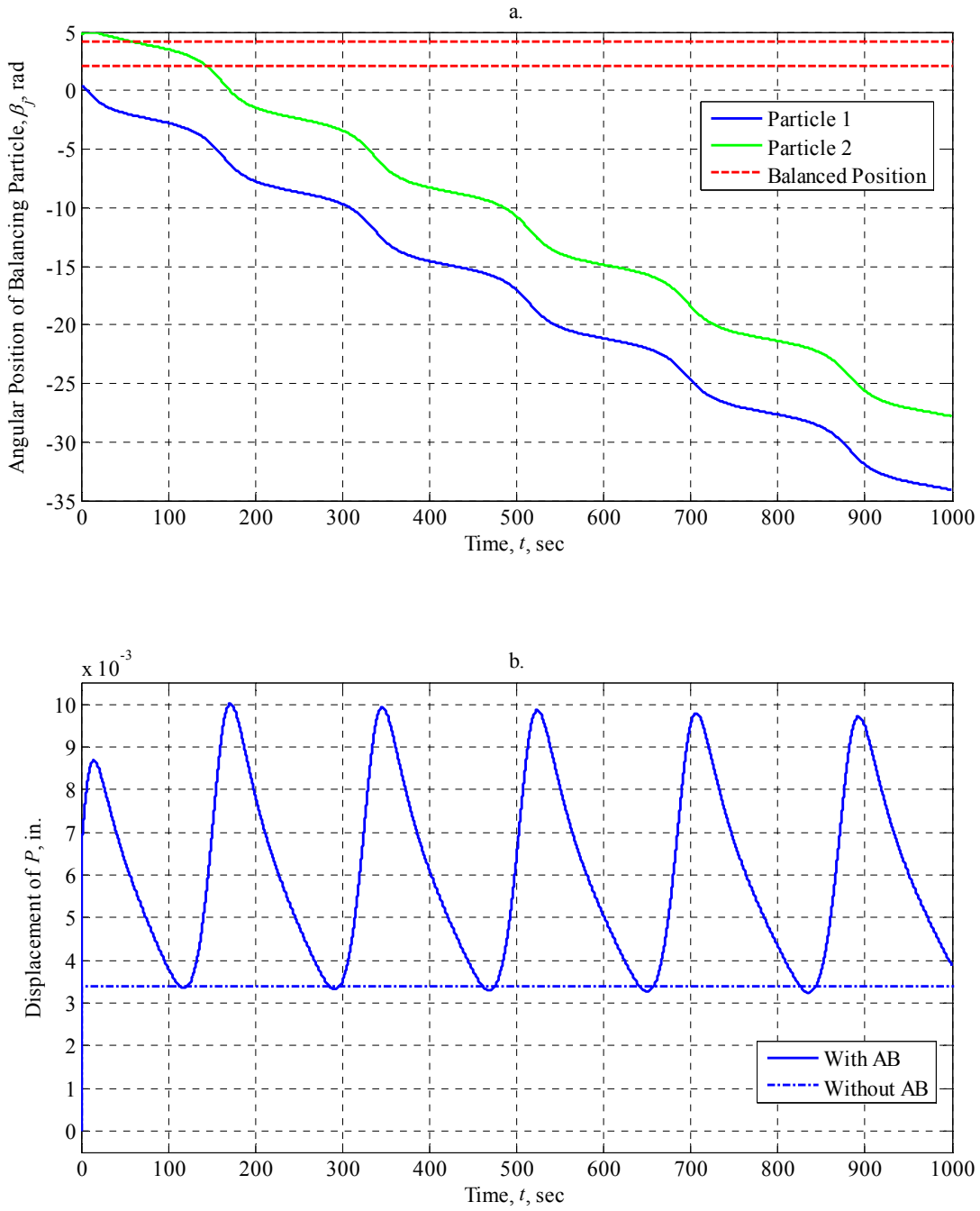

Figure 5.2.2.2.1: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

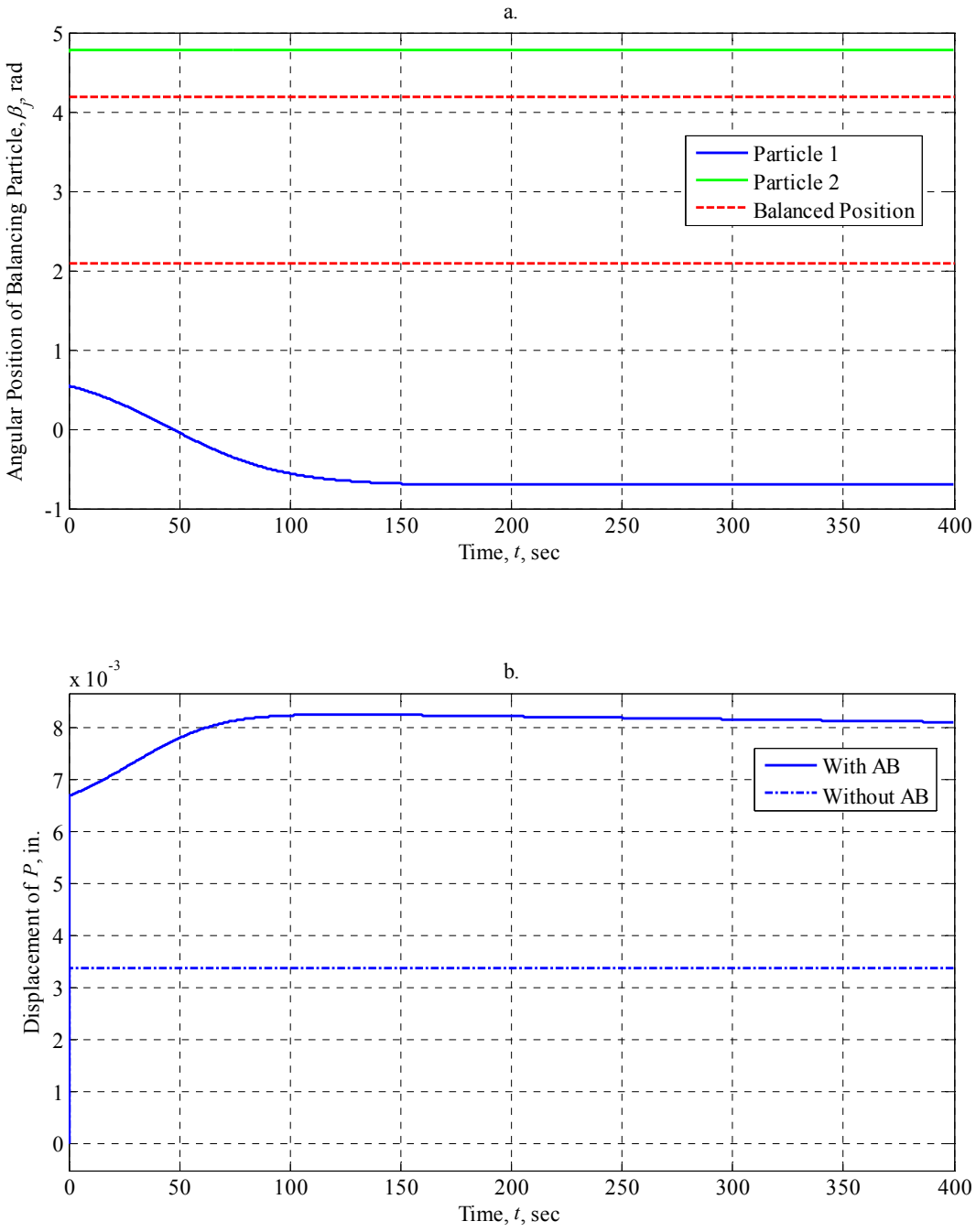

Figure 5.2.2.2.2: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

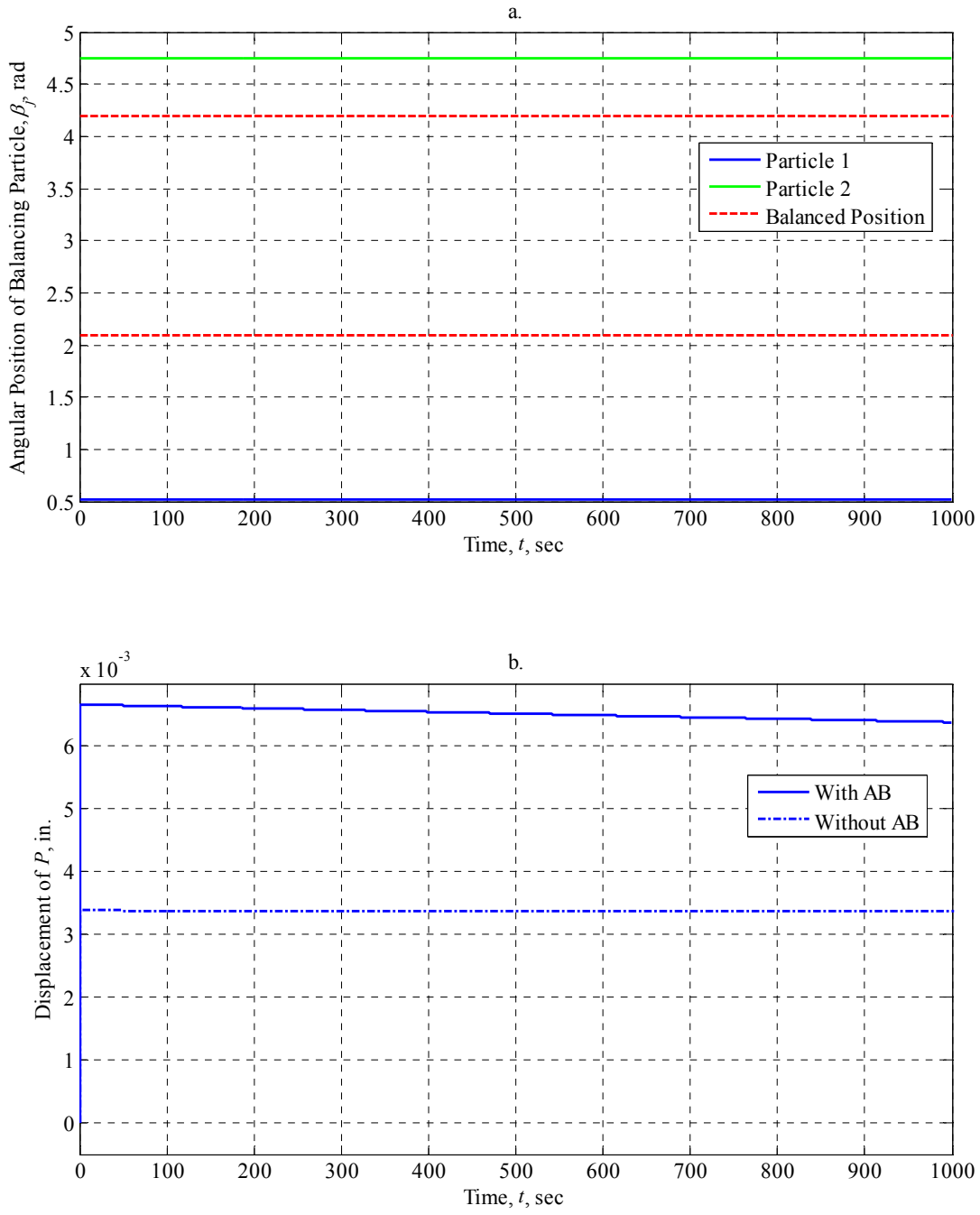

Figure 5.2.2.2.3: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

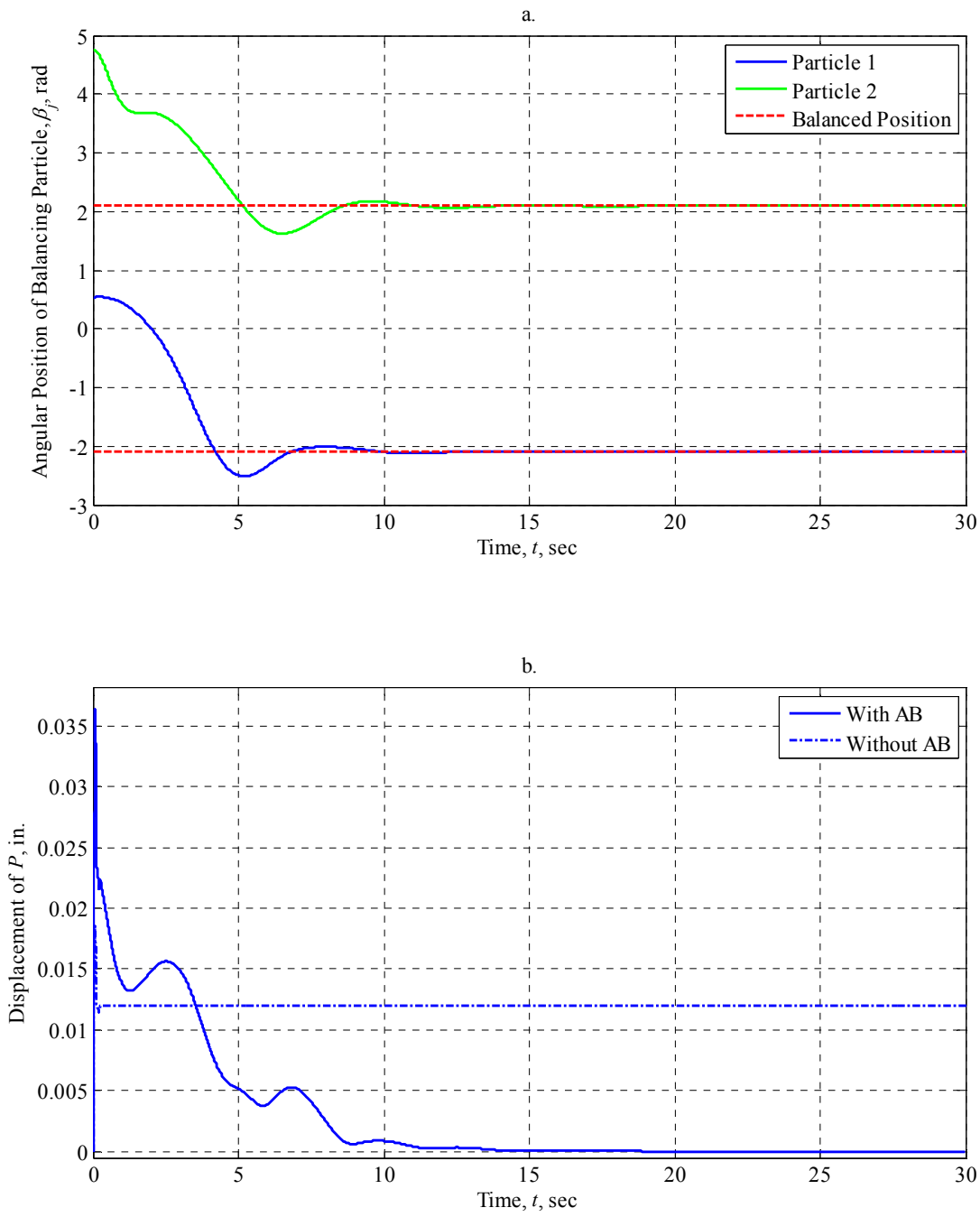

Figure 5.2.2.2.4: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

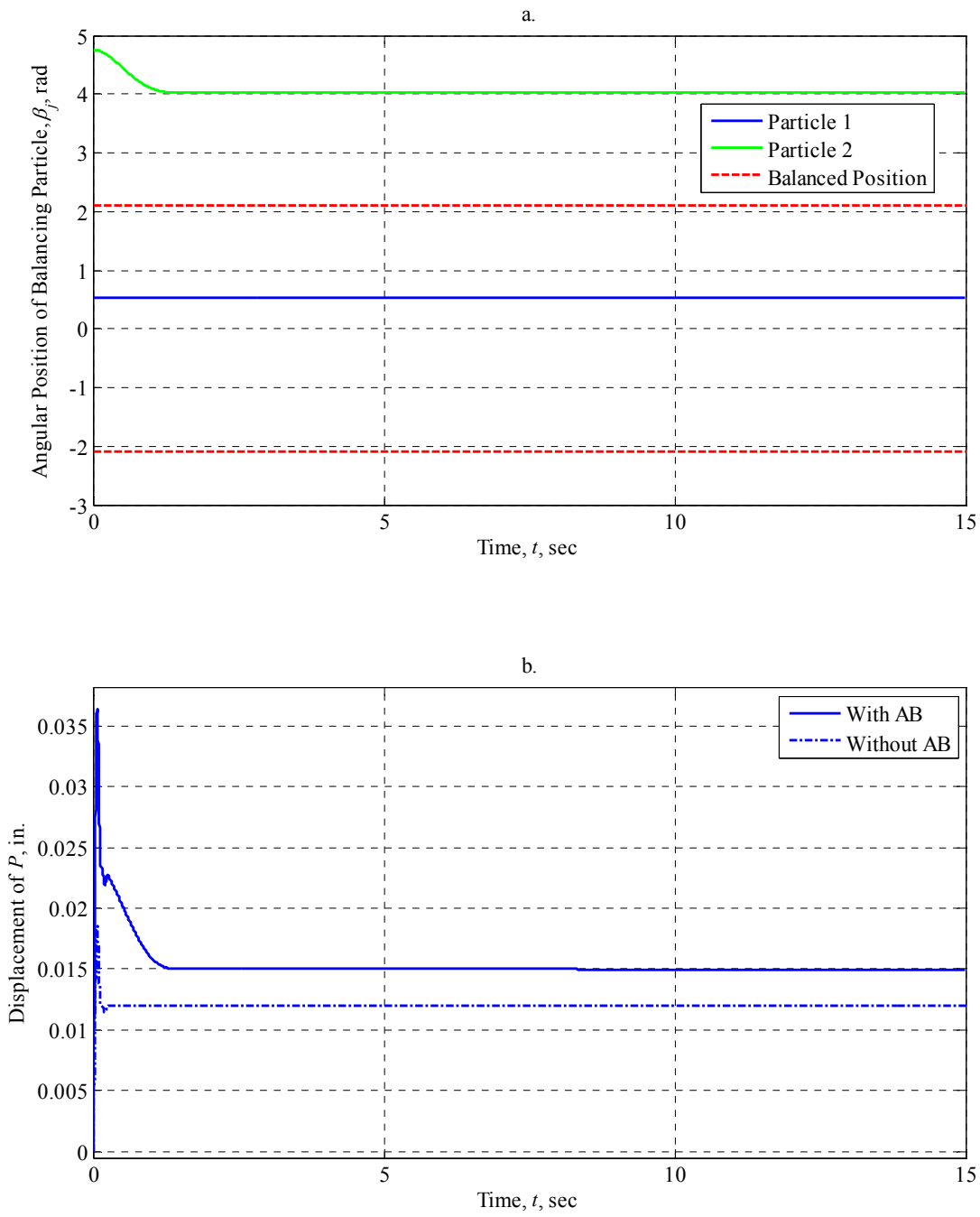

Figure 5.2.2.2.5: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

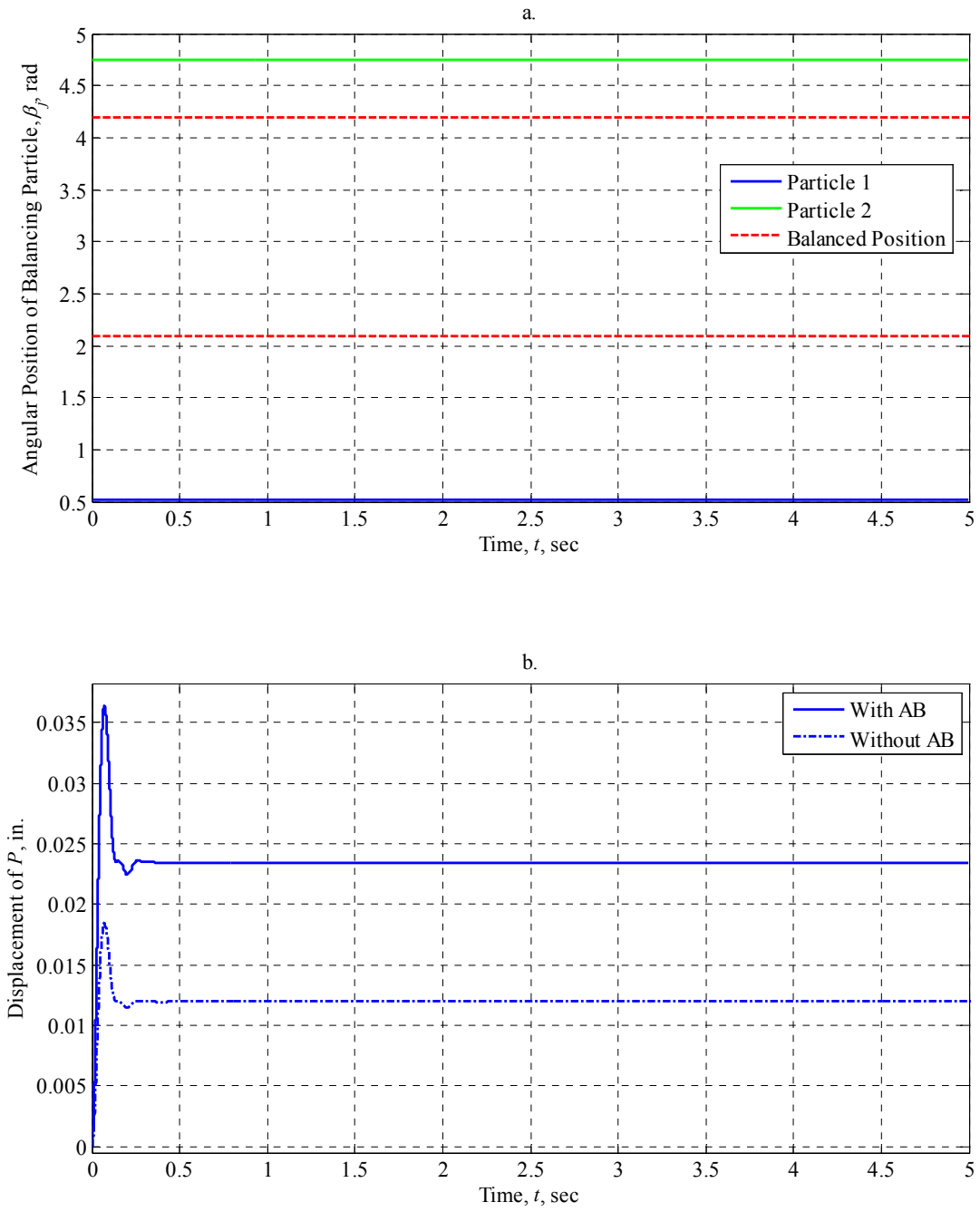

Figure 5.2.2.2.6: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

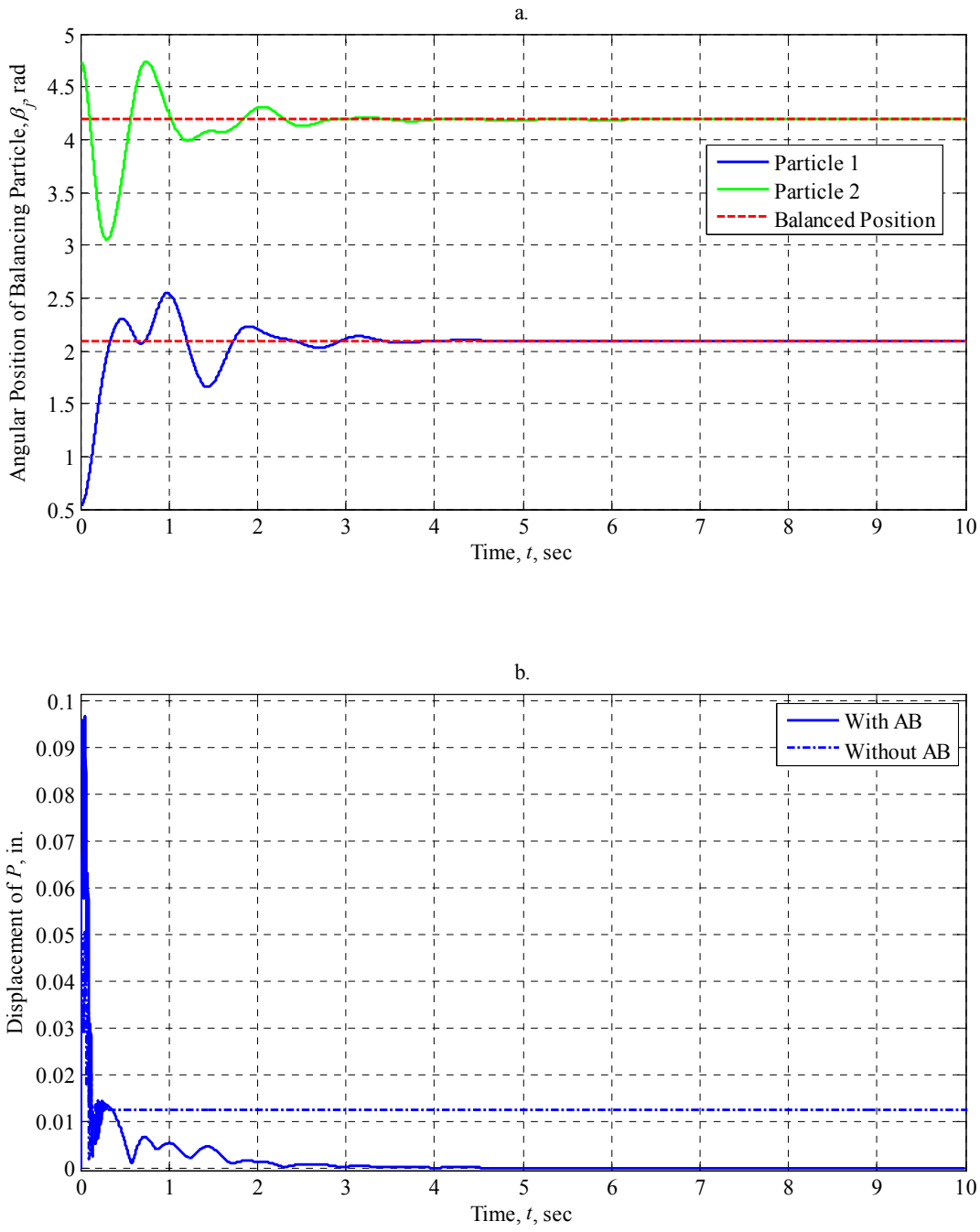

Figure 5.2.2.2.7: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

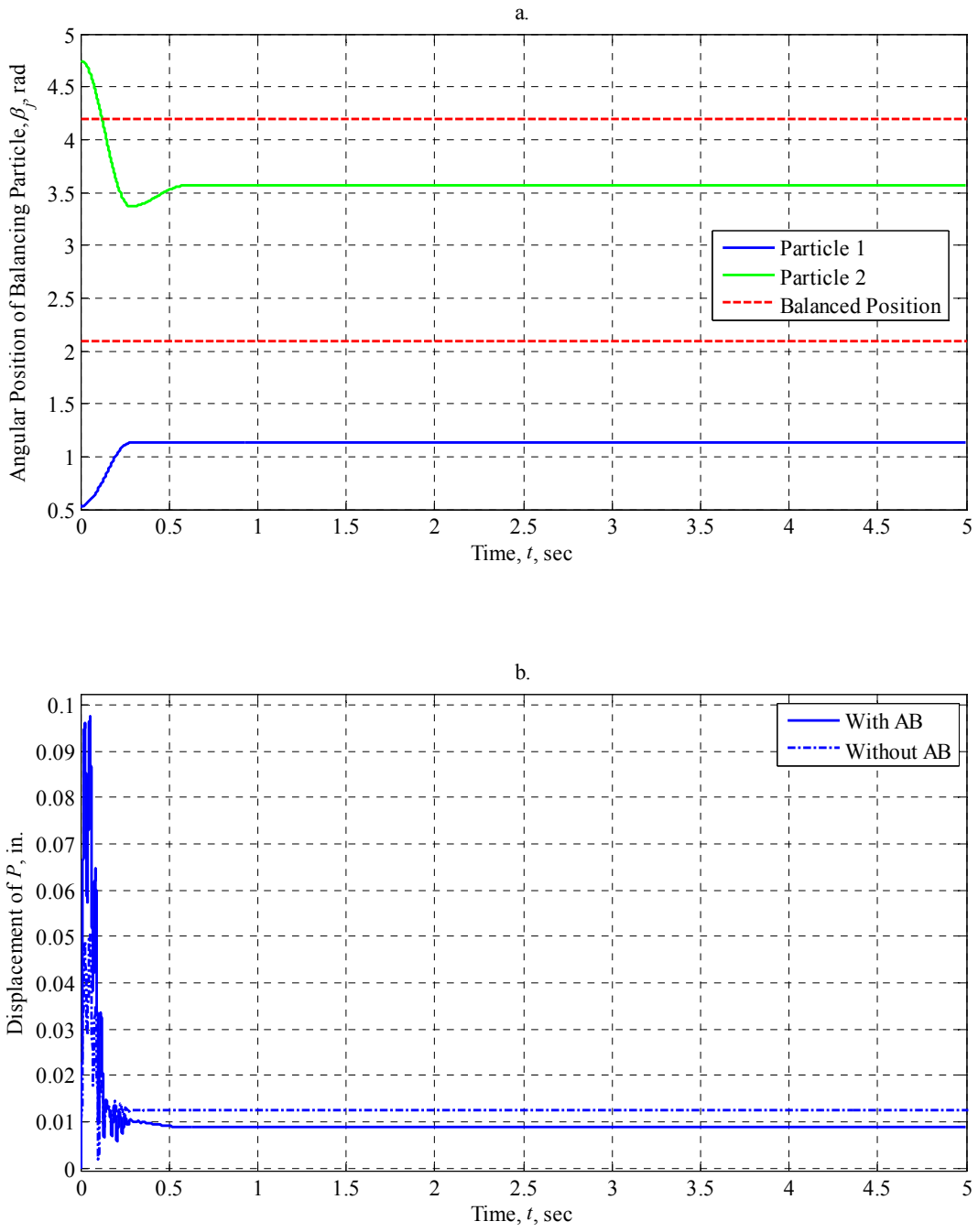

Figure 5.2.2.2.8: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

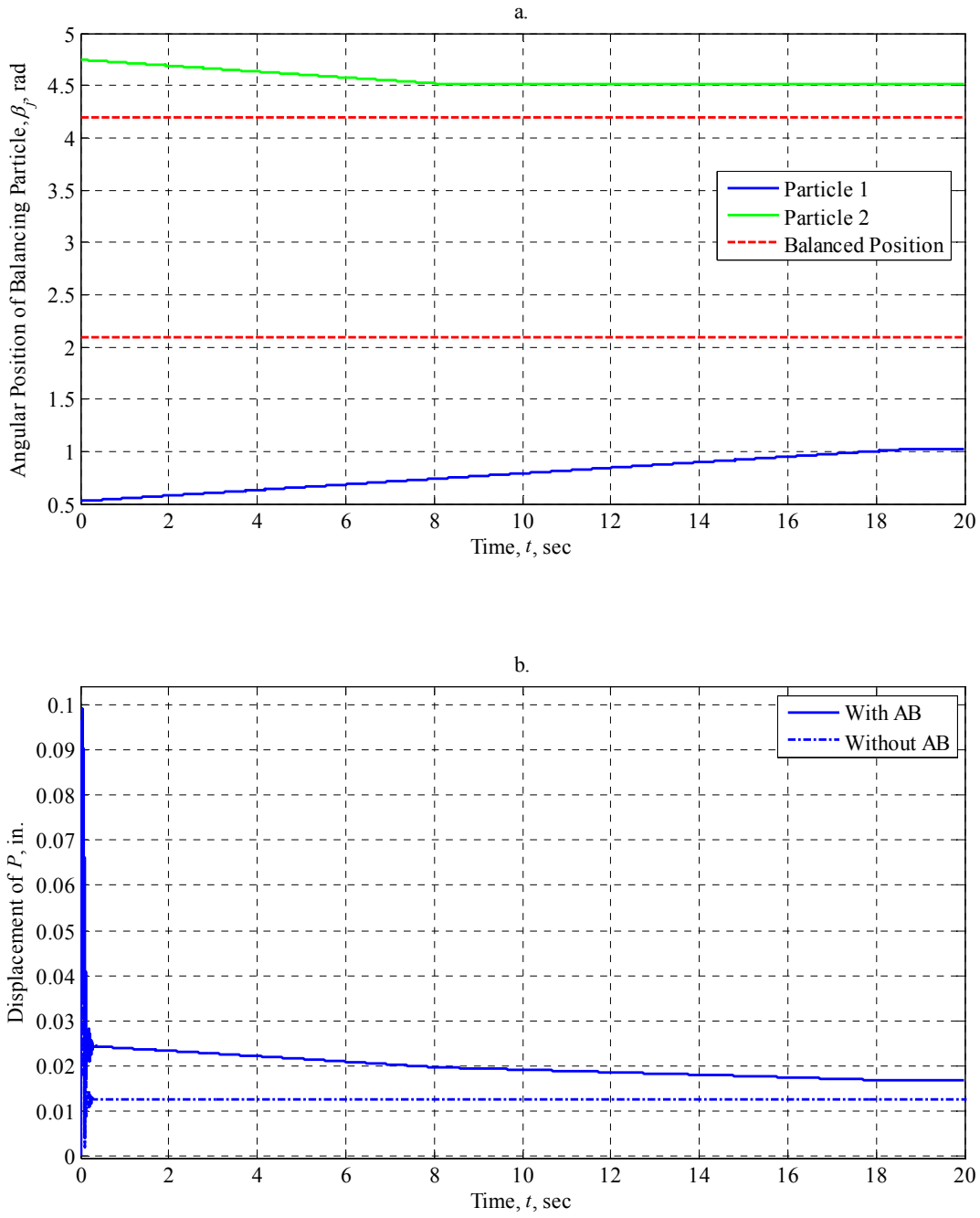

Figure 5.2.2.2.9: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

5.2.3 Horizontally-Oriented Rotor with One Balancing Particle

Now, we will examine the case a horizontally-oriented rotor having a single balancing particle. We will determine the performance of the AB for a range of Coulomb friction levels as previously outlined in Table 5.2.4 and a range of operating speeds as outlined in Table 5.2.5. The performance of the AB will be evaluated on whether or not it is able to eliminate or reduce rotor vibration as compared to the unbalanced rotor performance alone. We will also explore the final resting position of the balancing particle and its motion relative to the rotor. Figures 5.2.3.2.1 through 5.2.3.2.9, grouped at the end of Section 5.2.3, show the behavior of the balancing particle and the displacement of the geometric center *P*. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. As before, the a-part of the figure shows the behavior of the balancing particle while the b-part shows the magnitude of the displacement of the geometric center *P*, in inches.

Table 5.2.3.1 summarizes the behavior of the balancing particle for the selected operating speeds and Coulomb friction levels by listing the final angular position and ultimate stopping time of the balancing particle. Table 5.2.3.2 summarizes the behavior of the geometric center *P* for the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of *P* and the steady-state amplitude of the displacement of *P*. Table 5.2.3.3 shows the maximum and steady-state amplitudes of the displacement of *P* in the absence of a balancer to establish a baseline dataset for evaluating the performance of the rotor and automatic balancer together. Table 5.2.3.4 compares these baseline values to the values obtained in Table 5.2.3.2 by examining the difference in the maximum and steady-state amplitudes of *P*.

If no steady-state amplitude is obtained because *P* exhibits oscillatory behavior, then an average steady-state amplitude will be considered as the final steady-state amplitude. A similar scenario will be utilized for the final angular position of the balancing particle since it can also exhibit oscillatory behavior. In these instances, the settling time for the particle will be defined as the time when the particle first enters an envelope of 1° about its steady-state value.

Table 5.2.3.1: Behavior of the Balancing Particle (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees	Ultimate Stopping Time, sec
15	0	—‡	—‡
	0.001	—‡	—‡
	0.1	—‡	—‡
50	0	-180.59*	5.3748†
	0.001	-182.70*	7.0779†
	0.1	29.99	0.987
200	0	180.00*	3.7006†
	0.001	179.98*	10.5895†
	0.1	32.16	45.2415

† Indicates that $\pm$ 1° tolerance was used to determine the particle stop time.
‡ Indicates that no steady-state value could be determined.
* Indicates that the steady-state value was determined using an average value.

Table 5.2.3.2: Behavior of the Geometric Center (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0	0.7667	—‡
	0.001	0.7658	—‡
	0.1	0.7123	—‡
50	0	3.533	0.0012*
	0.001	3.532	0.0035*
	0.1	3.521	2.230*
200	0	9.691	0.000*
	0.001	9.696	0.0001*
	0.1	9.697	2.320*

‡ Indicates that no steady-state value could be determined.
* Indicates that the steady-state value was determined using an average displacement.

Table 5.2.3.3: Rotor Performance in the Absence of an Automatic Balancer (Horizontally-Oriented Statically-Imbalanced Rotor)

Initial z-axis Angular Velocity, rad/sec	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0.3383	0.3379
50	1.853	1.194
200	5.052	1.248

Table 5.2.3.4: Comparison of Rotor Performance With and Without an Automatic Balancer (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Displacement of P, (10^{-2}) inches	Difference in Steady-State Displacement of P, (10^{-2}) inches
15	0	+0.4283	— ‡
	0.001	+0.4274	— ‡
	0.1	+0.3739	— ‡
50	0	+1.680	-1.193*
	0.001	+1.679	-1.191*
	0.1	+1.668	+1.036*
200	0	+4.639	-1.248*
	0.001	+4.644	-1.248*
	0.1	+4.645	+1.072*

‡ Indicates that no steady-state value could be determined.
* Indicates that an average steady-state value was used.

A positive (+) sign in the above table indicates an increase (worsening) in the absolute displacement of P. A negative (-) sign indicates a decrease (improvement) in the absolute displacement of P. It should be noted that the difference in steady-state values of the displacement of P are calculated from the new static equilibrium point of the rotor caused by the addition of a 1-oz. balancing particle. For a spring stiffness of 1000 lb/ft and a 1-oz. balancing particle, this additional deflection is $7.5(10^{-4})$ in.

5.2.3.1 Summary of Findings

For the cases where $\omega_3 = 15$ rad/sec, we observe that there is no steady-state position for the balancing particle regardless of the Coulomb friction level. The particle is seen to exhibit an oscillatory motion while it continues to travel around the race. Further, there is no steady-state displacement amplitude for the geometric center. The geometric center is seen to exhibit periodic behavior where it spends a small portion of time below the amplitude level of the rotor without a balancer, but the majority of its time is spent at amplitude levels greater than this level. It is very interesting to note that as the Coulomb friction level is increased, the frequency of periodic motions for the geometric center increases. For $\mu_s = 0$, the period is approximately 16.6 seconds. For $\mu_s = 0.001$, the period shortens to 15.7 seconds. Finally, when $\mu_s = 0.1$, the period shortens to 6.43 seconds. It is unclear why this phenomenon occurs, but a similar phenomenon occurred at intermediate friction levels not indicated in the paper.

The maximum displacement of the geometric center increased in all three cases by an average of 121% going from $0.3383(10^{-2})$ inches to an average of $0.7483(10^{-2})$ inches. The smallest increase in maximum displacement occurred at the highest Coulomb friction level of $\mu_s = 0.1$ resulting in an overall increase of 111% from $0.3383(10^{-2})$ inches to $0.7123(10^{-2})$ inches. The oscillatory motion of the particle stems from a gravitational force which now acts in the plane of the rotor. This force was absent in the previous simulations and overwhelms the small levels of Coulomb friction present between the balancing particle and the race resulting in the particle never coming to rest relative to the rotor.

For the cases where $\omega_3 = 50$ rad/sec, we get results similar to the case where gravity was absent, but the behavior is slightly different. For zero Coulomb friction levels, the particle is able to reach a steady-state oscillation around its balanced position after 5.3748 seconds which results in a very small average steady-state amplitude of displacement for the geometric center of $0.0012(10^{-2})$ inches, a reduction of effectively 100%. When the Coulomb friction level is increased to $\mu_s = 0.001$, the particle reaches a steady-state oscillation about its balanced position after 7.0779 seconds resulting in an average amplitude of steady-state displacement of the geometric center of $0.0035(10^{-2})$ inches, a reduction of 99.7%. This is a dramatic improvement in performance over the case of a single particle in the absence of gravity and highlights the effect of gravity overwhelming the friction force. The performance of the rotor is worsened once the Coulomb friction level increases to $\mu_s = 0.1$. The particle comes to rest after only 0.987 seconds, resulting in an average steady-state amplitude of $2.230(10^{-2})$ inches which is an increase of 86.8%.

The maximum displacement increases as usual for all three cases by an average of 90.4% going from $1.853(10^{-2})$ inches to an average of $3.529(10^{-2})$ inches. Again, this is no surprise since we have effectively doubled the imbalance present in the rotor with the addition of a balancing particle.

For the cases where $\omega_3 = 200$ rad/sec, the rotor shows improved performance for Coulomb friction levels of $\mu_s = 0$ and 0.001. In both cases, the displacement of the geometric center P is effectively eliminated. This is a significant improvement in the behavior of a single particle in the absence of gravity where the rotor performance was negatively impacted at a Coulomb friction level of $\mu_s = 0.001$ and the balancing particle was unable to reach its correct position. In both cases here, the balancing particle never came to a complete stop relative to the

rotor, and its settling times to within a 1° band around its steady-state angular position are 3.7006 seconds for $\mu_s = 0$ and 10.5895 seconds for $\mu_s = 0.001$. While these times are on the same order of magnitude with one another, it is interesting to note that the Coulomb friction level of $\mu_s = 0.001$ results in a settling time that is 2.86 times longer than the case of no Coulomb friction.

Once the Coulomb friction level reaches a value of $\mu_s = 0.1$, the rotor performance is negatively impacted resulting in an average steady-state amplitude of $2.320(10^{-2})$ inches, an increase of 85.9% from $1.248(10^{-2})$ inches. Additionally, the particle is seen to come to a complete stop after 45.2415 seconds. This time is dramatically longer than the 0.987 seconds required to stop the particle at the same Coulomb friction level at an operating speed of 50 rad/sec, and also longer than the required time periods for the other 200 rad/sec simulations. Figure 5.2.3.2.9 shows that the majority of the particle's angular displacement has been completed after approximately 0.35 seconds, and it is possible the remaining time seen in Table 5.2.3.1 is attributed to numerical errors within MatLab. As before, the maximum displacement of the geometric center increases for all three cases. The increase is approximately 91.9% going from $5.052(10^{-2})$ inches to an average of $9.694(10^{-2})$ inches.

5.2.3.2 Plots of Rotor Performance and Balancing Particle Behavior

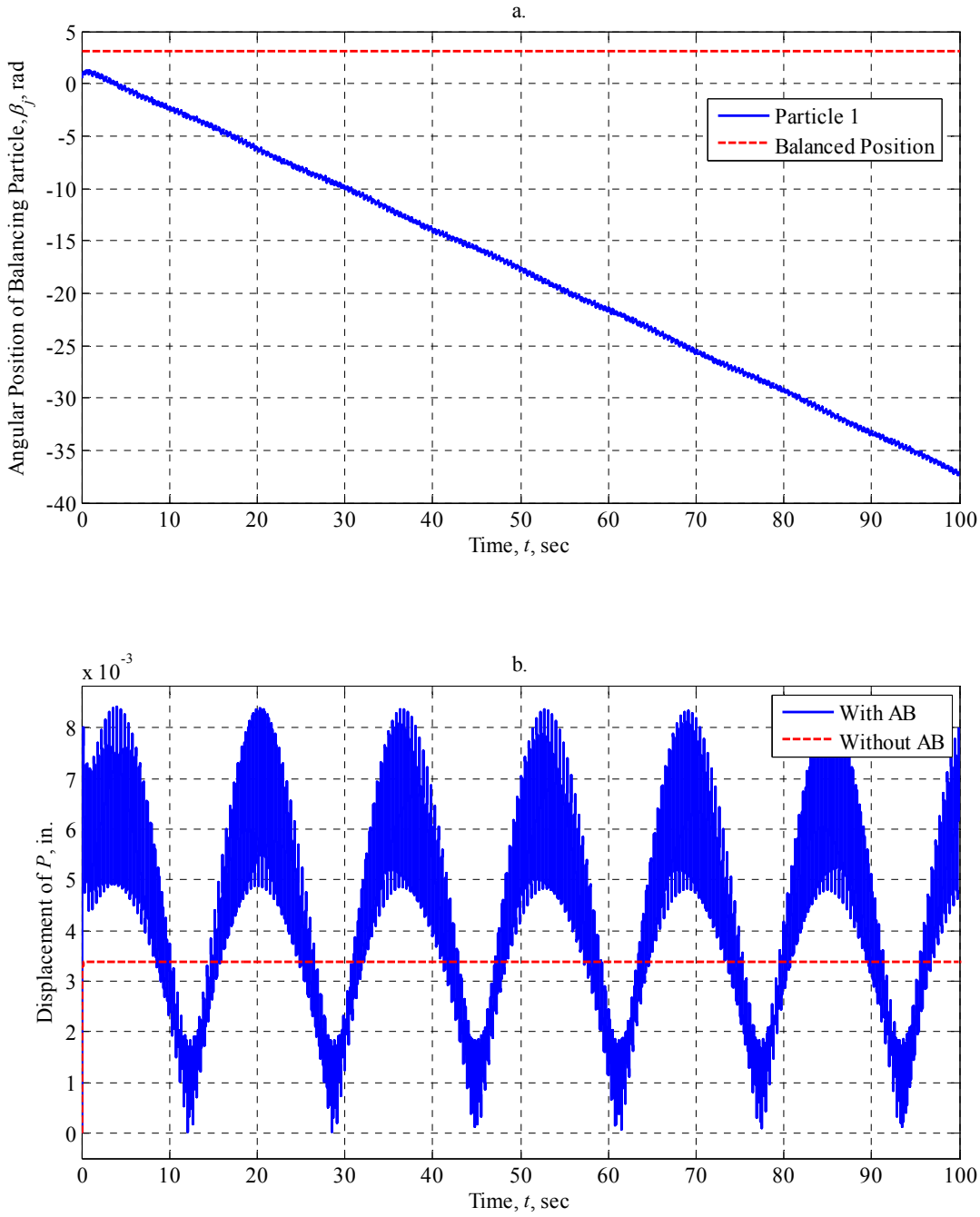

Figure 5.2.3.2.1: AB Performance at 15 rad/sec with $\mu_s = 0$ (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

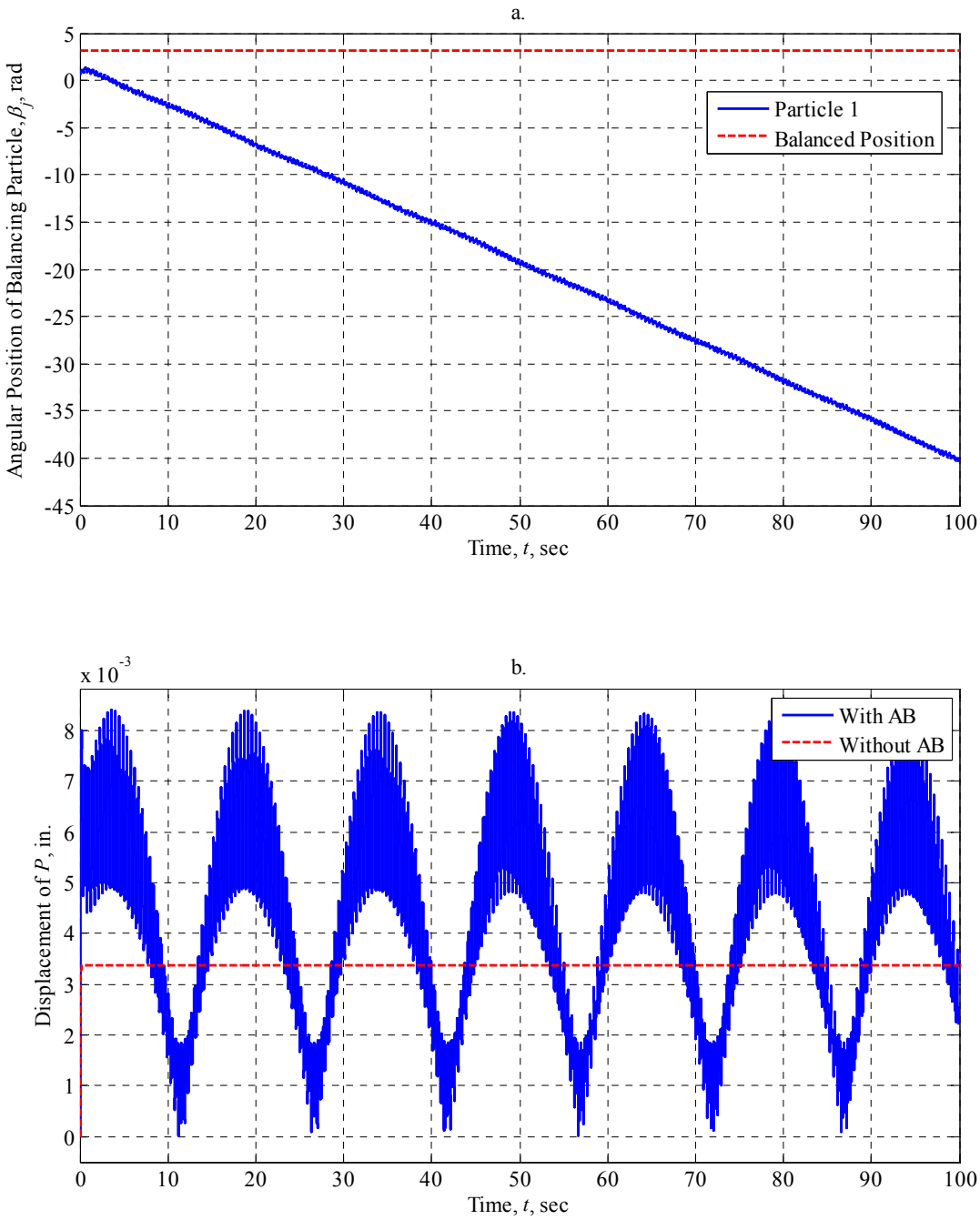

Figure 5.2.3.2.2: AB Performance at 15 rad/sec with μ_s = 0.001 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

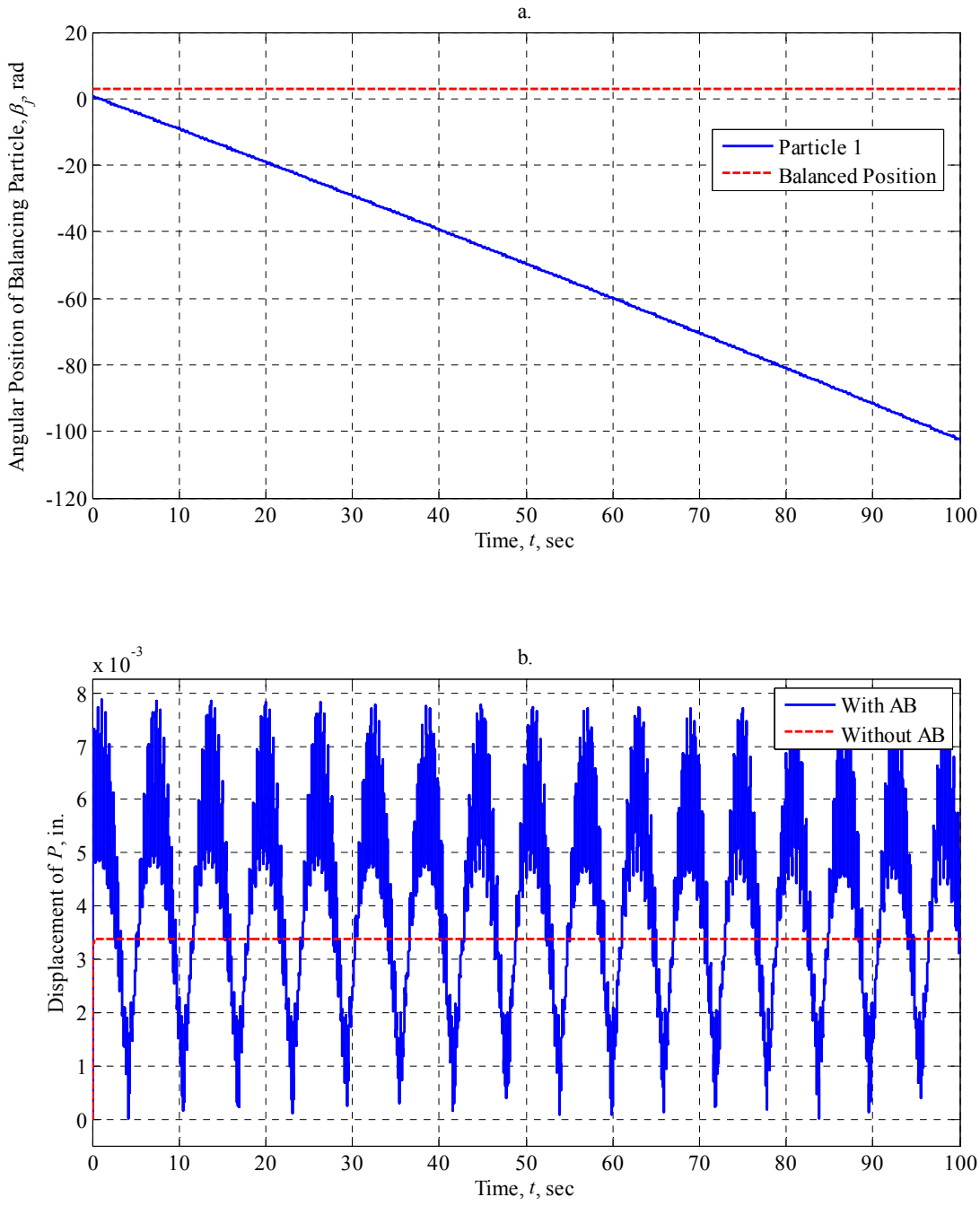

Figure 5.2.3.2.3: AB Performance at 15 rad/sec with μ_s = 0.1 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

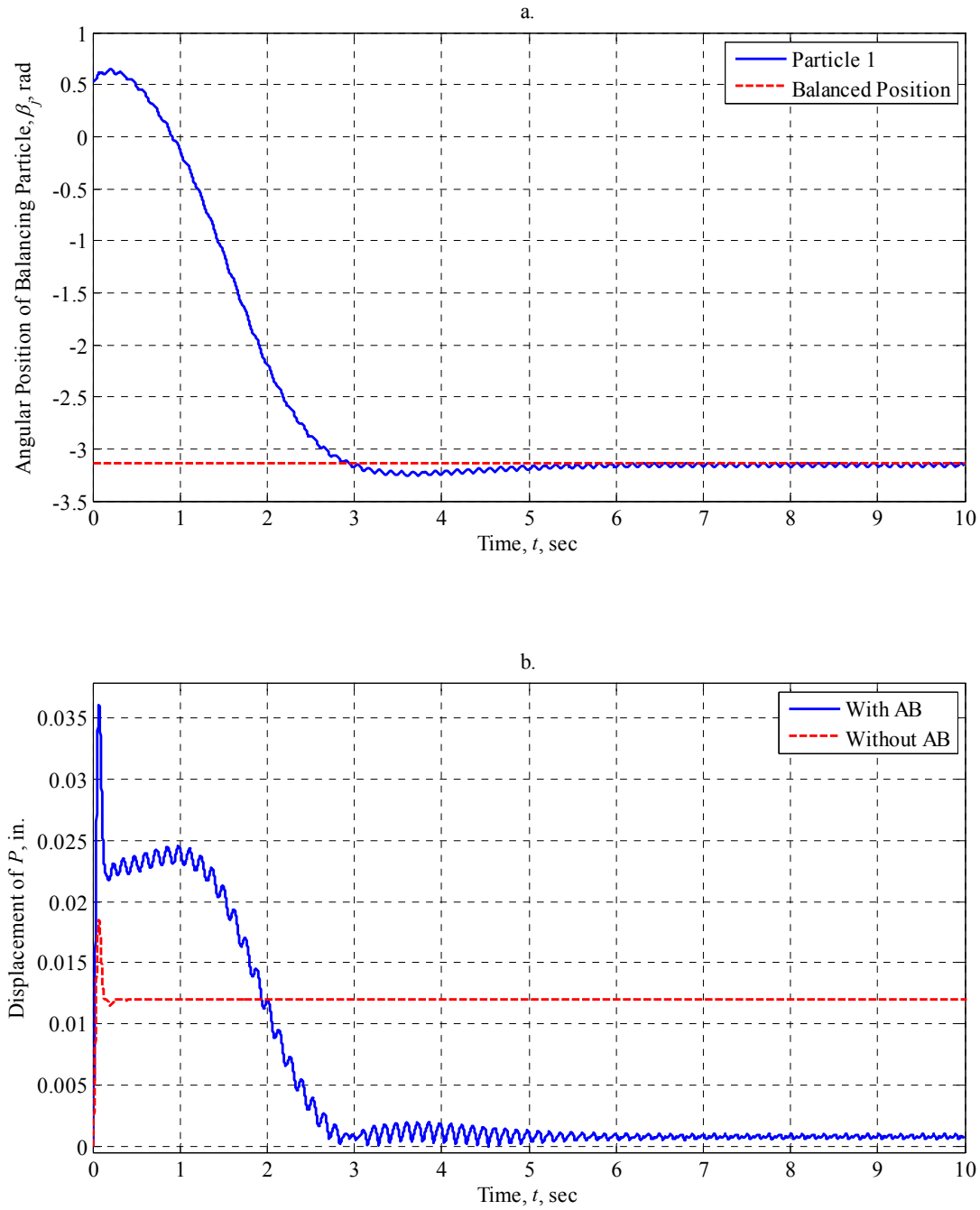

Figure 5.2.3.2.4: AB Performance at 50 rad/sec with $\mu_s = 0$ (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

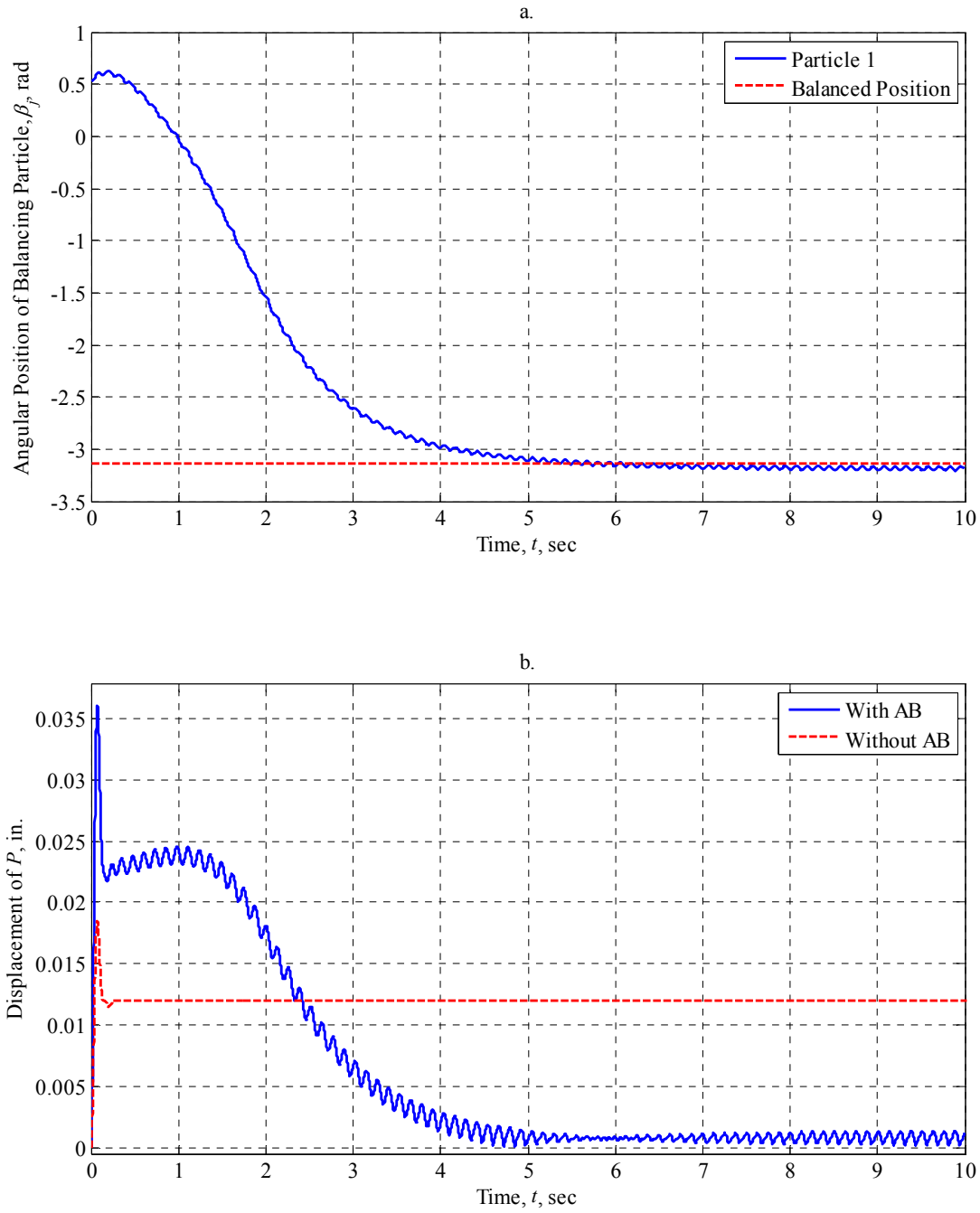

Figure 5.2.3.2.5: AB Performance at 50 rad/sec with μ_s = 0.001 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

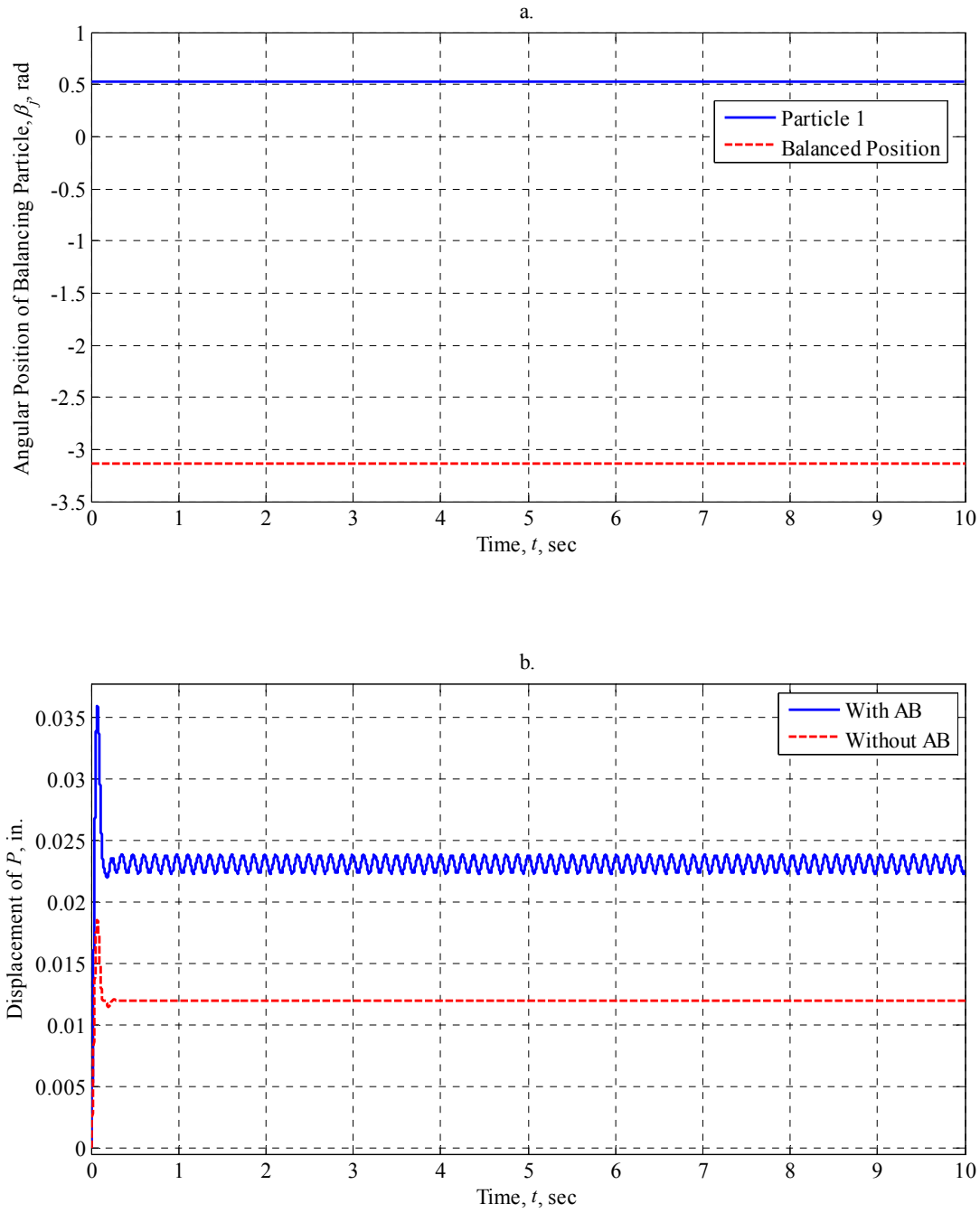

Figure 5.2.3.2.6: AB Performance at 50 rad/sec with μ_s = 0.1 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

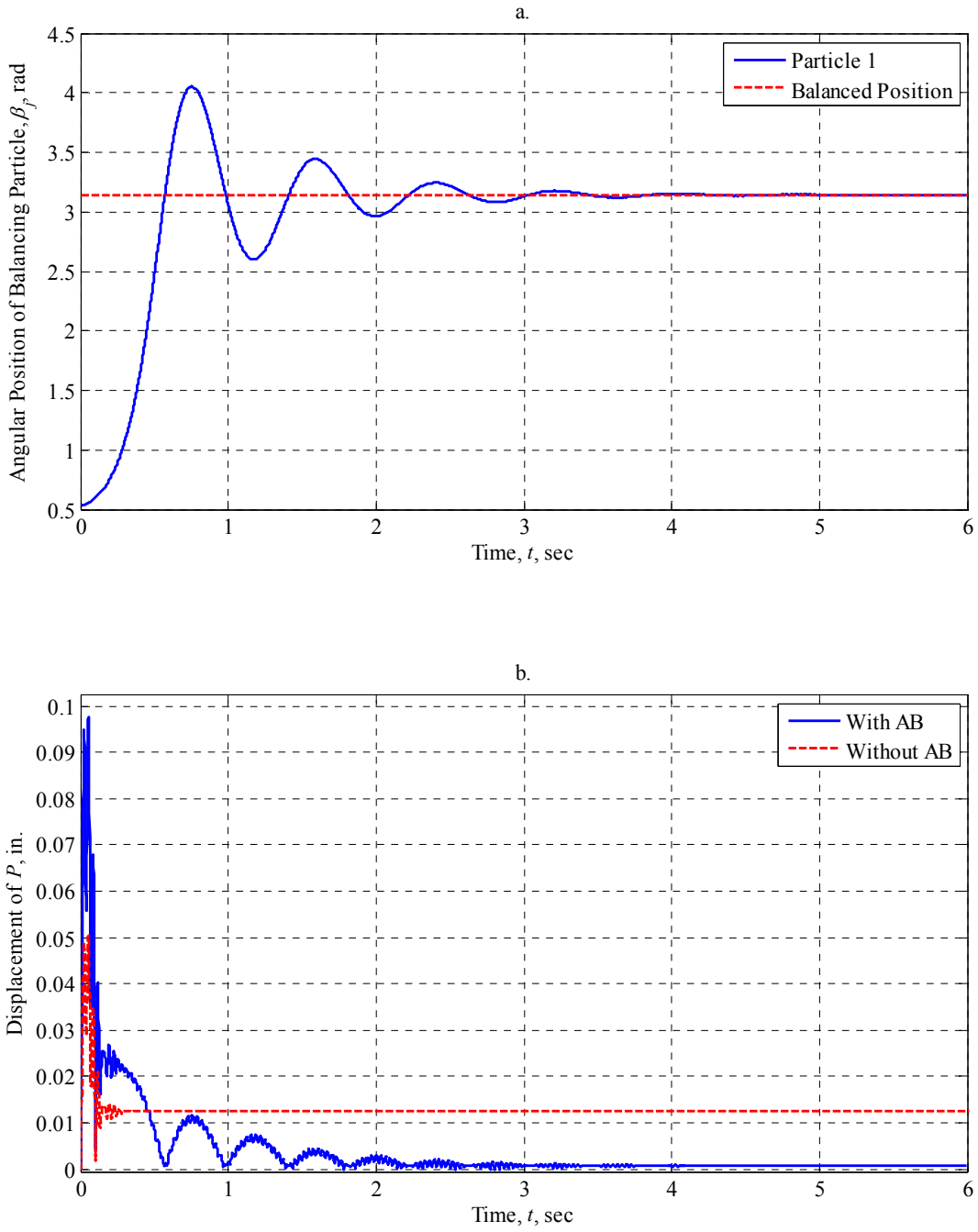

Figure 5.2.3.2.7: AB Performance at 200 rad/sec with $\mu_s = 0$ (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

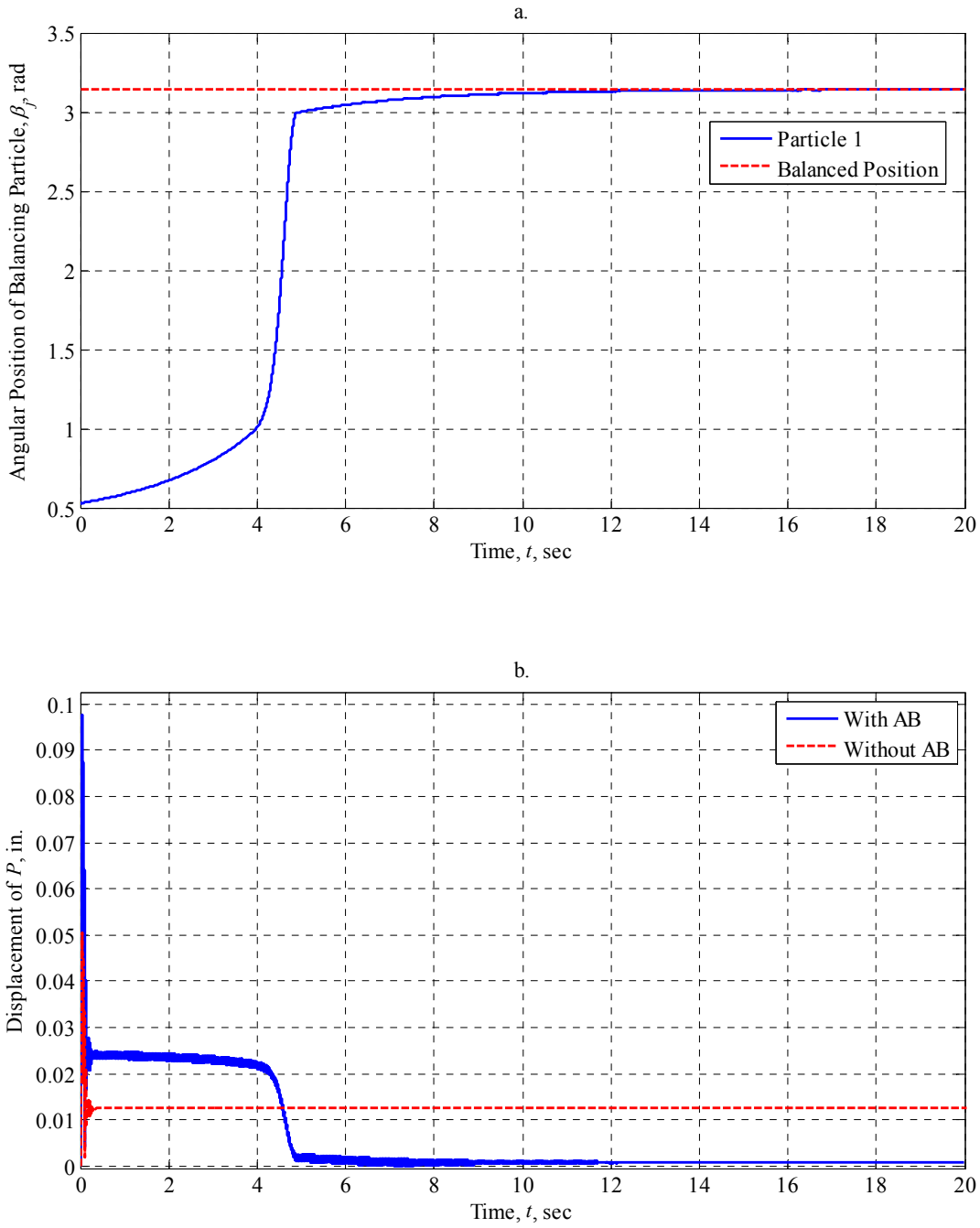

Figure 5.2.3.2.8: AB Performance at 200 rad/sec with μ_s = 0.001 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

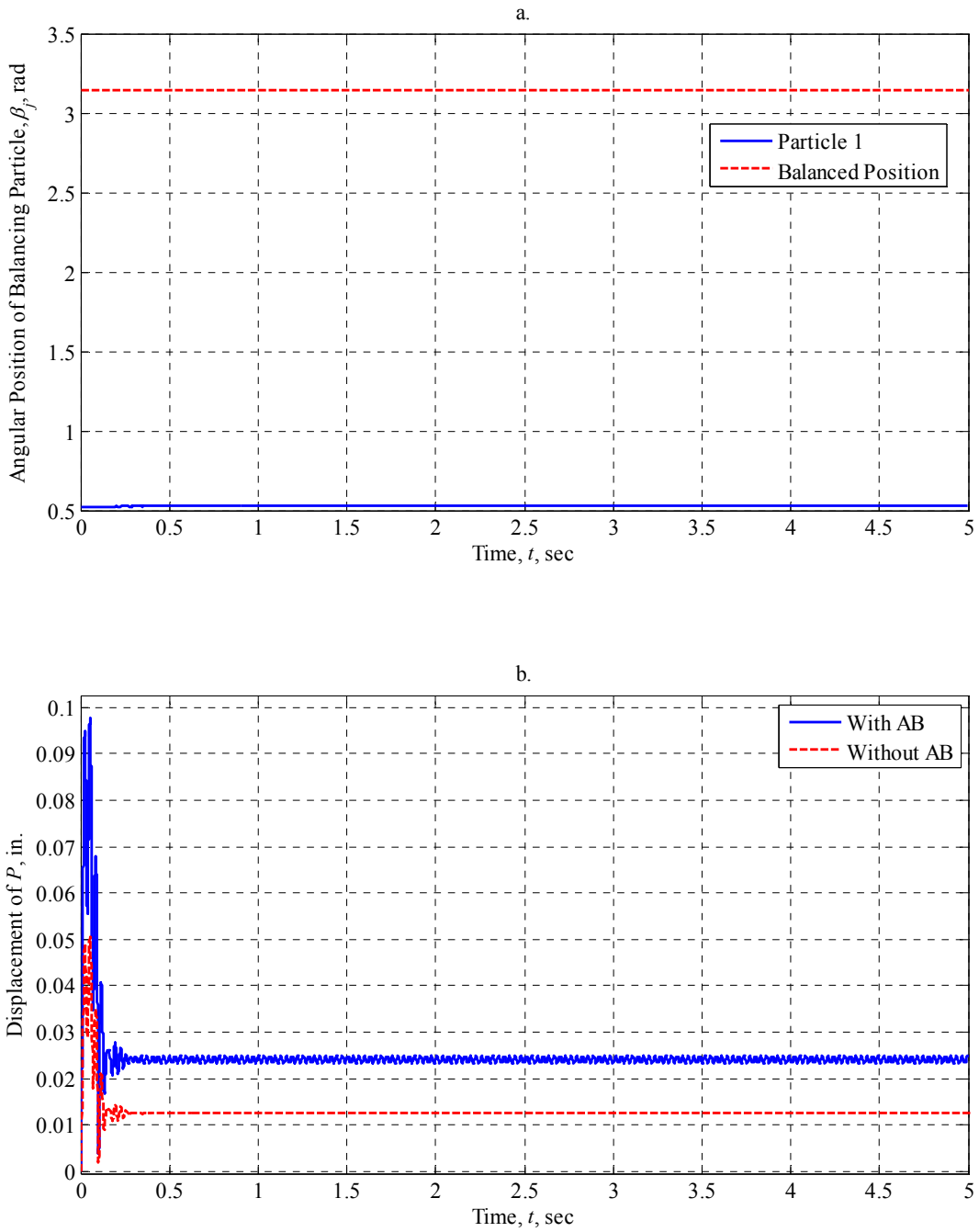

Figure 5.2.3.2.9: AB Performance at 200 rad/sec with μ_s = 0.1 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with One Balancing Particle)

5.2.4 Horizontally-Oriented Rotor with Two Balancing Particles

Our last static imbalance case will explore balancing a horizontally-oriented rotor using two balancing particles. We will determine the performance of the AB for a range of Coulomb friction levels as outlined in Table 5.2.4 and a range of operating speeds as outlined in Table 5.2.5. The performance of the AB will again be evaluated on whether or not it is able to eliminate or reduce rotor vibration as compared to the unbalanced performance alone. We will also explore the final resting position of the balancing particles and their motion relative to the rotor. Figures 5.2.4.2.1 through 5.2.4.2.9, grouped at the end of Section 5.2.4, show the behavior of the balancing particles and the displacement of the geometric center *P*. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. As before, the a-part of the figure shows the behavior of the balancing particles while the b-part shows the magnitude of the displacement of the geometric center *P*, in inches.

Table 5.2.4.1 summarizes the behavior of the balancing particles at the selected operating speeds and Coulomb friction levels by listing their steady-state angular positions and ultimate stopping times. As before if a particle never comes to rest relative to the cylinder during the time interval considered, the ultimate stopping time is determined as the time when the particle reaches and stays within 1° of its final steady-state angular position. Table 5.2.4.2 summarizes the behavior of the geometric center *P* at the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of *P* and the steady-state amplitude of the displacement of *P*. Table 5.2.4.3 illustrates the maximum and final amplitudes of the displacement of *P* in the absence of a balancer, as previously listed in Table 5.2.3.3, to serve as a baseline dataset for evaluating the performance of the rotor and automatic balancer together. Table 5.2.4.4 compares these baseline values to the values obtained in Table 5.2.4.2 by examining the difference in the maximum and steady-state amplitudes of *P*.

Table 5.2.4.1: Behavior of the Balancing Particles (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees		Ultimate Stopping Time, sec	
		β_1	β_2	β_1	β_2
15	0	—‡	—‡	—‡	—‡
	0.001	—‡	—‡	—‡	—‡
	0.1	—‡	—‡	—‡	—‡
50	0	-120.21*	119.15*	20.703†	22.006†
	0.001	-120.80*	116.05*	14.222†	13.702†
	0.1	29.87	271.79	7.119	0.079
200	0	120.00*	240.00*	3.321†	2.720†
	0.001	119.76*	239.90*	20.044†	18.811†
	0.1	50.03	250.39	13.026	13.488

† Indicates that $\pm$ 1° tolerance was used to determine the particle stop time.
‡ Indicates that no steady-state value could be determined.
* Indicates that the steady-state value was determined using an average value.

Table 5.2.4.2: Behavior of the Geometric Center (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0	1.159	—‡
	0.001	1.157	—‡
	0.1	0.987	—‡
50	0	3.413	0.0052*
	0.001	3.417	0.0106*
	0.1	3.466	2.181*
200	0	9.502	0.000*
	0.001	9.576	0.0001*
	0.1	9.756	1.491*

‡ Indicates that no steady-state value could be determined.
* Indicates that the steady-state value was determined using an average value.

Table 5.2.4.3: Rotor Performance in the Absence of an Automatic Balancer (Horizontally-Oriented Statically-Imbalanced Rotor)

Initial z-axis Angular Velocity, rad/sec	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0.3383	0.3379
50	1.853	1.194
200	5.052	1.248

Table 5.2.4.4: Comparison of Rotor Performance With and Without an Automatic Balancer (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Displacement of P, (10^{-2}) inches	Difference in Steady-State Displacement of P, (10^{-2}) inches
15	0	+0.821	— ‡
	0.001	+0.8181	— ‡
	0.1	+0.6484	— ‡
50	0	+1.561	-1.189*
	0.001	+1.564	-1.183*
	0.1	+1.613	+0.987*
200	0	+4.450	-1.248*
	0.001	+4.524	-1.248*
	0.1	+4.703	+0.243*

‡ Indicates that no steady-state value could be determined.
* Indicates that the steady-state value was determined using an average value.

A positive (+) sign in the above table indicates an increase (worsening) in the absolute displacement of P. A negative (-) sign indicates a decrease (improvement) in the absolute displacement of P. It should be noted that the difference in steady-state values of the displacement of P are calculated from the new static equilibrium point of the rotor caused by the addition of two 1-oz. balancing particles. For a spring stiffness of 1000 lb/ft and two balancing particles each weighing 1 oz., this additional deflection is $15.0(10^{-4})$ in.

5.2.4.1 Summary of Findings

For the cases where $\omega_3 = 15$ rad/sec, we see that the balancing particles never reach a steady-state position, and are observed to travel around the race together, at the same angular position. This means the geometric center will not exhibit a single steady-state amplitude, but will instead exhibit periodic behavior where it spends the majority of its time above the steady-state amplitude of the rotor in the absence of a balancer. There are times when the performance of the balancer is improved, but overall, the balancer serves to severely degrade performance. The maximum displacement amplitude of the geometric center is $1.159(10^{-2})$ inches and occurs when the Coulomb friction level is $\mu_s = 0$. This is an increase of 243% over an amplitude of $0.3383(10^{-2})$ inches in the absence of a balancer. This significant increase results from the addition of two 1-oz. imbalances along with the effect of having gravitational forces acting in the plane of the rotor.

As the Coulomb friction level increases, the period for the motion of the geometric center shortens. When $\mu_s = 0$, the period is approximately 14.7 seconds. When $\mu_s = 0.001$, the period decreases to 13.16 seconds. When $\mu_s = 0.1$, the period shortens to 5.26 seconds. Again, it is unclear why this phenomenon occurs, but it is similar to the behavior found with a single balancing particle and can certainly be attributed to gravitational effects.

For the cases where $\omega_3 = 50$ rad/sec, the two balancing particles are seen to get very close to their balanced positions for Coulomb friction levels of $\mu_s = 0$ and 0.001. For these simulations, Particle 1 settles to its average steady-state value after 20.7028 seconds and 14.2216 seconds respectively; while Particle 2 settles to its average steady-state value after 22.006 seconds and 13.7021 seconds. Both particles have settling times that are very similar to one another, but Particle 1 is observed to settle closer to its balanced position than Particle 2 is. Particle 1 settles to a value of -120.21° when $\mu_s = 0$, a difference of 0.21° with the balanced location; and it settles to -120.80° when $\mu_s = 0.001$, a difference of 0.80°. Particle 2 settles to a value of 119.15° when $\mu_s = 0$, a difference of 0.85°; and it settles to a value of 116.05° when $\mu_s = 0.001$, a difference of 2.95°. It is unclear why Particle 1 is able to get closer to its balanced position than Particle 2.

When $\mu_s = 0$, the rotor is seen to reach an average steady-state amplitude of displacement of the geometric center of $0.0052(10^{-2})$ inches, a decrease of 99.6% from a value of $1.194(10^{-2})$ inches in the absence of a balancer. When $\mu_s = 0.001$, the rotor reaches an average steady-state amplitude of $0.0106(10^{-2})$ inches, a decrease of 99.1% from the unbalanced value.

Once the Coulomb friction level reaches $\mu_s = 0.1$, the balancer is seen to degrade performance of the rotor by increasing the average steady-state amplitude of displacement of the geometric center to a value of $2.181(10^{-2})$ inches, an increase of 82.7% from the unbalanced value of $1.194(10^{-2})$ inches. For all three Coulomb friction levels, the maximum amplitude of displacement of the geometric center is increased from a value of $1.853(10^{-2})$ inches to an average value of 3.432, an increase of 85.2%. This is different from the 243% increase we saw when $\omega_3 = 15$ rad/sec, and can only be attributed to the fact that the system is nonlinear and the behavior below the translational natural frequency of the rotor is vastly different than behavior above the translational natural frequency. This is especially true when gravitation forces act in the plane of the rotor. It is difficult to make overarching statements about several of the observed phenomenon given the complexity of the system.

It is again interesting to note the stoppage times for the particles at the highest Coulomb friction level. Particle 1 stops in 7.1190 seconds, while Particle 2 stops in only 0.07940 seconds. This is a difference of two orders of magnitude, and successive simulations with varying time steps yielded the same results. When one looks at Figure 5.2.4.2.6, it is noted that both particles appear to stop very quickly. There is little change in their angular position over the duration of the simulation, so pronounced stoppage times can likely be attributed to some numerical artifacts within MatLab. It is possible, and indeed likely, that our choice of initial conditions has a dramatic effect on the particles' behavior, and overall result for each simulation.

For the cases where $\omega_3 = 200$ rad/sec, we again see rotor performance improved for Coulomb friction levels of $\mu_s = 0$ and 0.001. At a zero friction level, Particle 1 reaches a 1° envelope around its steady-state position within 3.321 seconds, while Particle 2 reaches its envelope in 2.7201 seconds. At a friction level of 0.001, both particles take longer, with Particle 1 reaching its envelope in 20.0442 seconds and Particle 2 taking 18.8114 seconds. Both particles settle very close to their balanced positions, and end up closer than for cases where $\omega_3 = 50$ rad/sec. The particles reach their exact balanced position when Coulomb friction is zero, and get within 0.24° and 0.10° of the balanced positions for particles 1 and 2, respectively. It is possible that the higher operating speed transmits more energy to the balancing particles with the help of gravitational forces acting in the plane of the rotor.

For both $\mu_s = 0$ and 0.001, the average steady-state amplitude of displacement of the geometric center is effectively eliminated. Once the Coulomb friction level reaches a value of $\mu_s = 0.1$, the rotor's performance is degraded and the average steady-state amplitude increases 19.5% over the unbalanced value to $1.491(10^{-2})$ inches. This is a much smaller percentage than was observed for the case of $\omega_3 = 50$ rad/sec where the increase when $\mu_s = 0.1$ was seen to jump to 82.7% of the unbalanced value. The maximum amplitude for all three Coulomb friction levels increased by an average of 90.2% from $5.052(10^{-2})$ inches to an average value of $9.611(10^{-2})$ inches.

One final item of interest is the stoppage times at the highest Coulomb friction level and highest operating speed. When $\omega_3 = 200$ rad/sec and $\mu_s = 0.1$, Particle 1 took 13.0256 seconds to come to a complete stop and Particle 2 took 13.4877 seconds. Again, it is unclear why the stoppage times do not decrease dramatically for orders of magnitude increases in the Coulomb friction values.

5.2.4.2 Plots of Rotor Performance and Balancing Particle Behavior

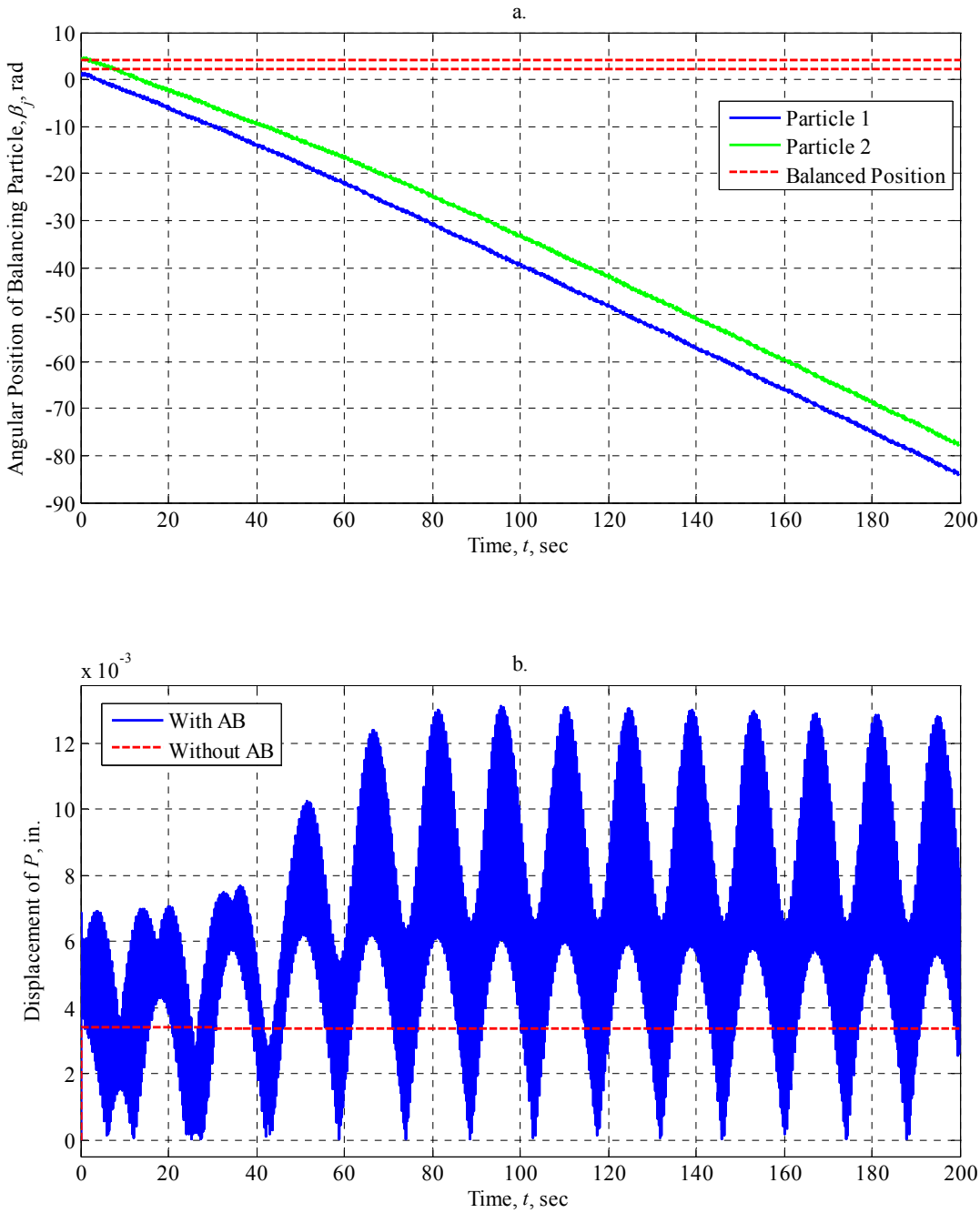

Figure 5.2.4.2.1: AB Performance at 15 rad/sec with $\mu_s = 0$ (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

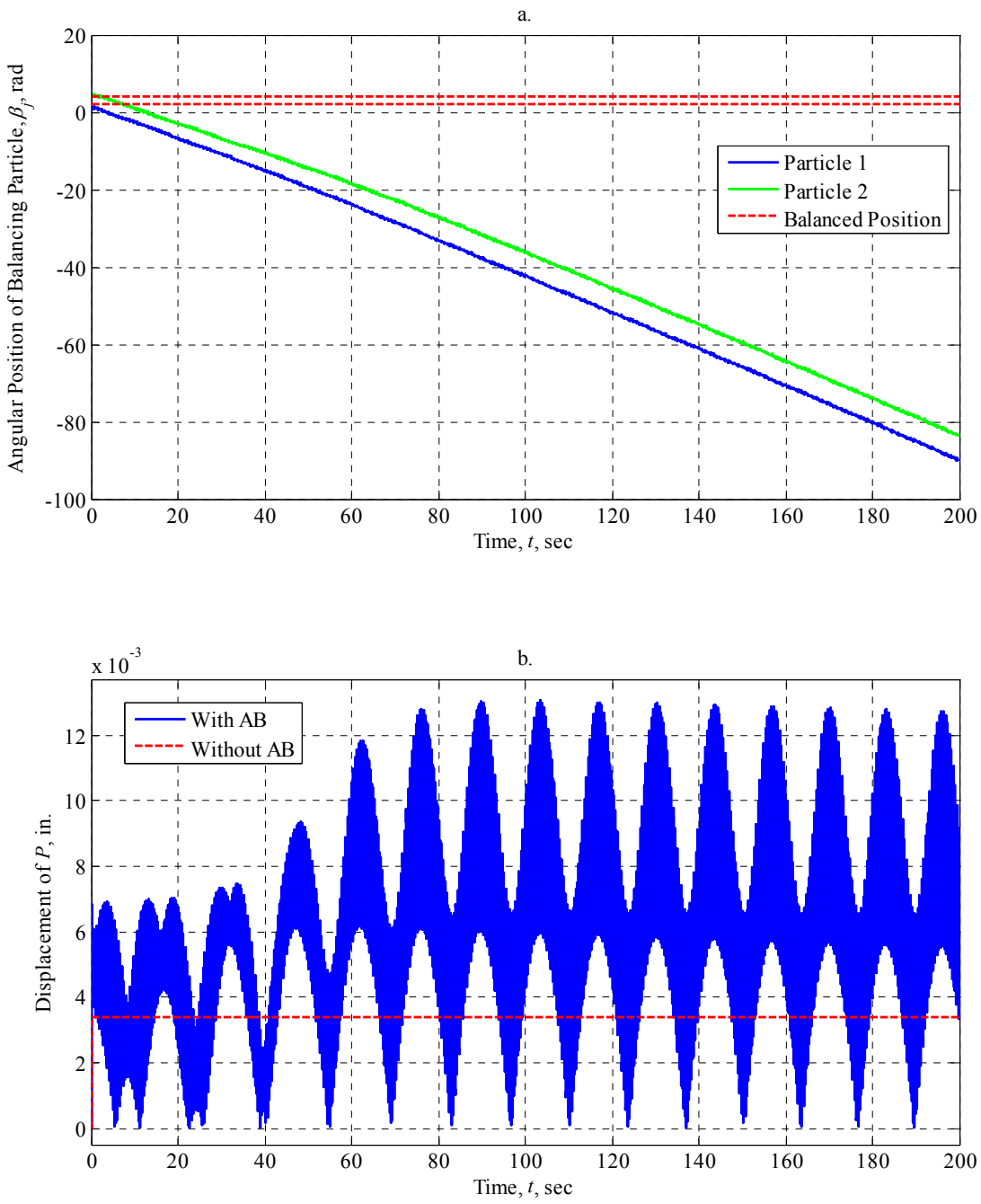

Figure 5.2.4.2.2: AB Performance at 15 rad/sec with μ_s = 0.001 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

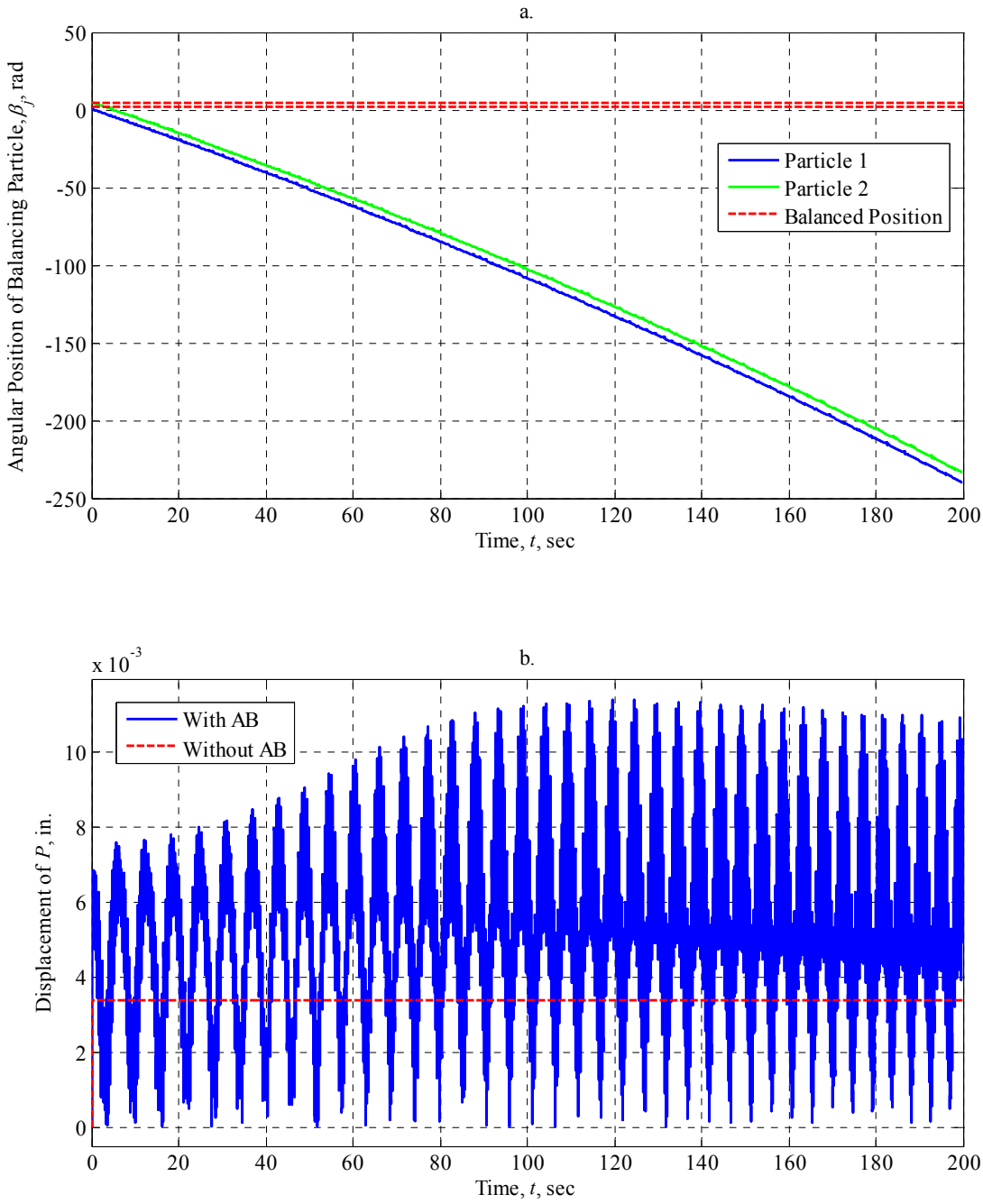

Figure 5.2.4.2.3: AB Performance at 15 rad/sec with μ_s = 0.1 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

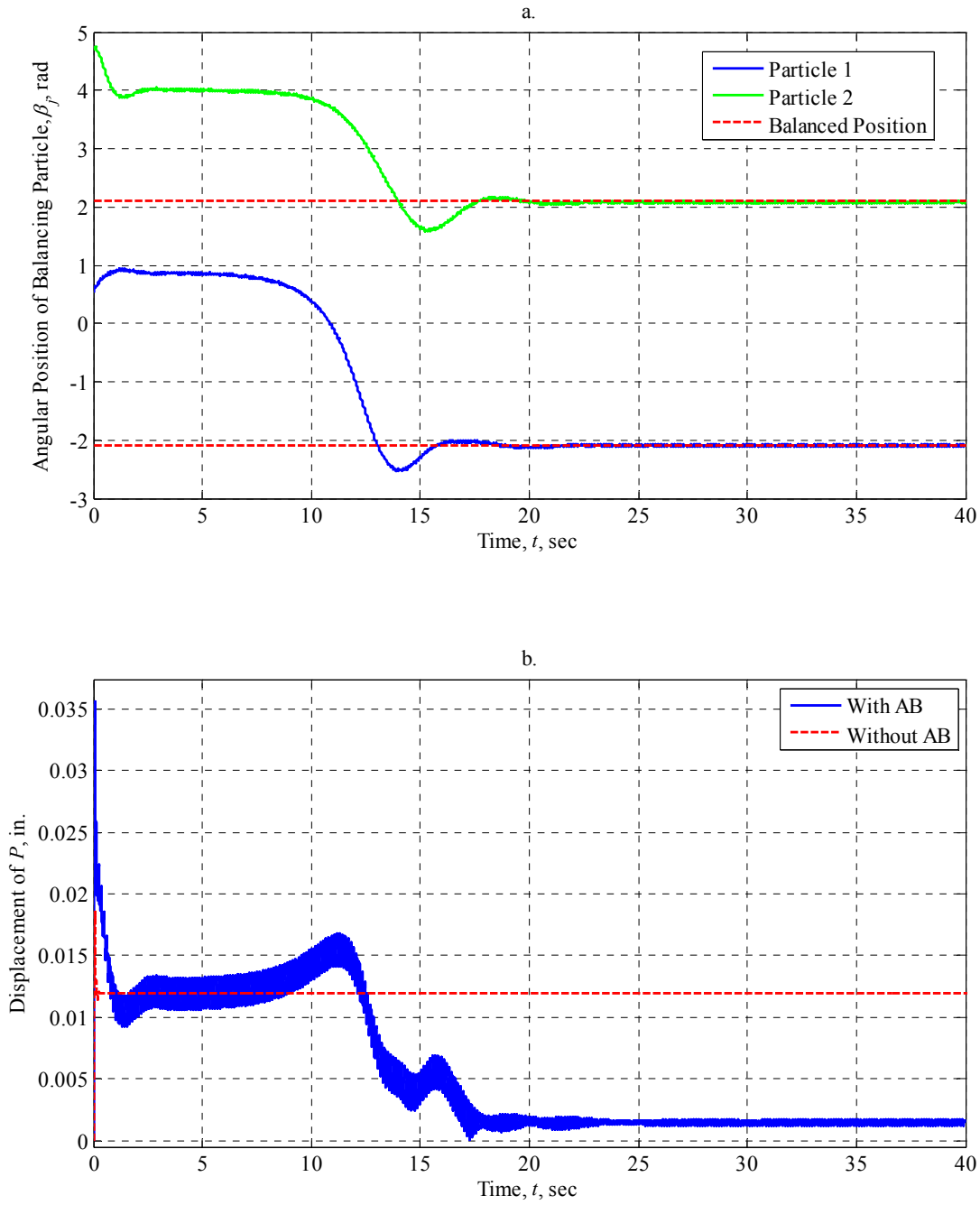

Figure 5.2.4.2.4: AB Performance at 50 rad/sec with $\mu_s = 0$ (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

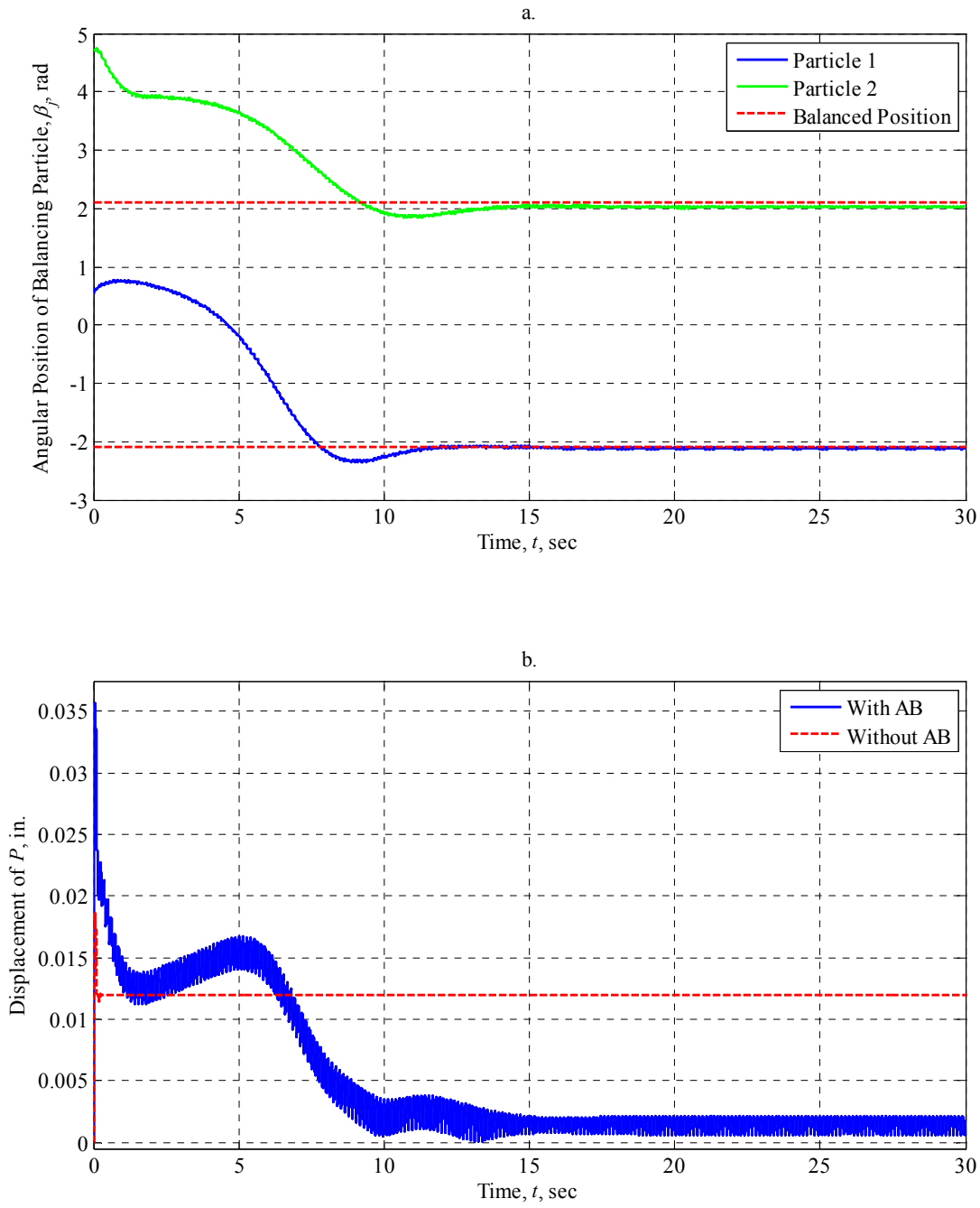

Figure 5.2.4.2.5: AB Performance at 50 rad/sec with μ_s = 0.001 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

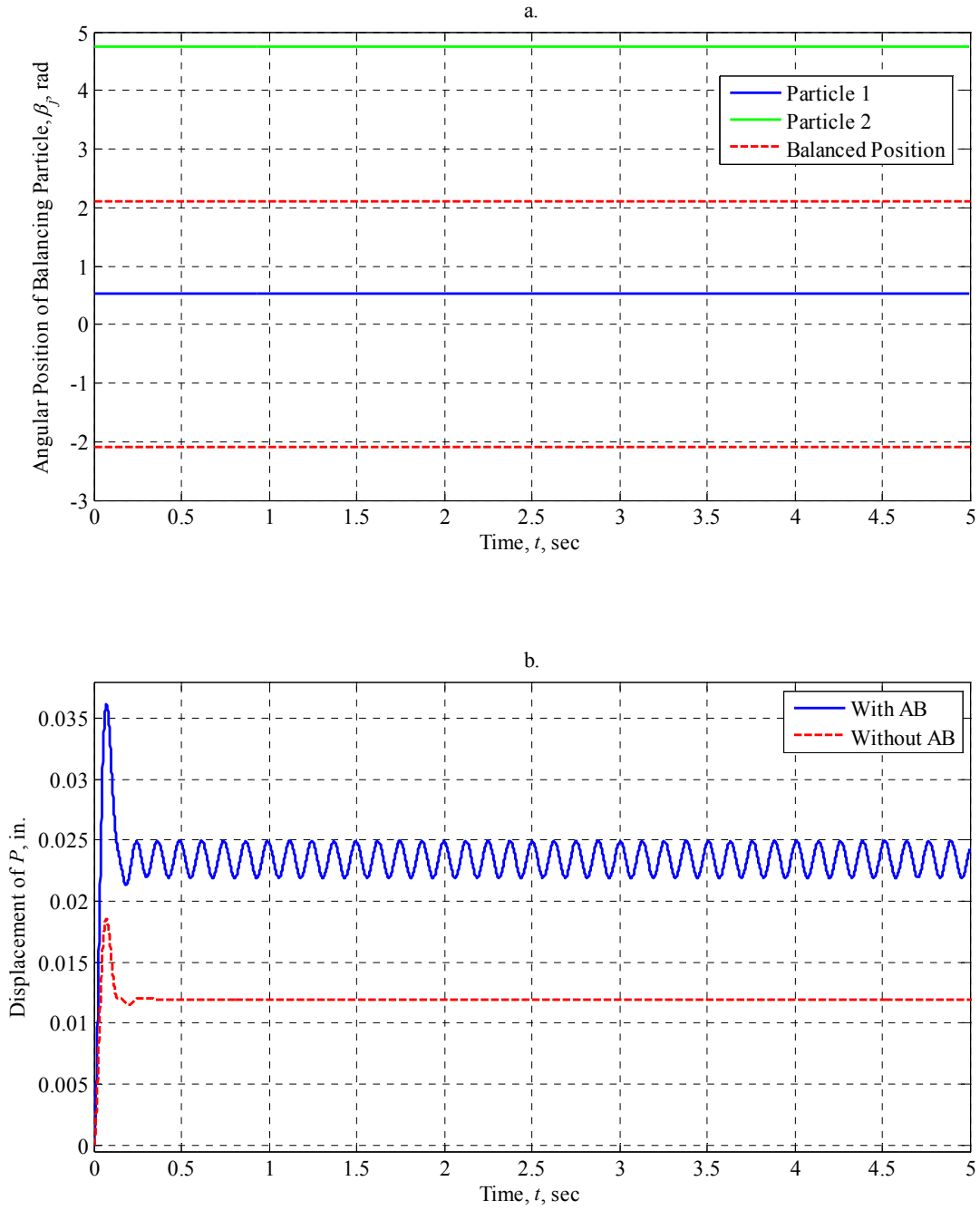

Figure 5.2.4.2.6: AB Performance at 50 rad/sec with μ_s = 0. 1 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

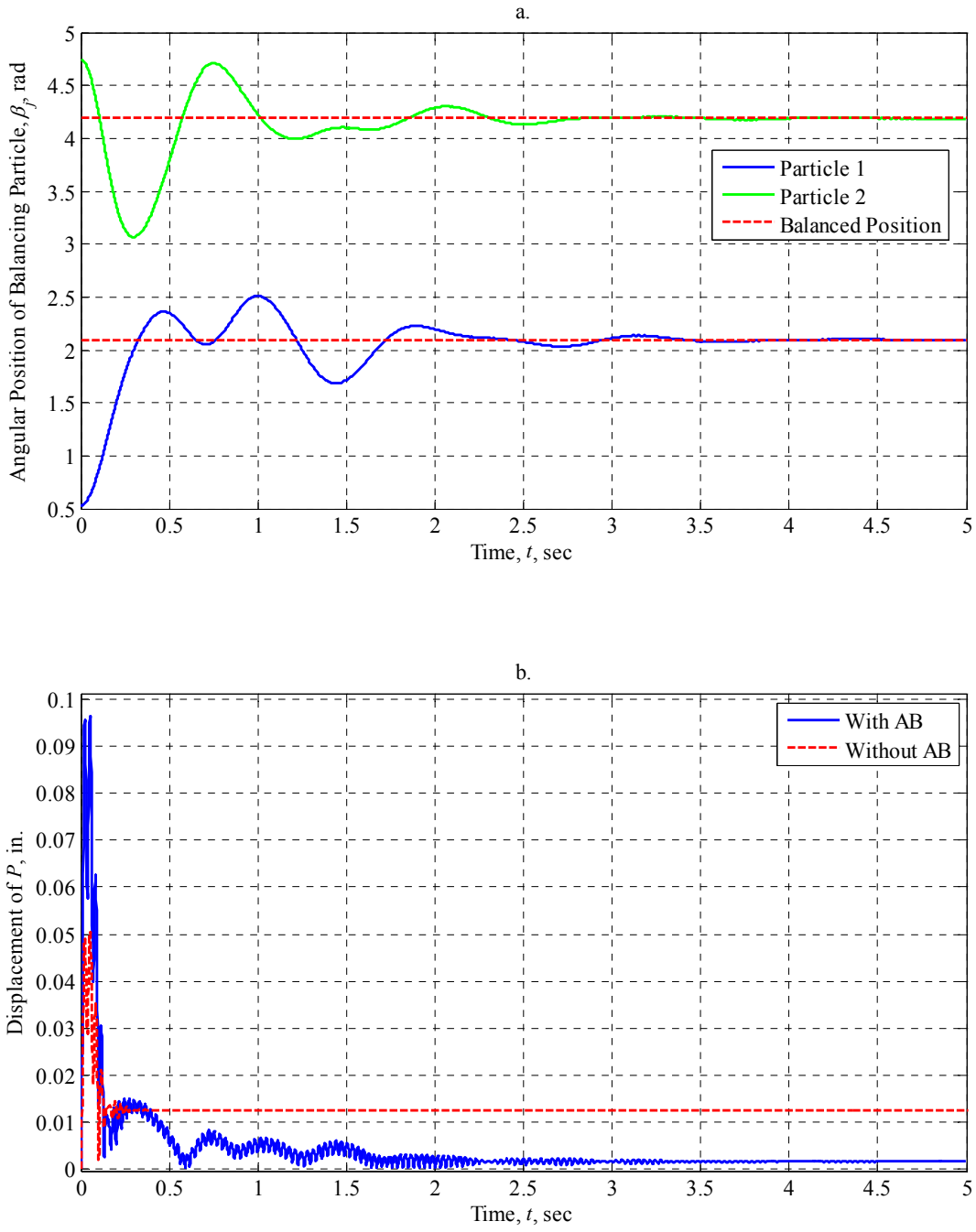

Figure 5.2.4.2.7: AB Performance at 200 rad/sec with $\mu_s = 0$ (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

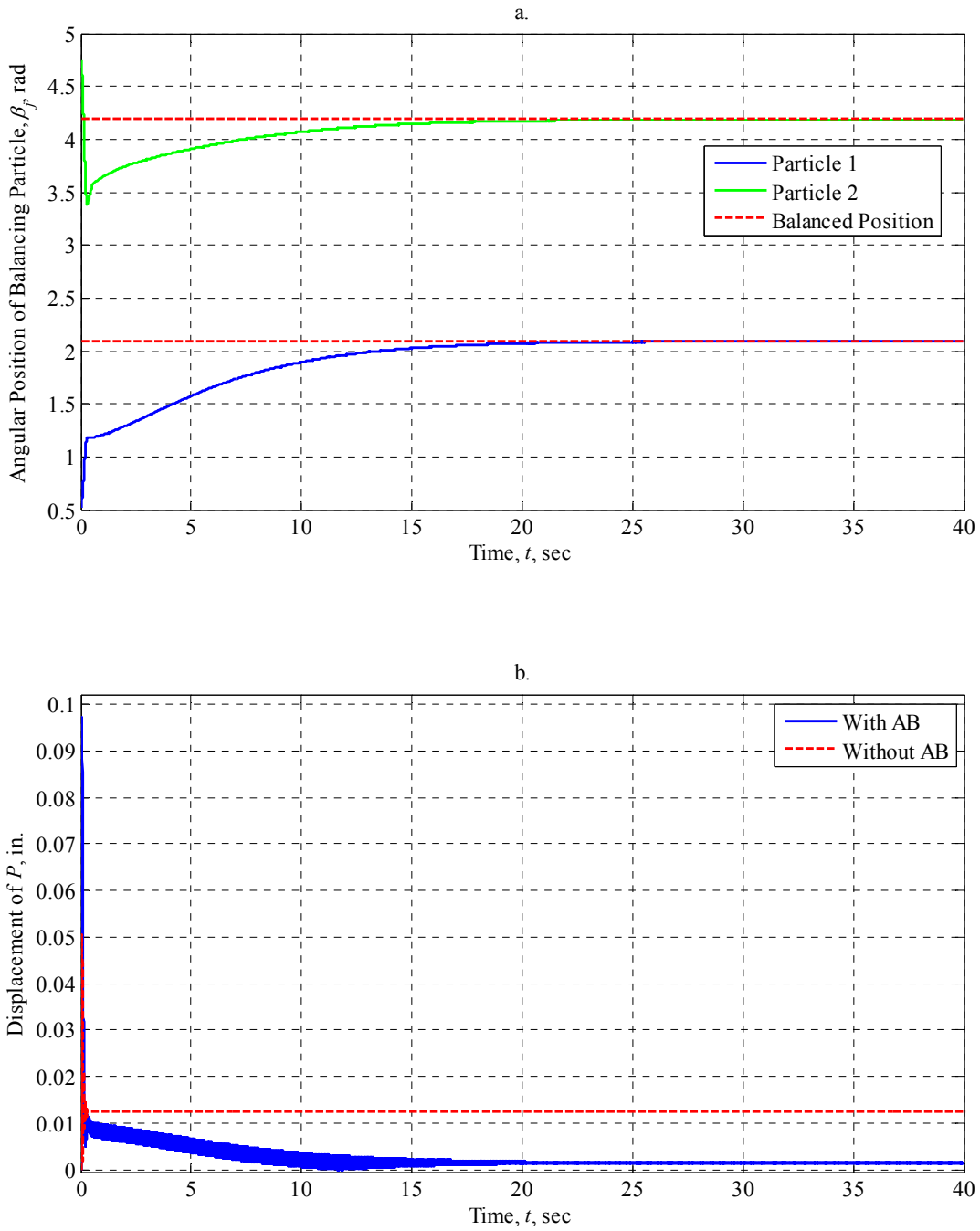

Figure 5.2.4.2.8: AB Performance at 200 rad/sec with μ_s = 0.001 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

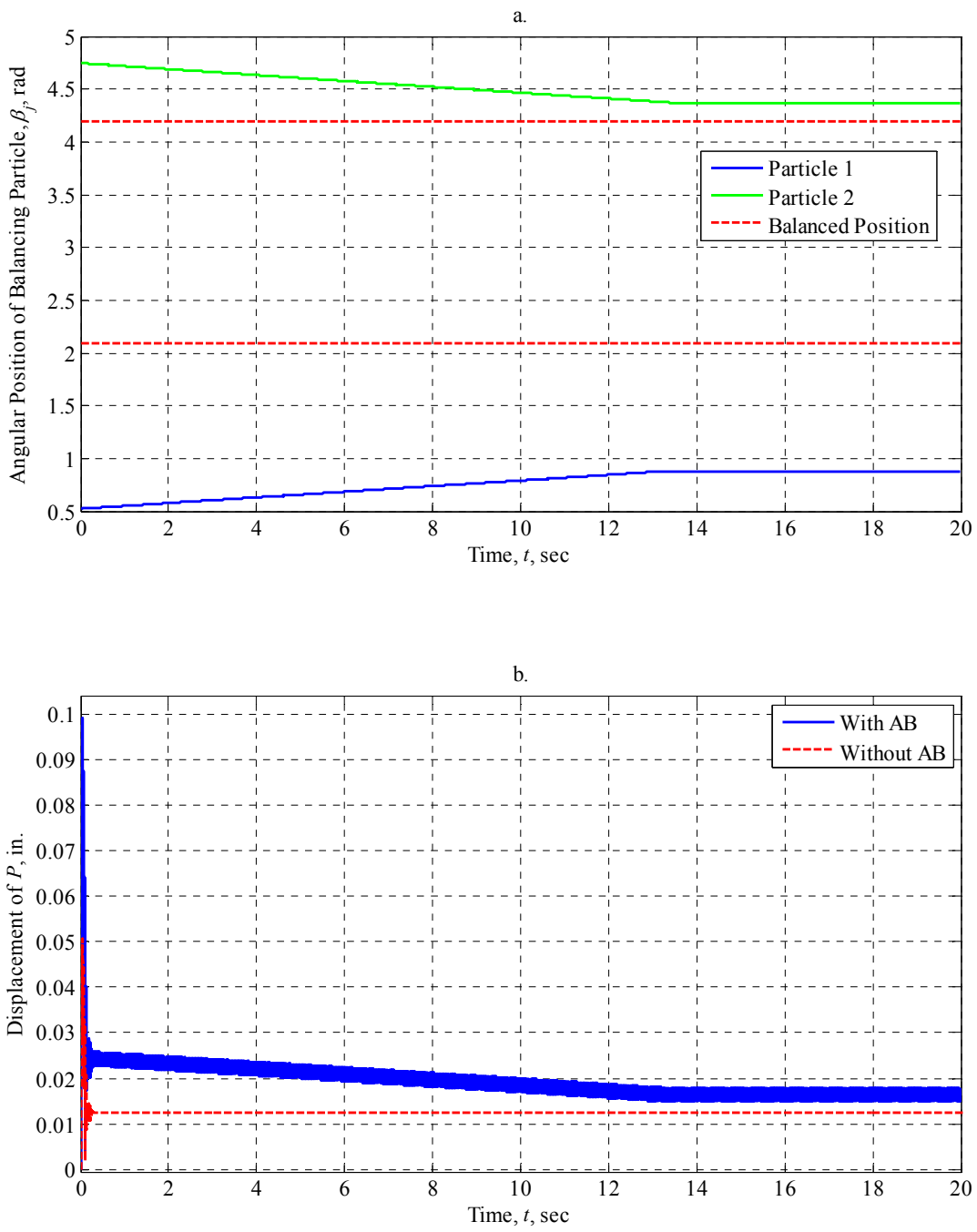

Figure 5.2.4.2.9: AB Performance at 200 rad/sec with μ_s = 0.1 (Horizontally-Oriented Statically-Imbalanced Rotor, Single Plane Balancing with Two Balancing Particles)

5.3 Dual-Plane Balancing of a Dynamically Imbalanced Rotor

The first dynamically-imbalanced rotor we will examine is shown in Figure 5.3.1. The rotor possesses two imbalance masses, both denoted m and shown in blue, located on the inner radius of the cylindrical rotor. The first mass is located a distance $d/2$ in the positive z-direction at an angular position of 0° relative to the body-fixed coordinate axes, while the second mass is located a distance $d/2$ in the negative z-direction at an angular position of 180° relative to the body-fixed coordinate axes . This equal and opposite positioning of the imbalance masses places the center of mass for the system at the geometric center P and ensures the rotor will only wobble about its intermediate axes during operation. Our first balancing scenario will feature the introduction two balancing particles m_1 and m_3, shown in red, having identical mass with each imbalance and located at the same radial distance from P. Mass m_1 lies in a race at the front plane of the rotor in the positive z-direction, while mass m_3 lies in a race at the rear plane of the rotor in the negative z-direction. To balance the rotor, particle m_1 must travel to the 180° position, shown by the dark red particle labeled m_1', relative to the body-fixed coordinate system and particle m_3 must travel to the 0° position, shown by the dark red particle labeled m_3', relative to the body-fixed coordinate system. This positioning will ensure that the combined center of mass lies at the geometric center P and that all products of inertia will go to zero. The calculation for this is shown below.

$$\left.\begin{aligned}
\bar{x} &= mr_i + m\left(-r_i\right) + m_1 r_1 \cos\beta_1 + m_3 r_3 \cos\beta_3 = 0 \\
\bar{y} &= m\left(0\right) + m\left(0\right) + m_1 r_1 \sin\beta_1 + m_3 r_3 \sin\beta_3 = 0 \\
\bar{z} &= m\frac{d}{2} + m\left(\frac{-d}{2}\right) + m_1\frac{d}{2} + m_3\left(\frac{-d}{2}\right) = 0 \\
I_{xy} &= mr_i\left(0\right) + m\left(-r_i\right)\left(0\right) + m_1 r_1 \cos\beta_1\left(r_1 \sin\beta_1\right) + m_3 r_3 \cos\beta_3\left(r_3 \sin\beta_3\right) = 0 \\
I_{xz} &= mr_i\frac{d}{2} + m\left(-r_i\right)\left(\frac{-d}{2}\right) + m_1 r_1 \cos\beta_1\left(\frac{d}{2}\right) + m_3 r_3 \cos\beta_3\left(\frac{-d}{2}\right) = 0 \\
I_{yz} &= m\left(0\right)\frac{d}{2} + m\left(0\right)\left(\frac{-d}{2}\right) + m_1 r_1 \sin\beta_1\left(\frac{d}{2}\right) + m_3 r_3 \sin\beta_3\left(\frac{-d}{2}\right) = 0
\end{aligned}\right\} \rightarrow \begin{cases} \beta_1 = \pm\pi \\ \beta_3 = 0 \end{cases} \quad (5.3)$$

Note that the condition on the z-coordinate for the center of mass is satisfied by default since there is an equal amount of particle mass in the front and rear races.

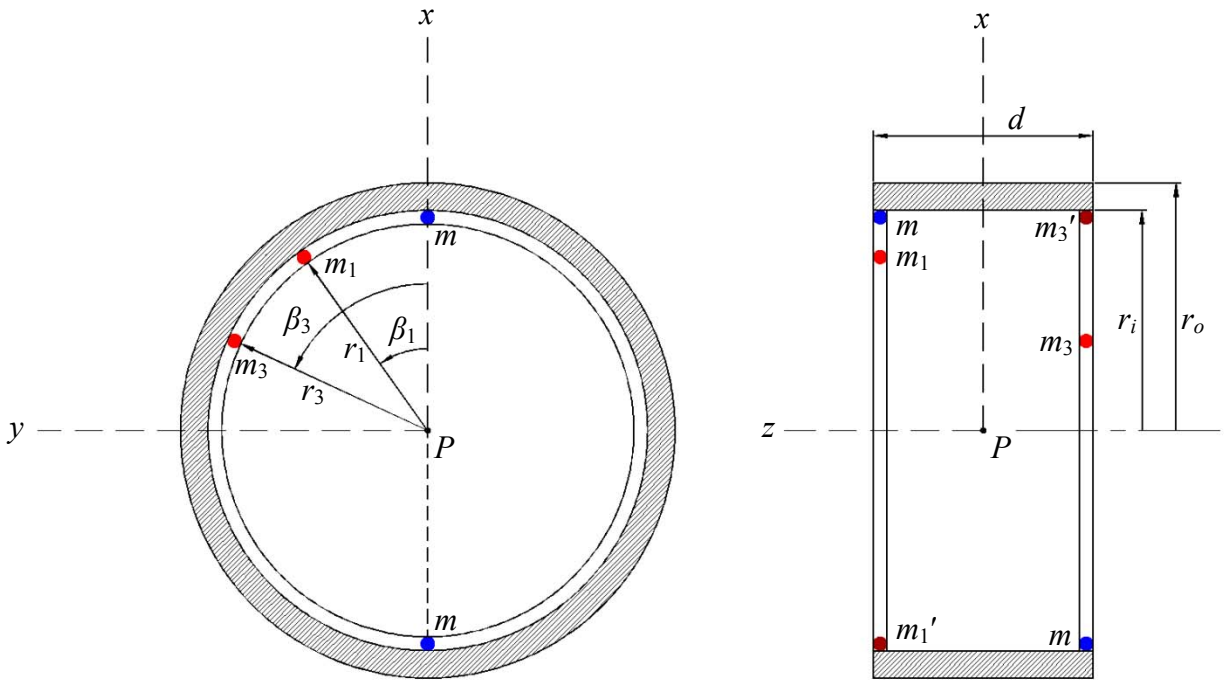

Figure 5.3.1: Dual-Plane Balancing of a Dynamically-Imbalanced Rotor Utilizing One Balancing Particle per Race

Next, we examine the case of two balancing particles per race for a total of four particles. The particles located in the front plane of the cylinder, at $d/2$ in the positive z-direction, are m_1 and m_2, and the particles in the rear plane, at $d/2$ in the negative z-direction, are m_3 and m_4. All particles have the same mass and radial distance from P as the imbalance masses. The situation is illustrated in Figure 5.3.2 with the balancing particles shown in red. To balance the rotor, particles 1 and 2 in the front race must travel to positions symmetric about the 180° position and be separated by an angle of 120°, while particles 3 and 4 in the rear race must travel to positions symmetric about the 0° position with 120° of separation between them. This situation is shown in Figure 5.3.3 with the balancing particles shown in dark red and denoted with primes. The calculation for this follows.

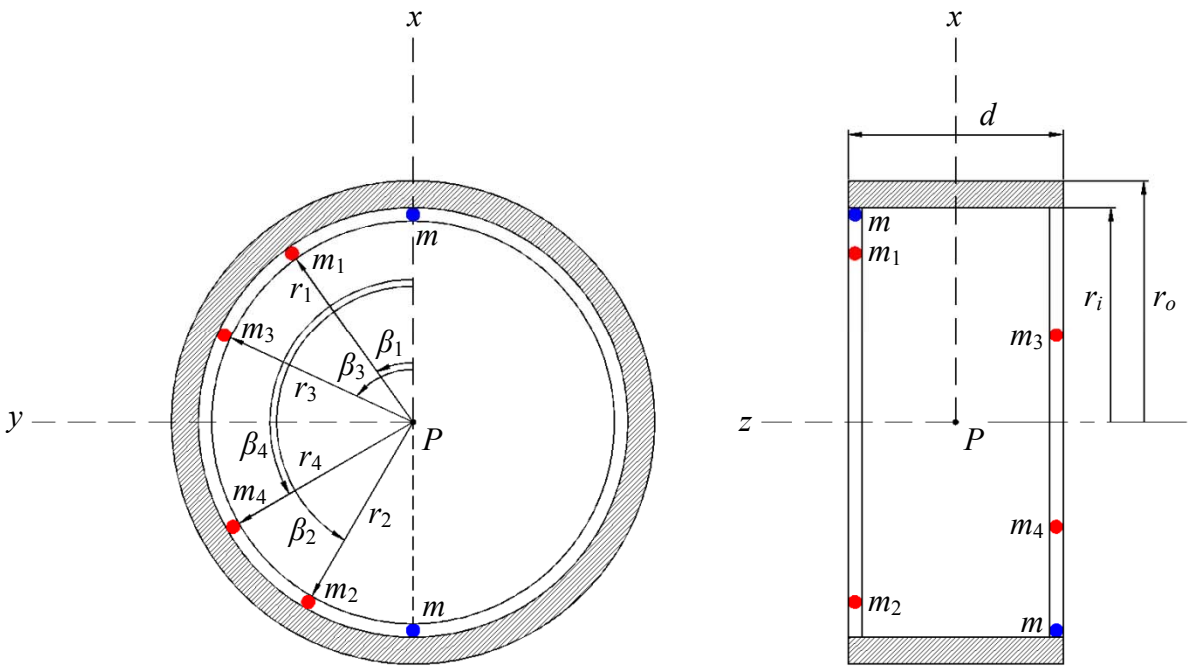

Figure 5.3.2: Dual-Plane Balancing of a Dynamically-Imbalanced Rotor Utilizing Two Balancing Particles per Race, Unbalanced Configuration

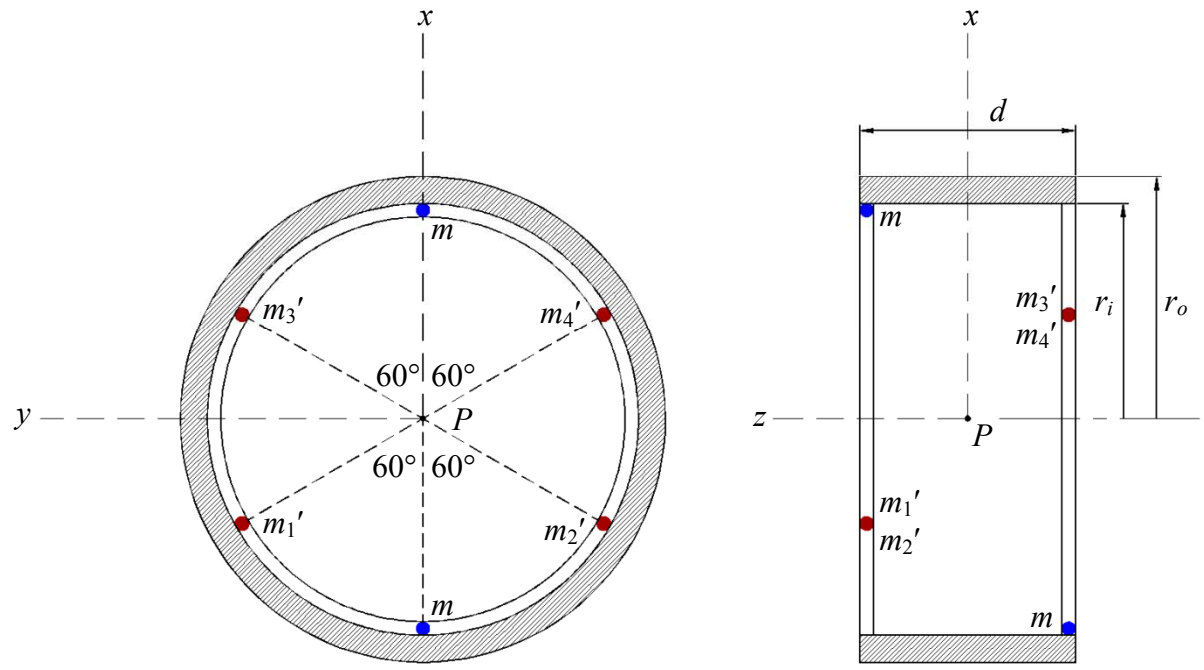

Figure 5.3.3: Dual-Plane Balancing of a Dynamically-Imbalanced Rotor Utilizing Two Balancing Particles per Race, Balanced Configuration

$$\left.\begin{aligned}
&\bar{x} = mr_i + m\left(-r_i\right) + m_1 r_1 \cos\beta_1 + m_2 r_2 \cos\beta_2 + m_3 r_3 \cos\beta_3 + m_4 r_4 \cos\beta_4 = 0 \\
&\bar{y} = m\left(0\right) + m\left(0\right) + m_1 r_1 \sin\beta_1 + m_2 r_2 \sin\beta_2 + m_3 r_3 \sin\beta_3 + m_4 r_4 \sin\beta_4 = 0 \\
&\bar{z} = m\frac{d}{2} + m\left(\frac{-d}{2}\right) + m_1\frac{d}{2} + m_2\frac{d}{2} + m_3\left(\frac{-d}{2}\right) + m_4\left(\frac{-d}{2}\right) = 0 \\
&I_{xy} = mr_i\left(0\right) + m\left(-r_i\right)\left(0\right) + m_1 r_1 \cos\beta_1\left(r_1 \sin\beta_1\right) + m_2 r_2 \cos\beta_2\left(r_2 \sin\beta_2\right) + \\
&m_3 r_3 \cos\beta_3\left(r_3 \sin\beta_3\right) + m_4 r_4 \cos\beta_4\left(r_4 \sin\beta_4\right) = 0 \\
&I_{xz} = mr_i\frac{d}{2} + m\left(-r_i\right)\left(\frac{-d}{2}\right) + m_1 r_1 \cos\beta_1\left(\frac{d}{2}\right) + m_2 r_2 \cos\beta_2\left(\frac{d}{2}\right) + \\
&m_3 r_3 \cos\beta_3\left(\frac{-d}{2}\right) + m_4 r_4 \cos\beta_4\left(\frac{-d}{2}\right) = 0 \\
&I_{yz} = m\left(0\right)\frac{d}{2} + m\left(0\right)\left(\frac{-d}{2}\right) + m_1 r_1 \sin\beta_1\left(\frac{d}{2}\right) + m_2 r_2 \sin\beta_2\left(\frac{d}{2}\right) + \\
&m_3 r_3 \sin\beta_3\left(\frac{-d}{2}\right) + m_4 r_4 \sin\beta_4\left(\frac{-d}{2}\right) = 0
\end{aligned}\right\} \rightarrow \begin{cases}
\beta_1 = \pm\dfrac{2\pi}{3} \\
\beta_2 = \mp\dfrac{2\pi}{3} \\
\beta_3 = \pm\dfrac{\pi}{3} \\
\beta_4 = \mp\dfrac{\pi}{3}
\end{cases} \quad (5.4)$$

Table 5.3.1 lists the selected system properties for cases involving a single balancing particle or two balancing particles.

Table 5.3.1: Selected Values for System Properties (Dynamically-Imbalanced Rotor)

Property	Value	Property	Value
Mass of the rotor, M	1.242 slugs	Spring stiffness along X-axis, k_X	1000 lb/ft
Mass of the imbalance, m	0.001941 slugs	Spring stiffness along Y-axis, k_Y	1000 lb/ft
Mass of a balancing particle, m_j	0.001941 slugs	Spring stiffness along Z-axis, k_Z	100,000 lb/ft
Acceleration of gravity, g	32.2 ft/sec^2	Torsional stiffness along X-axis, k_{θ_1}	5000 lb-ft/rad
Inner radius of rotor, r_i	8 in.	Torsional stiffness along y_2-axis, k_{θ_2}	5000 lb-ft/rad
Outer radius of rotor, r_o	9 in.	Torsional stiffness along z-axis, k_{θ_3}	0
Radial distance of imbalance, r	8 in.	Damping coefficient along X-axis, c_X	49.3 lb-sec/ft[†]
Radial distance of balancing particle, r_j	8 in.	Damping coefficient along Y-axis, c_Y	49.3 lb-sec/ft[†]
Depth of the rotor, d	8 in.	Damping coefficient along Z-axis, c_Z	500 lb-sec/ft
Internal offset of the imbalance from rotor mid-plane, d_m	0 in.	Damping coefficient along X-axis, c_{θ_1}	59.3 lb-ft-sec/rad[†]
Internal offset of the race containing the balancing particles, $d_{1,2}$	4 in.	Damping coefficient along y_2-axis, c_{θ_2}	59.3 lb-ft-sec/rad[†]
Internal offset of the race containing the balancing particles, $d_{3,4}$	-4 in.	Damping coefficient along z-axis, c_{θ_3}	0
Angular position of the imbalance from the x-axis, β_m	0°	Viscous drag coefficient for the balancing particles in the race, μ_c	0.005 lb-sec/ft

[†] Damping values were selected to provide a viscous damping factor of 0.7.

To examine the balancing characteristics of the AB, the products of inertia for the cylinder and imbalance must be examined. To effectively balance the cylinder and eliminate unwanted vibrations, the products of inertia must vanish, transforming the body-fixed coordinate axes into principal axes. Additionally, care must be taken when choosing an operating speed for the rotor to ensure that a resonant frequency is not selected. Based on the chosen values for the system properties, Table 5.3.2 summarizes the pertinent system characteristics resulting from the above properties. Please note that the table lists characteristics based solely on the wheel and imbalance masses with no consideration of any balancing particles. Tables 5.3.3 and 5.3.4 list the selected initial conditions for the simulations involving one balancing particle per race or two balancing particles per race, respectively. Table 5.3.5 lists the range of Coulomb friction levels that will be explored for all numerical simulations. As before, the value of the coefficient of kinetic friction has been taken to be 75% of the value of the static friction. Table 5.3.5 lists the range of operating speeds that will be explored for each Coulomb friction level. The operating speeds were chosen based on the natural frequencies of the rotor listed in Table 5.3.2 and were staggered below and above each translational natural frequency with the exception of the highest natural frequency along the *Z*-axis.

Table 5.3.2: Undamped Natural Frequencies and Mass Moments of Inertia (Dynamically-Imbalanced Rotor)

Undamped Natural Frequency	Mass Moment of Inertia, lb-ft-sec^2
$\omega_{n_x} \simeq \sqrt{\frac{k_X}{M+m}} = 28.4$ rad/sec	$I_{xx} = 0.3592$
$\omega_{n_y} \simeq \sqrt{\frac{k_Y}{M+m}} = 28.4$ rad/sec	$I_{yy} = 0.3609$
$\omega_{n_z} \simeq \sqrt{\frac{k_Z}{M+m}} = 284$ rad/sec	$I_{zz} = 0.6272$
$\omega_{n_{\theta_1}} \simeq \sqrt{\frac{k_{\theta_1}}{I_{xx}}} = 118.1$ rad/sec	$I_{xy} = 0$
$\omega_{n_{\theta_2}} \simeq \sqrt{\frac{k_{\theta_2}}{I_{yy}}} = 117.9$ rad/sec	$I_{xz} = 8.627\left(10^{-4}\right)$
$\omega_{n_{\theta_3}} \simeq \sqrt{\frac{k_{\theta_3}}{I_{zz}}} = 0$	$I_{yz} = 0$

Table 5.3.3: Initial Conditions with One Balancing Particle per Race (Dynamically-Imbalanced Rotor)

Rotor Position	Rotor Velocity	Rotor Orientation	Rotor Angular Velocity	Balancing Mass Angular Position	Balancing Mass Angular Velocity
$X_0 = 0$	$\dot{X}_0 = 0$	$\theta_{1_0} = 0$	$\dot{\theta}_{1_0} = 0$	$\beta_{1_0} = 30°$	$\dot{\beta}_{1_0} = 0.15$ rad/sec
$Y_0 = 0$	$\dot{Y}_0 = 0$	$\theta_{2_0} = 0$	$\dot{\theta}_{2_0} = 0$	$\beta_{3_0} = 300°$	$\dot{\beta}_{3_0} = 0.15$ rad/sec
$Z_0 = 0$	$\dot{Z}_0 = 0$	$\theta_{3_0} = 0$	$\dot{\theta}_{3_0} = \omega_{3_0}$ (varies)		

Table 5.3.4: Initial Conditions with Two Balancing Particles per Race (Dynamically-Imbalanced Rotor)

Cylinder Position	Cylinder Velocity	Cylinder Orientation	Cylinder Angular Velocity	Balancing Mass Angular Position	Balancing Mass Angular Velocity
$X_0 = 0$	$\dot{X}_0 = 0$	$\theta_{1_0} = 0$	$\dot{\theta}_{1_0} = 0$	$\beta_{1_0} = 20°$	$\dot{\beta}_{1_0} = -0.10$ rad/sec
$Y_0 = 0$	$\dot{Y}_0 = 0$	$\theta_{2_0} = 0$	$\dot{\theta}_{2_0} = 0$	$\beta_{2_0} = 265°$	$\dot{\beta}_{2_0} = 0.15$ rad/sec
$Z_0 = 0$	$\dot{Z}_0 = 0$	$\theta_{3_0} = 0$	$\dot{\theta}_{3_0} = \omega_{3_0}$ (varies)	$\beta_{3_0} = 80°$	$\dot{\beta}_{3_0} = -0.20$ rad/sec
				$\beta_{4_0} = 130°$	$\dot{\beta}_{4_0} = 0.15$ rad/sec

Table 5.3.5: Coulomb Friction Levels of Interest for Numerical Simulations

Coefficient of Static Friction, μ_s	Coefficient of Kinetic Friction, μ_k
0	0
0.001	0.00075
0.1	0.075

Table 5.3.6: Operating Speeds for Dynamically-Imbalanced Rotor

Initial *z*-axis Angular Velocity, rad/sec
15
50
200

5.3.1 Vertically-Oriented Rotor with One Balancing Particle per Race

Our first dynamic imbalance case will explore balancing a vertically-oriented rotor using two total balancing particles with one particle in the front race and one particle in the rear race. We will determine the performance of the AB for a range of Coulomb friction levels as outlined in Table 5.3.5 and a range of operating speeds as outlined in Table 5.3.6. Since the dynamically-imbalanced rotor features only wobble motions about intermediate coordinate axes, and no

translational motions of its geometric center P, the performance of the AB will be evaluated on whether or not it is able to eliminate or reduce these unwanted rotational effects as compared to the unbalanced rotor performance alone. The motion of the geometric center will still be explored because the addition of extra masses that are not symmetric about P will result in translational motions of the geometric center. These must vanish or the performance of the rotor and AB is worsened. The final resting position of the balancing particles and their motion relative to the rotor will be examined along with the corresponding angular displacements of the rotor about intermediate coordinate axes and products of inertia. Figures 5.3.1.2.1 through 5.3.1.2.9, grouped at the end of Section 5.3.1, illustrate these behaviors. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. As before, the a-part of the figure shows the behavior of the balancing particles while the b-part shows the magnitude of the displacement of the geometric center P, in inches. The c-part of the figure shows the orientation of the rotor about the inertial X-axis and the intermediate y_2-axis, while the d-part of the figure shows the products of inertia for the system of the rotor, imbalance masses, and balancing particles. Note that the figures will span two pages per simulation and will feature the a- and b-parts on the first page and the c- and d-parts on the second. The labeling will be consistent between figures with the a- and b-parts post scripted with a "I" and the c- and d-parts post scripted with a "II" to denote the first and second pair of figures for the simulation, respectively.

Table 5.3.1.1 summarizes the behavior of the balancing particles at the selected operating speeds and Coulomb friction levels by listing their steady-state angular positions and ultimate stopping times. If a particles never comes to rest relative to the cylinder during the time interval considered, the ultimate stopping time is determined as the time when the particle reaches and stays within 1° of its final steady-state angular position. Table 5.3.1.2 summarizes the behavior of the geometric center P at the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of P and the steady-state amplitude of the displacement of P. Table 5.3.1.3 summarizes the orientation of the rotor by listing the maximum angular displacement and steady-state angular displacement about the inertial X- and intermediate y_2-axes. Table 5.3.1.4 summarizes the angular displacements of the rotor in the absence of an AB to establish a baseline for comparison purposes. Note that it is not necessary to illustrate the motion of the geometric center P in the absence of an AB since a dynamic

imbalance gives rise to only angular displacements. Table 5.3.1.5 summarizes the effect of the AB on the rotor performance by comparing the angular displacements of the rotor both before and after the addition of a balancer by comparing the differences in the maximum and steady-state angular displacements of the rotor as found in Tables 5.3.1.3 and 5.3.1.4. Table 5.3.1.6 computes the maximum and steady-state values of the products of inertia for the rotor and balancer combination, and Table 5.3.1.7 compares these values to the ones in Table 5.3.2 for the case of an imbalanced rotor alone.

Table 5.3.1.1: Behavior of the Balancing Particles (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees		Ultimate Stopping Time, sec	
		β_1	β_3	β_1	β_3
15	0	—‡	—‡	—‡	—‡
	0.001	31.27	301.33	0.3261	0.4116
	0.1	30.04	300.04	0.0242	0.0140
50	0	-54.95	125.05	28.7525†	28.7525†
	0.001	30.34	230.25	0.0706	2.3726
	0.1	30.01	299.99	0.0156	0.1518
200	0	-75.22	104.78	17.2056†	17.2056†
	0.001	-77.96	111.45	0.7215	7.4701
	0.1	46.49	281.34	10.9580	11.5550

† Indicates that $\pm$ 1° tolerance was used to determine the particle stop time.

‡ Indicates that no steady-state value could be determined.

Table 5.3.1.2: Behavior of the Geometric Center (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0	0.6705	0.6671
	0.001	0.4809	0.4790
	0.1	0.4790	0.4787
50	0	2.600	0.000
	0.001	2.618	0.4123
	0.1	2.621	1.685
200	0	7.006	0.000
	0.001	7.054	0.2037
	0.1	7.143	1.145

Table 5.3.1.3: Angular Displacement of the Rotor in the Presence of an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Angular Displacement, (10^{-2}) degrees		Steady-State Amplitude of Angular Displacement, (10^{-2}) degrees	
		θ_1	θ_2	θ_1	θ_2
15	0	0.2933	0.2924	0.2135	0.2134
	0.001	0.2934	0.2931	0.2930	0.2930
	0.1	0.2956	0.2956	0.2955	0.2955
50	0	3.849	3.844	3.417	3.412
	0.001	3.598	3.593	3.594	3.589
	0.1	2.662	2.635	2.626	2.622
200	0	18.909	18.495	15.071	15.068
	0.001	18.867	19.159	15.912	15.909
	0.1	15.050	15.770	14.996	14.992

Table 5.3.1.4: Angular Displacement of the Rotor in the Absence of an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor)

Initial z-axis Angular Velocity, rad/sec	Maximum Angular Displacement, (10^{-2}) degrees		Steady-State Amplitude of Angular Displacement, (10^{-2}) degrees	
	θ_1	θ_2	θ_1	θ_2
15	0.2165	0.2164	0.2164	0.2164
50	1.935	1.929	1.929	1.926
200	10.992	11.699	9.9283	9.9263

Table 5.3.1.5: Comparison of Rotor Performance With and Without an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Angular Displacement of the Rotor, (10^{-2}) degrees		Difference in Steady-State Angular Displacement of the Rotor, (10^{-2}) degrees	
		θ_1	θ_2	θ_1	θ_2
15	0	+0.07691	+0.07596	-0.002959	-0.002972
	0.001	+0.07699	+0.07669	+0.07658	+0.07657
	0.1	+0.07917	+0.07916	+0.07905	+0.07904
50	0	+1.9138	+1.9145	+1.4888	+1.4867
	0.001	+1.6627	+1.6636	+1.6650	+1.6626
	0.1	+0.7268	+0.7058	+0.6969	+0.6960
200	0	+7.9173	+6.7964	+5.1432	+5.1418
	0.001	+7.875	+7.460	+5.984	+5.982
	0.1	+4.058	+4.072	+5.068	+5.065

A positive (+) sign in the above table indicates an increase (worsening) in the absolute displacement of P. A negative (-) sign indicates a decrease (improvement) in the absolute displacement of P.

Table 5.3.1.6: Products of Inertia With an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Product of Inertia, (10^{-4}) lb-ft-sec^2			Steady-State Product of Inertia, (10^{-4}) lb-ft-sec^2		
		I_{xy}	I_{xz}	I_{yz}	I_{xy}	I_{xz}	I_{yz}
15	0	8.621	11.186	5.935	— ‡	8.588	-0.0463
	0.001	-0.00458	10.205	5.926	-0.00458	10.070	5.923
	0.1	0.0000828	10.205	5.893	0.0000828	10.202	5.893
50	0	-8.627	17.253	-7.063	-8.111	13.580	-7.063
	0.001	8.002	15.107	6.492	8.002	15.107	5.495
	0.1	0.001209	10.206	5.893	0.001209	10.206	5.893
200	0	8.446	16.685	-8.626	-4.251	10.825	-8.342
	0.001	-8.594	16.791	-8.233	-4.696	11.104	-8.233
	0.1	2.644	10.748	7.358	2.644	10.748	7.358

‡ Indicates that no steady-state value could be determined.

Table 5.3.1.7: Comparison of Products of Inertia for the Rotor With and Without an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Product of Inertia, (10^{-4}) lb-ft-sec^2			Difference in Steady-State Product of Inertia, (10^{-4}) lb-ft-sec^2		
		I_{xy}	I_{xz}	I_{yz}	I_{xy}	I_{xz}	I_{yz}
15	0	+8.621	+2.559	+5.935	— ‡	-0.0382	+0.0463
	0.001	+0.00458	+1.579	+5.926	+0.00458	+1.444	+5.923
	0.1	+0.0000828	+1.579	+5.893	+0.0000828	+1.575	+5.893
50	0	+8.627	+8.627	+7.063	+8.111	+4.953	+7.063
	0.001	+8.002	+6.481	+6.492	+8.002	+6.481	+5.495
	0.1	+0.001209	+1.579	+5.893	+0.001209	+1.579	+5.893
200	0	+8.446	+8.058	+8.626	+4.251	+2.198	+8.342
	0.001	+8.594	+8.165	+8.233	+4.696	+2.477	+8.233
	0.1	+2.644	+2.121	+7.358	+2.644	+2.121	+7.358

‡ Indicates that no steady-state value could be determined.

Please note that in the table above, a positive (+) sign indicates that the magnitude of the product of inertia increased (worsened) compared with the unbalanced rotor. Similarly, a negative (-) sign indicates that the magnitude of the product of inertia decreased (improved) compared with the unbalanced rotor.

5.3.1.1 Summary of Findings

For the cases where $\omega_3 = 15$ rad/sec, when the Coulomb friction level is $\mu_s = 0$, the two balancing particles are unable to locate any steady-state position, and travel around the race together acting as a single particle. This gives a steady-state amplitude of displacement to the

geometric center of $0.6671(10^{-2})$ inches. This is a dramatic decrease in performance since the unbalanced rotor had no displacement of the geometric center. There is no need to discuss the effect on maximum displacement of the geometric center because the unbalanced rotor had no translational effects and therefore no displacement of the geometric center. It is only of interest to explore the steady-state displacement since the balancer should at least make improvements in the long-term behavior of the rotor, even if it worsens behavior in the transient period.

Once the Coulomb friction level is increased to $\mu_s = 0.001$, the two balancing particles come to rest relative to the rotor within a short period of time. Particle 1 stops within 0.3261 seconds and Particle 3 stops within 0.4116 seconds. Neither particle reaches its balanced position, so there is a steady-state amplitude for the geometric center of $0.4790(10^{-2})$ inches. While this is better than the case of no Coulomb friction and represents a 28.2% reduction, it is worse overall. When the Coulomb friction level is $\mu_s = 0.1$, the particles stop even faster needing 0.0242 seconds for Particle 1 and 0.0140 seconds for Particle 3. This gives a steady-state displacement of $0.4787(10^{-2})$ inches, a decrease of 28.2% when compared to the case when $\mu_s = 0$. The reduction in displacement cannot be attributed to any mechanism in the system that occurs when the rotor is operated below a translational natural frequency, but instead is owing to the fact that the chosen initial conditions allow the particles to stop sooner so they don't travel as far from the balanced positions to worsen performance. For the non-zero cases of Coulomb friction, the particles stop very close to their initial conditions with Particle 1 only moving 1.27° and 0.04° for $\mu_s = 0.001$ and $\mu_s = 0.1$, respectively; and Particle 3 moving only 1.33° and 0.04° for $\mu_s = 0.001$ and $\mu_s = 0.1$, respectively.

It is interesting to note that the steady-state angular displacements of the rotor about the inertial X-axis and the intermediate y_2-axis decrease slightly from $0.2164(10^{-2})$ degrees to $0.2135(10^{-2})$ degrees for the X-axis and $0.2134(10^{-2})$ degrees for the y_2-axis, a reduction of approximately 1.3%, when the Coulomb friction level is set to zero. This is most likely a result of adding mass to the system which slightly increases the terms in the inertia tensor. When the Coulomb friction level is set to $\mu_s = 0.001$, the steady-state angular displacements about the X- and y_2-axes are both $0.2930(10^{-2})$ degrees, an increase of 35.4% over the unbalanced values of $0.2164(10^{-2})$ degrees about each axis. When the Coulomb friction level is set to $\mu_s = 0.1$, the steady-state values increase to $0.2955(10^{-2})$ degrees about each axis, an increase of 36.6% over the unbalanced rotor values.

The unbalanced rotor features a maximum angular displacement about the X- and y_2-axes of $0.2165(10^{-2})$ degrees and $0.2164(10^{-2})$ degrees. Across the three Coulomb friction levels, the rotor with balancer sees an average maximum of $0.2941(10^{-2})$ degrees about the X-axis and $0.2937(10^{-2})$ degrees about the y_2-axis. This is an average increase of 35.8% about both the X- and y_2-axes, and is not unexpected because we have added two 1-oz. imbalances to the system. The maximum angular displacement observed took place when the Coulomb friction level was $\mu_s = 0.1$, and was $0.2956(10^{-2})$ degrees about both the X- and y_2-axes, an increase of 36.5% and 36.6% respectively.

The products of inertia for the system are not improved for any case. The unbalanced rotor features no I_{xy} or I_{yz} products of inertia, but the addition of balancing masses results in non-zero values for these terms. It appears that the value for I_{xz} is improved for the case where Coulomb friction is $\mu_s = 0$, but when one examines Figure 5.3.1.1-II, it shows that the value is simply oscillating about the unbalanced value of $8.627(10^{-4})$ lb-ft-sec^2. For this same friction value, I_{yz} is almost eliminated, but still exhibits an oscillatory behavior about the unbalanced level. I_{xy} is seen to exhibit large oscillations, reaching a maximum value of $8.621(10^{-4})$ lb-ft-sec^2. The oscillatory behavior is expected because the balancing particles are simply traveling around their respective races, changing the products of inertia in a continuous fashion.

When the Coulomb friction is increased to $\mu_s = 0.001$ or 0.1, the particles are seen to stop very quickly, so the products of inertia reach a steady-state value. I_{xy} is small for both cases becoming fixed at $-0.00458(10^{-4})$ lb-ft-sec^2 and $0.0000828(10^{-4})$ lb-ft-sec^2 respectively. These values are small, but still worse than the unbalanced rotor value of zero. Because the two particles were started on opposite sides of the body-fixed x-axis, their y-coordinates almost cancel when they come to rest so quickly. I_{yz} increases to $5.923(10^{-4})$ lb-ft-sec^2 and $5.893(10^{-4})$ lb-ft-sec^2 for the same Coulomb friction levels, which also represents a degradation in performance since the unbalanced rotor had $I_{yz} = 0$. I_{xz} becomes fixed at $10.070(10^{-4})$ lb-ft-sec^2 when $\mu_s = 0.001$, and $10.202(10^{-4})$ lb-ft-sec^2 when $\mu_s = 0.1$. These represent increases of 16.73% and 18.3% respectively from the unbalanced value of $8.627(10^{-4})$ lb-ft-sec^2.

The maximum values obtained for I_{xy} and I_{yz} are almost identical to their steady-state values when the Coulomb friction level is non-zero. These terms reach their largest value when there is no Coulomb friction present with I_{xy} reaching a maximum of $8.621(10^{-4})$ lb-ft-sec^2 and I_{yz} reaching a maximum of $5.935(10^{-4})$ lb-ft-sec^2. The same is true for I_{xz} where it reaches a

maximum of $11.186(10^{-4})$ lb-ft-sec^2. When the Coulomb friction is non-zero, I_{xz} reaches a maximum value of $10.205(10^{-4})$ lb-ft-sec^2, a decrease of 8.8% over the zero Coulomb friction level.

For the cases where $\omega_3 = 50$ rad/sec, we observe very different behavior when $\mu_s = 0$. For this case, the two balancing particles reach a steady-state separation of 180°, effectively cancelling their imbalance resulting in a zero steady-state amplitude of displacement for the geometric center. This is not surprising because we are above the translational natural frequency of the support and the mechanism to drive the balancing particles to their balanced positions is present. However, there is no tendency to drive the particles to a position which cancels the effect of the two dynamically imbalanced masses. The balancing particles reach a 1° envelope about their steady-state value after 28.7525 seconds, but are able to effectively eliminate any motion of the geometric center after 4.8 seconds when they reach positions roughly 180° opposite one another. There is a period of their motion where they move around the race at the same speed maintaining a 180° angle of separation.

When the Coulomb friction is increased to $\mu_s = 0.001$, Particle 1 stops within 0.0706 seconds and Particle 3 stops in 2.3726 seconds. This short time interval does not allow the particles to move to opposing positions and results in a steady-state displacement of the geometric center of $0.4790(10^{-2})$ inches. A similar trend occurs when $\mu_s = 0.1$, where Particles 1 and 3 stop in 0.0156 seconds and 0.1518 seconds, respectively, resulting in a steady-state amplitude of the geometric center of $0.4787(10^{-2})$ inches.

The steady-state amplitudes for both angular displacements about the inertial X- and intermediate y_2-axes are worsened in all cases indicating that there is no improvement in the products of inertia for the rotor. When $\mu_s = 0$, the steady-state amplitudes are $3.417(10^{-2})$ degrees and $3.412(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 77.1% and 77.2% respectively from the unbalanced values of $1.929(10^{-2})$ degrees about the X-axis and $1.926(10^{-2})$ degrees about the y_2-axis. These differences increase as the Coulomb friction level increases. When $\mu_s = 0.001$, the values are $3.594(10^{-2})$ degrees and $3.589(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 86.3% from the unbalanced values. When $\mu_s = 0.1$, the values are $3.626(10^{-2})$ degrees and $3.622(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 88.0% from the unbalanced values.

The maximum amplitudes for the angular displacements also increase in every case, but are seen to decrease as the Coulomb friction level increases. When $\mu_s = 0$, the maximum values are $3.849(10^{-2})$ degrees about the X-axis and $3.844(10^{-2})$ degrees about the y_2-axis, representing increases of 98.9% and 99.3% respectively from the unbalanced values of $1.935(10^{-2})$ degrees for the X-axis and $1.929(10^{-2})$ degrees about the y_2-axis. When $\mu_s = 0.001$, the maximum values are $3.598(10^{-2})$ degrees about the X-axis and $3.593(10^{-2})$ degrees about the y_2-axis, representing increases of 85.9% and 86.3% respectively from the unbalanced values. Once $\mu_s = 0.1$ there is a marked reduction where the maximum value is $2.662(10^{-2})$ degrees about the X-axis and $2.635(10^{-2})$ degrees about the y_2-axis, representing increases of 37.6% and 36.6% respectively from the unbalanced values. It is no surprise that the changes are similar for both axes because the support is nearly isotropic in the plane of the rotor, and the overall angular displacement about intermediate axes is small. Additionally, the particles were released at 30° and 300° from the body-fixed x-axis, which places a line of symmetry between the particles at 15° off the body-fixed x-axis in the fourth quadrant. This renders the system close to symmetric with respect to the body-fixed x-axis when the motion begins.

The products of inertia are worsened for all values of Coulomb friction, but the effect is most pronounced when $\mu_s = 0$. At this friction level, I_{xy} reaches a steady state value of $-8.111(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a value of $-7.063(10^{-4})$ lb-ft-sec^2. These are much worse than their unbalanced values of zero. At this friction level, these products of inertia have maximum values of $-8.627(10^{-4})$ lb-ft-sec^2 and $-7.063(10^{-4})$ lb-ft-sec^2 for I_{xy} and I_{yz} respectively. For the same friction level, I_{xz} reaches a steady-state value of $13.580(10^{-4})$ lb-ft-sec^2 representing a 57.4% increase over the unbalanced value. The maximum value of I_{xz} is $17.253(10^{-4})$ lb-ft-sec^2 which is an increase of 100.0% over the unbalanced rotor. Because the mass has been effectively doubled as viewed with respect to the body-fixed z-axis, this is not an unexpected result.

When $\mu_s = 0.001$, I_{xy} reaches a steady state value of $8.002(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a value of $5.495(10^{-4})$ lb-ft-sec^2. These are still worse than their unbalanced values of zero, but represent decreases of 1.3% and 22.2% over the zero Coulomb friction values at the same operating speed. At this Coulomb friction level, these products of inertia have maximum values of $8.002(10^{-4})$ lb-ft-sec^2 and $6.492(10^{-4})$ lb-ft-sec^2 for I_{xy} and I_{yz} respectively. These are reductions of 7.2% and 8.1% from the zero Coulomb friction values. I_{xz} reaches a maximum and

steady-state value of $15.107(10^{-4})$ lb-ft-sec^2 representing a 75.1% increase over the unbalanced values.

When $\mu_s = 0.1$, I_{xy} reaches a maximum and steady state value of $0.001209(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a maximum and steady-state value of $5.893(10^{-4})$ lb-ft-sec^2. This is a significant improvement for I_{xy}, because it is almost eliminated. For I_{yz}, this is a decrease of 16.6% over the zero Coulomb friction value. I_{xz} reaches a maximum and steady-state value of $10.206(10^{-4})$ lb-ft-sec^2 representing an 18.3% increase over the unbalanced values, which is an improvement over the value when $\mu_s = 0.001$, but worse overall.

For the cases where $\omega_3 = 200$ rad/sec, we see results similar to the 50 rad/sec cases. When Coulomb friction is zero, the two balancing particles reach a steady-state where they are 180° apart, effectively cancelling their imbalance resulting in a zero steady-state amplitude of displacement for the geometric center. We again see no tendency to drive the particles to positions which cancel the effect of the two dynamically imbalanced masses. The balancing particles reach a 1° envelope about their steady-state value after 17.2056 seconds, but are able to effectively eliminate any motion of the geometric center after approximately 3.75 seconds when they reach positions roughly 180° opposite one another. There is no clear period of their motion where they move around the race at the same speed maintaining a 180° angle of separation as was seen in the 50 rad/sec case when synchronous motion was obvious.

When the friction is increased to $\mu_s = 0.001$, Particle 1 stops within 0.7215 seconds and Particle 3 stops in 7.4701 seconds. This short time interval does not allow the particles to move to opposing positions and results in a steady-state displacement of the geometric center of $0.2037(10^{-2})$ inches. This is an improvement over the 50 rad/sec case representing a 50.6% reduction from the steady-state amplitude at the same Coulomb friction level. It is interesting to note that similar to previous simulations of a statically imbalanced rotor, the particles are able to move for longer periods of time at higher operating speeds in the presence of increased friction.

A similar trend occurs when $\mu_s = 0.1$, where particles 1 and 3 stop in 10.958 seconds and 11.555 seconds, respectively, resulting in a steady-state amplitude of the geometric center of $1.145(10^{-2})$ inches. This is a reduction of 32.0% from the same Coulomb friction level at an operating speed of 50 rad/sec. Again, we note the significant difference in stopping times for the particles at higher operating speeds.

The steady-state amplitudes for both angular displacements about the inertial X- and intermediate y_2-axes are worsened in all cases but are smaller increases as compared with the 50 rad/sec simulations. When $\mu_s = 0$, the steady-state amplitudes are $15.071(10^{-2})$ degrees and $15.068(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 51.8% from the unbalanced values of $9.9283(10^{-2})$ degrees about the X-axis and $9.9263(10^{-2})$ degrees about the y_2-axis. When $\mu_s = 0.001$, the values are $15.912(10^{-2})$ degrees and $15.909(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 60.3% from the unbalanced values. When $\mu_s = 0.1$, the values are $14.996(10^{-2})$ degrees and $14.992(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 51.0% from the unbalanced values.

The maximum amplitudes for the angular displacements tend to decrease as the friction level increases, but tend to be smaller percentage increases as compared with the 50 rad/sec simulations. When $\mu_s = 0$, the maximum values are $18.909(10^{-2})$ degrees about the X-axis and $18.495(10^{-2})$ degrees about the y_2-axis, representing increases of 72.0% and 58.1% respectively from the unbalanced values of $10.992(10^{-2})$ degrees for the X-axis and $11.699(10^{-2})$ degrees about the y_2-axis. When $\mu_s = 0.001$, the maximum values are $18.867(10^{-2})$ degrees about the X-axis and $19.159(10^{-2})$ degrees about the y_2-axis, representing increases of 71.6% and 63.8% respectively from the unbalanced values. Once $\mu_s = 0.1$ there is a marked reduction where the maximum value is $15.050(10^{-2})$ degrees about the X-axis and $15.770(10^{-2})$ degrees about the y_2-axis, representing increases of 36.9% and 34.8% respectively from the unbalanced values. Overall, the maximum amplitudes about the X-axis in the presence of a balancer increase an average of 1046% from $0.2941(10^{-2})$ degrees at 15 rad/sec to $3.3697(10^{-2})$ degrees at 50 rad/sec; and increase an average of 423% to $17.6087(10^{-2})$ degrees from 50 rad/sec to 200 rad/sec. For y_2, the situation is similar with increases of 1043% from $0.2937(10^{-2})$ degrees at 15 rad/sec to $3.3573(10^{-2})$ degrees at 50 rad/sec; and increase an average of 430% to $17.808(10^{-2})$ degrees from 50 rad/sec to 200 rad/sec.

The products of inertia are worsened for all values of Coulomb friction, but the effect is most pronounced when $\mu_s = 0$. At this friction level, I_{xy} reaches a steady state value of $-4.251(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a value of $-8.342(10^{-4})$ lb-ft-sec^2. These are again worse than their unbalanced values of zero. At this Coulomb friction level, these products of inertia have maximum values of $8.446\ (10^{-4})$ lb-ft-sec^2 and $-8.626\ (10^{-4})$ lb-ft-sec^2 for I_{xy} and I_{yz} respectively. For the same friction level, I_{xz} reaches a steady-state value of $10.825(10^{-4})$

lb-ft-sec^2 representing a 25.5% increase over the unbalanced value. This is a marked improvement over the same friction level at 50 rad/sec. The maximum value of I_{xz} is 16.685(10^{-4}) lb-ft-sec^2, which is an increase of 93.4% over the unbalanced rotor. Again, because the mass has been effectively doubled as viewed with respect to the body-fixed z-axis, this is not an unexpected result.

When $\mu_s = 0.001$, I_{xy} reaches a steady state value of 4.696(10^{-4}) lb-ft-sec^2 while I_{yz} reaches a value of 8.233(10^{-4}) lb-ft-sec^2. At this Coulomb friction level, these products of inertia have maximum values of -8.594(10^{-4}) lb-ft-sec^2 and -8.233(10^{-4}) lb-ft-sec^2 for I_{xy} and I_{yz} respectively. I_{xz} reaches a steady-state value of 11.104(10^{-4}) lb-ft-sec^2 representing a 28.7% increase over the unbalanced value of 8.627(10^{-4}) lb-ft-sec^2, but a 26.4% reduction over the value at the same friction level with an operating speed of 50 rad/sec. The maximum amplitude of I_{xz} is 16.791(10^{-4}) lb-ft-sec^2 which is an increase of 94.6% over the unbalanced value.

When $\mu_s = 0.1$, I_{xy} reaches a maximum and steady state value of 2.644(10^{-4}) lb-ft-sec^2 while I_{yz} reaches a maximum and steady-state value of 7.358(10^{-4}) lb-ft-sec^2. These are much worse than the values obtained at 50 rad/sec for the same friction levels, where I_{xy} has increased from approximately zero, and I_{yz} has increased 24.9%. I_{xz} reaches a maximum and steady-state value of 10.748(10^{-4}) lb-ft-sec^2 representing a 24.6% increase over the unbalanced values, which is an improvement over the previous friction levels, but worse overall.

In general, we find that the particles have no affinity for selecting balancing positions opposite the dynamically imbalanced pair of masses. The balancing particles instead focus on each other and at zero Coulomb friction levels will move to positions 180° apart to cancel their collective effect if the rotor is operated above the translational natural frequency of the support in the plane of the rotor. This leaves the wheel dynamically unbalanced, but statically balanced. If the Coulomb friction levels are increased only slightly, the situation is exacerbated because the particles are not allowed to completely reach a balanced state and the rotor has a combined static and dynamic imbalance which worsens the rotor performance in all six degrees of freedom. There tends to be an improvement in particle behavior as the operating speed is increased and the Coulomb friction level is increased, but it is difficult to tell if this is a general occurrence or a happenstance based on initial conditions and system properties. Additionally, we see the same phenomenon in the dynamic imbalance cases as was present in the static imbalance cases, where particles at very high operating speeds and high Coulomb friction levels can move for longer

periods of time than at lower operating speeds. This seems counterintuitive, but the trend has been observed in numerous cases and the results are consistent across the operating speeds.

5.3.1.2 Plots of Rotor Performance and Balancing Particle Behavior

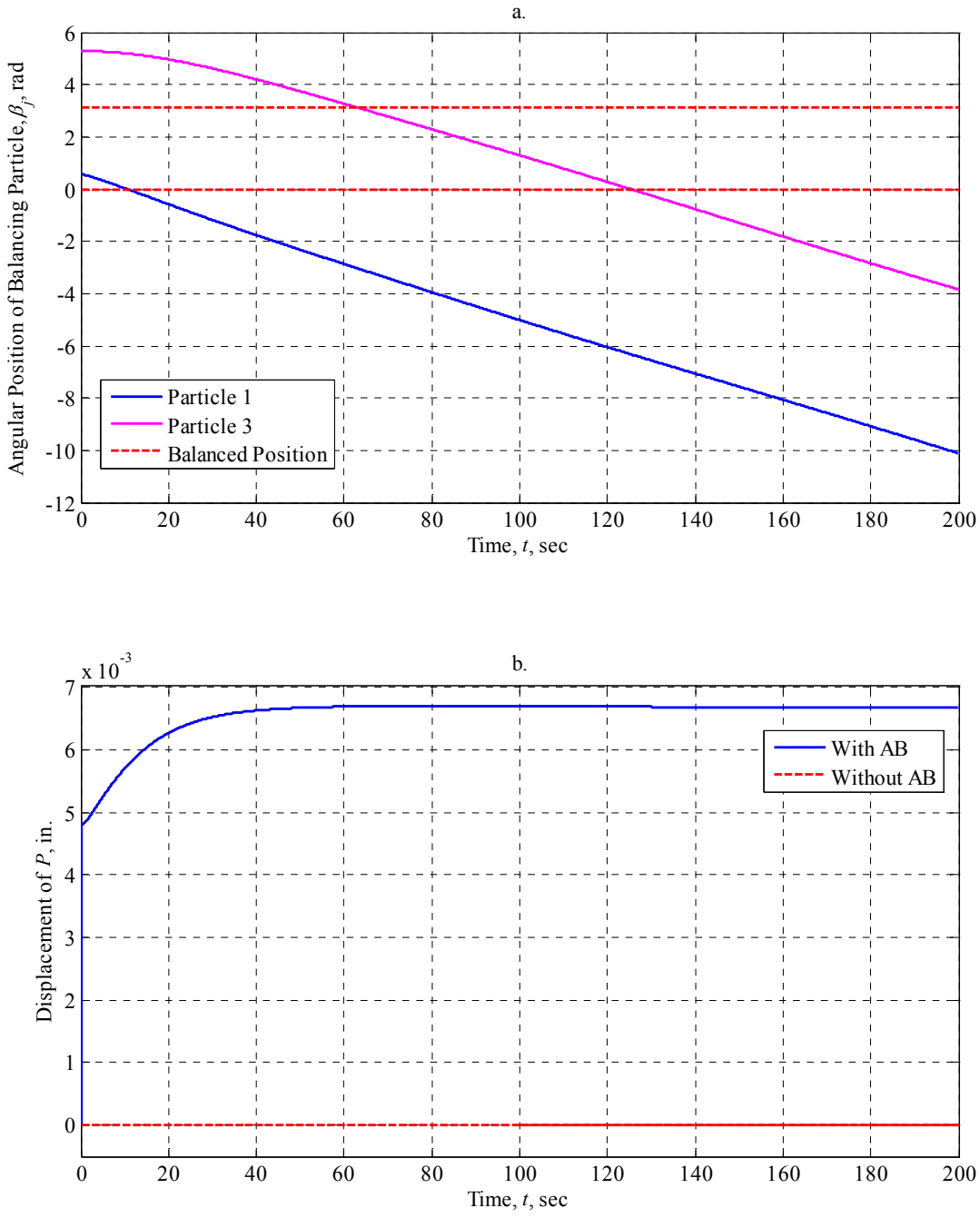

Figure 5.3.1.2.1-I: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

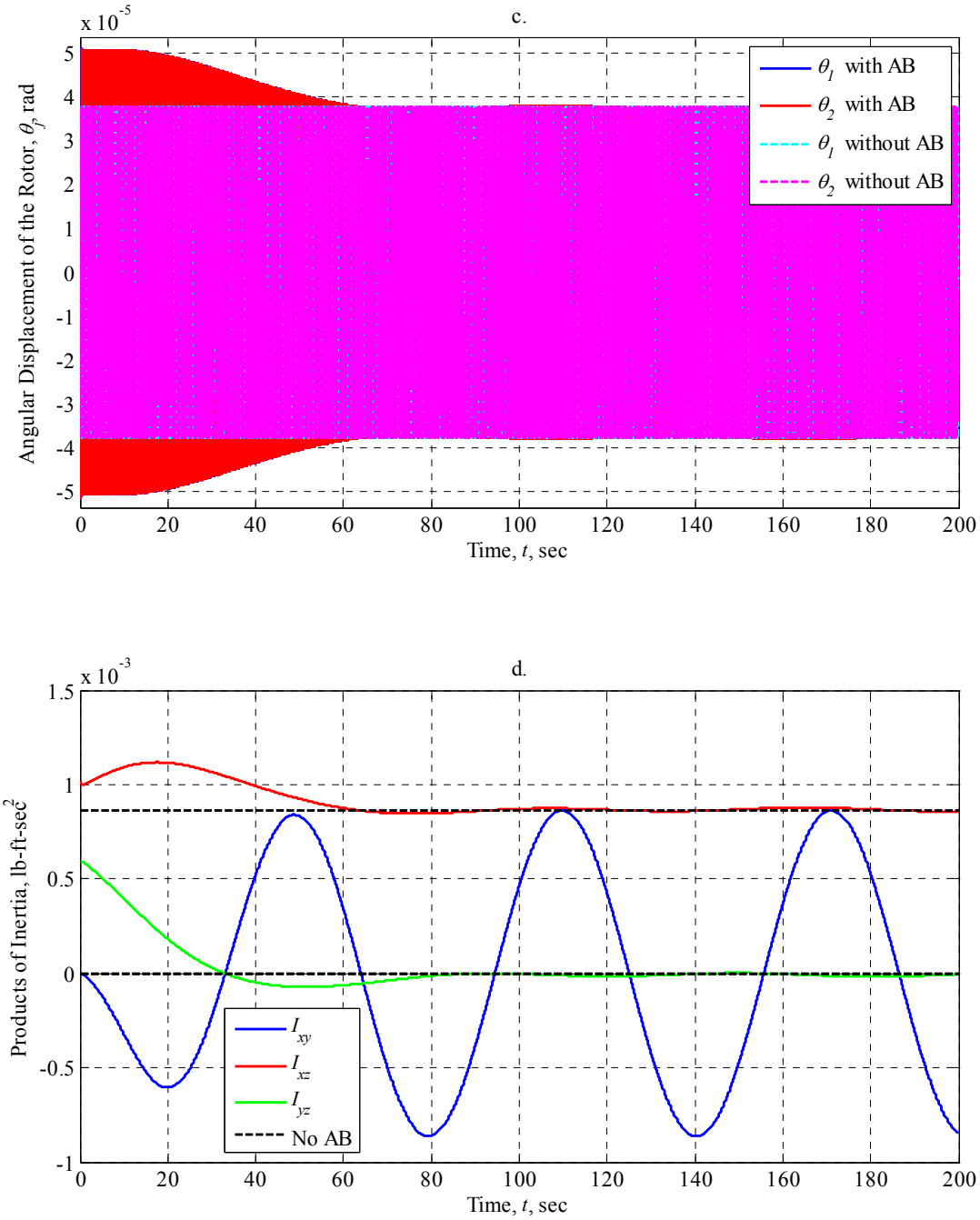

Figure 5.3.1.2.1-II: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

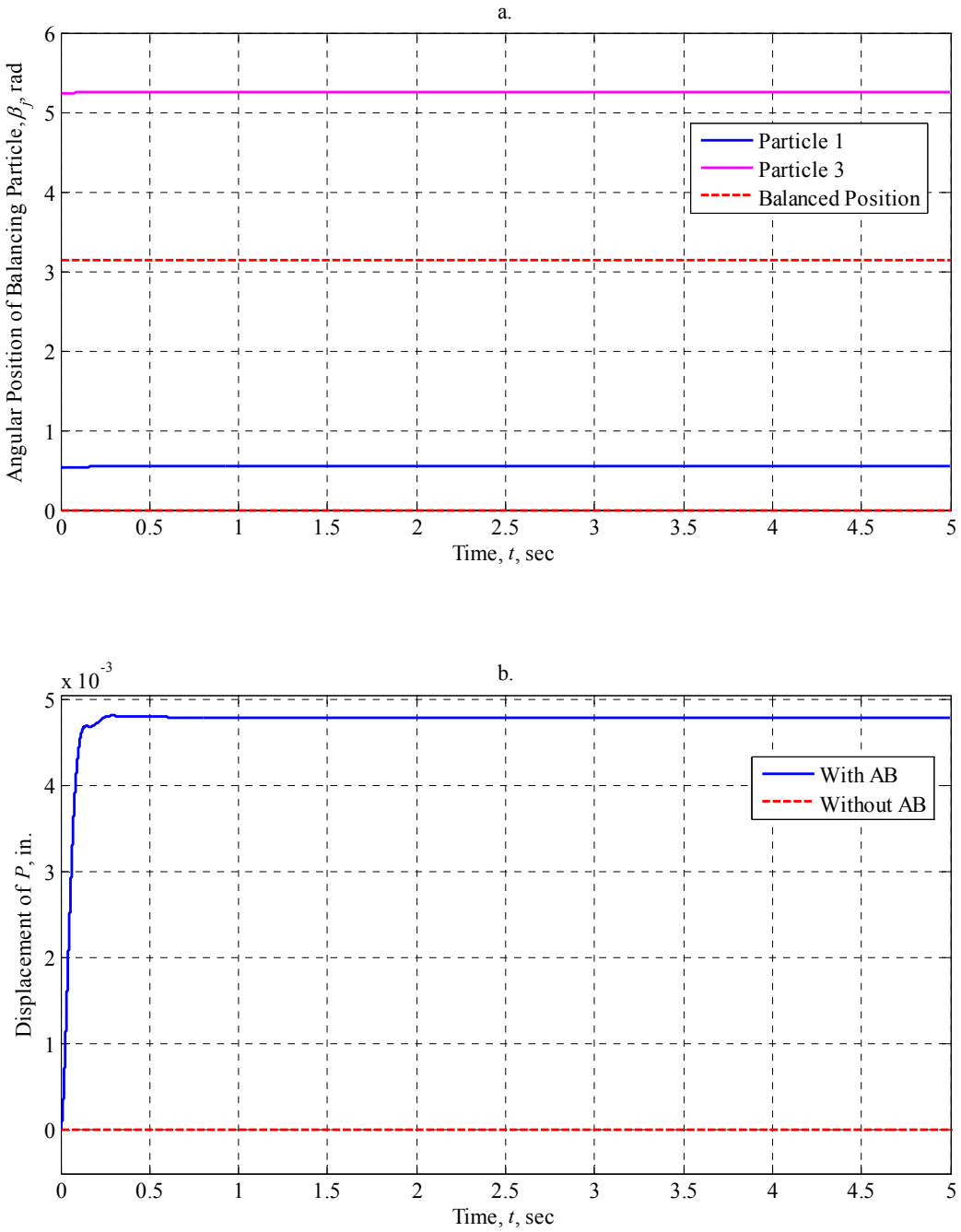

Figure 5.3.1.2.2-I: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

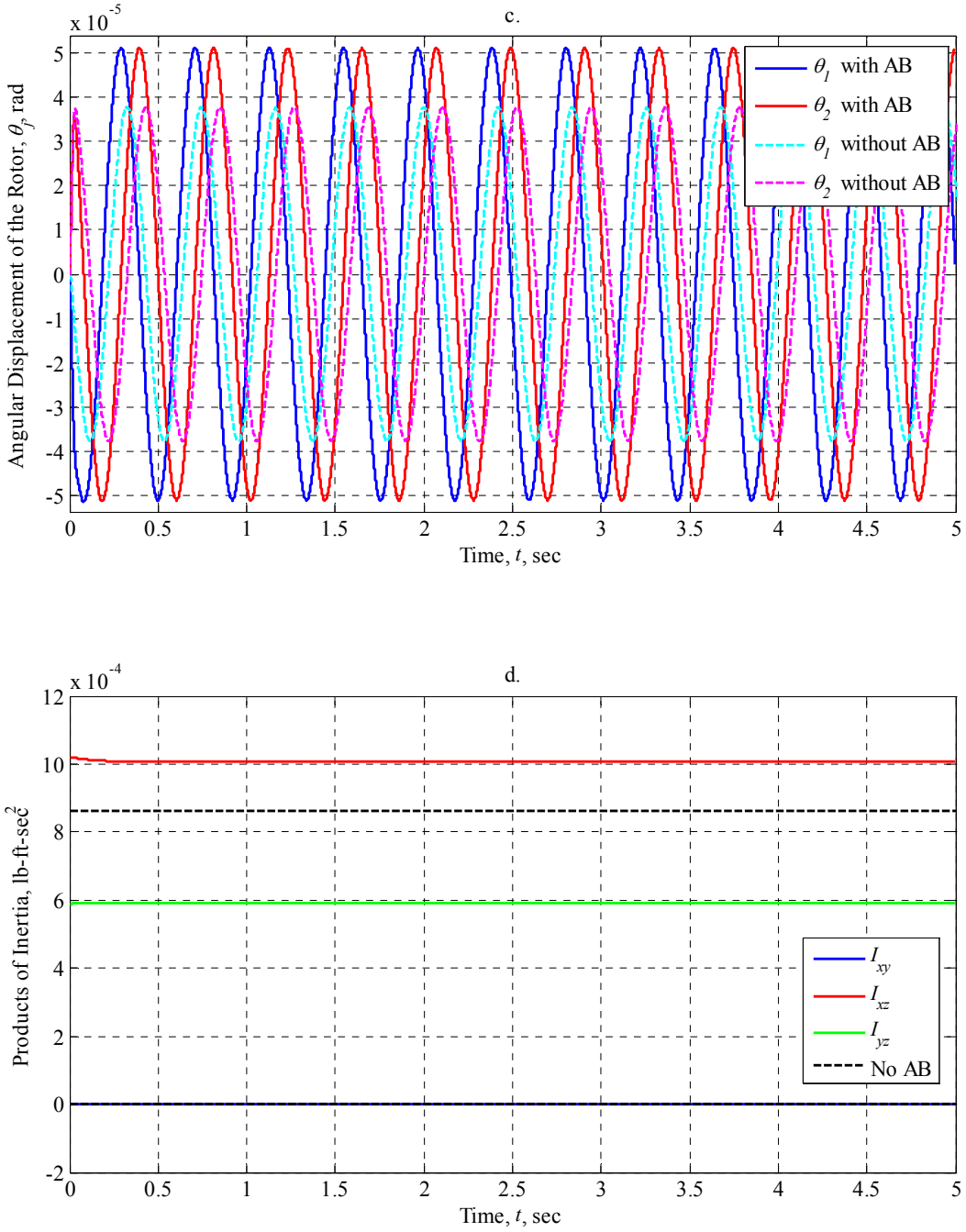

Figure 5.3.1.2.2-II: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

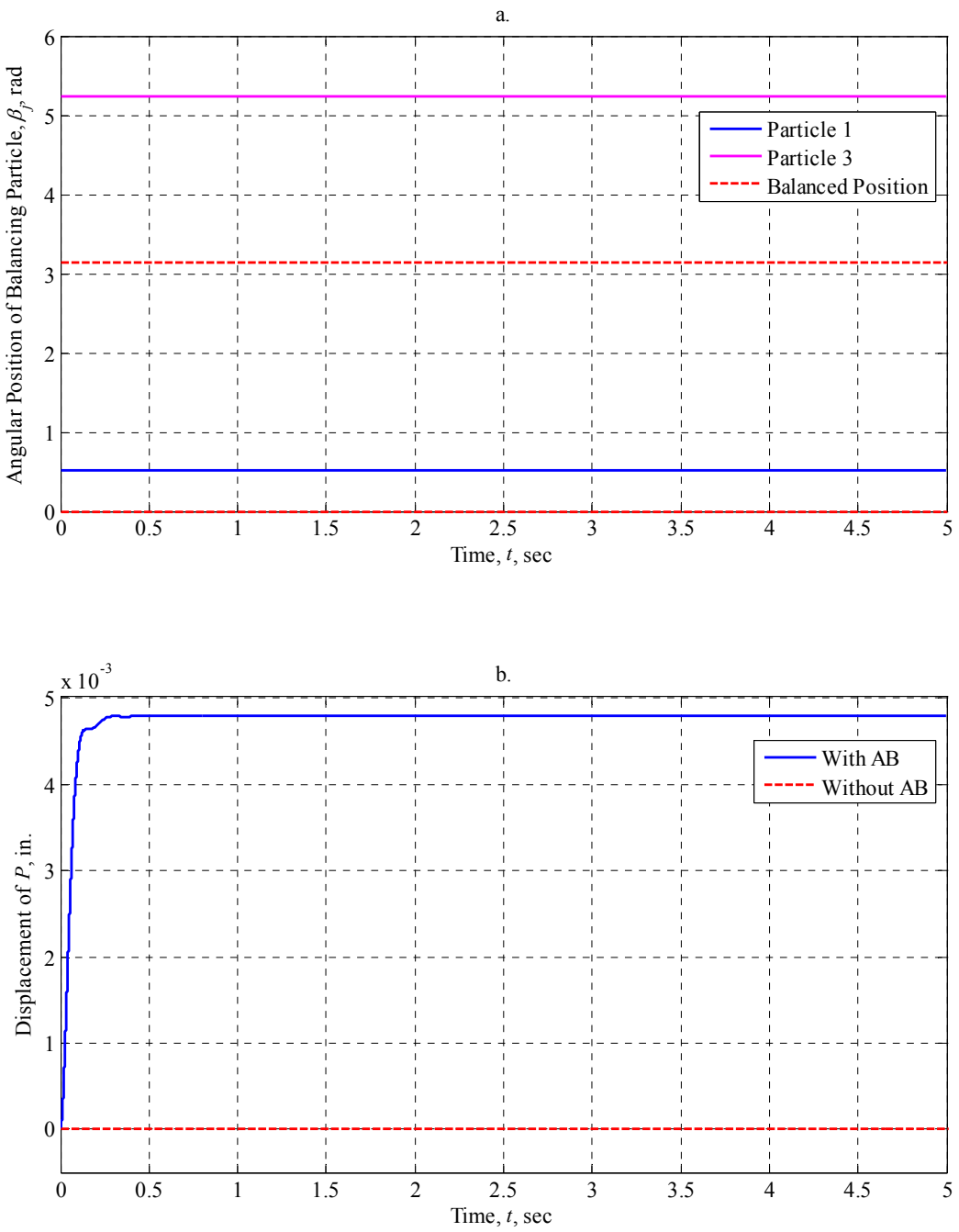

Figure 5.3.1.2.3-I: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

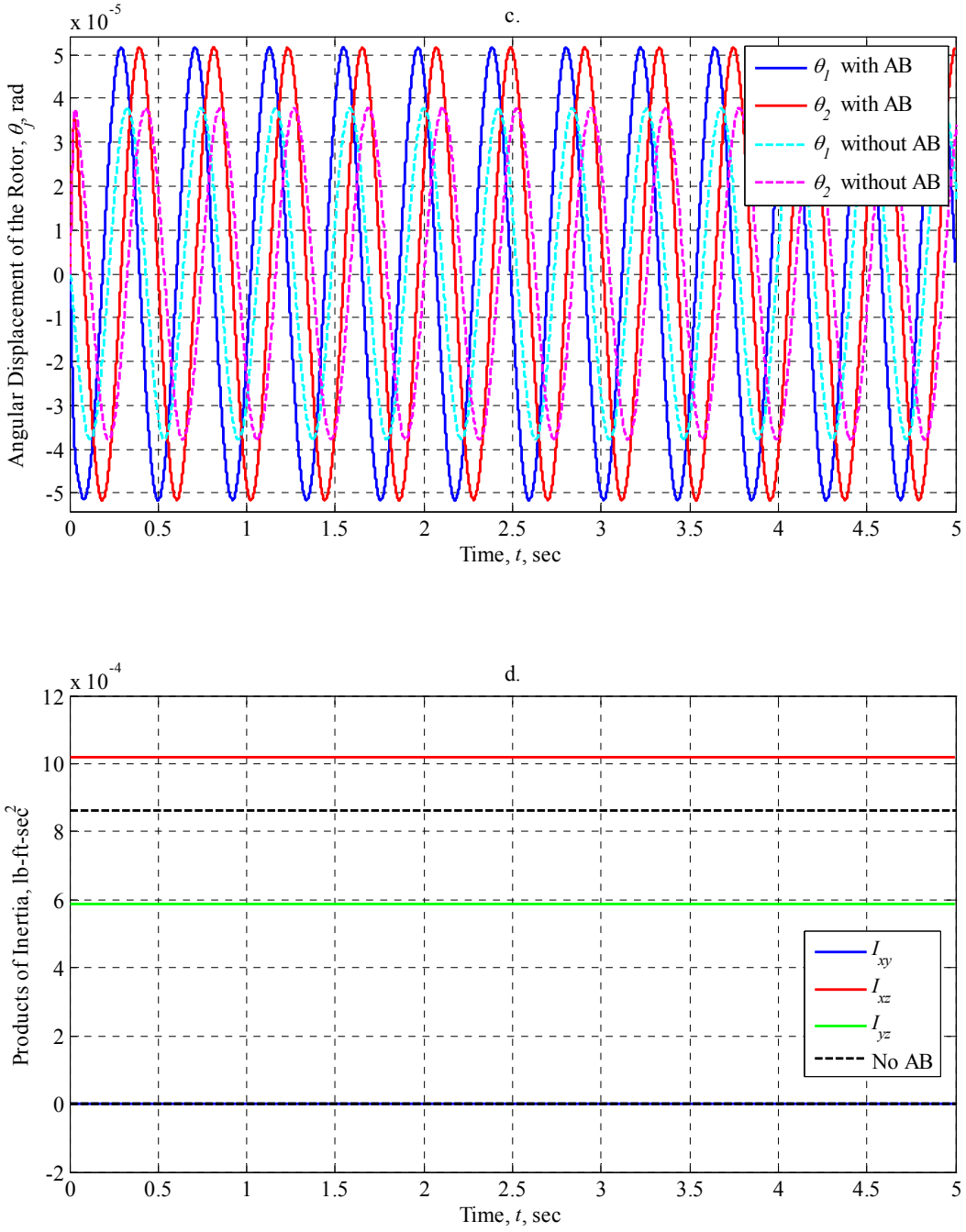

Figure 5.3.1.2.3-II: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

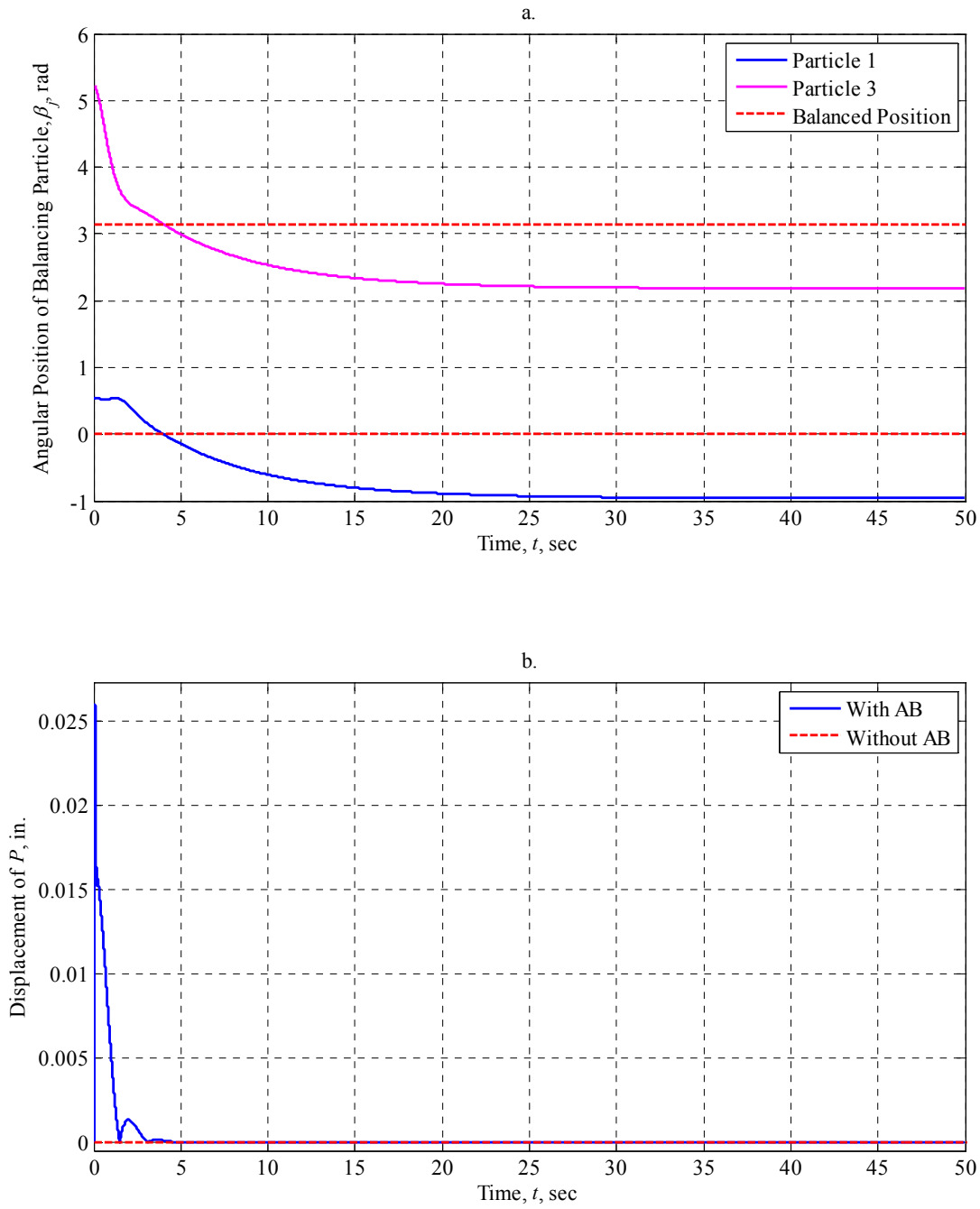

Figure 5.3.1.2.4-I: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

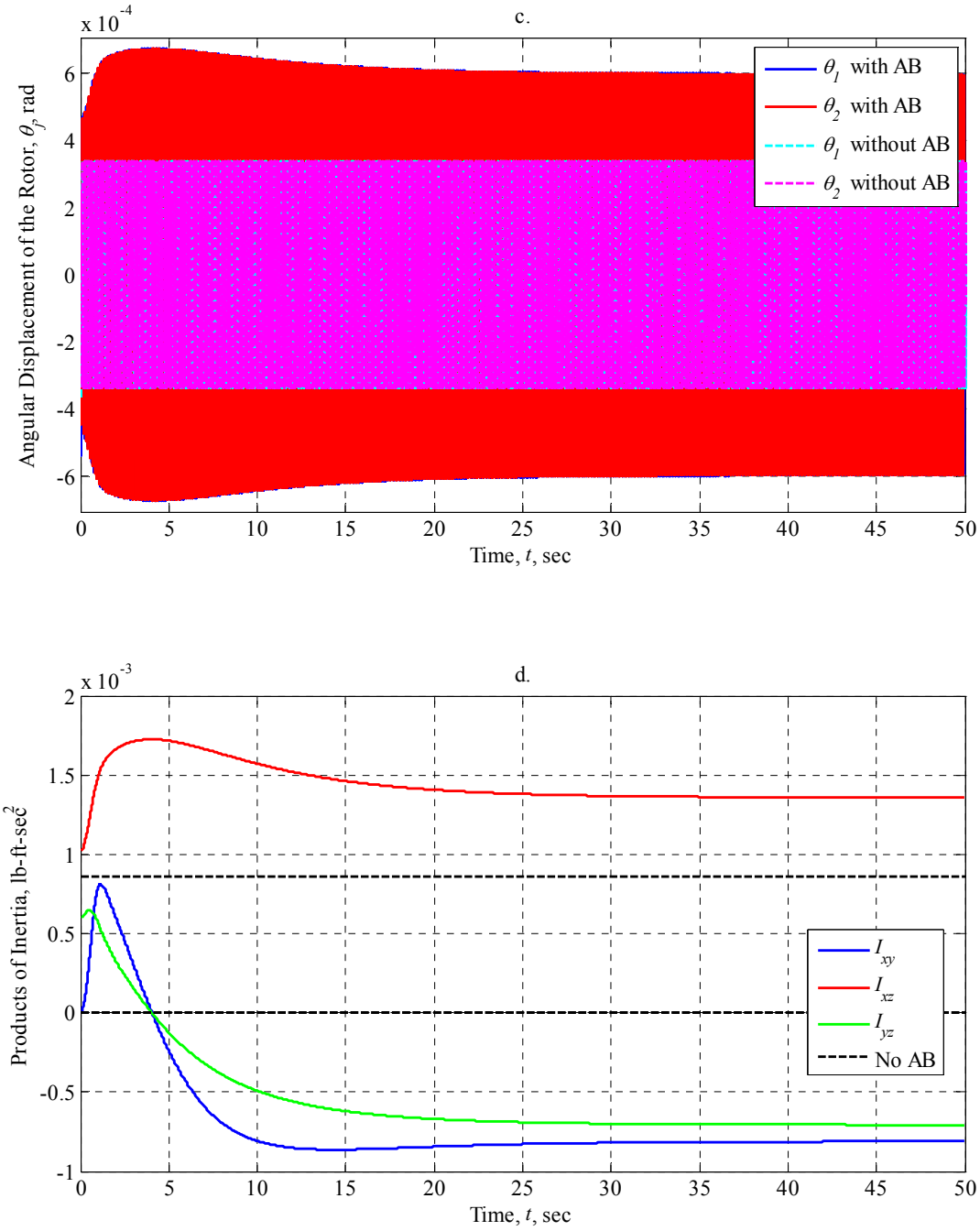

Figure 5.3.1.2.4-II: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

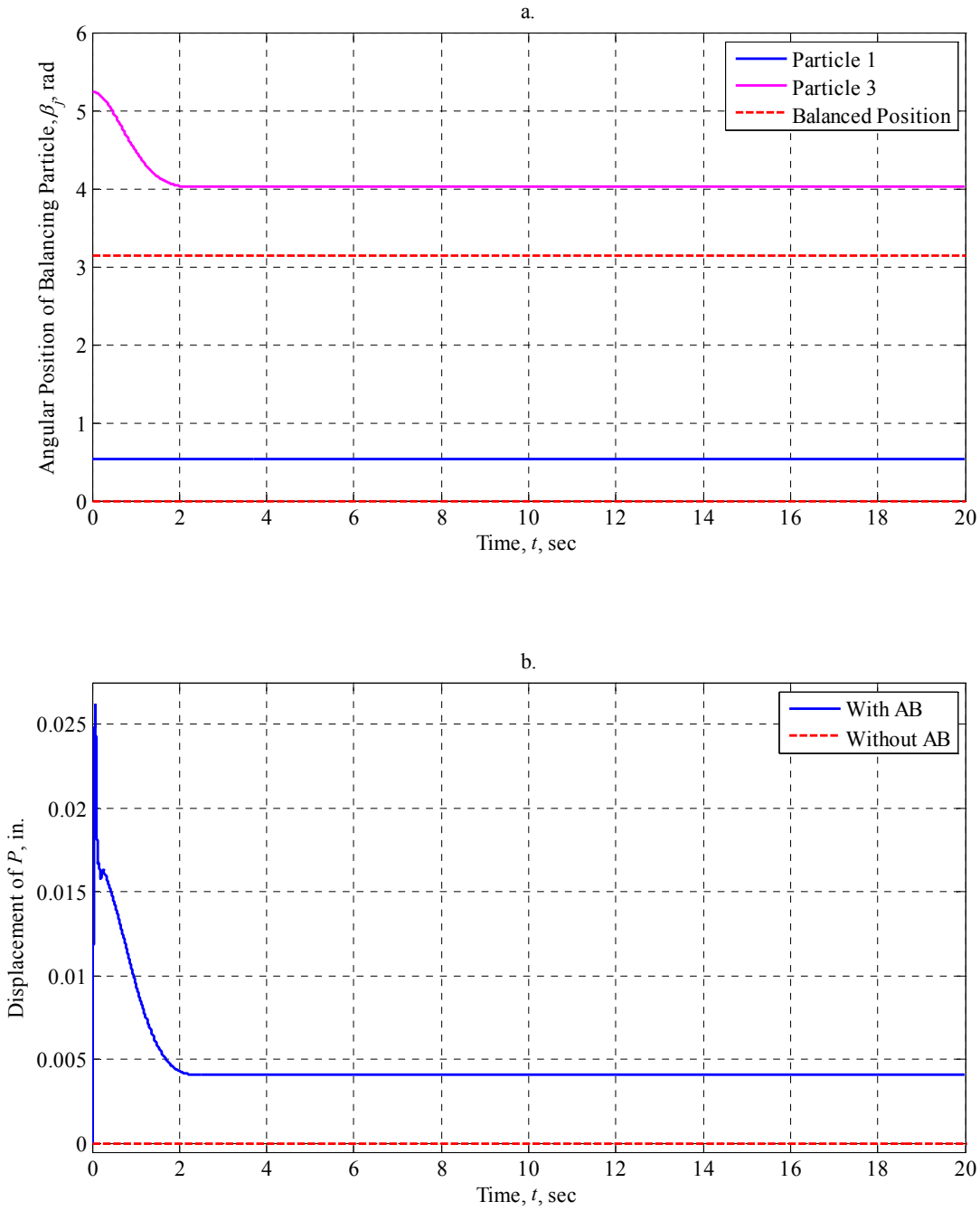

Figure 5.3.1.2.5-I: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

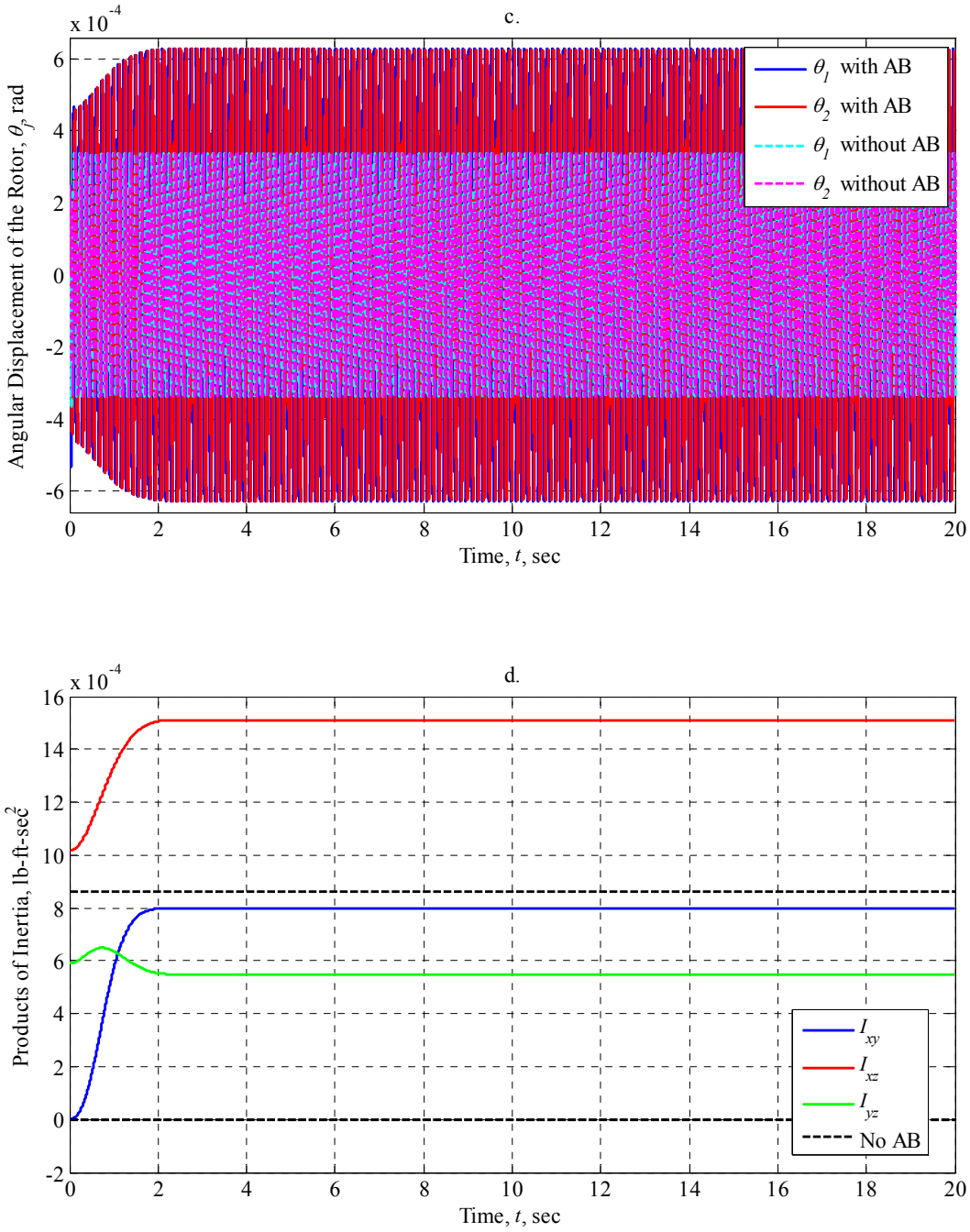

Figure 5.3.1.2.5-II: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

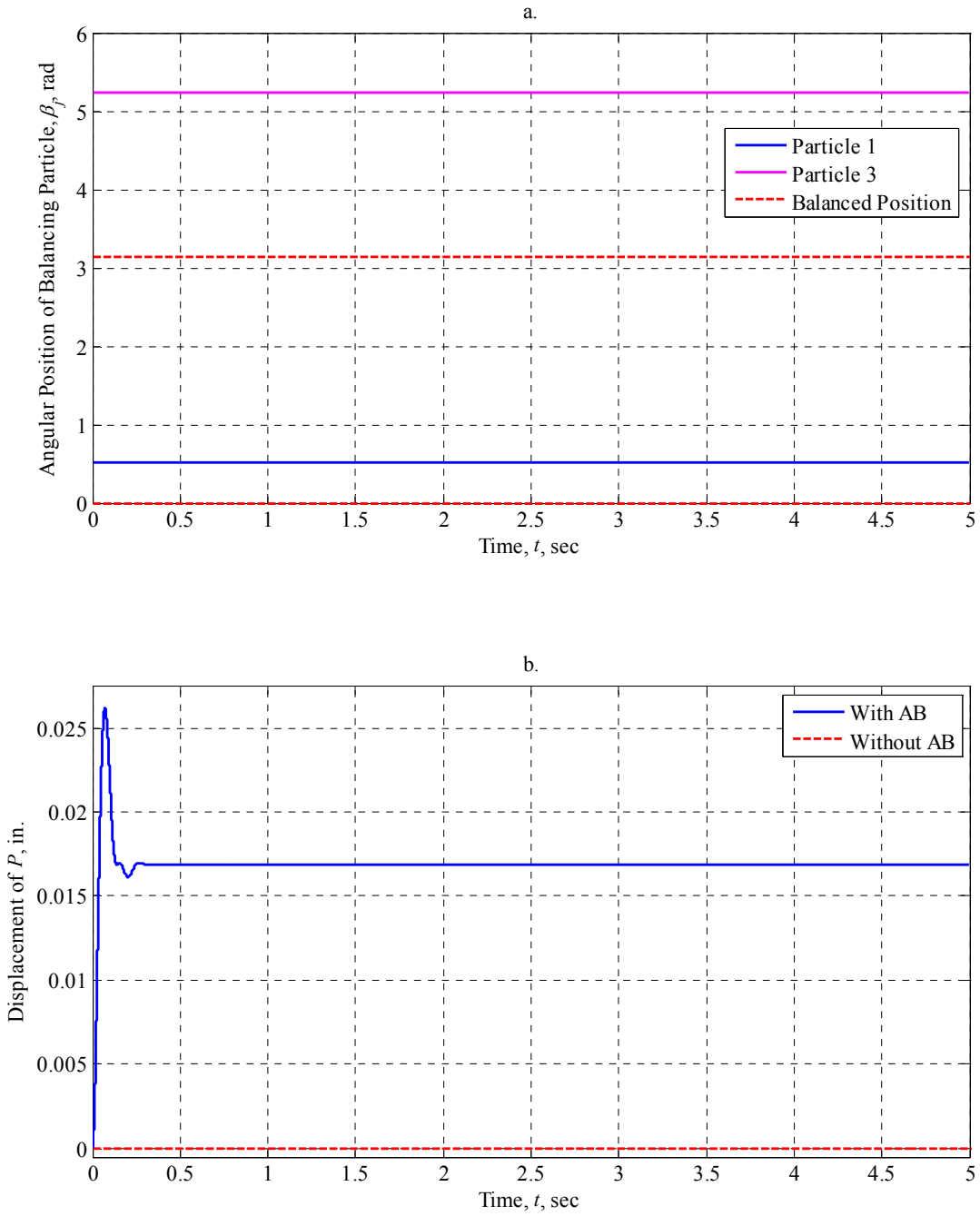

Figure 5.3.1.2.6-I: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

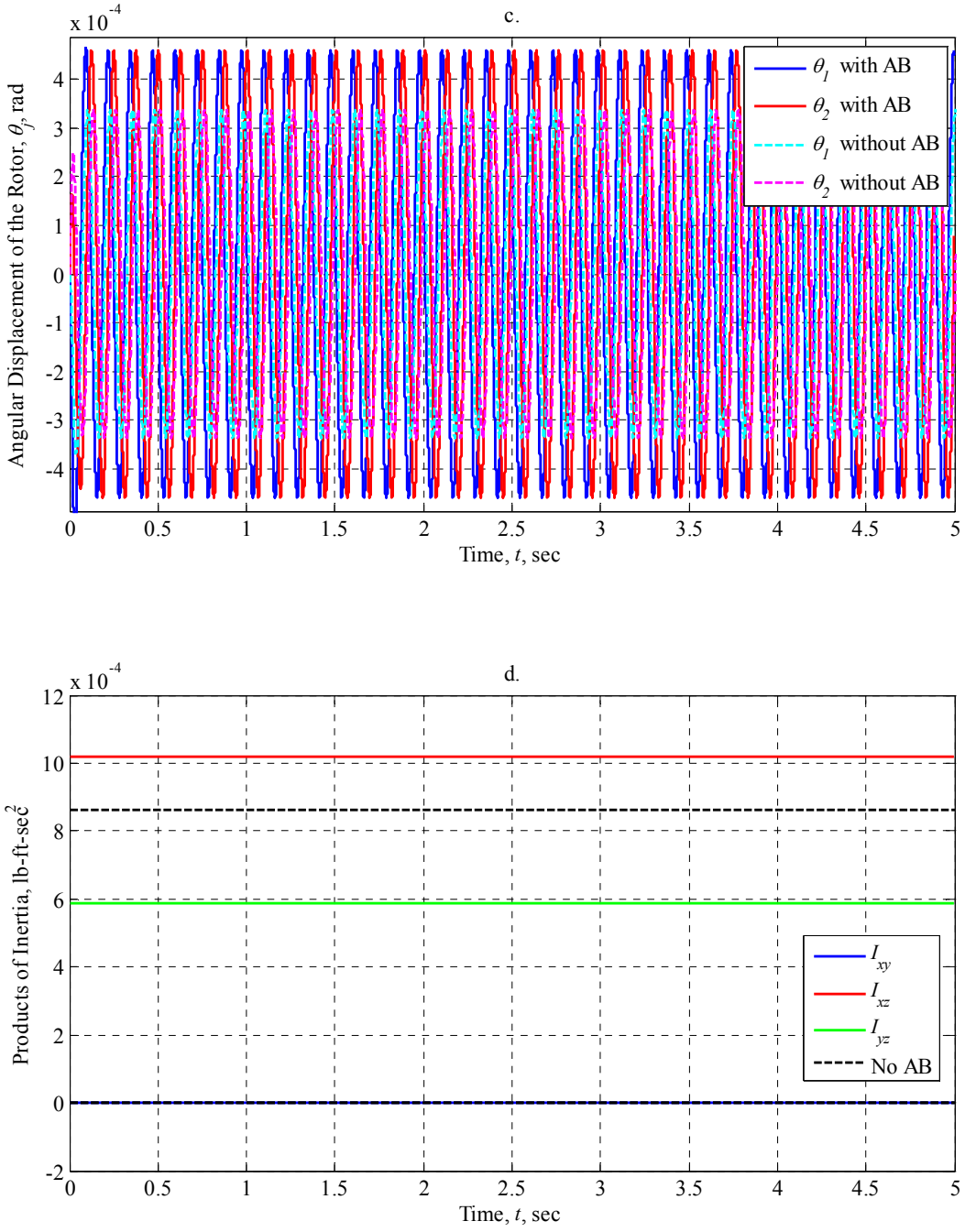

Figure 5.3.1.2.6-II: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

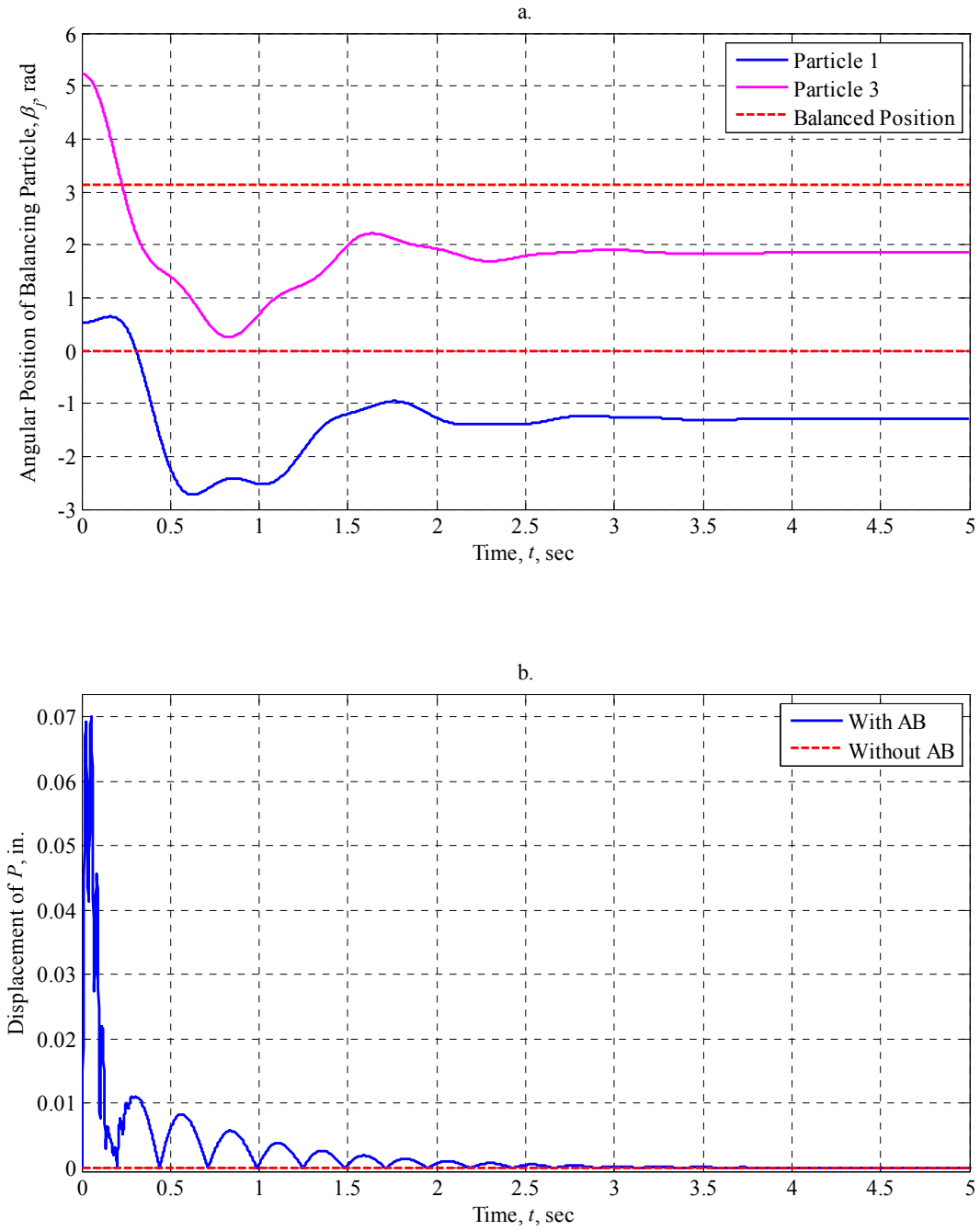

Figure 5.3.1.2.7-I: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

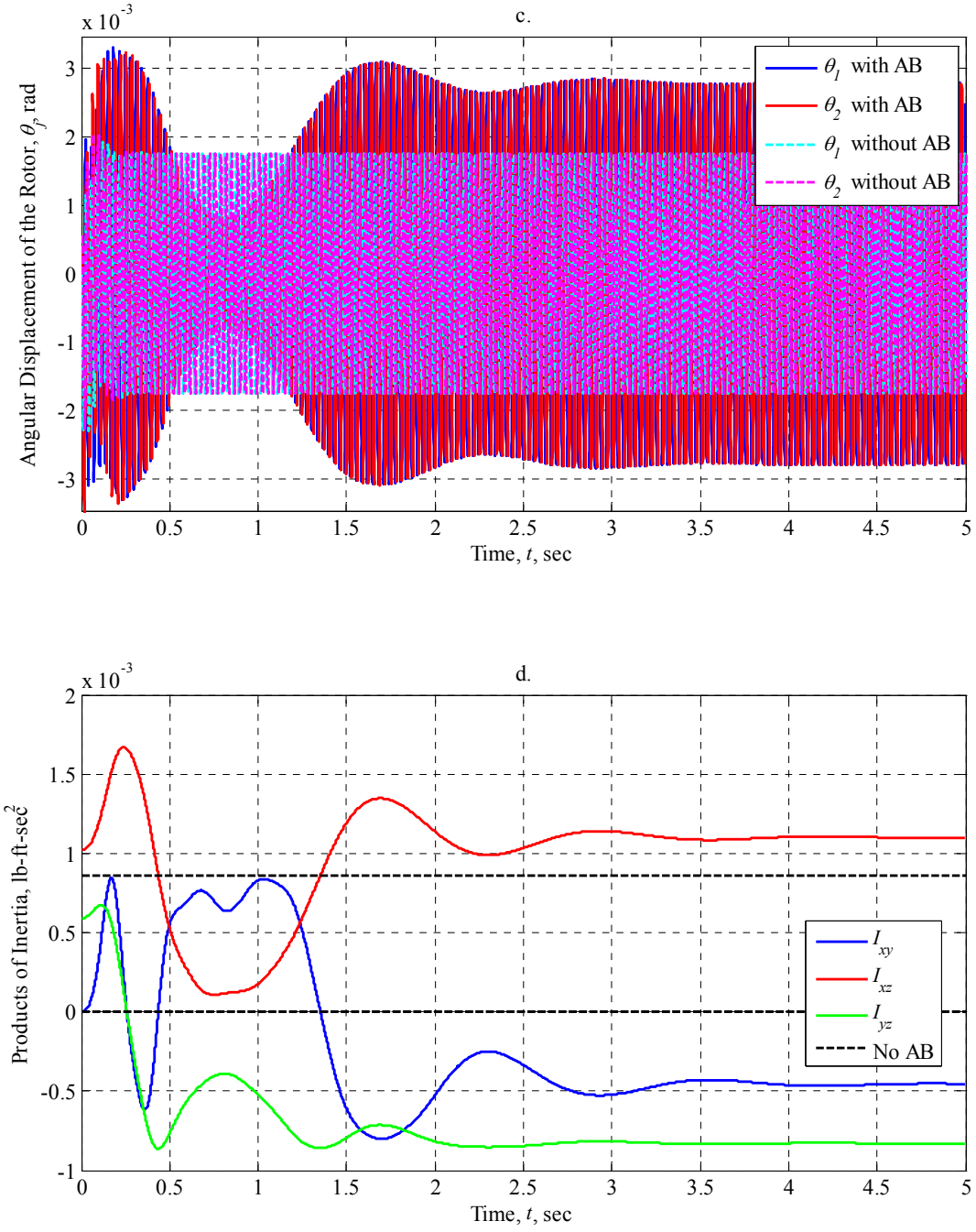

Figure 5.3.1.2.7-II: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

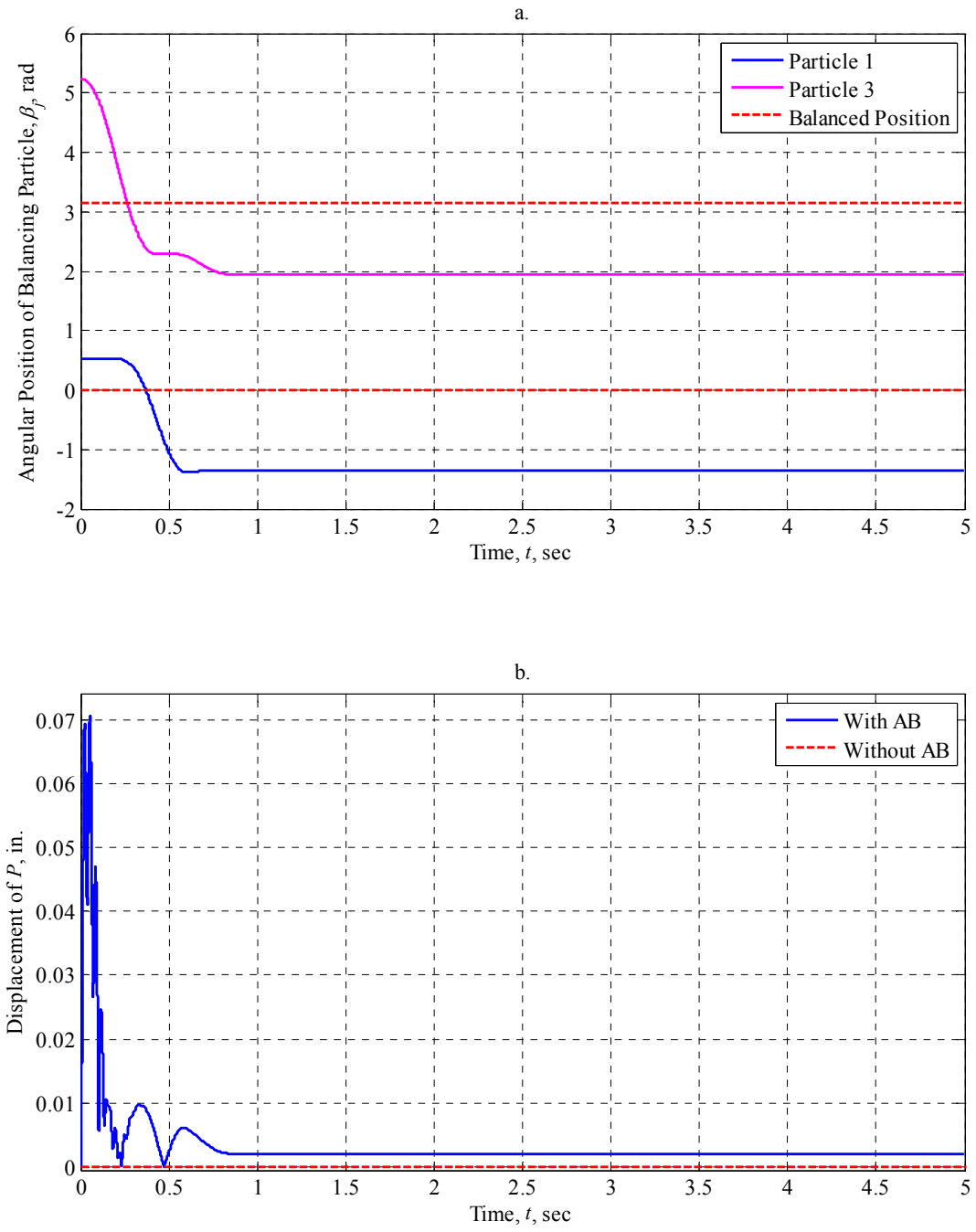

Figure 5.3.1.2.8-I: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

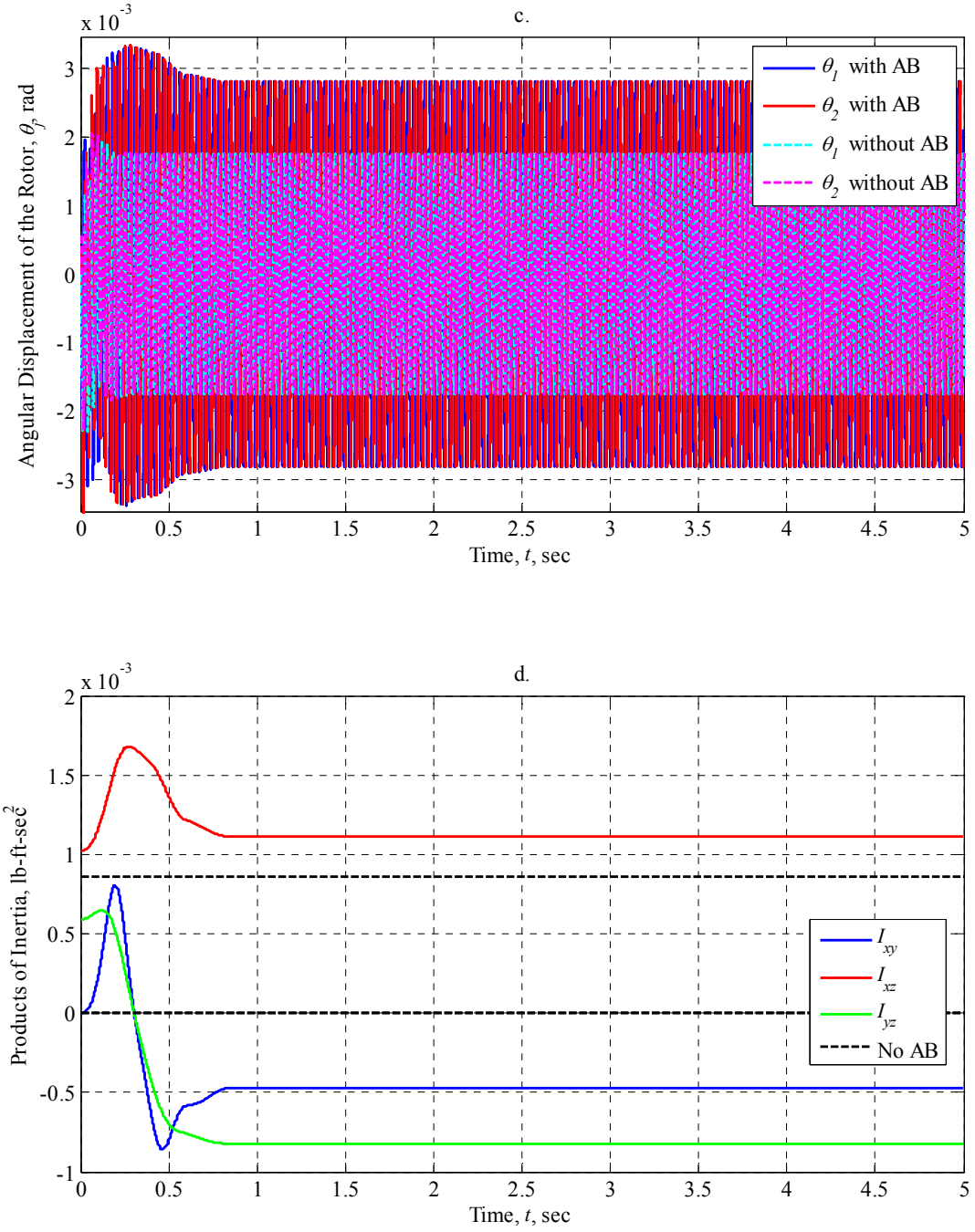

Figure 5.3.1.2.8-II: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

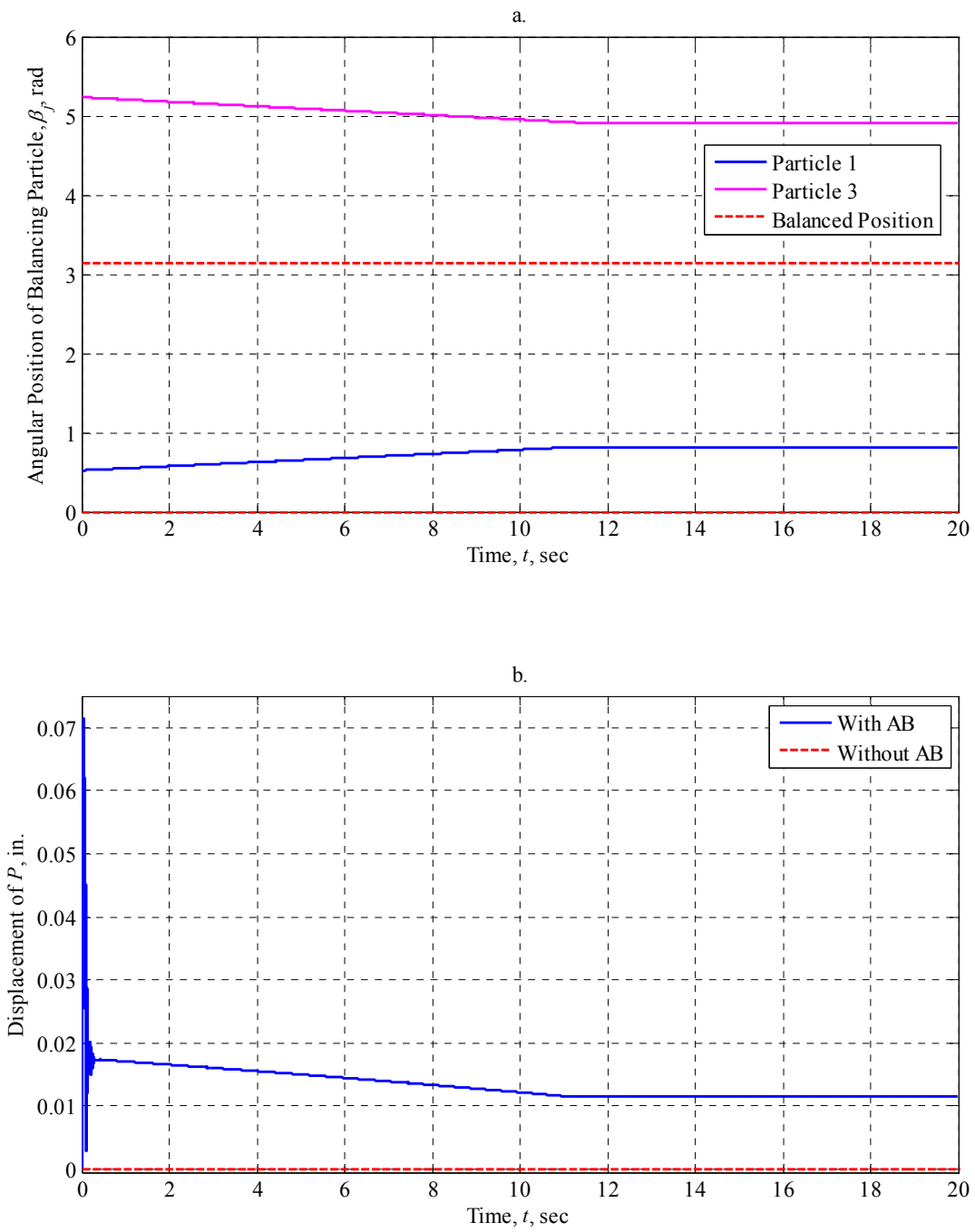

Figure 5.3.1.2.9-I: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

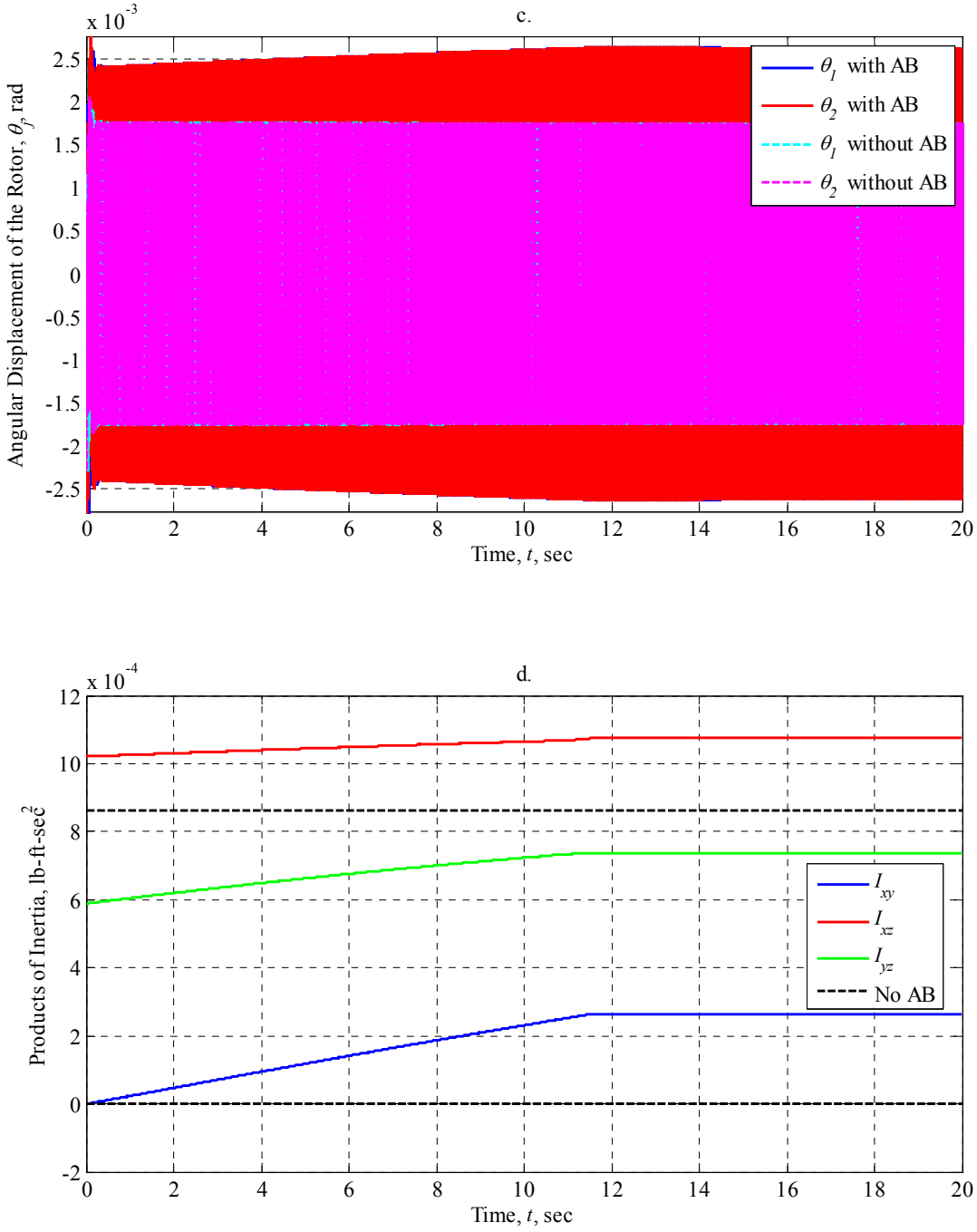

Figure 5.3.1.2.9-II: AB Performance at 200 rad/sec with $\mu_s = 0.1$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with One Particle per Race)

5.3.2 Vertically-Oriented Rotor with Two Balancing Particles per Race

Our second and final dynamic imbalance case will explore balancing a vertically-oriented rotor using four total balancing particles with two particles in the front race and two particles in the rear race. We will determine the performance of the AB for a range of Coulomb friction levels as outlined in Table 5.3.5 and a range of operating speeds as outlined in Table 5.3.6. As already stated, because the dynamically-imbalanced rotor features only wobble motions about intermediate coordinate axes, and no translational motions of its geometric center P, the performance of the AB will be evaluated on whether or not it is able to eliminate or reduce these unwanted rotational effects as compared to the unbalanced performance alone. The motion of the geometric center will still be examined because the addition of extra masses, that are not symmetric about P, will cause translational motions of the geometric center. These motions must vanish or the performance of the rotor and AB is worsened. The final resting position of the balancing particles, and their motion relative to the rotor will be examined along with the corresponding angular displacements of the rotor about intermediate coordinate axes and changes in the system products of inertia. Figures 5.3.2.2.1 through 5.3.2.2.9, grouped at the end of Section 5.3.2, show these behaviors. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. As before, the a-part of the figure shows the behavior of the balancing particles while the b-part shows the magnitude of the displacement of the geometric center P, in inches. The c-part of the figure shows the orientation of the rotor about the inertial X-axis and the intermediate y_2-axis, while the d-part of the figure shows the products of inertia for the system of the rotor, imbalance masses, and balancing particles. Note that the figures will span two pages per simulation and will feature the a- and b-parts on the first page and the c- and d-parts on the second. The labeling will be consistent between figures with the a- and b-parts post scripted with a "I" and the c- and d-parts post scripted with a "II" to denote the first and second pair of figures for the simulation, respectively.

Table 5.3.2.1 summarizes the behavior of the balancing particles at the selected operating speeds and Coulomb friction levels by listing their steady-state angular positions and ultimate stopping times. If a particles never come to rest relative to the cylinder during the time interval considered, the ultimate stopping time is determined as the time when the particle reaches and stays within 1° of its final steady-state angular position. Table 5.3.2.2 summarizes the behavior

of the geometric center P at the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of P and the steady-state amplitude of the displacement of P. Table 5.3.2.3 summarizes the orientation of the rotor by listing the maximum angular displacement and steady-state angular displacement about the inertial X- and intermediate y_2-axes. Table 5.3.1.4 is repeated here as Table 5.3.2.4 and summarizes the rotational motions of the rotor in the absence of an AB to establish a baseline for comparison purposes. Table 5.3.2.5 summarizes the effect of the AB on the rotor performance by comparing the angular displacements of the rotor before and after the addition of a balancer by comparing the differences in the maximum and steady-state angular displacements of the rotor as found in Tables 5.3.2.3 and 5.3.2.4. Table 5.3.2.6 computes the maximum and steady-state values of the products of inertia for the rotor and balancer combination and Table 5.3.2.7 compares these values to the ones in Table 5.3.2 for the case of an imbalanced rotor alone.

Table 5.3.2.1: Behavior of the Balancing Particles (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees				Ultimate Stopping Time, sec			
		β_1	β_2	β_3	β_4	β_1	β_2	β_3	β_4
15	0	—‡	—‡	—‡	—‡	—‡	—‡	—‡	—‡
	0.001	18.95	267.21	76.95	131.37	0.4020	0.7864	0.7703	0.7723
	0.1	19.98	265.04	79.93	130.04	0.0496	0.03640	0.0152	0.1093
50	0	-93.89	266.11	86.11	86.11	31.923†	31.923†	31.923†	31.923†
	0.001	-41.58	269.28	76.62	130.38	2.3984	0.6298	0.5330	0.0872
	0.1	19.99	265.02	79.63	130.09	0.0969	0.1824	3.7826	0.6416
200	0	—‡	—‡	—‡	—‡	—‡	—‡	—‡	—‡
	0.001	-114.33	265.14	80.57	56.42	10.2689	0.4418	0.1956	2.3558
	0.1	10.84	277.19	96.60	135.55	5.4670	8.9261†	11.1411	3.6405

† Indicates that $\pm$ 1° tolerance was used to determine the particle stop time.

‡ Indicates that no steady-state value could be determined.

Table 5.3.2.2: Behavior of the Geometric Center (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0	1.329	1.297
	0.001	0.3888	0.3887
	0.1	0.3935	0.3934
50	0	2.120	0.000
	0.001	2.128	0.3887
	0.1	2.144	1.381
200	0	5.659	0.000
	0.001	5.718	0.2939
	0.1	5.854	1.1549

Table 5.3.2.3: Angular Displacement of the Rotor in the Presence of an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Angular Displacement, (10^{-2}) degrees		Steady-State Amplitude of Angular Displacement, (10^{-2}) degrees	
		θ_1	θ_2	θ_1	θ_2
15	0	0.4460	0.4605	0.2115	0.2115
	0.001	0.4447	0.4605	0.4442	0.4442
	0.1	0.4438	0.4592	0.4437	0.4436
50	0	4.538	4.533	4.133	4.127
	0.001	4.474	4.468	4.465	4.459
	0.1	3.961	4.300	3.940	3.934
200	0	27.398	27.291	—‡	—‡
	0.001	24.619	25.823	18.739	18.733
	0.1	23.095	25.269	18.724	18.715

‡ Indicates that no steady-state value could be determined.

Table 5.3.2.4: Angular Displacement of the Rotor in the Absence of an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor)

Initial z-axis Angular Velocity, rad/sec	Maximum Angular Displacement, (10^{-2}) degrees		Steady-State Amplitude of Angular Displacement, (10^{-2}) degrees	
	θ_1	θ_2	θ_1	θ_2
15	0.2165	0.2164	0.2164	0.2164
50	1.935	1.929	1.929	1.926
200	10.992	11.699	9.928	9.926

Table 5.3.2.5: Comparison of Rotor Performance With and Without an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Angular Displacement of the Rotor, (10^{-2}) degrees		Difference in Steady-State Angular Displacement of the Rotor, (10^{-2}) degrees	
		θ_1	θ_2	θ_1	θ_2
15	0	+0.2296	+0.2440	-0.004932	-0.004893
	0.001	+0.2283	+0.2441	+0.2278	+0.2278
	0.1	+0.2274	+0.2428	+0.2272	+0.2272
50	0	+2.602	+2.603	+2.204	+2.201
	0.001	+2.539	+2.539	+2.536	+2.533
	0.1	+2.026	+2.371	+2.011	+2.008
200	0	+16.406	+15.593	— ‡	— ‡
	0.001	+13.627	+14.125	+8.810	+8.807
	0.1	+12.103	+13.571	+8.795	+8.788

‡ Indicates that no steady-state value could be determined.

A positive (+) sign in the above table indicates an increase (worsening) in the absolute displacement of P. A negative (-) sign indicates a decrease (improvement) in the absolute displacement of P.

Table 5.3.2.6: Products of Inertia for the Rotor in the Presence of an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Product of Inertia, (10^{-4}) lb-ft-sec^2			Steady-State Product of Inertia, (10^{-4}) lb-ft-sec^2		
		I_{xy}	I_{xz}	I_{yz}	I_{xy}	I_{xz}	I_{yz}
15	0	-17.253	14.447	-10.721	— ‡	8.735	0.002768
	0.001	0.7490	14.387	-10.346	0.6880	14.373	-10.346
	0.1	0.7504	14.329	-10.373	0.7503	14.328	-10.374
50	0	-6.373	14.997	-17.253	2.330	6.370	-10.372
	0.001	-6.490	14.682	-14.658	-6.489	13.595	-14.658
	0.1	0.7945	14.329	-10.374	0.7945	14.307	-10.365
200	0	17.253	25.880	17.253	— ‡	— ‡	— ‡
	0.001	9.336	14.582	16.813	9.336	3.392	-16.076
	0.1	-4.827	17.004	-10.827	-4.775	16.978	-10.773

‡ Indicates that no steady-state value could be determined.

Table 5.3.2.7: Comparison of Products of Inertia for the Rotor With and Without an Automatic Balancer (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Product of Inertia, (10^{-4}) lb-ft-sec^2			Difference in Steady-State Product of Inertia, (10^{-4}) lb-ft-sec^2		
		I_{xy}	I_{xz}	I_{yz}	I_{xy}	I_{xz}	I_{yz}
15	0	+17.253	+5.821	+10.721	— ‡	+0.1087	+0.002768
	0.001	+0.7490	+5.760	+10.346	+0.6880	+5.747	+10.346
	0.1	+0.7504	+5.702	+10.373	+7.503	+5.701	+10.374
50	0	+6.373	+6.370	+17.253	+2.330	-1.168	+10.372
	0.001	+6.490	+6.055	+14.658	+6.489	+4.969	+14.658
	0.1	+0.7945	+5.703	+10.374	+0.7945	+5.681	+10.365
200	0	+17.253	+17.253	+17.253	— ‡	— ‡	— ‡
	0.001	+9.336	+5.956	+16.813	+9.336	-5.235	+16.076
	0.1	+4.827	+8.378	+10.827	+4.775	+8.351	+10.773

‡ Indicates that no steady-state value could be determined.

As before, a positive (+) sign indicates that the magnitude of the product of inertia increased (worsened) compared with the unbalanced rotor. Similarly, a negative (-) sign indicates that the magnitude of the product of inertia decreased (improved) compared with the unbalanced rotor.

5.3.2.1 Summary of Findings

For the cases where $\omega_3 = 15$ rad/sec, when the Coulomb friction level is $\mu_s = 0$, none of the balancing particles are able to locate a steady-state position. Instead, the particles travel around the races together, acting as a single particle. Particles 1 and 2 are together in the outer race while particles 3 and 4 move together in the inner race; but all four particles occupy the same angular position and speed. This gives a steady-state amplitude of displacement to the geometric center of $1.297(10^{-2})$ inches. Since the dynamically unbalanced rotor had no displacement of the geometric center this is a drastic decrease in performance.

Once the Coulomb friction level is increased to $\mu_s = 0.001$, the particles come to rest relative to the rotor within a short period of time and occupy distinct positions. All four particles stop within 0.7723 seconds. None of the particle reaches a balanced position, so there is a steady-state amplitude for the geometric center of $0.3887(10^{-2})$ inches. While this is better than the case of zero Coulomb friction and represents a 70.0% reduction, it is worse overall. When the Coulomb friction level is $\mu_s = 0.1$, the particles stop even faster needing only 0.1093 seconds before they all stop. This gives a steady-state displacement of $0.3934\ (10^{-2})$ inches, a decrease of 69.7% when compared to the case when $\mu_s = 0$, but slightly larger than the case of $\mu_s = 0.001$.

The reduction in displacement cannot be attributed to any mechanism in the system that occurs when the rotor is operated below a translational natural frequency, but instead is owing to the fact that the chosen initial conditions allow the particles to stop sooner so they don't travel as far from the balanced positions to worsen performance. For the non-zero cases of Coulomb friction, the particles stop very close to their initial conditions. When $\mu_s = 0.001$ and 0.1, Particle 1 moves 1.05° and 0.02°; Particle 2 moves 2.21° and 0.04°; Particle 3 moves 3.05° and 0.07°; and Particle 4 moves 1.37° and 0.04°, respectively.

As was observed for the case of only two balancing particles, the steady-state angular displacements of the rotor about the inertial X-axis and the intermediate y_2-axis decrease slightly from $0.2164(10^{-2})$ degrees to $0.2115(10^{-2})$ degrees for both the X- and y_2-axes, a reduction of approximately 2.3%, when the Coulomb friction level is set to zero. It is difficult to pinpoint why the use of four particles, compared with only two particles or zero particles, is able to lower the steady-state values when there is no Coulomb friction. Perhaps it can be attributed to having more mass in the system that is able to adjust and slightly increase the values of the diagonal entries in the inertia tensor, lowering the maximum achievable angular displacements. When the Coulomb friction level is set to $\mu_s = 0.001$, the steady-state angular displacements about the X- and y_2-axes are both $0.4442(10^{-2})$ degrees, an increase of 105.3% over the unbalanced values of $0.2164(10^{-2})$ degrees about each axis. When the Coulomb friction level is set to $\mu_s = 0.1$, the steady-state values are slightly lower than previous at approximately $0.4437(10^{-2})$ degrees about each axis, an increase of 105.0% over the unbalanced rotor values.

The unbalanced rotor features a maximum angular displacement about the X- and y_2-axes of $0.2165(10^{-2})$ degrees and $0.2164(10^{-2})$ degrees. Across the three Coulomb friction levels, the rotor with balancer sees an average maximum of $0.4448(10^{-2})$ degrees about the X-axis and $0.4601(10^{-2})$ degrees about the y_2-axis. This is an average increase of 105.5% about the X-axis and 112.6% about the y_2-axis. While an increase is not unexpected since we have added four 1-oz. imbalances to the system, it is a slight surprise to see a four-fold percent increase when compared to the case of using two balancing particles, where we now have an average maximum amplitude percent increase of 35.8%.

The products of inertia for the system are not improved for any case. The unbalanced rotor features no I_{xy} or I_{yz} products of inertia, but the addition of four balancing masses results in non-zero values for these terms. When $\mu_s = 0$, Figure 5.3.2.2.1-II shows that I_{xz} and I_{yz} oscillate

about the unbalanced rotor values, which does not necessarily degrade performance, but it does not improve it. I_{xy} is seen to exhibit large oscillations reaching a maximum value of $17.253(10^{-4})$ lb-ft-sec^2. The oscillatory behavior is expected since the balancing particles are simply traveling around the races of the rotor, changing the products of inertia in a continuous fashion.

When the Coulomb friction is increased to $\mu_s = 0.001$ or 0.1, the particles are seen to stop very quickly, so the products of inertia reach a steady-state value. I_{xy} is small for both cases, becoming fixed at $0.6880(10^{-4})$ lb-ft-sec^2 when $\mu_s = 0.001$, and $0.7503(10^{-4})$ lb-ft-sec^2 when $\mu_s = 0.1$. I_{yz} increases dramatically to a steady-state value of $-10.346(10^{-4})$ lb-ft-sec^2 and $-10.374(10^{-4})$ lb-ft-sec^2 for $\mu_s = 0.001$ and 0.1, respectively. This is a severe reduction in performance since the unbalanced rotor had $I_{yz} = 0$. I_{xz} becomes fixed at $14.373(10^{-4})$ lb-ft-sec^2 when $\mu_s = 0.001$, and $14.328(10^{-4})$ lb-ft-sec^2 when $\mu_s = 0.1$. These represent increases of 66.6% and 66.1%, respectively, from the unbalanced value of $8.627(10^{-4})$ lb-ft-sec^2.

The maximum values obtained for I_{xy} and I_{yz} occur when there is no Coulomb friction present with I_{xy} reaching a maximum of $-17.253(10^{-4})$ lb-ft-sec^2 and I_{yz} reaching a maximum of $-10.721(10^{-4})$ lb-ft-sec^2. The same is true for I_{xz} where it reaches a maximum of $14.447(10^{-4})$ lb-ft-sec^2. When the Coulomb friction is non-zero, I_{xz} reaches a maximum value of $14.329(10^{-4})$ lb-ft-sec^2, a decrease of 0.817% over the zero Coulomb friction level.

For the cases where $\omega_3 = 50$ rad/sec, we observe very different behavior when $\mu_s = 0$. For this case, particles 3 and 4 converge to a steady-state angle of 86.11° while particles 1 and 2 end up at -93.89° and 267.21°, which is the same location on the wheel but 180° removed from particles 3 and 4. Thus, the particles in each race pair up and cancel their combined effects. This results in a zero steady-state amplitude of displacement for the geometric center. This is not surprising since we are above the translational natural frequency of the support and the mechanism to drive the balancing particles to their balanced positions is present. As with only two balancing particles, there is no tendency to drive the particles to positions which cancel the effect of the two dynamically imbalanced masses. All four balancing particles reach a 1° envelope about their steady-state value after 31.923 seconds indicating they are moving in synchronous motion. They are able to effectively eliminate any motion of the geometric center after 6.2 seconds when they reach positions roughly 180° opposite one another.

When the friction is increased to $\mu_s = 0.001$, all four particles stop rather quickly, with Particle 1 taking the longest at 2.3984 seconds. This short time interval does not allow the

particles to move to opposing positions and results in a steady-state displacement of the geometric center of 0.3887(10^{-2}) inches. A similar trend occurs when $\mu_s = 0.1$, but Particle 3 takes the longest to stop at 3.7826 seconds. This is an increase in overall stoppage time of 57.7% from what was seen when $\mu_s = 0.001$. This trend has been observed previously where increased operating speeds and increased Coulomb friction levels do not necessarily mean the particles will stop sooner. At this friction level, there is a steady-state amplitude of the geometric center of 1.381 (10^{-2}) inches which is an increase of 256.4% over the value when $\mu_s = 0.001$.

The steady-state amplitudes for both angular displacements about the inertial X- and intermediate y_2-axes are worsened in all cases indicating there is no improvement in the products of inertia for the rotor. When $\mu_s = 0$, the steady-state amplitudes are 4.133(10^{-2}) degrees and 4.127(10^{-2}) degrees about the X- and y_2-axes, respectively, representing increases of 114.3% from the unbalanced values of 1.929(10^{-2}) degrees about the X-axis and 1.926(10^{-2}) degrees about the y_2-axis. These differences increase when the Coulomb friction level increases to $\mu_s = 0.001$ where the values are 4.465(10^{-2}) degrees and 4.459(10^{-2}) degrees about the X- and y_2-axes, respectively. These numbers represent increases of 131.5% from the unbalanced values. When $\mu_s = 0.1$, the values are 3.940(10^{-2}) degrees and 3.934(10^{-2}) degrees about the X- and y_2-axes, respectively, representing increases of 104.3% from the unbalanced values.

The maximum amplitudes for the angular displacements also increase in every case, but are seen to decrease as the Coulomb friction level increases. When $\mu_s = 0$, the maximum values are 4.538(10^{-2}) degrees about the X-axis and 4.533(10^{-2}) degrees about the y_2-axis, representing increases of 134.5% and 135.0%, respectively, from the unbalanced values of 1.935(10^{-2}) degrees for the X-axis and 1.929(10^{-2}) degrees about the y_2-axis. When $\mu_s = 0.001$, the maximum values are 4.474(10^{-2}) degrees about the X-axis and 4.468(10^{-2}) degrees about the y_2-axis, representing increases of 131.2% and 131.6%, respectively, from the unbalanced values. Once $\mu_s = 0.1$ there is a slight reduction where the maximum value is 3.961(10^{-2}) degrees about the X-axis and 4.300(10^{-2}) degrees about the y_2-axis, representing increases of 104.7% and 122.9% from the unbalanced values. It is no surprise that the changes are similar for both axes since the support characteristics are identical about the plane of the rotor, and the overall angular displacement about intermediate axes is small.

The products of inertia are worsened for all values of Coulomb friction except for I_{xz} when $\mu_s = 0$. At this friction level, I_{xz} reaches a steady state value of 6.370(10^{-4}) lb-ft-sec^2 which

is a decrease of 26.2% over the unbalanced value. At this same friction level, I_{xy} reaches a steady-state value of $2.330(10^{-4})$ lb-ft-sec^2 and I_{yz} reaches a steady-state value of $-10.0372(10^{-4})$ lb-ft-sec^2. These are worse overall since these products of inertia are zero for the unbalanced rotor alone. The maximum values of I_{xy} and I_{yz} at this friction level are $-6.373(10^{-4})$ lb-ft-sec^2 and $-17.253(10^{-4})$ lb-ft-sec^2, respectively. The maximum value for I_{xz} is $14.997(10^{-4})$ lb-ft-sec^2 representing a 73.8% increase over the unbalanced value. It is interesting to note that this is a smaller increase than was observed for the same parameters using two balancing particles. It is difficult to compare the two simulations exactly since the initial conditions for four balancing particles differ from that for two, but a doubling of the mass did not produce a doubling of the maximum product of inertia value.

When $\mu_s = 0.001$, all three products of inertia worsen with I_{xy} reaching a steady state value of $-6.489(10^{-4})$ lb-ft-sec^2 and I_{yz} reaches a value of $-14.658(10^{-4})$ lb-ft-sec^2. These represent increases of 178.5% and 41.3% over the zero-friction levels at the same operating speed. At this friction level, these products of inertia have maximum values of $-6.490(10^{-4})$ lb-ft-sec^2 and $-14.658(10^{-4})$ lb-ft-sec^2 for I_{xy} and I_{yz}, respectively. Where I_{xy} slightly increases, I_{yz} decreases by 15.0%. I_{xz} reaches a steady-state value of $13.595(10^{-4})$ lb-ft-sec^2 representing a 57.6% increase over the unbalanced value which is a significant improvement over the results found using only two balancing particles which featured a 75.1% increase over the unbalanced value. I_{xz} reaches has a maximum value of $14.682(10^{-4})$ lb-ft-sec^2 representing a 70.2% increase over the unbalanced value.

When $\mu_s = 0.1$, I_{xy} reaches a maximum and steady state value of $0.7945(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a maximum value of $-10.374(10^{-4})$ lb-ft-sec^2 and a steady-state value of $-10.365(10^{-4})$ lb-ft-sec^2. This is a significant improvement over the values obtained with a friction level of $\mu_s = 0.001$, and for I_{xy}, a significant improvement over the zero Coulomb friction level. I_{xz} reaches a maximum value of $14.329(10^{-4})$ lb-ft-sec^2 and a steady-state value of $14.307(10^{-4})$ lb-ft-sec^2 representing increases of 66.1% and 65.9% respectively over the unbalanced values.

For the cases where $\omega_3 = 200$ rad/sec, we see results similar to both the 15 rad/sec and 50 rad/sec simulations. When Coulomb friction is zero, the four balancing particles are unable to reach steady-state values. They instead travel around the races in synchronous motion with the particles in each race exhibiting small oscillations about one another. The pairs of particles in

the front and rear races are separated by 180°, effectively cancelling their imbalance resulting in a zero steady-state amplitude of displacement for the geometric center after approximately 2.6 seconds. There is see no tendency to drive the particles to a position which cancels the effect of the two dynamically imbalanced masses.

When the friction is increased to $\mu_s = 0.001$, particles 2 and 3 stop in 0.4418 seconds and 0.1956 seconds respectively. It takes particles 4 and 1 2.3558 seconds and 10.2689 seconds to stop. This short time interval for particles 2 and 3 does not allow the particles to assume opposing positions and results in a steady-state displacement of the geometric center of $0.2939(10^{-2})$ inches. This is an improvement over the 50 rad/sec case representing a 24.4% reduction from the steady-state amplitude at the same Coulomb friction level. It is interesting to note that as seen before with the statically imbalanced rotor, the particles are able to move for longer periods of time at higher operating speeds in the presence of increased friction.

A similar trend occurs when $\mu_s = 0.1$, where all four particles take significant amount of time to stop. The stop times are 5.4670, 8.9261, 11.1411, and 3.6405 seconds for Particles 1 through 4, respectively. At this friction level, there is a resulting steady-state amplitude of the geometric center of $1.1549(10^{-2})$ inches. This is a reduction of 16.4% from the same friction levels at an operating speed of 50 rad/sec. Again, we note the significant difference in stopping times for the particles at higher operating speeds. As an example, the stop time for Particle 1 from 50 rad/sec to 200 rad/sec increased from 0.0969 seconds to 5.4670 seconds, an increase of 5540%.

The steady-state amplitudes for both angular displacements about the inertial X- and intermediate y_2-axes are worsened in all cases and were significantly larger percentage increases than the 50 rad/sec results. When $\mu_s = 0$, no steady-state values could be obtained since the two angular displacements were seen to exhibit periodic motion as seen in Figure 5.3.2.7-II. When $\mu_s = 0.001$, the steady-state values are $18.739(10^{-2})$ degrees and $18.733(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 88.7% from the unbalanced values. When $\mu_s = 0.1$, the values are $18.724(10^{-2})$ degrees and $18.715(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of approximately 88.6% from the unbalanced values.

The maximum amplitudes for the angular displacements tend to decrease as the friction level increases. When $\mu_s = 0$, the maximum values are $27.398(10^{-2})$ degrees about the X-axis and $27.291(10^{-2})$ degrees about the y_2-axis, representing increases of 149.3% and 133.3%

respectively from the unbalanced values of $10.992(10^{-2})$ degrees for the X-axis and $11.699(10^{-2})$ degrees about the y_2-axis. When $\mu_s = 0.001$, the maximum values drop slightly to $24.619(10^{-2})$ degrees about the X-axis and $25.823(10^{-2})$ degrees about the y_2-axis, representing increases of 124.0% and 120.7% respectively from the unbalanced values. When $\mu_s = 0.1$, the maximum value is $23.095(10^{-2})$ degrees about the X-axis and $25.269(10^{-2})$ degrees about the y_2-axis, representing increases of 110.1% and 129.9%, respectively, from the unbalanced values. Overall, the maximum amplitudes about the X-axis in the presence of a balancer increase an average of 872% from $0.4601(10^{-2})$ degrees at 15 rad/sec to $4.324(10^{-2})$ degrees at 50 rad/sec; and increase an average of 479% to $25.037(10^{-2})$ degrees from 50 rad/sec to 200 rad/sec. For the y_2-axis, the situation is similar with increases of 864% from $0.4601\ (10^{-2})$ degrees at 15 rad/sec to $4.434(10^{-2})$ degrees at 50 rad/sec; and increase an average of 830% to $26.123(10^{-2})$ degrees from 50 rad/sec to 200 rad/sec. These percentages are lower for the jump from 15 rad/sec to 50 rad/sec compared to the balancer utilizing only two masses. The percentage of increase for the maximum angular displacement about the X-axis is similar to the two particle balancer when going from 50 rad/sec to 200 rad/sec; but, the increase about the y_2-axis is significantly higher at 830% versus only 430% with a two particle balancer. It is unclear why these percentages are so different.

The products of inertia worsened for all values of Coulomb friction except in regards to I_{xz} at a Coulomb friction level of $\mu_s = 0.001$. Overall, the products of inertia seem to become worse at low Coulomb friction levels. When $\mu_s = 0$, all three products of inertia exhibit oscillatory motion as seen in Figure 5.3.2.2.7-II. Since the particles never come to rest, it is no surprise that the products of inertia never reach a steady-state. At this friction level, these products of inertia have maximum values of $17.253(10^{-4})$ lb-ft-sec^2 for I_{xy} and I_{yz} respectively, and $25.880(10^{-4})$ lb-ft-sec^2 for I_{xz}. Amazingly enough, the increase over the unbalanced values for these three terms are all $17.253(10^{-4})$ lb-ft-sec^2. For I_{xz} this is an increase of 100.0% over the unbalanced value. This is a severe reduction in performance for the unbalanced rotor.

When $\mu_s = 0.001$, I_{xy} reaches a steady state value of $9.336(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a value of $-16.076(10^{-4})$ lb-ft-sec^2. These are the two highest values recorded for these products of inertia over this series of simulations. Interestingly enough, while I_{xy} and I_{yz} record their highest simulation values, I_{xz} reaches its smallest at a value of $3.392(10^{-4})$ lb-ft-sec^2 representing a reduction of 60.7% over the unbalanced value of $8.627(10^{-4})$ lb-ft-sec^2. At this Coulomb

friction level, all three products of inertia have decreased maximum values when compared with the zero friction levels. I_{xy} has a value of $9.336(10^{-4})$ lb-ft-sec^2 down 45.9% from $17.253(10^{-4})$ lb-ft-sec^2. I_{yz} has a value of $16.813(10^{-4})$ lb-ft-sec^2 down 2.62% from $17.253(10^{-4})$ lb-ft-sec^2. I_{xz} has a value of $14.582(10^{-4})$ lb-ft-sec^2 down 43.7% from $25.880(10^{-4})$ lb-ft-sec^2.

When $\mu_s = 0.1$, I_{xy} reaches a steady state value of $-4.775(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a value of $10.773(10^{-4})$ lb-ft-sec^2. I_{xz} reaches its largest recorded steady-state value for this series of simulations at $16.978(10^{-4})$ lb-ft-sec^2 representing an increase of 96.8% over the unbalanced value of $8.627(10^{-4})$ lb-ft-sec^2. At this friction level, the maximum value of I_{xy} decreases again to $-4.827(10^{-4})$ lb-ft-sec^2 down 48.3% from $9.336(10^{-4})$ lb-ft-sec^2. I_{yz} also drops to $-10.827(10^{-4})$ lb-ft-sec^2 down 35.6% from $16.813(10^{-4})$ lb-ft-sec^2. I_{xz} has a slight increase to $17.004(10^{-4})$ lb-ft-sec^2 up 16.6% from $14.582(10^{-4})$ lb-ft-sec^2.

In general, we again find that the particles have no affinity for selecting balancing positions opposite the dynamically imbalanced pair of masses. The balancing particles instead focus on each other and at zero friction levels will move to positions 180° apart to cancel their collective effect provided the rotor is operated above the translational natural frequency of the support in the plane of the rotor. This leaves the wheel dynamically unbalanced, but statically balanced. If the Coulomb friction levels are increased only slightly, the situation is exacerbated since the particles are not allowed to completely reach a balanced state and the rotor has a combined static and dynamic imbalance which tends to worsen the rotor performance in all six degrees of freedom. There tends to be an improvement in particle balancing behavior as the operating speed is increased and the Coulomb friction level is increased, but it is difficult to tell if this is a general occurrence or a happenstance based on initial conditions and system properties. Additionally, we see the same phenomenon in the dynamic imbalance cases as was present in the static imbalance cases, where particles at very high operating speeds and high Coulomb friction levels can move for longer periods of time than at lower operating speeds. This seems counterintuitive, but the trend has been observed in numerous cases. The results found using four balancing particles tend to parallel what was observed using two balancing particles, but different phenomena can arise, especially when Coulomb friction is zero.

5.3.2.2 Plots of Rotor Performance and Balancing Particle Behavior

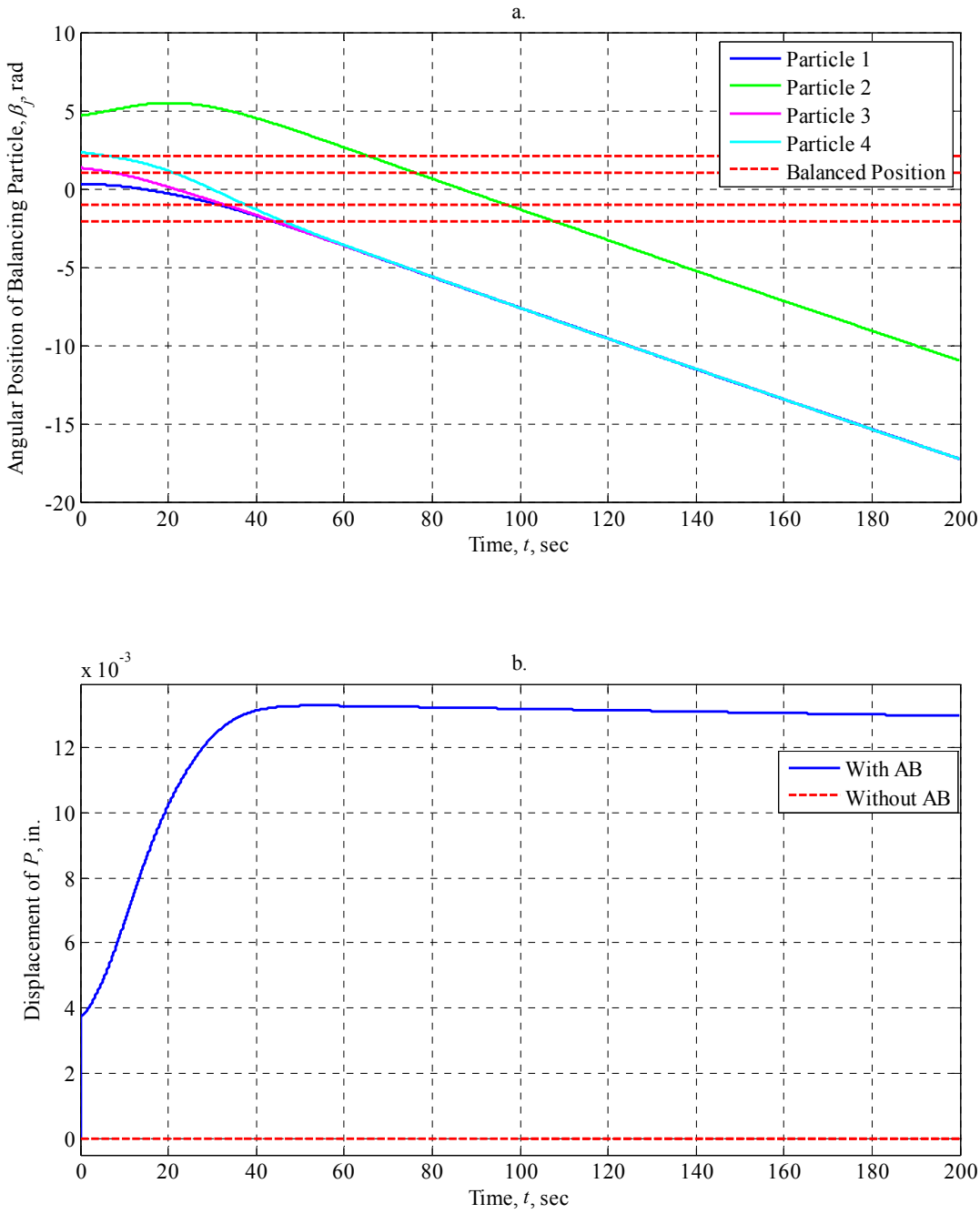

Figure 5.3.2.2.1-I: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

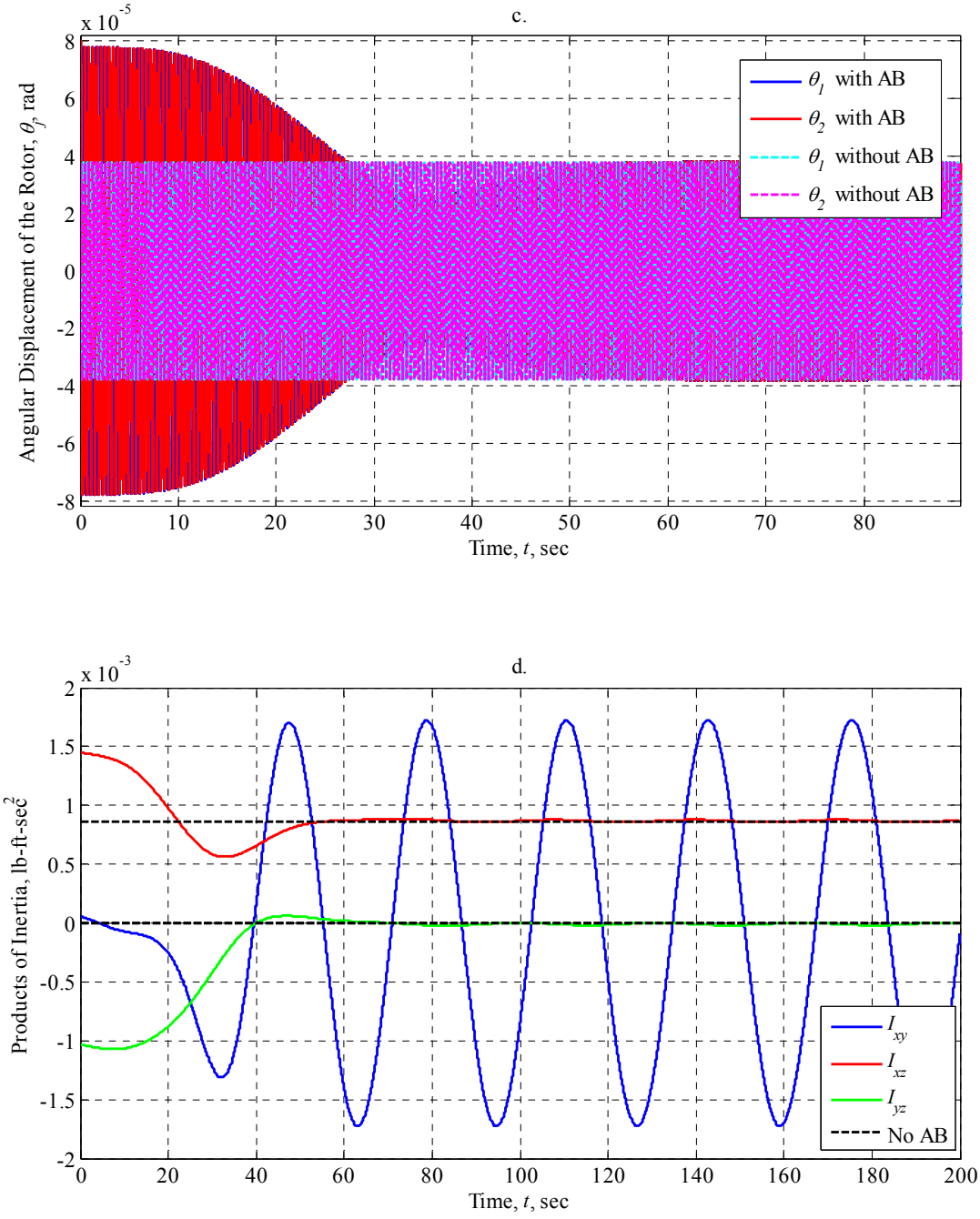

Figure 5.3.2.2.1-II: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

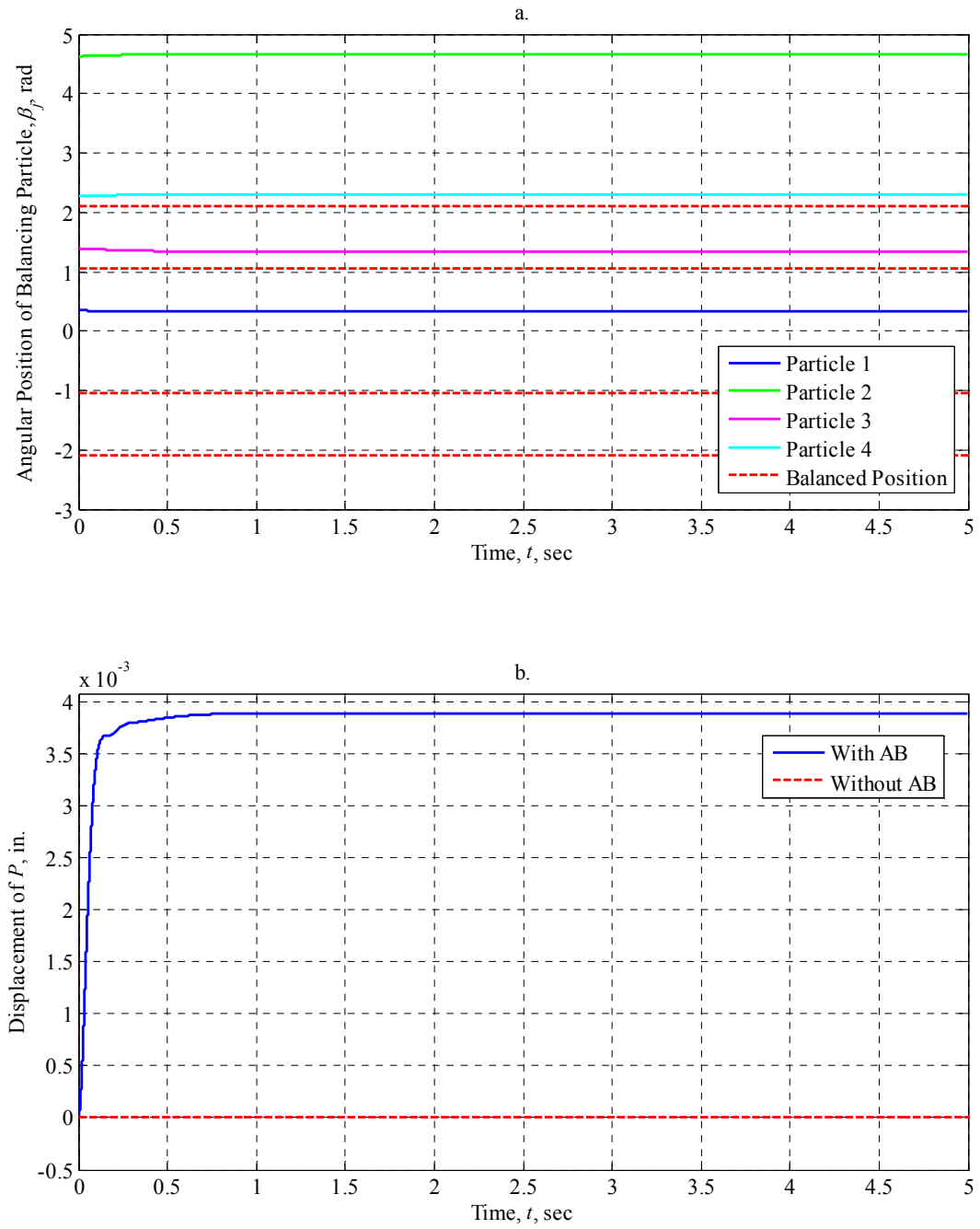

Figure 5.3.2.2.2-I: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

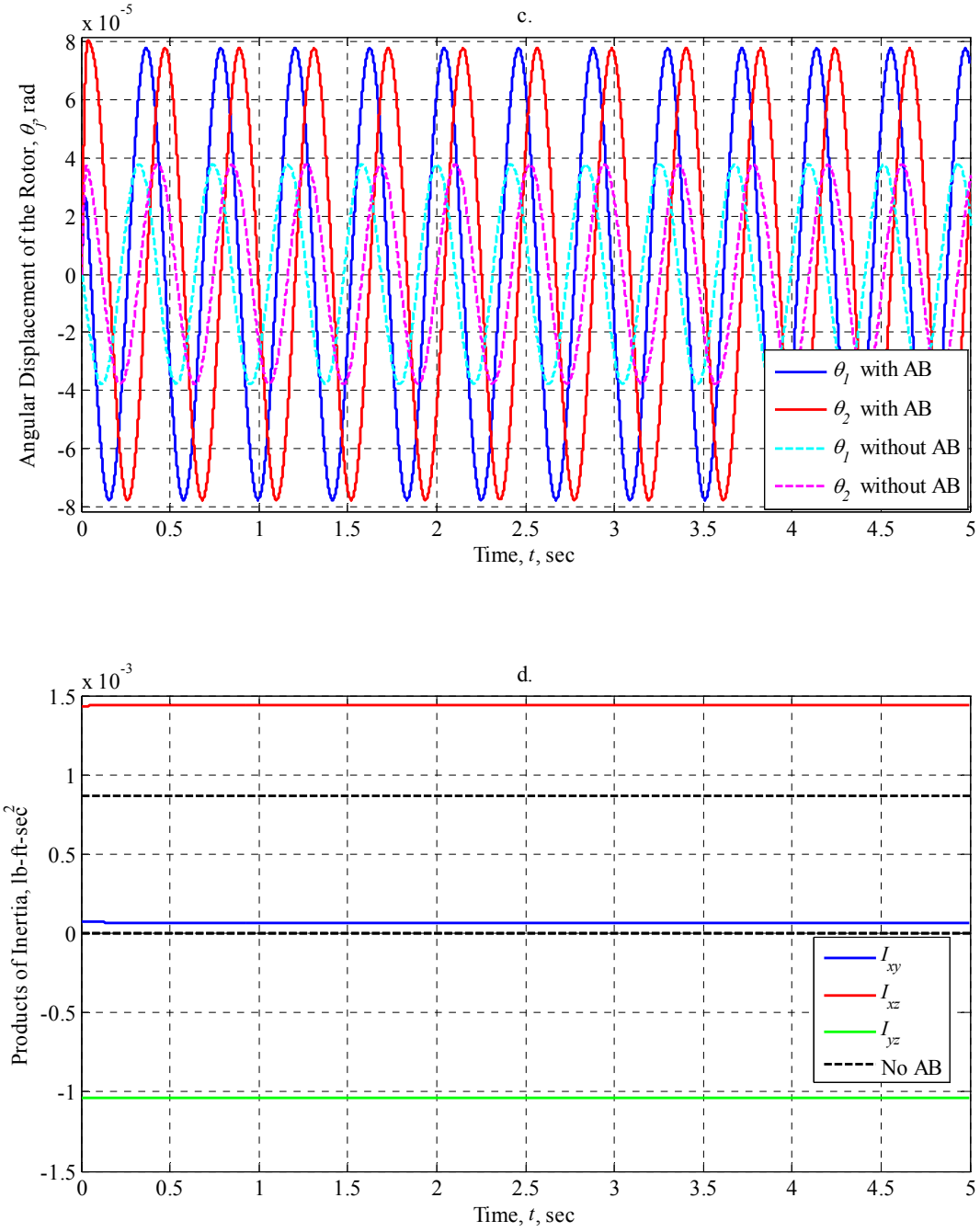

Figure 5.3.2.2.2-II: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

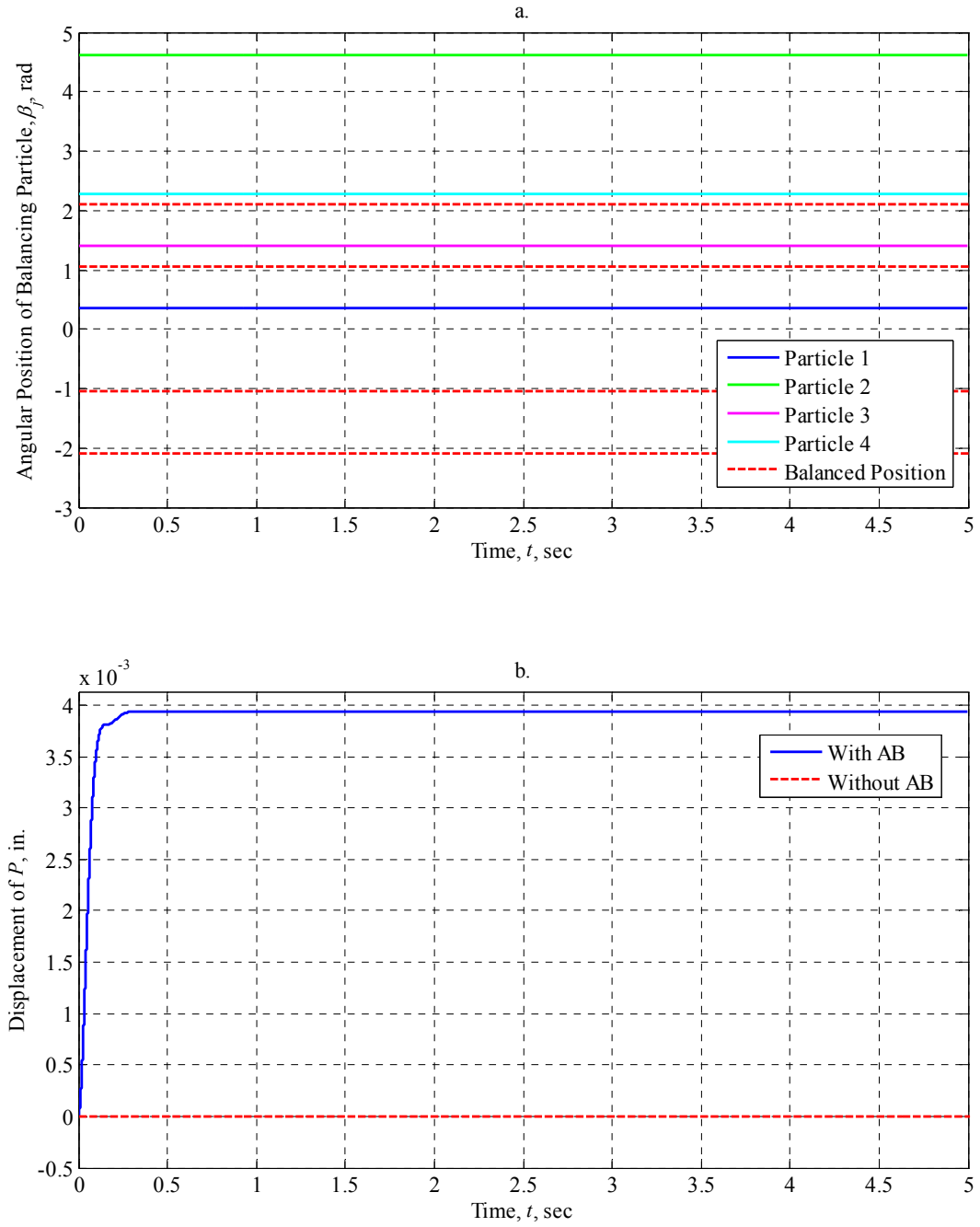

Figure 5.3.2.2.3-I: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

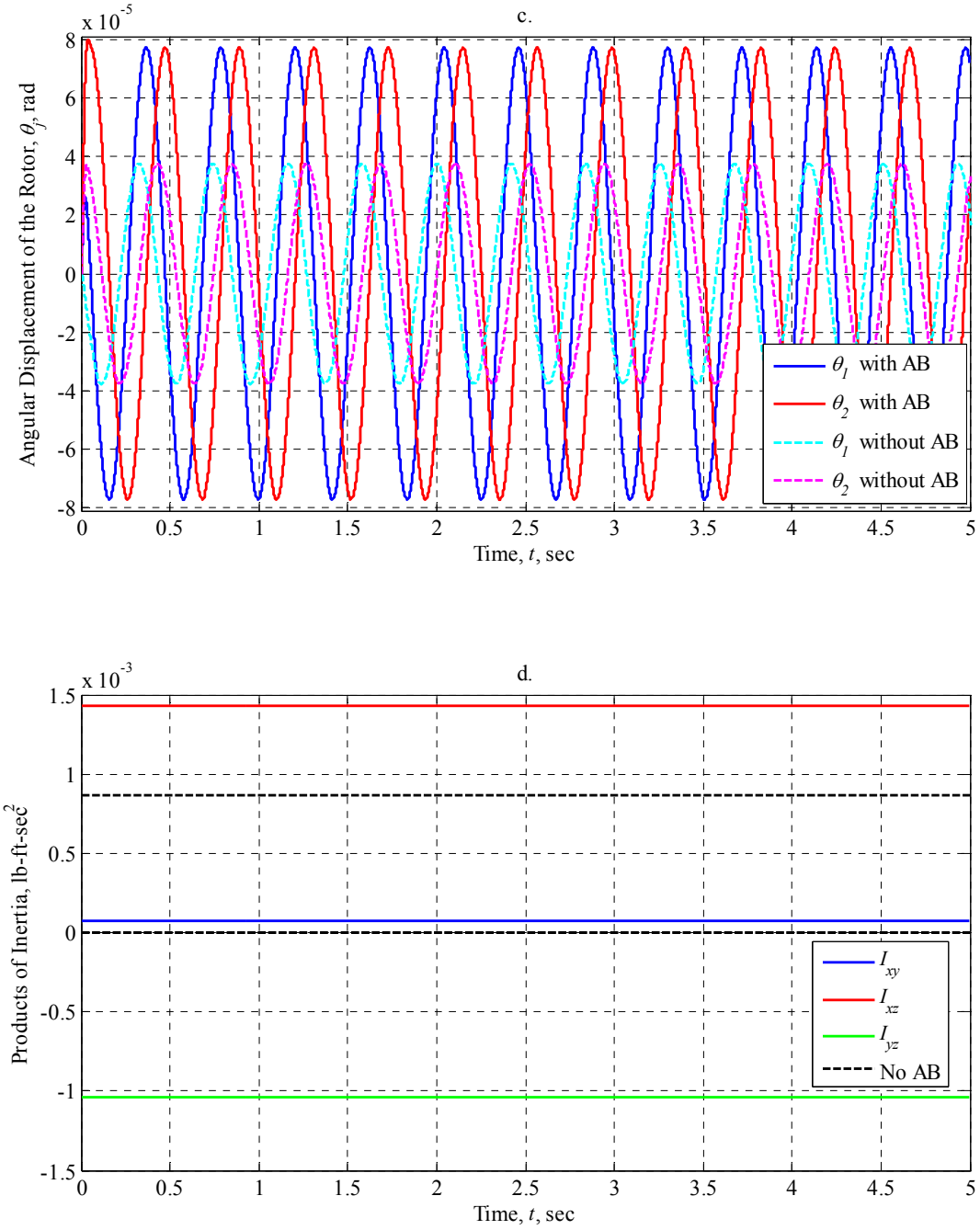

Figure 5.3.2.2.3-II: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

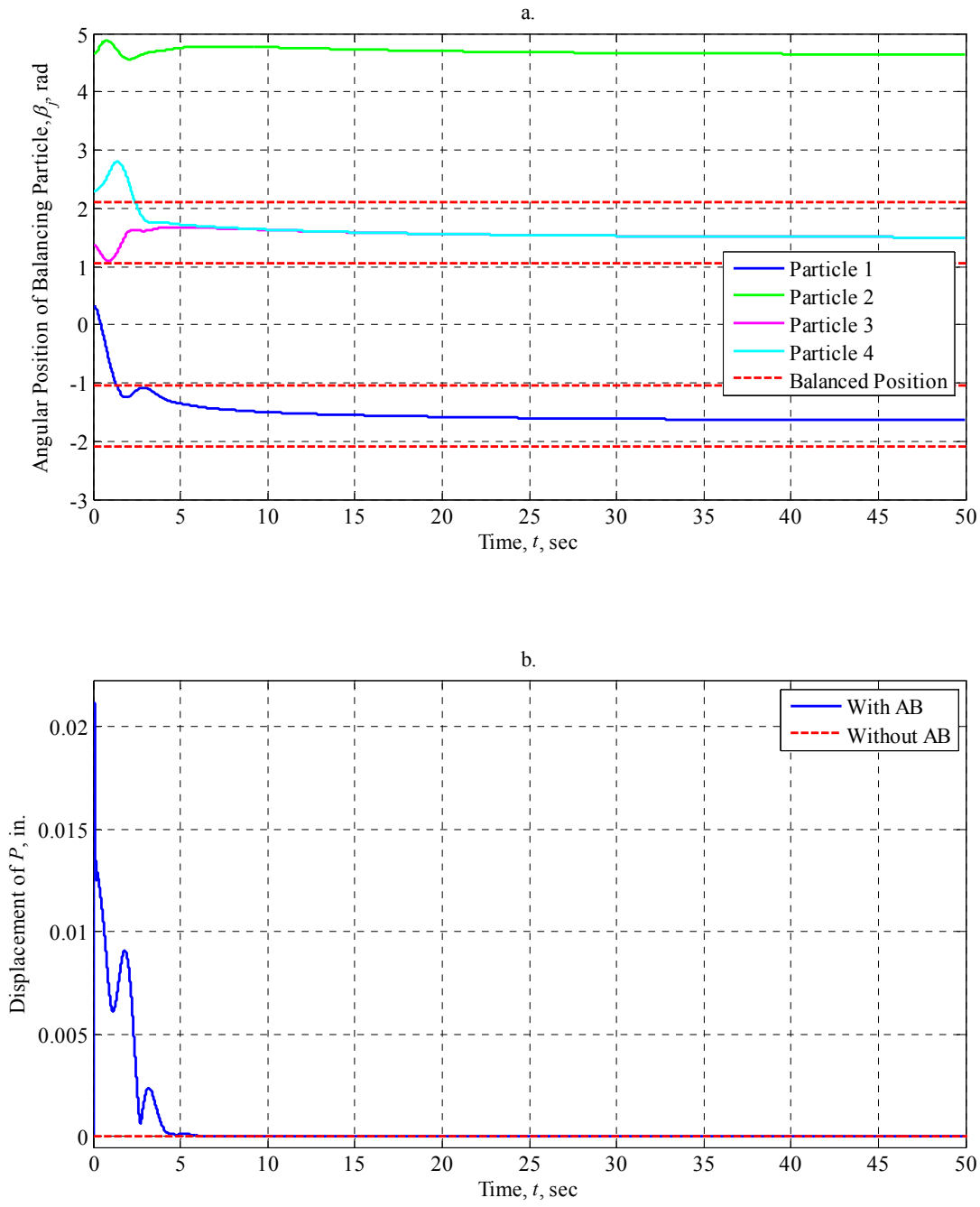

Figure 5.3.2.2.4-I: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

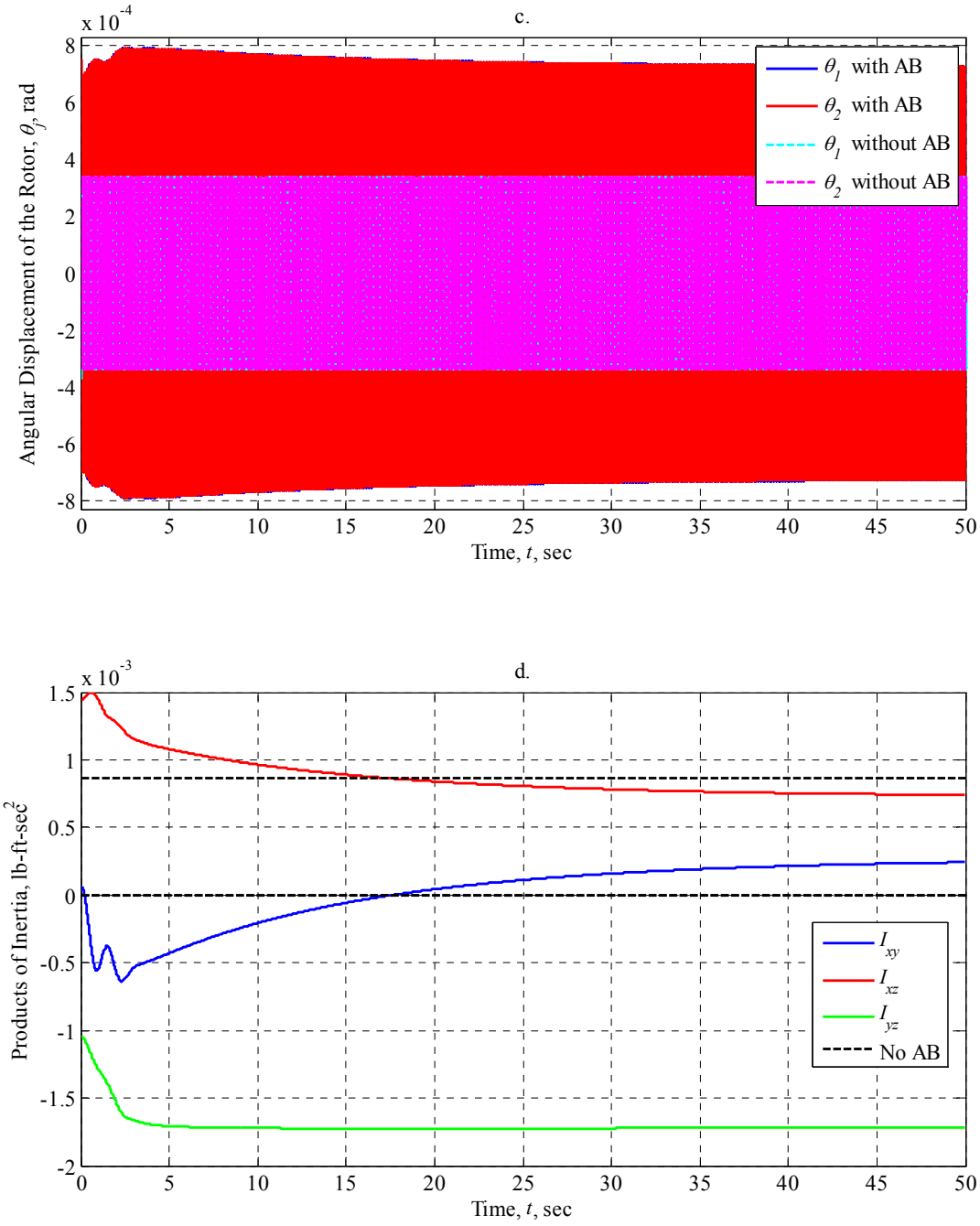

Figure 5.3.2.2.4-II: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

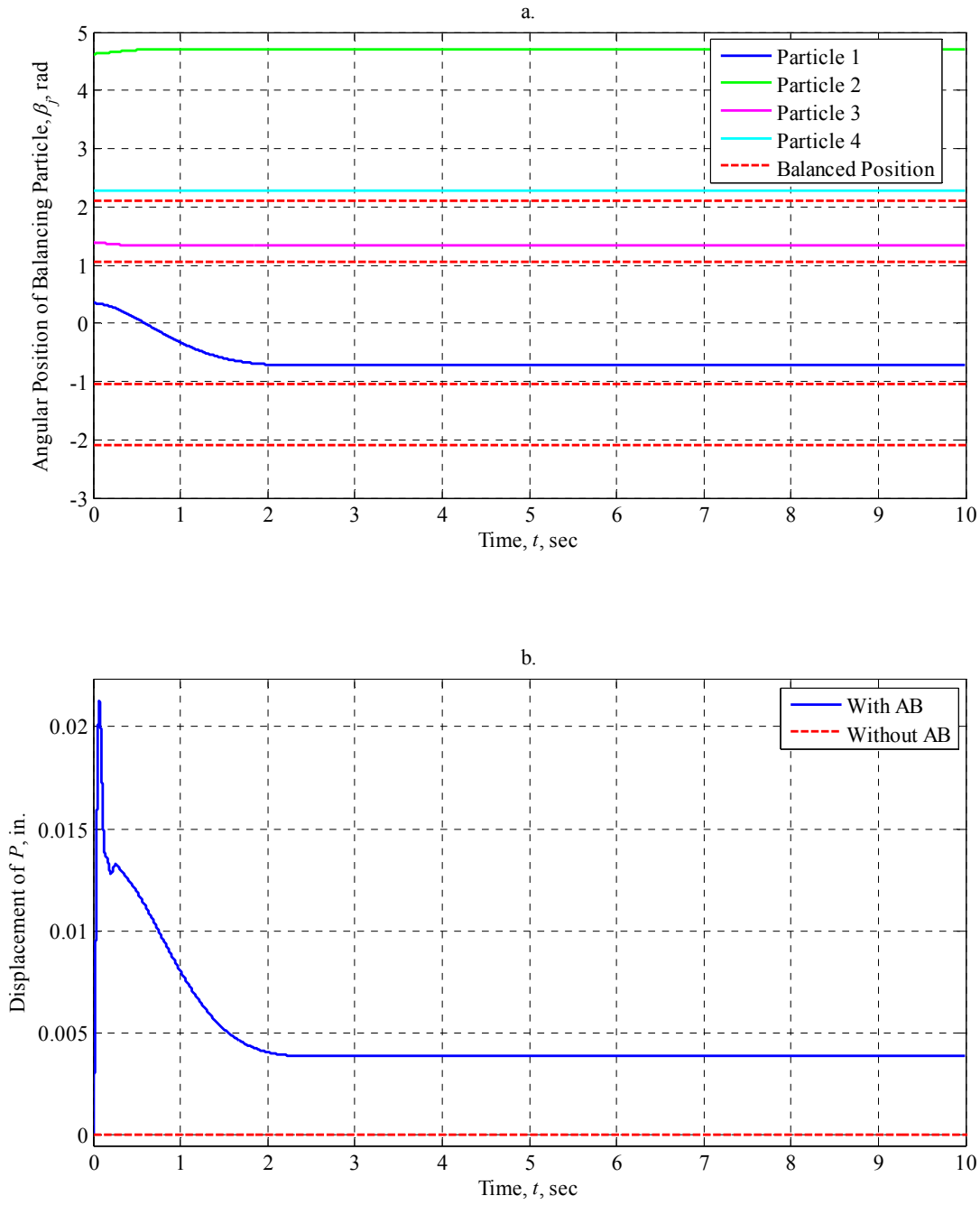

Figure 5.3.2.2.5-I: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

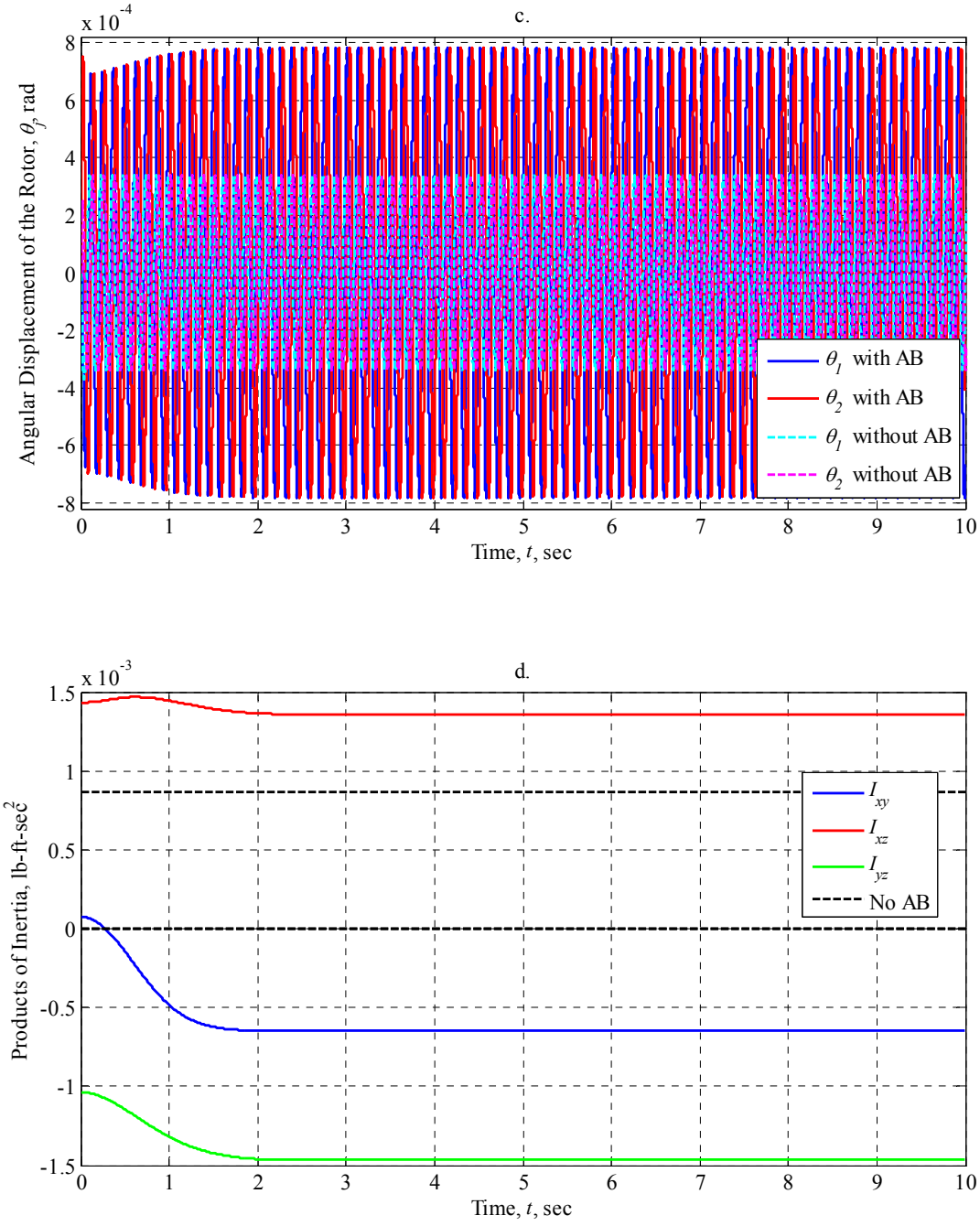

Figure 5.3.2.2.5-II: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

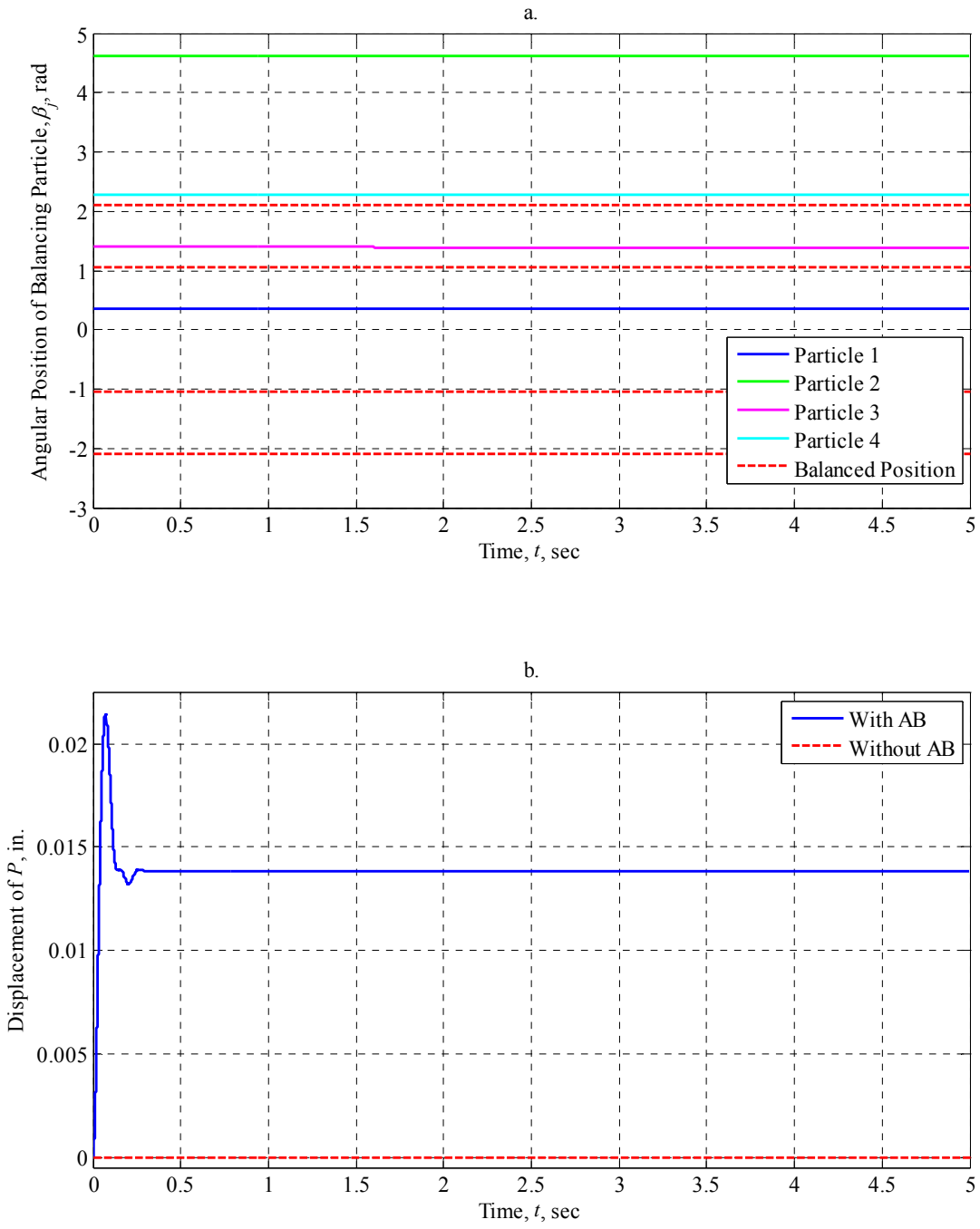

Figure 5.3.2.2.6-I: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

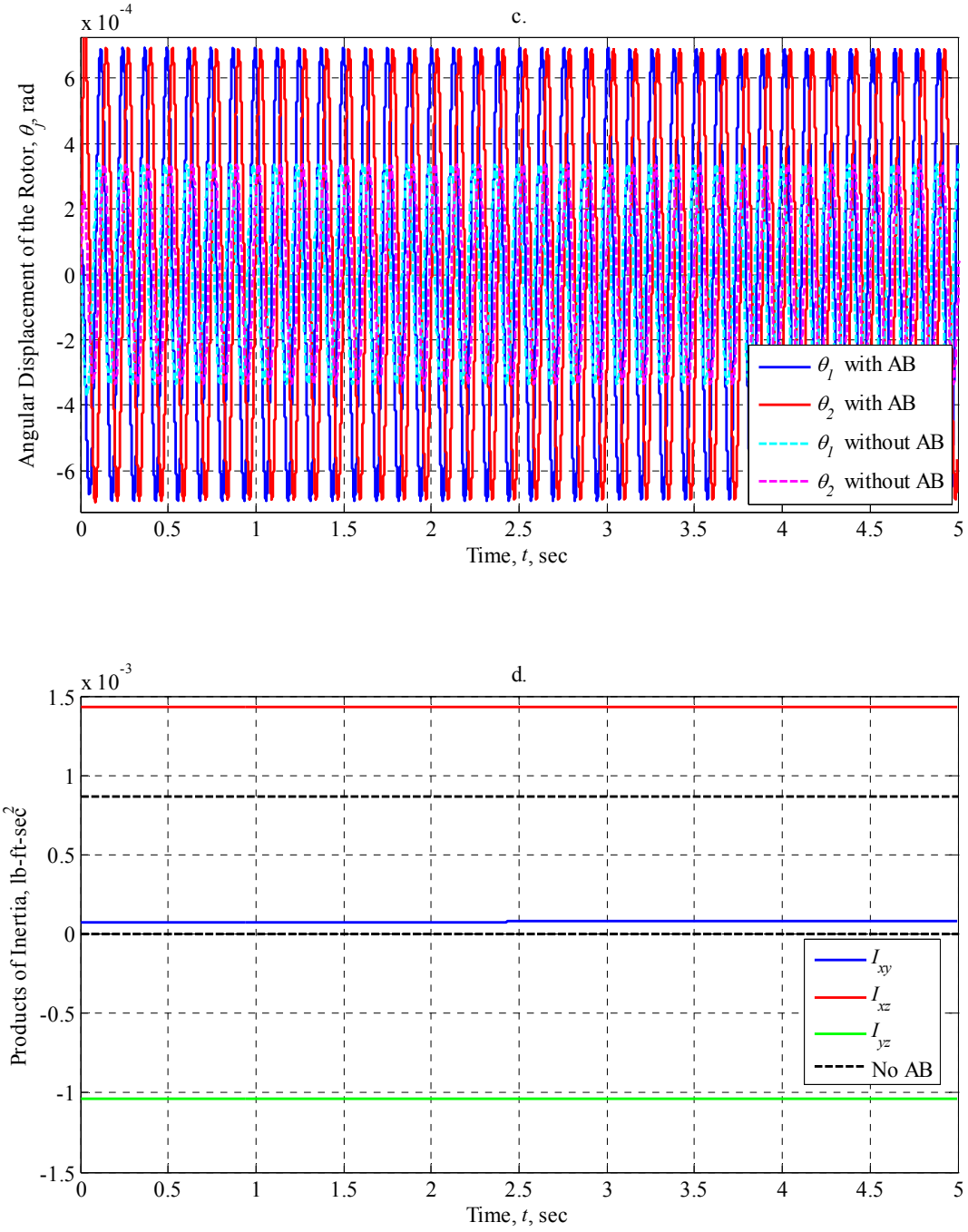

Figure 5.3.2.2.6-II: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

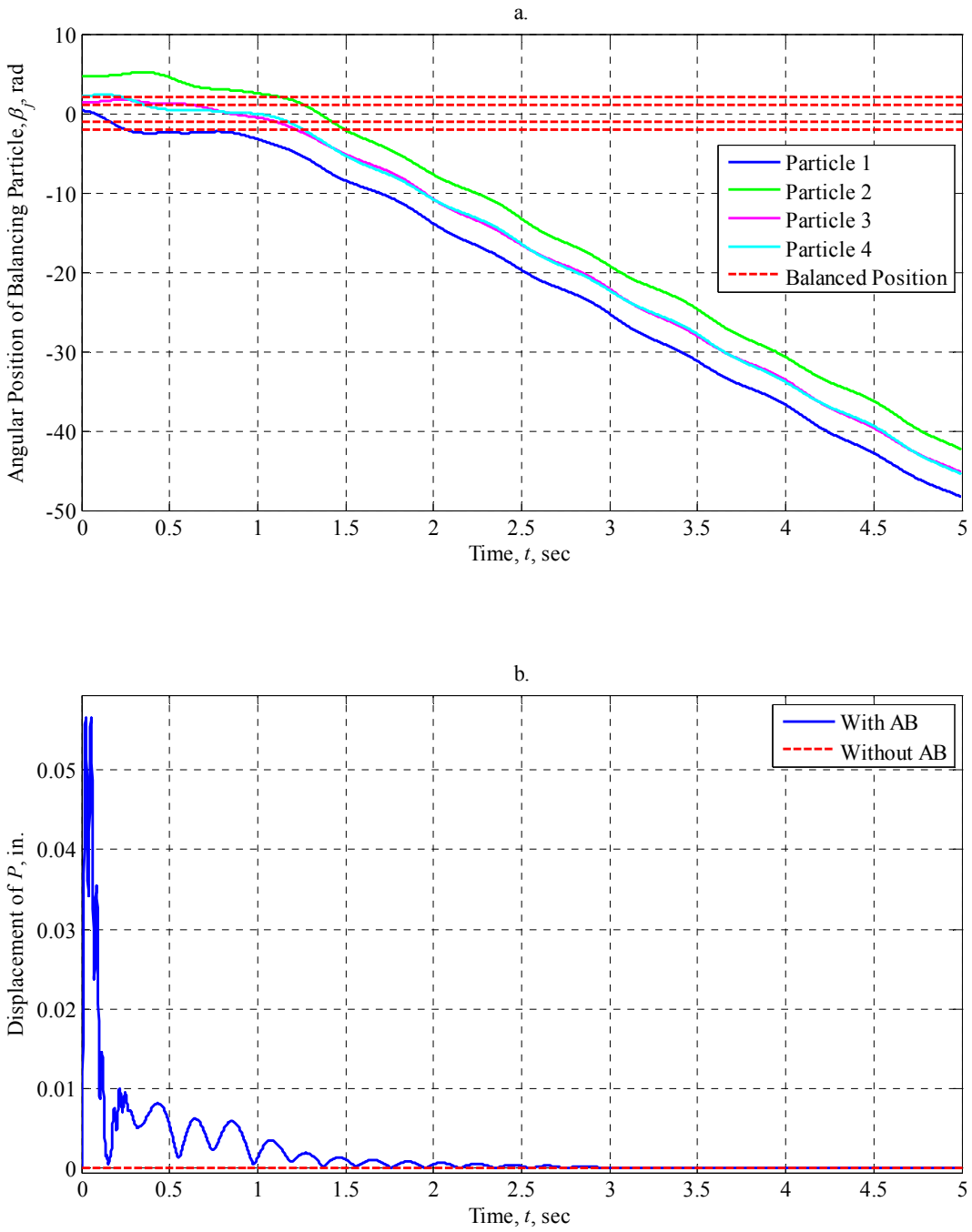

Figure 5.3.2.2.7-I: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

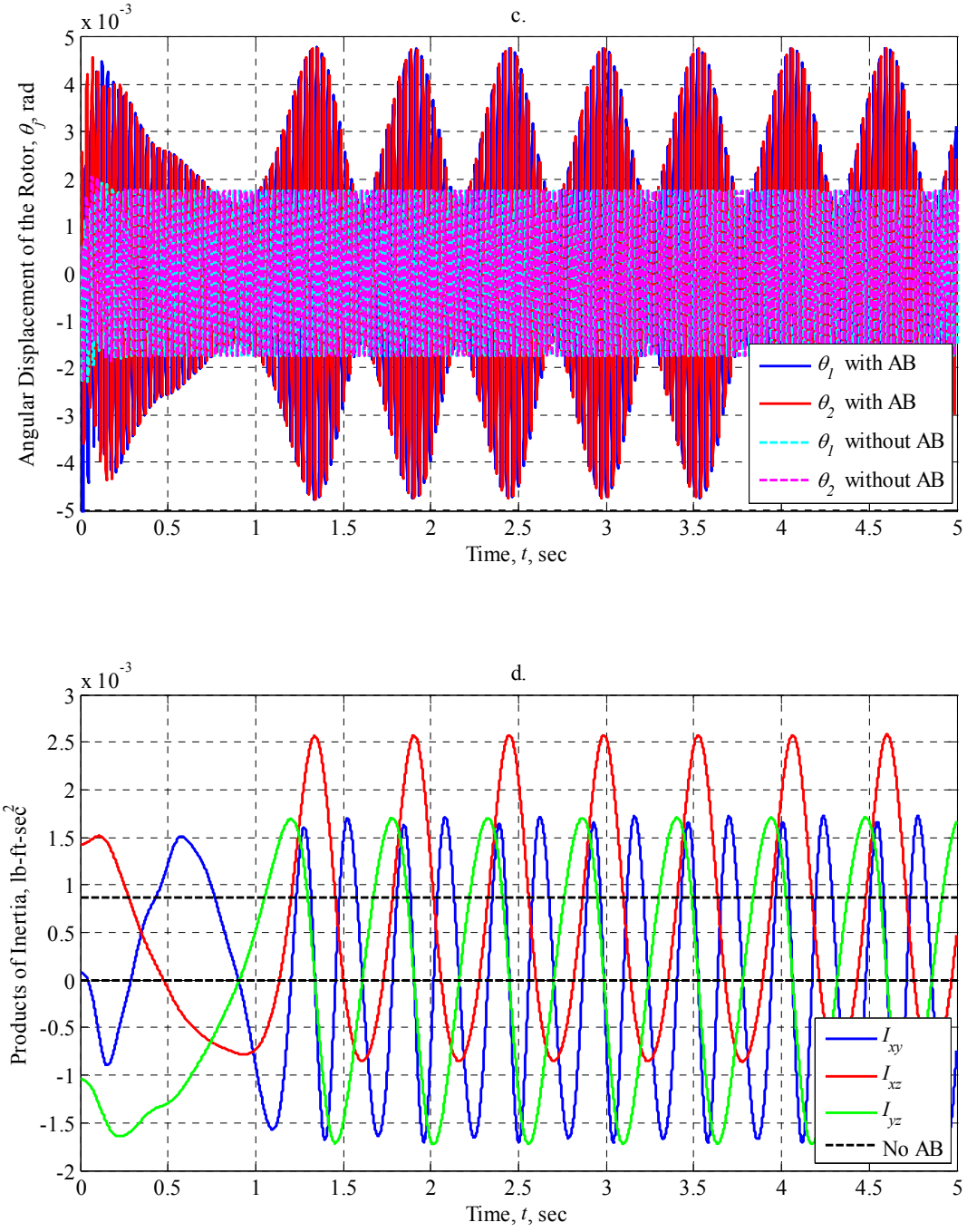

Figure 5.3.2.2.7-II: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

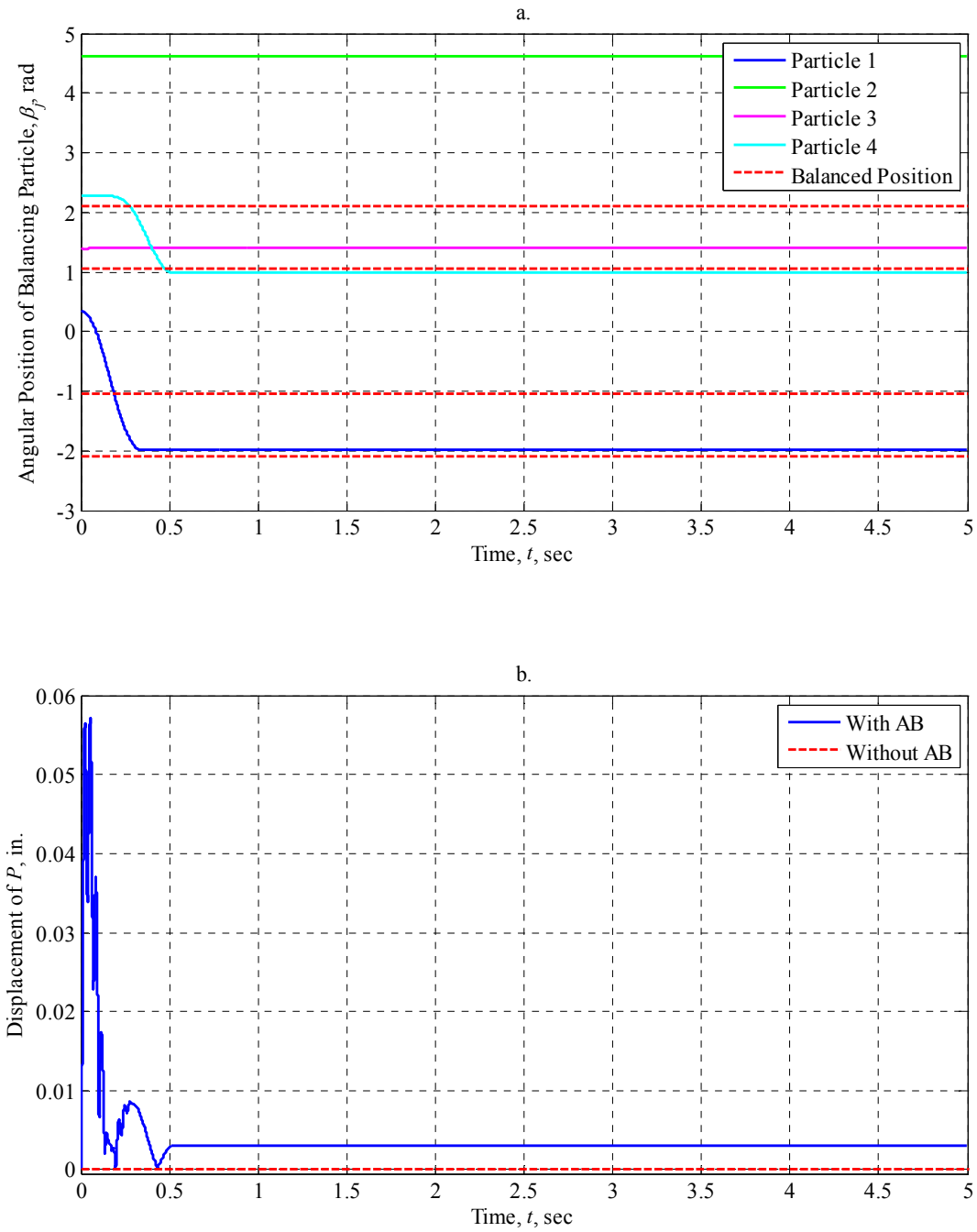

Figure 5.3.2.2.8-I: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

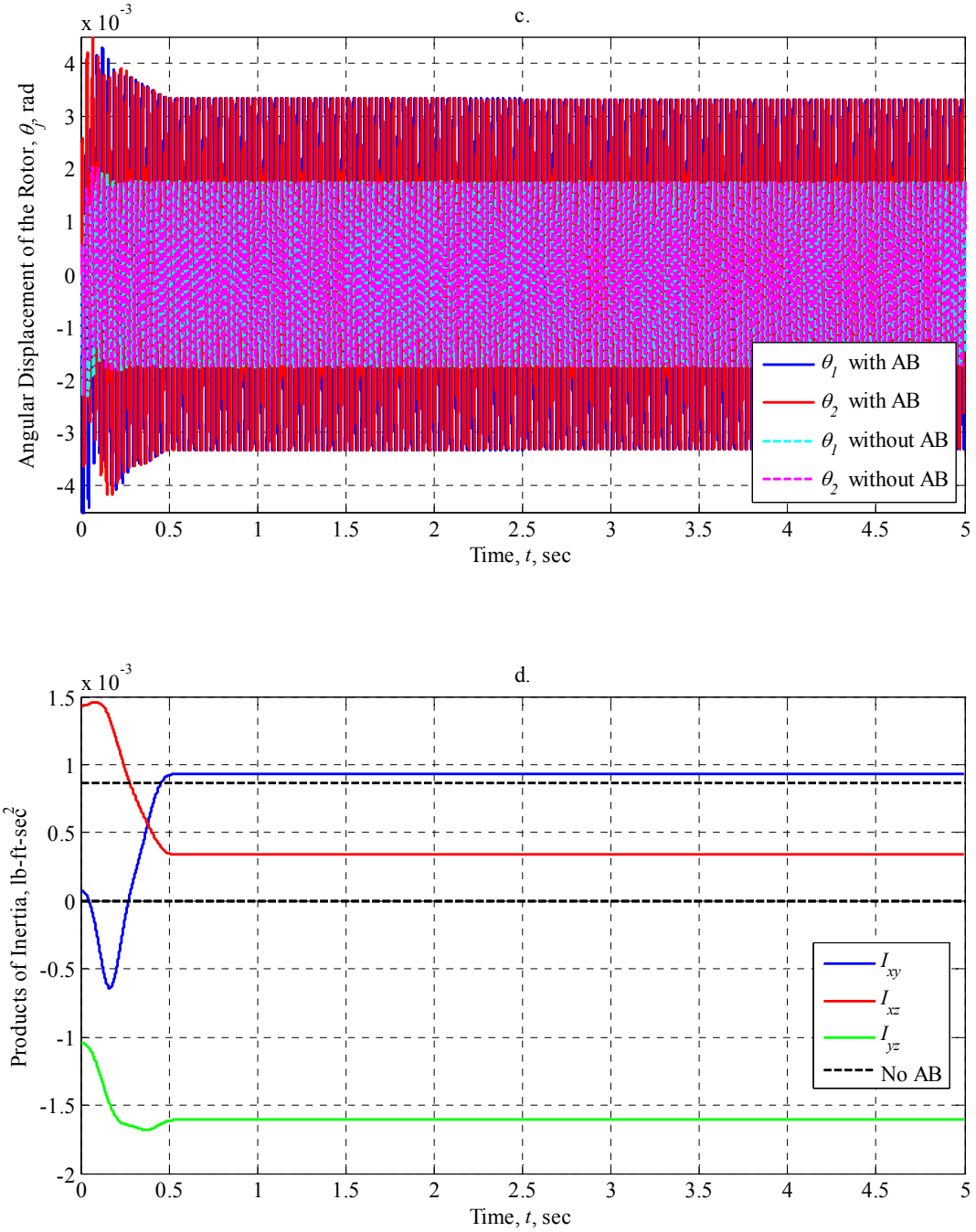

Figure 5.3.2.2.8-II: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

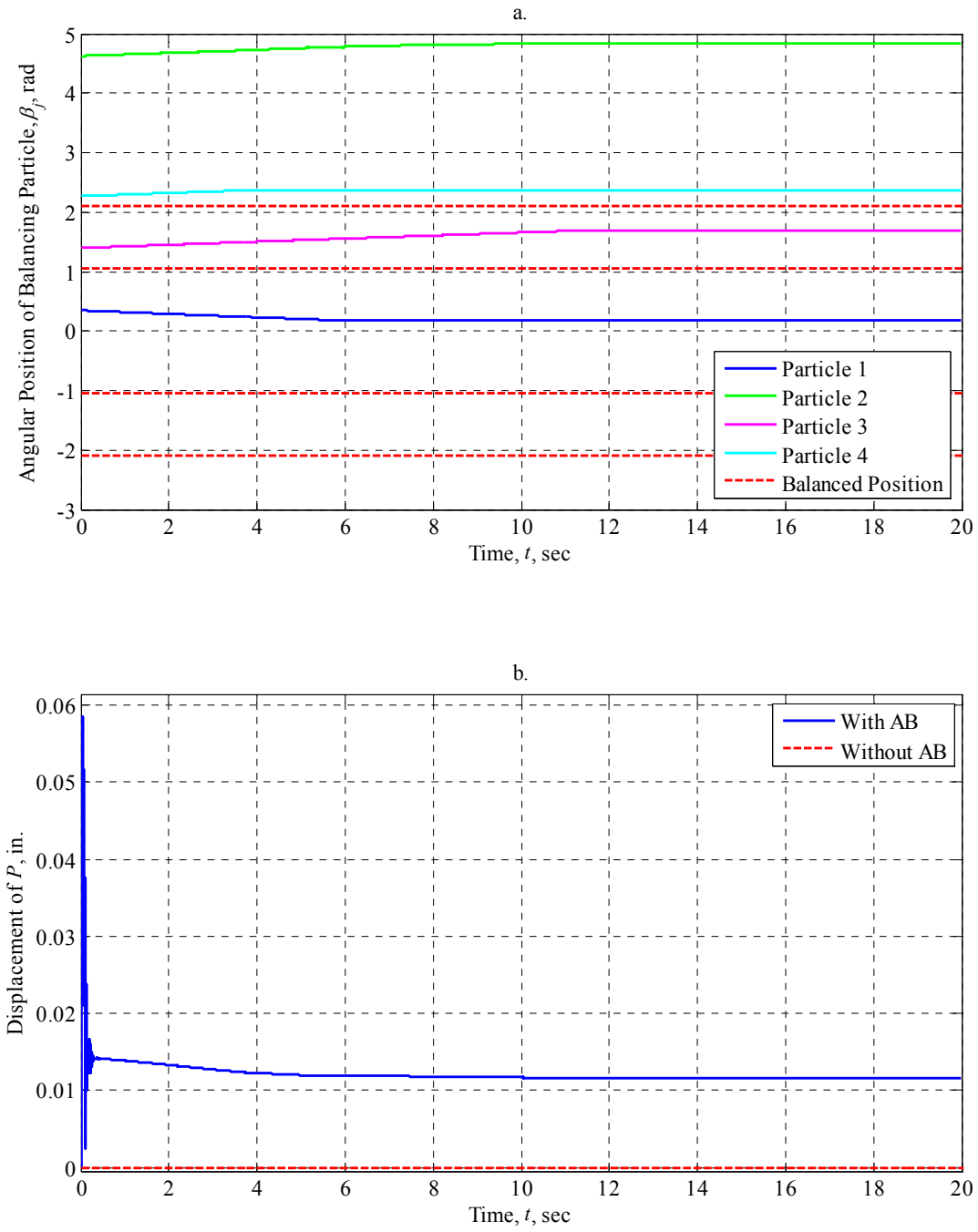

Figure 5.3.2.2.9-I: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

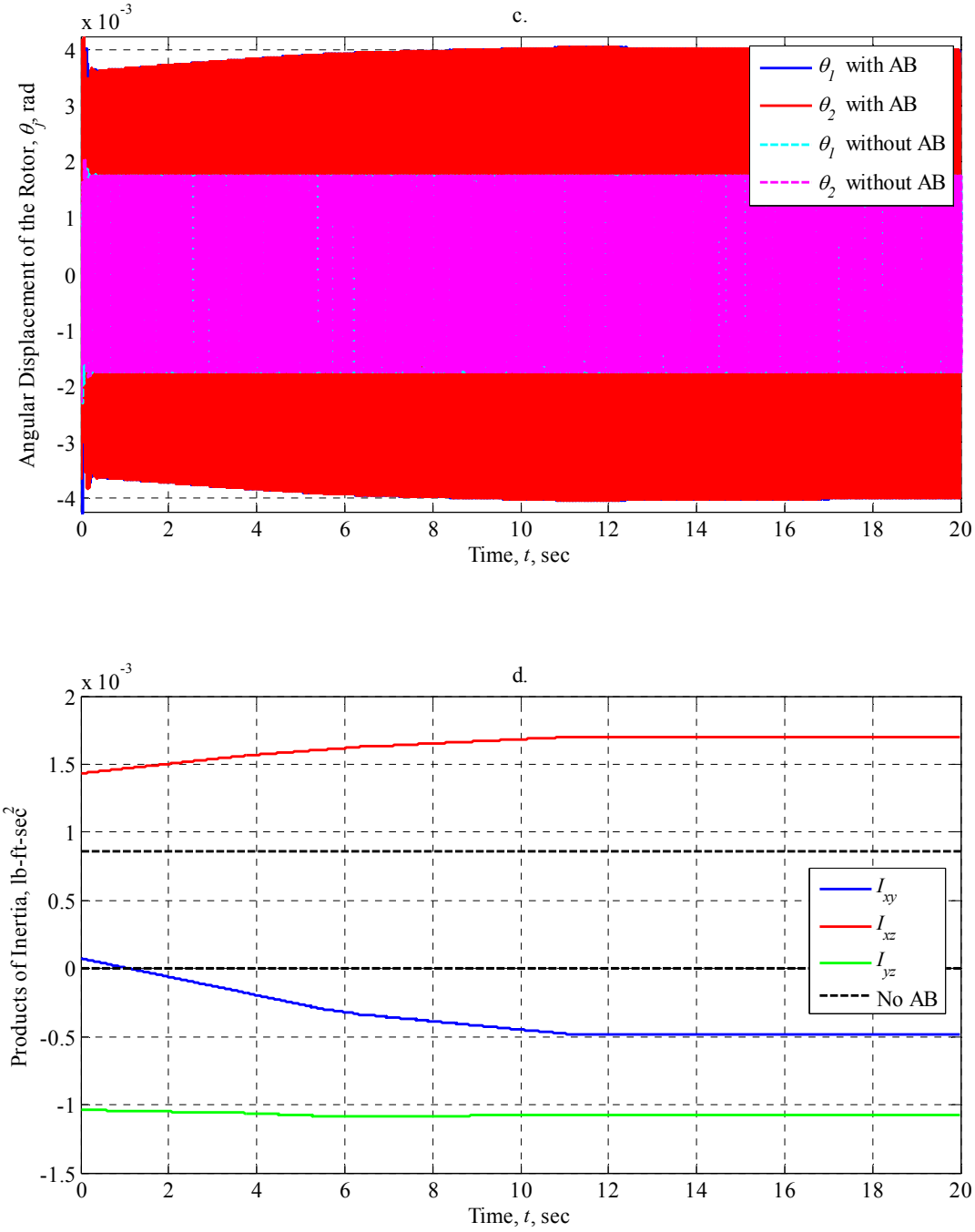

Figure 5.3.2.2.9-II: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Dynamically-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

5.4 Dual-Plane Balancing of an Arbitrarily Imbalanced Rotor

The final rotor we will consider features an arbitrarily located mass imbalance that creates both a dynamic imbalance effect and a static imbalance effect. The rotor is depicted in Figure 5.4.1, and possesses a single imbalance mass, denoted m and shown in blue, located at an arbitrary location within the rotor. The imbalance was taken to be offset one-fourth of the rotor depth d in the positive z-direction, and displaced radially three-fourths of the inner radius r_i from the geometric center of the rotor. The imbalance was given a zero angular displacement (or corresponding zero y-coordinate) from the body-fixed x-axis for simplification reasons. Four balancing particles are introduced in an attempt to balance the rotor. Masses m_1 and m_2 lie in a race at the front plane of the cylinder in the positive z-direction, while masses m_3 and m_4 lie in a race at the rear plane of the cylinder in the negative z-direction. Both front and rear races lie along the inner radius of the wheel. The balancing positions for the masses are found by again requiring the center of mass for the rotor, imbalance, and balancing masses to lie at the geometric center of the rotor, and for all products of inertia for the system to vanish. It is apparent that with the mass distribution for the balancing particles in the two races, it is impossible to eliminate a z-coordinate for the center of mass. It is possible to eliminate the remaining five conditions if the balancing masses seek out particular locations.

For balancing, particles m_1 and m_2 must travel to positions m_1' and m_2' located at ±106.34° off of the body-fixed x-axis. Particles m_3 and m_4 must travel to positions m_3' and m_4' located at ±95.38° off of the body-fixed x-axis. The balanced position of the particles in each race are symmetric about the body-fixed x-axis. In this configuration, the x- and y-coordinates for the center of mass are eliminated, as are the product of inertia terms. For these conditions the z-coordinate for the center of mass is non-zero.

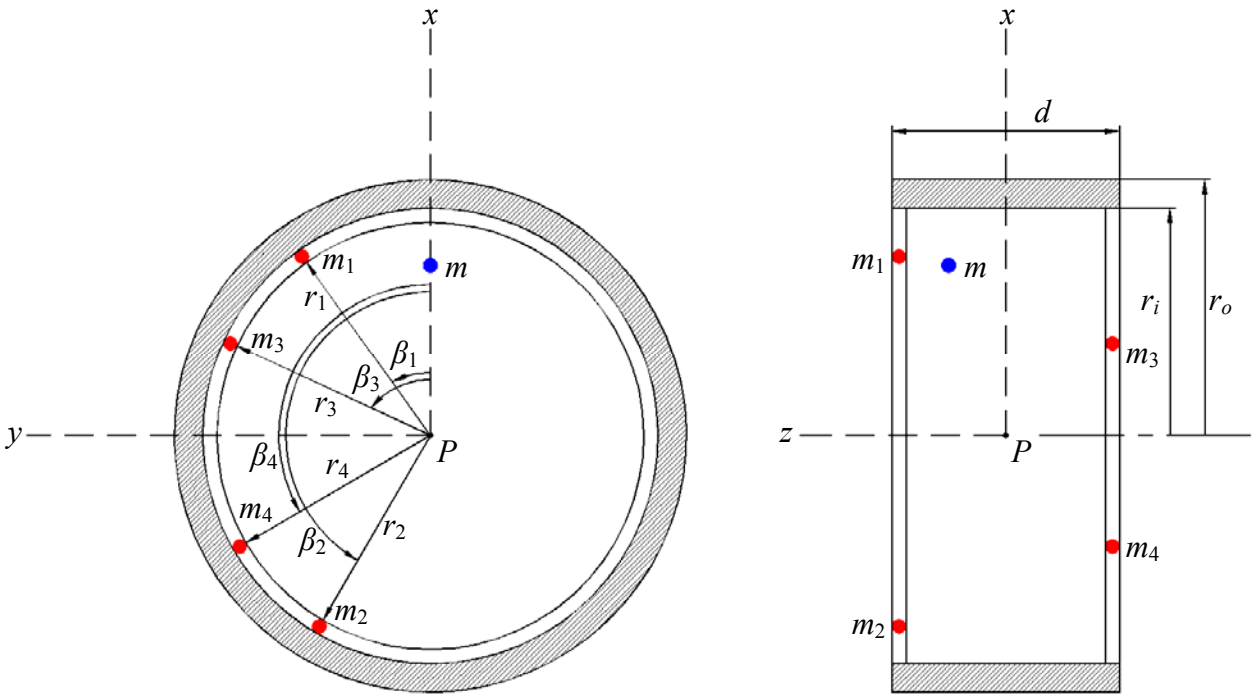

Figure 5.4.1: Dual-Plane Balancing of an Arbitrarily-Imbalanced Rotor Utilizing Two Balancing Particles per Race, Unbalanced Configuration

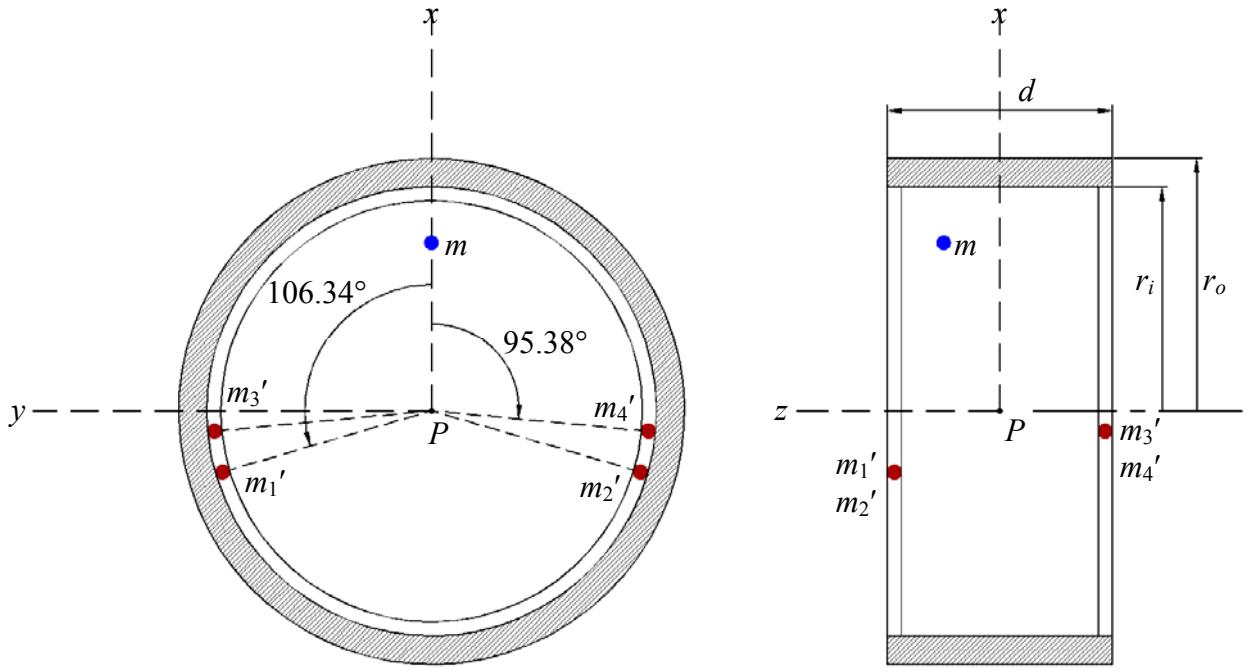

Figure 5.4.2: Dual-Plane Balancing of an Arbitrarily-Imbalanced Rotor Utilizing Two Balancing Particles per Race, Balanced Configuration

The calculation of the balanced angular positions follows.

$$\left.\begin{aligned}
\bar{x} &= m\left(\frac{3r_i}{4}\right)+m_1r_1\cos\beta_1+m_2r_2\cos\beta_2+m_3r_3\cos\beta_3+m_4r_4\cos\beta_4=0\\
\bar{y} &= m(0)+m_1r_1\sin\beta_1+m_2r_2\sin\beta_2+m_3r_3\sin\beta_3+m_4r_4\sin\beta_4=0\\
\bar{z} &= m\frac{d}{4}+m_1\frac{d}{2}+m_2\frac{d}{2}+m_3\left(\frac{-d}{2}\right)+m_4\left(\frac{-d}{2}\right)=0\\
I_{xy} &= m\left(\frac{3r_i}{4}\right)(0)+m_1r_1\cos\beta_1\left(r_1\sin\beta_1\right)+m_2r_2\cos\beta_2\left(r_2\sin\beta_2\right)+\\
& m_3r_3\cos\beta_3\left(r_3\sin\beta_3\right)+m_4r_4\cos\beta_4\left(r_4\sin\beta_4\right)=0\\
I_{xz} &= m\left(\frac{3r_i}{4}\right)\frac{d}{4}+m_1r_1\cos\beta_1\left(\frac{d}{2}\right)+m_2r_2\cos\beta_2\left(\frac{d}{2}\right)+\\
& m_3r_3\cos\beta_3\left(\frac{-d}{2}\right)+m_4r_4\cos\beta_4\left(\frac{-d}{2}\right)=0\\
I_{yz} &= m(0)\frac{d}{4}+m_1r_1\sin\beta_1\left(\frac{d}{2}\right)+m_2r_2\sin\beta_2\left(\frac{d}{2}\right)+\\
& m_3r_3\sin\beta_3\left(\frac{-d}{2}\right)+m_4r_4\sin\beta_4\left(\frac{-d}{2}\right)=0
\end{aligned}\right\}\rightarrow\begin{cases}\beta_1=\pm106.34^\circ\\ \beta_2=\mp106.34^\circ\\ \beta_3=\pm95.38^\circ\\ \beta_4=\mp95.38^\circ\end{cases}\tag{5.5}$$

Table 5.4.1 lists the selected system properties for the use of four balancing particles. To examine the balancing characteristics of the AB, the products of inertia for the cylinder and imbalance will also examined. To effectively balance the cylinder and eliminate unwanted vibrations, the products of inertia must vanish, transforming the body-fixed coordinate axes into principal axes. Additionally, care must be taken when choosing an operating speed for the rotor to ensure that a resonant frequency is not selected. Based on the chosen values for the system properties, Table 5.4.2 summarizes the pertinent natural frequencies and mass moment of inertia terms. Please note that the table lists characteristics based solely on the wheel and imbalance mass with no consideration of any balancing particles.

Table 5.4.1: Selected Values for System Properties (Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Property	Value	Property	Value
Mass of the rotor, M	1.242 slugs	Spring stiffness along X-axis, k_X	1000 lb/ft
Mass of the imbalance, m	0.001941 slugs	Spring stiffness along Y-axis, k_Y	1000 lb/ft
Mass of a balancing particle, m_j	0.001941 slugs	Spring stiffness along Z-axis, k_Z	100,000 lb/ft
Acceleration of gravity, g	32.2 ft/sec^2	Torsional stiffness along X-axis, k_{θ_1}	5000 lb-ft/rad
Inner radius of rotor, r_i	8 in.	Torsional stiffness along y_2-axis, k_{θ_2}	5000 lb-ft/rad
Outer radius of rotor, r_o	9 in.	Torsional stiffness along z-axis, k_{θ_3}	0
Radial distance of imbalance, r	6 in.	Damping coefficient along X-axis, c_X	49.3 lb-sec/ft †
Radial distance of balancing particle, r_j	8 in.	Damping coefficient along Y-axis, c_Y	49.3 lb-sec/ft †
Depth of the rotor, d	8 in.	Damping coefficient along Z-axis, c_Z	500
Internal offset of the imbalance from rotor mid-plane, d_m	2 in.	Damping coefficient along X-axis, c_{θ_1}	59.3 lb-ft-sec/rad †
Internal offset of the race containing the balancing particles, $d_{1,2}$	4 in.	Damping coefficient along y_2-axis, c_{θ_2}	59.3 lb-ft-sec/rad †
Internal offset of the race containing the balancing particles, $d_{3,4}$	-4 in.	Damping coefficient along z-axis, c_{θ_3}	0
Angular position of the imbalance from the x-axis, β_m	0°	Viscous drag coefficient for the balancing particles in the race, μ_c	0.005 lb-sec/ft

† Damping values were selected to provide a viscous damping factor of 0.7.

Table 5.4.3 lists the selected initial conditions for the simulations. Table 5.4.4 lists the range of Coulomb friction levels that will be explored for all numerical simulations. As before, the value of the coefficient of kinetic friction has been taken to be 75% of the value of the static friction. Table 5.4.5 lists the range of operating speeds that will be explored. The operating speeds were chosen based on the natural frequencies of the rotor listed in Table 5.4.2 and were staggered below and above each translational natural frequency with the exception of the highest natural frequency along the Z-axis.

Table 5.4.2: Undamped Natural Frequencies and Mass Moments of Inertia (Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Undamped Natural Frequency	Mass Moment of Inertia, lb-ft-sec^2
$\omega_{n_x} \simeq \sqrt{\frac{k_X}{M+m}} = 28.4$ rad/sec	$I_{xx} = 0.3592$
$\omega_{n_y} \simeq \sqrt{\frac{k_Y}{M+m}} = 28.4$ rad/sec	$I_{yy} = 0.3609$
$\omega_{n_z} \simeq \sqrt{\frac{k_Z}{M+m}} = 284$ rad/sec	$I_{zz} = 0.6272$
$\omega_{n_{\theta_1}} \simeq \sqrt{\frac{k_{\theta_1}}{I_{xx}}} = 118.1$ rad/sec	$I_{xy} = 0$
$\omega_{n_{\theta_2}} \simeq \sqrt{\frac{k_{\theta_2}}{I_{yy}}} = 117.9$ rad/sec	$I_{xz} = 1.617\left(10^{-4}\right)$
$\omega_{n_{\theta_3}} \simeq \sqrt{\frac{k_{\theta_3}}{I_{zz}}} = 0$	$I_{yz} = 0$

Table 5.4.3: Selected Values for Initial Conditions (Arbitrarily-Imbalanced Rotor)

Rotor Position	Rotor Velocity	Rotor Orientation	Rotor Angular Velocity	Balancing Mass Angular Position	Balancing Mass Angular Velocity
$X_0 = 0$	$\dot{X}_0 = 0$	$\theta_{1_0} = 0$	$\dot{\theta}_{1_0} = 0$	$\beta_{1_0} = 30°$	$\dot{\beta}_{1_0} = 0.15$ rad/sec
$Y_0 = 0$	$\dot{Y}_0 = 0$	$\theta_{2_0} = 0$	$\dot{\theta}_{2_0} = 0$	$\beta_{2_0} = 272°$	$\dot{\beta}_{2_0} = 0.15$ rad/sec
$Z_0 = 0$	$\dot{Z}_0 = 0$	$\theta_{3_0} = 0$	$\dot{\theta}_{3_0} = \omega_{3_0}$ (varies)	$\beta_{3_0} = 20°$	$\dot{\beta}_{3_0} = 0$
				$\beta_{4_0} = 300°$	$\dot{\beta}_{4_0} = -0.15$ rad/sec

Table 5.4.4: Coulomb Friction Levels of Interest for Numerical Simulations

Coefficient of Static Friction, μ_s	Coefficient of Kinetic Friction, μ_k
0	0
0.001	0.00075
0.1	0.075

Table 5.4.5: Operating Speeds for Arbitrarily-Imbalanced Rotor

Initial *z*-axis Angular Velocity, rad/sec
15
50
200

We will now determine the performance of the AB for the specified range of Coulomb friction levels and operating speeds. Since the general imbalance will create both translational and rotational motions of the rotor geometric center, the performance of the AB will be evaluated on whether or not it is able to eliminate or reduce these unwanted motions as compared to the unbalanced performance alone. Table 5.4.6 summarizes the behavior of the balancing particles at the selected operating speeds and Coulomb friction levels by listing their steady-state angular positions and ultimate stopping times. If a particle never come to rest relative to the cylinder during the time interval considered, the ultimate stopping time is determined as the time when the particle reaches and stays within 1° of its final steady-state angular position. Table 5.4.7 summarizes the behavior of the geometric center P at the selected operating speeds and Coulomb friction levels by listing the maximum amplitude of the displacement of P and the steady-state amplitude of the displacement of P. Table 5.4.8 summarizes the orientation of the rotor by listing the maximum angular displacement and steady-state angular displacement about the inertial X- and intermediate y_2-axes. Table 5.4.9 summarizes the translational and rotational motions of the rotor in the absence of an AB to establish a baseline for comparison purposes. Table 5.4.10 summarizes the effect of the AB on the rotor performance by comparing the translational and angular displacements of the rotor before and after the addition of a balancer by comparing the differences in the maximum and steady-state translational and angular displacements of the rotor as found in Tables 5.4.7, 5.4.8, and 5.4.9. Table 5.4.11 computes the maximum and steady-state values of the products of inertia for the rotor and balancer combination and Table 5.4.12 compares these values to the ones in Table 5.4.2 for the case of an unbalanced rotor alone.

Table 5.4.6: Behavior of the Balancing Particles (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Steady-State Angular Position, degrees				Ultimate Stopping Time, sec			
		β_1	β_2	β_3	β_4	β_1	β_2	β_3	β_4
15	0	—‡	—‡	—‡	—‡	—‡	—‡	—‡	—‡
	0.001	31.24	273.59	120.00	298.08	0.3143	0.5558	0.0673	0.5120
	0.1	30.04	272.04	120.00	299.96	0.020	0.2649	0.00*	0.0404
50	0	—‡	—‡	—‡	—‡	—‡	—‡	—‡	—‡
	0.001	30.43	232.22	131.84	211.10	0.0726	1.8402	5.1083	3.5393
	0.1	29.97	271.96	120.00	299.94	0.3605	0.4873	0.00*	0.5187
200	0	—‡	—‡	—‡	—‡	—‡	—‡	—‡	—‡
	0.001	—‡	—‡	—‡	—‡	—‡	—‡	—‡	—‡
	0.1	44.73	269.84	120.00	293.57	9.8059	1.3481	0.00*	4.0131

‡ Indicates that no steady-state value could be determined.

* Indicates that the particle never started moving.

Table 5.4.7: Behavior of the Geometric Center (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches
15	0	1.575	—‡
	0.001	0.5798	0.5778
	0.1	0.5842	0.5792
50	0	3.166	—‡
	0.001	3.173	0.6271
	0.1	3.192	2.052
200	0	8.377	—‡
	0.001	8.471	—‡
	0.1	8.7026	1.7386

‡ Indicates that no steady-state value could be determined.

Table 5.4.8: Angular Displacement of the Rotor (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Angular Displacement, (10^{-2}) degrees		Steady-State Amplitude of Angular Displacement, (10^{-2}) degrees	
		θ_1	θ_2	θ_1	θ_2
15	0	0.2436	0.2436	0.03535	0.03535
	0.001	0.1518	0.1614	0.1511	0.1511
	0.1	0.1484	0.1581	0.1471	0.1471
50	0	4.1234	4.1217	—‡	—‡
	0.001	2.2027	2.2017	2.1305	2.1296
	0.1	1.3226	1.3228	1.3188	1.3182
200	0	19.587	19.620	—‡	—‡
	0.001	21.040	21.041	—‡	—‡
	0.1	7.6359	8.3491	6.0252	6.0248

‡ Indicates that no steady-state value could be determined.

Table 5.4.9: Translational and Angular Displacement of the Rotor in the Absence of an Automatic Balancer (Vertically-Oriented Arbitrarily-Imbalanced Rotor)

Initial z-axis Angular Velocity, rad/sec	Maximum Displacement of P, (10^{-2}) inches	Steady-State Displacement of P, (10^{-2}) inches	Maximum Angular Displacement, (10^{-2}) degrees		Steady-State Amplitude of Angular Displacement, (10^{-2}) degrees	
			θ_1	θ_2	θ_1	θ_2
15	0.2538	0.2534	0.04060	0.04060	0.04054	0.04054
50	1.3896	0.8953	0.3626	0.3618	0.3568	0.3567
200	3.789	0.9363	2.0619	2.1950	1.8778	1.8780

Table 5.4.10: Comparison of Rotor Performance With and Without an Automatic Balancer (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Displacement of P, (10^{-2}) inches	Difference in Steady-State Displacement of P, (10^{-2}) inches	Difference in Maximum Angular Displacement of the Rotor, (10^{-2}) degrees		Difference in Steady-State Angular Displacement of the Rotor, (10^{-2}) degrees	
				θ_1	θ_2	θ_1	θ_2
15	0	+1.3212	—‡	+0.2030	+0.2030	-0.00519	-0.00519
	0.001	+0.3260	+0.3244	+0.1112	+0.1208	+0.1106	+0.1106
	0.1	+0.3304	+0.3258	+0.1078	+0.1175	+0.1065	+0.1065
50	0	+1.7764	—‡	+3.7608	+3.7599	—‡	—‡
	0.001	+1.7834	-0.2682	+1.8401	+1.8399	+1.7736	+1.7729
	0.1	+1.8024	+1.1567	+0.9600	+0.9610	+0.9619	+0.9615
200	0	+4.588	—‡	+17.5250	+17.4246	—‡	—‡
	0.001	+4.682	—‡	+18.978	+18.846	—‡	—‡
	0.1	+4.9136	+0.8023	+5.5740	+6.1541	+4.1473	+4.1467

‡ Indicates that no steady-state value could be determined.

A positive (+) sign in the above table indicates an increase (worsening) in the indicated quantities. A negative (-) sign indicates a decrease (improvement) in the indicated quantities.

Table 5.4.11: Products of Inertia for the Rotor in the Presence of an Automatic Balancer (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Maximum Product of Inertia, (10^{-4}) lb-ft-sec^2			Steady-State Product of Inertia, (10^{-4}) lb-ft-sec^2		
		I_{xy}	I_{xz}	I_{yz}	I_{xy}	I_{xz}	I_{yz}
15	0	17.253	8.618	-4.497	—‡	1.644	-0.0226
	0.001	-4.036	5.702	-2.154	-4.032	5.702	-1.997
	0.1	-4.036	5.507	-2.154	-4.036	5.507	-2.150
50	0	17.251	18.582	16.958	—‡	—‡	—‡
	0.001	8.519	9.808	-2.209	7.473	9.265	-2.209
	0.1	-4.036	5.505	-2.154	-4.027	5.505	-2.154
200	0	-15.908	18.841	-17.017	—‡	—‡	—‡
	0.001	16.986	18.865	-17.005	—‡	—‡	—‡
	0.1	-4.036	5.524	-2.154	-2.560	5.102	-1.059

‡ Indicates that no steady-state value could be determined.

Table 5.4.12: Comparison of Products of Inertia for the Rotor With and Without an Automatic Balancer (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

Initial z-axis Angular Velocity, rad/sec	Coefficient of Static Friction, μ_s	Difference in Maximum Product of Inertia, (10^{-4}) lb-ft-sec^2			Difference in Steady-State Product of Inertia, (10^{-4}) lb-ft-sec^2		
		I_{xy}	I_{xz}	I_{yz}	I_{xy}	I_{xz}	I_{yz}
15	0	+17.253	+7.000	+4.497	—‡	+0.027	+0.0226
	0.001	+4.036	+4.085	+2.154	+4.032	+4.085	+1.997
	0.1	+4.036	+3.889	+2.154	+4.036	+3.889	+2.150
50	0	+17.251	+16.965	+16.958	—‡	—‡	—‡
	0.001	+8.519	+8.191	+2.209	+7.473	+7.647	+2.209
	0.1	+4.036	+3.887	+2.154	+4.027	+3.887	+2.154
200	0	+15.908	+17.223	+17.017	—‡	—‡	—‡
	0.001	+16.986	+17.248	+17.005	—‡	—‡	—‡
	0.1	+4.036	+3.907	+2.154	+2.560	+3.485	+1.059

‡ Indicates that no steady-state value could be determined.

As before, a positive (+) sign indicates that the magnitude of the product of inertia increased (worsened) compared with the unbalanced rotor. Similarly, a negative (-) sign indicates that the magnitude of the product of inertia decreased (improved) compared with the unbalanced rotor.

Figures 5.4.2.1 through 5.4.2.9, grouped at the end of Section 5.4, show the behavior of the balancing masses and rotor. The plots are arranged in order of increasing operating speed and then in order of increasing Coulomb friction. As before, the a-part of the figure shows the behavior of the balancing particles while the b-part shows the magnitude of the displacement of the geometric center P, in inches. The c-part of the figure shows the orientation of the rotor

about the inertial X-axis and the intermediate y_2-axis, while the d-part of the figure shows the products of inertia for the system of the rotor, imbalance masses, and balancing particles. Note that the figures will span two pages per simulation and will feature the a- and b-parts on the first page and the c- and d-parts on the second. The labeling will be consistent between figures with the a- and b-parts post scripted with a "I" and the c- and d-parts post scripted with a "II" to denote the first and second pair of figures for the simulation, respectively.

5.4.1 Summary of Findings

For the cases where $\omega_3 = 15$ rad/sec, when the Coulomb friction level is $\mu_s = 0$, none of the balancing particles are able to locate any steady-state position, and instead, travel around the races together acting as a single particle. Particles 1 and 2 are together in the outer race while particles 3 and 4 move together in the inner race. All four particles occupy the same angular position and speed with respect to the body-fixed coordinate frame. This gives a periodic motion to the amplitude of displacement for the geometric center since there is also a static imbalance present in the rotor. It is clear from Figure 5.4.2.1-I that the overall performance of the rotor is degraded at this friction level with an average amplitude that is approximately 4.8 times larger than the unbalanced rotor alone. Additionally, the rotor experiences a maximum displacement of the geometric center of $1.575(10^{-2})$ inches, an increase of 521% over the unbalanced value of $0.2538(10^{-2})$ inches.

Once the Coulomb friction level is increased to $\mu_s = 0.001$, the particles come to rest relative to the rotor within a short period of time and occupy distinct positions. All four particles stop within 0.5558 seconds. None of the particle reaches a balanced position, so there is a steady-state amplitude for the geometric center of $0.5778(10^{-2})$ inches. While this is certainly better than the case of zero Coulomb friction, it still represents a 128.0% increase over the unbalanced value of $0.2534(10^{-2})$ inches. The maximum displacement of the geometric center at this friction level is $0.5798(10^{-2})$ inches, an increase of 128.4% over the unbalanced value.

When the Coulomb friction level is $\mu_s = 0.1$, the particles stop even faster needing only 0.2649 seconds for all four to come to rest. The particles are again unable to reach a balanced position which gives a steady-state displacement to the geometric center of $0.5792(10^{-2})$ inches, a slight increase over the values when $\mu_s = 0.001$, for an overall increase of 128.6%. The maximum displacement of the geometric center at this Coulomb friction level is $0.5842(10^{-2})$

inches which is an increase of 130.2% over the unbalanced value, but only slightly higher than the value at $\mu_s = 0.001$.

As was observed for the case of only two balancing particles the steady-state angular displacements of the rotor about the inertial X-axis and the intermediate y_2-axis decrease slightly from $0.04054(10^{-2})$ degrees to $0.03535(10^{-2})$ degrees for both the X- and y_2-axes, a reduction of approximately 12.8%, when the Coulomb friction level is set to zero. It is difficult to say for certain why the addition of balancing masses lowers the value slightly since there is no net improvement in the products of inertia for the system. When the Coulomb friction level is set to $\mu_s = 0.001$, the steady-state angular displacements about the X- and y_2-axes are both $0.1511(10^{-2})$ degrees, an increase of 273% over the unbalanced values. When the Coulomb friction level is set to $\mu_s = 0.1$, the steady-state values are slightly lower than the previous Coulomb friction level having a value of approximately $0.1471(10^{-2})$ degrees about each axis, an increase of 263% over the unbalanced rotor values.

The maximum angular displacement about the X- and y_2-axes is greatest when the Coulomb friction is set to zero, and has a value of $0.2436(10^{-2})$ degrees. This is an increase of 500% from the unbalanced value of $0.04060(10^{-2})$ degrees. At a Coulomb friction level of $\mu_s = 0.001$, the percentage increase drops to an average of 286% with an average maximum angular displacement of $0.1566(10^{-2})$ degrees about the X- and y_2-axes. The increase is similar at a friction level of $\mu_s = 0.1$, having an average maximum angular displacement of $0.1533(10^{-2})$ degrees, an increase of 277%. It is unclear why the increases in maximum angular displacement are so large, but it could owe to the location of the balancing masses being much farther from the geometric center than the imbalance mass.

The products of inertia for the system are not improved for any case. The unbalanced rotor features no I_{xy} or I_{yz} products of inertia, but the addition of four balancing masses results in non-zero values for these terms. When $\mu_s = 0$, Figure 5.3.2.1-II shows that I_{xz} and I_{yz} oscillate slightly about the unbalanced rotor values of $1.617(10^{-4})$ lb-ft-sec^2 and 0, which does not severely degrade performance, but certainly does not improve it. I_{xy} is seen to exhibit large oscillations reaching a maximum value of $17.253(10^{-4})$ lb-ft-sec^2. The oscillatory behavior is expected since the balancing particles are simply traveling around the races of the rotor, changing the products of inertia in a continuous fashion.

When the Coulomb friction is increased to $\mu_s = 0.001$ or 0.1, the particles are seen to stop very quickly, so the products of inertia reach steady-state values. I_{xy} becomes fixed at -4.032(10^{-4}) lb-ft-sec^2 when $\mu_s = 0.001$, and -4.036(10^{-4}) lb-ft- sec^2 when $\mu_s = 0.1$. I_{yz} increases dramatically from its previous friction level value to a steady-state value of -1.997(10^{-4}) lb-ft-sec^2 and -2.150 (10^{-4}) lb-ft-sec^2 for $\mu_s = 0.001$ and 0.1, respectively. I_{xz} becomes fixed at 5.702(10^{-4}) lb-ft-sec^2 when $\mu_s = 0.001$, and 5.507(10^{-4}) lb-ft-sec^2 when $\mu_s = 0.1$. These represent increases of 253% and 241% respectively from the unbalanced value of 8.627(10^{-4}) lb-ft-sec^2. The maximum values obtained for I_{xy} and I_{yz} occur when there is no Coulomb friction present with I_{xy} reaching a maximum of -17.253(10^{-4}) lb-ft-sec^2 and I_{yz} reaching a maximum of 4.497(10^{-4}) lb-ft-sec^2. The same is true for I_{xz} which reaches a maximum of 8.618(10^{-4}) lb-ft-sec^2.

For the cases where $\omega_3 = 50$ rad/sec, we observe a different type of behavior than seen in the 15 rad/sec simulations. When $\mu_s = 0$, the particles are unable to reach steady-state positions, and travel around the races in synchronous motion. The particles in each race come together to act as a single particle, but the particles in the two races are separated by 180°. This effectively cancels their imbalance, but is unable to eliminate the unbalance in the rotor caused by the presence of a static imbalance. As a result, the rotor geometric center again exhibits periodic motion. The difference between the 15 rad/sec simulations is that for the majority of the time, the amplitude of the displacement of the geometric center is lower than for the unbalanced rotor alone. It does have peak values that are higher than the value seen by the unbalanced rotor alone, as is clearly shown in Figure 5.4.2.4-I. Further, the period of the geometric center is characterized by two distinct saw tooth waves of different time length. The first wave has a period of approximately 14.47 seconds while the second has a period of approximately 17.89 seconds. The peak value of this motion is approximately 1.229(10^{-2}) inches, which is an increase of 37.3% over the unbalanced value of 0.8953(10^{-2}) inches. It is reasonable to consider this a slight improvement overall since the majority of the displacement amplitude of P is kept below the steady-state value for the unbalanced rotor. As far as maximum value is concerned, the rotor reaches 3.166(10^{-2}) inches, an increase of 127.8% over the unbalanced value of 1.3896(10^{-2}) inches.

The fact that the rotor experiences some improvement in motion characteristics is not surprising since it is operating above the translational natural frequency of the support and the

mechanism to drive the balancing particles to their balanced positions is present. As we have seen previously, there is no tendency to drive the particles to a position which cancels the dynamic imbalance effect caused by the mass imbalance.

When the Coulomb friction is increased to $\mu_s = 0.001$, the particles come to rest relative to the rotor, in distinct positions, but not in the required balanced positions. Particle 1 stops the quickest, needing only 0.0726 seconds. Ironically, Particle 3, which begins from rest, requires the longest amount of time at 5.1083 seconds before it stops. Since all four particles come to rest, the displacement of the geometric center assumes a steady-state value of $0.6271(10^{-2})$ inches, which is a decrease of 30.0% from the unbalanced value of $0.8953(10^{-2})$ inches. The rotor geometric center has a maximum displacement of $3.173(10^{-2})$ inches for this friction level which is an increase of 128.3% over the unbalanced value.

Once the Coulomb friction increases to $\mu_s = 0.1$, all four particles stop much faster, requiring only 0.5187 seconds until all relative motion is arrested. Particle 3 never moves from its initial position. This results in a steady-state amplitude of the geometric center of $2.052(10^{-2})$ inches which is an increase of 129.2% over the unbalanced value and much worse than the value observed when $\mu_s = 0.001$. The maximum displacement of the geometric center at this Coulomb friction level is $3.192(10^{-2})$ inches, an increase of 129.7% over the unbalanced value.

The steady-state amplitudes for both angular displacements about the inertial X- and intermediate y_2-axes are worsened in all cases indicating there is no improvement in the products of inertia for the rotor. When $\mu_s = 0$, no steady state amplitudes are achieved as shown in Figure 5.4.6-II, as the motions about the X- and y_2-axes exhibit periodic motion. The motion for each angle reaches a periodic maximum value of approximately $4.122(10^{-2})$ degrees, an increase of 1055% over the unbalanced value of $0.3568(10^{-2})$ degrees. Once the Coulomb friction level increases to $\mu_s = 0.001$, a steady state is achieved where the values are $2.1305(10^{-2})$ degrees and $2.1296(10^{-2})$ degrees about the X- and y_2-axes, respectively. These numbers represent increases of 497% from the unbalanced values of approximately $0.3568(10^{-2})$ degrees. When $\mu_s = 0.1$, the values drop significantly to $1.3188(10^{-2})$ degrees and $1.3182(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of 270% from the unbalanced values. This is essentially the same phenomenon observed at increased friction levels when the operating speed was 15 rad/sec.

The maximum amplitudes for the angular displacements also increase in every case, but are seen to decrease as the Coulomb friction level increases. When $\mu_s = 0$, the maximum values are $4.1234(10^{-2})$ degrees about the X-axis and $4.1217(10^{-2})$ degrees about the y_2-axis, representing increases of 1037% and 1039% respectively from the unbalanced values of $0.3626(10^{-2})$ degrees for the X-axis and $0.3618(10^{-2})$ degrees about the y_2-axis. When $\mu_s = 0.001$, the maximum values drop significantly from the previous Coulomb friction level to $2.2027(10^{-2})$ degrees about the X-axis and $2.2017(10^{-2})$ degrees about the y_2-axis, representing overall increases of 507% and 509% respectively from the unbalanced values. Once $\mu_s = 0.1$ there is another significant reduction as compared to the previous friction value where the maximum values are $1.3226(10^{-2})$ degrees about the X-axis and $1.3228(10^{-2})$ degrees about the y_2-axis, representing increases of 265% and 266%, respectively, from the unbalanced values. It is no surprise that the changes are similar for both axes since the support characteristics are identical in the plane of the rotor, and the overall angular displacement about intermediate axes is small. The values for the highest Coulomb friction level are similar to what was observed at 15 rad/sec, but the lower two friction levels have much different effects.

The products of inertia are worsened for all values of Coulomb friction at this operating speed. When $\mu_s = 0$, the three products of inertia are seen to have periodic behavior with amplitudes that far exceed the unbalanced values for the rotor. The maximum values reached by the products of inertia at this operating speed all occur at the zero Coulomb friction level, and are $17.251(10^{-4})$ lb-ft-sec^2, $18.582(10^{-4})$ lb-ft-sec^2, and $16.598(10^{-4})$ lb-ft-sec^2 for I_{xy}, I_{xz}, and I_{yz}, respectively.

When $\mu_s = 0.001$, the particles come to rest and the products of inertia reach steady state values. I_{xy} reaches a value of $7.473(10^{-4})$ lb-ft-sec^2 and I_{yz} reaches a value of $-2.209(10^{-4})$ lb-ft-sec^2. At this Coulomb friction level, these products of inertia have maximum values of $8.519(10^{-4})$ lb-ft-sec^2 and $-2.209(10^{-4})$ lb-ft-sec^2 for I_{xy} and I_{yz} respectively. These are significant reductions from the zero Coulomb friction levels of 50.6% and 87.0%, for I_{xy} and I_{yz} respectively. I_{xz} reaches a steady-state value of $9.265(10^{-4})$ lb-ft-sec^2 representing an increase of 473% over the unbalanced value. I_{xz} reaches has a maximum value of $9.808(10^{-4})$ lb-ft-sec^2 representing a 507% increase over the unbalanced value.

When $\mu_s = 0.1$, I_{xy} reaches a value lower than at the previous Coulomb friction level of $-4.027(10^{-4})$ lb-ft-sec^2. I_{yz} reaches a maximum and steady-state value of $-2.154(10^{-4})$ lb-ft-sec^2.

At this friction level, I_{xy} reaches a maximum value of -4.036(10^{-4}) lb-ft-sec^2. I_{xz} reaches a value lower than at the previous Coulomb friction level of 5.505(10^{-4}) lb-ft-sec^2 representing a massive increase of 240% over the unbalanced value. The maximum value which I_{xz} reaches is the same as its steady-state value.

For the cases where $\omega_3 = 200$ rad/sec, we see results similar to the 50 rad/sec simulations. When Coulomb friction is zero, the four balancing particles are unable to reach steady-state values. They instead travel around the races in synchronous motion with the particles in each race exhibiting small oscillations about one another. The pairs of particles in the front and rear races are separated by 180°, effectively cancelling their imbalance which has the effect of keeping the long-term value of the amplitude of displacement of the geometric center below its unbalanced value. This is shown in Figure 5.4.2.7-I where again we see the motion of the geometric center consisting of two distinct wave forms. The first wave, beginning at approximately 2.91 seconds has a period of 0.297 seconds with a peak value of approximately 1.110(10^{-2}) inches, a modest increase of 18.6% over the unbalanced value of 0.9363(10^{-2}) inches. The second wave has a period of approximately 0.259 seconds with a peak value of 0.871(10^{-2}) inches, a slight decrease of 6.97% from the unbalanced value. The maximum displacement of the geometric center at this friction level is 8.377(10^{-2}) inches, an increase of 121.1% over the unbalanced value of 3.789(10^{-2}) inches.

A similar scenario is observed once the friction is increased to $\mu_s = 0.001$. Again, none of the particles reach a steady-state value, but instead move together in synchronous motion in the races. The pair of particles in each race occupies the same angular positions, and the pairs of particles situate themselves 180° apart and effectively cancel their own imbalance. This results in a periodic motion of the geometric center where its amplitude of displacement stays at or below the unbalanced level for the duration of the simulation. This is shown in Figure 5.4.2.8-I, where we again see the presence of two distinct wave forms in the motion. The shorter of the two waves has a period of approximately 0.725 seconds with a peak amplitude of 0.936(10^{-2}) inches. The longer wave has a period of approximately 1.181 seconds with approximately the same peak amplitude as the shorter wave.

When the Coulomb friction is increased to $\mu_s = 0.1$, no improvement in the behavior of the geometric center is observed. As noted previously, the stop times for the particles increase when compared to the same Coulomb frictional values at an operating speed of 50 rad/sec. The

times are 9.8059, 1.3481, and 4.0131 seconds for particles 1, 2, and 4, respectively. Since Particle 3 was given zero initial angular velocity, it was never able to move from its initial position. The stop times for the particles at 50 rad/sec were 0.3605, 0.4873, and 0.5187 seconds for particles 1, 2, and 4, respectively. These represent increases of 2620% for Particle 1, 176.6% for Particle 2, and 674% for Particle 4. At this Coulomb friction level, there is a resulting steady-state amplitude of the geometric center of $1.7386(10^{-2})$ inches. This is an increase of 85.7% over the unbalanced value, but a decrease of 15.3% over the value observed at the same Coulomb friction level but an operating speed of 50 rad/sec.

The steady-state amplitudes for both angular displacements about the inertial X- and intermediate y_2-axes are worsened in all cases at the increased operating speed, and the two lowest friction levels result in oscillatory behavior of the angles. When $\mu_s = 0$ and 0.001, no steady-state values could be obtained since the two angular displacements were seen to exhibit periodic motion as seen in Figures 5.4.2.7-II and 5.4.2.8-II. When $\mu_s = 0.1$, the steady-state amplitudes are $6.0252(10^{-2})$ degrees and $6.0248(10^{-2})$ degrees about the X- and y_2-axes, respectively, representing increases of approximately 221% from the unbalanced values. The percent increase at this Coulomb friction level has been observed in all three cases.

The maximum amplitudes for the angular displacements tend to decrease as the friction level increases. When $\mu_s = 0$, the maximum values are $19.587(10^{-2})$ degrees about the X-axis and $19.620(10^{-2})$ degrees about the y_2-axis, representing increases of 850% and 794% respectively from the unbalanced values of $2.0619(10^{-2})$ degrees for the X-axis and $2.1950(10^{-2})$ degrees about the y_2-axis. When $\mu_s = 0.001$, the maximum values increase slightly to $21.040(10^{-2})$ degrees about the X-axis and $21.041(10^{-2})$ degrees about the y_2-axis, representing increases of 920% and 859% respectively from the unbalanced values. When $\mu_s = 0.1$, the maximum values drop significantly from the previous friction level to $7.6359(10^{-2})$ degrees about the X-axis and $8.3491(10^{-2})$ degrees about the y_2-axis, representing increases of 270% and 280% respectively from the unbalanced values.

For this operating speed, the products of inertia only settle to steady values when the Coulomb friction is at its highest level. When $\mu_s = 0$ or 0.001, all three products of inertia exhibit oscillatory motion as seen in Figures 5.4.2.7-II and 5.4.2.8-II. When $\mu_s = 0$, the products of inertia have maximum values of $15.908(10^{-4})$ lb-ft-sec^2 for I_{xy}, $17.017(10^{-4})$ lb-ft-sec^2 for I_{xz}, and

$17.223(10^{-4})$ lb-ft-sec^2 for I_{xz}. When $\mu_s = 0.001$, the products of inertia have maximum values of $16.986(10^{-4})$ lb-ft-sec^2 for I_{xy}, $17.005(10^{-4})$ lb-ft-sec^2 for I_{xz}, and $17.248(10^{-4})$ lb-ft-sec^2 for I_{xz}.

When $\mu_s = 0.1$, I_{xy} reaches a steady state value of $-2.560(10^{-4})$ lb-ft-sec^2 while I_{yz} reaches a steady-state value of $-1.059(10^{-4})$ lb-ft-sec^2. I_{xz} reaches a steady-state value of $5.102(10^{-4})$ lb-ft-sec^2 representing an increase of 216% over the unbalanced value of $1.617(10^{-4})$ lb-ft-sec^2. At this Coulomb friction level, the maximum value of I_{xy} decreases significantly from previous friction levels to $-4.036(10^{-4})$ lb-ft-sec^2. I_{yz} also drops significantly to $-2.154(10^{-4})$ lb-ft-sec^2. I_{xz} has a large decrease over previous friction levels as well, falling to $5.524(10^{-4})$ lb-ft-sec^2.

In general, we find that the particles have no affinity for selecting balancing positions to place the center of mass for the combined rotor/balancer system at the geometric center of the rotor while simultaneously eliminating the products of inertia for the system. When operated above the translational natural frequency of the rotor, the balancing particles are able to offer some improvement over the unbalanced case, but there is still a steady-state motion of the geometric center than can be larger than the unbalanced amplitude or smaller depending on the operating speed selected. A speed of 50 rad/sec in the absence of Coulomb friction allowed the geometric center to spend much of its time below the unbalanced vibration level but still exhibit a periodic behavior. An increase to 200 rad/sec allowed the geometric center to spend nearly all of its time below the unbalanced amplitude. At these speeds, the products of inertia and angular displacements about intermediate axes were dramatically increased showing little or no improvement in balancing properties for the angular properties of the rotor/imbalance system. Further, at zero Coulomb friction levels, the particles exhibited synchronous motion where the pairs of particles in each race would move in phase with one another below the translational natural frequency of the support, or out of phase with each other above the translational natural frequency. This led to exacerbated steady-state amplitude of displacement for the geometric center below the natural frequency, and an improved state when above the natural frequency. Once the Coulomb friction levels were increased, the balancing particles were seen to move for longer periods of time at higher operating speeds. Overall behavior of the rotor was worsened because friction opposes the motion of the balancing particles as they seek out their balanced positions when operated above the natural frequency.

5.4.2 Plots of Rotor Performance and Balancing Particle Behavior

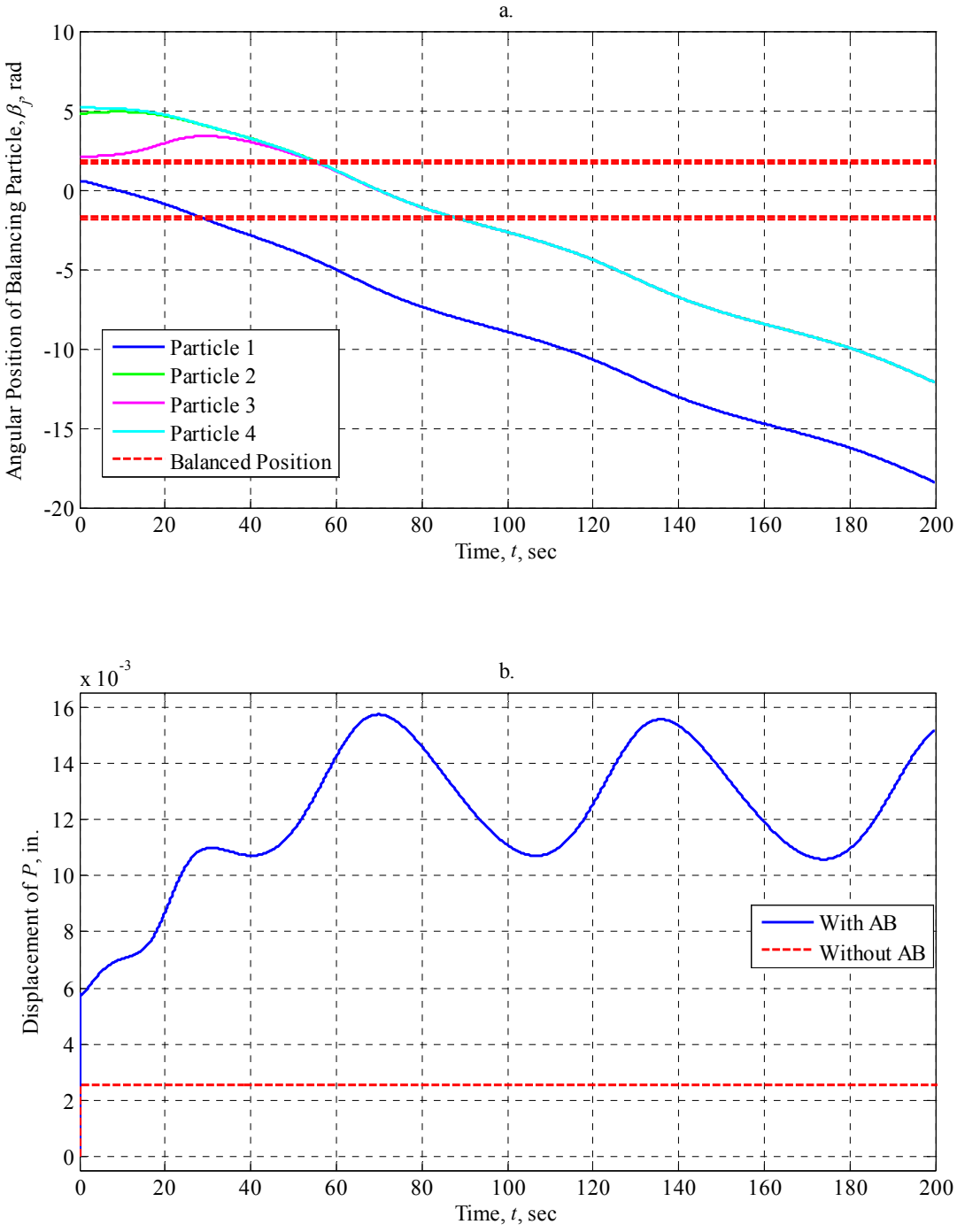

Figure 5.4.2.1-I: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

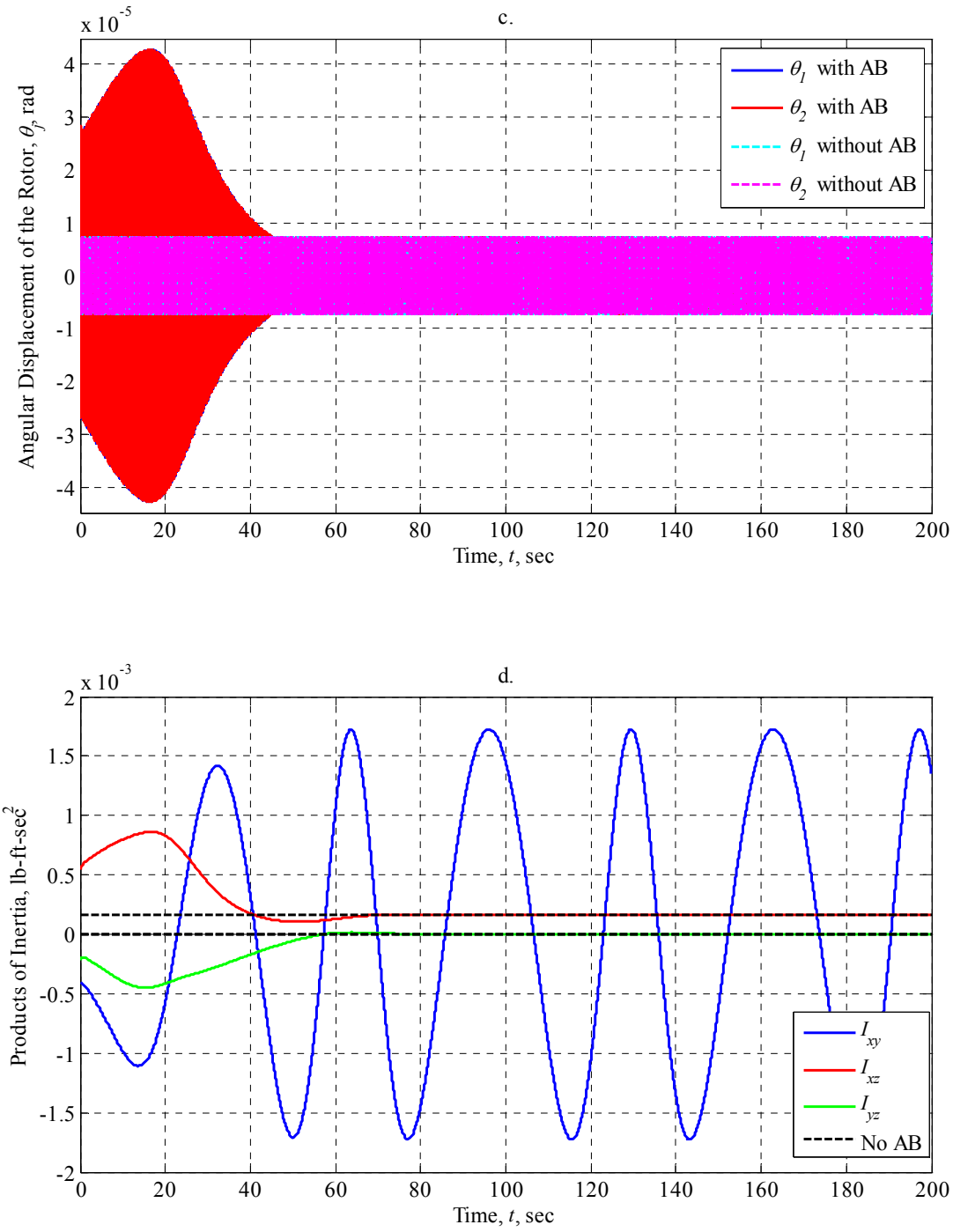

Figure 5.4.2.1-II: AB Performance at 15 rad/sec with $\mu_s = 0$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

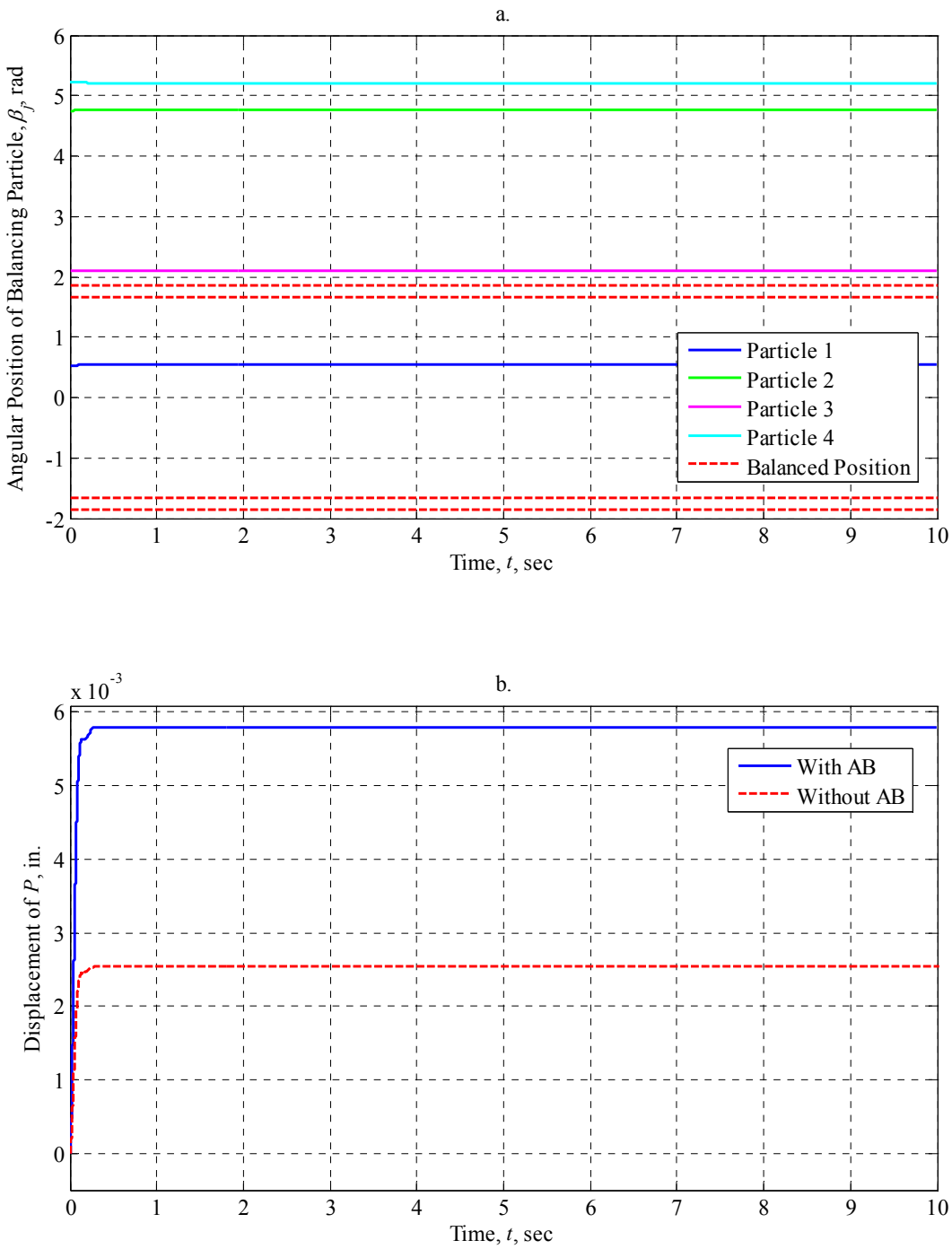

Figure 5.4.2.2-I: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

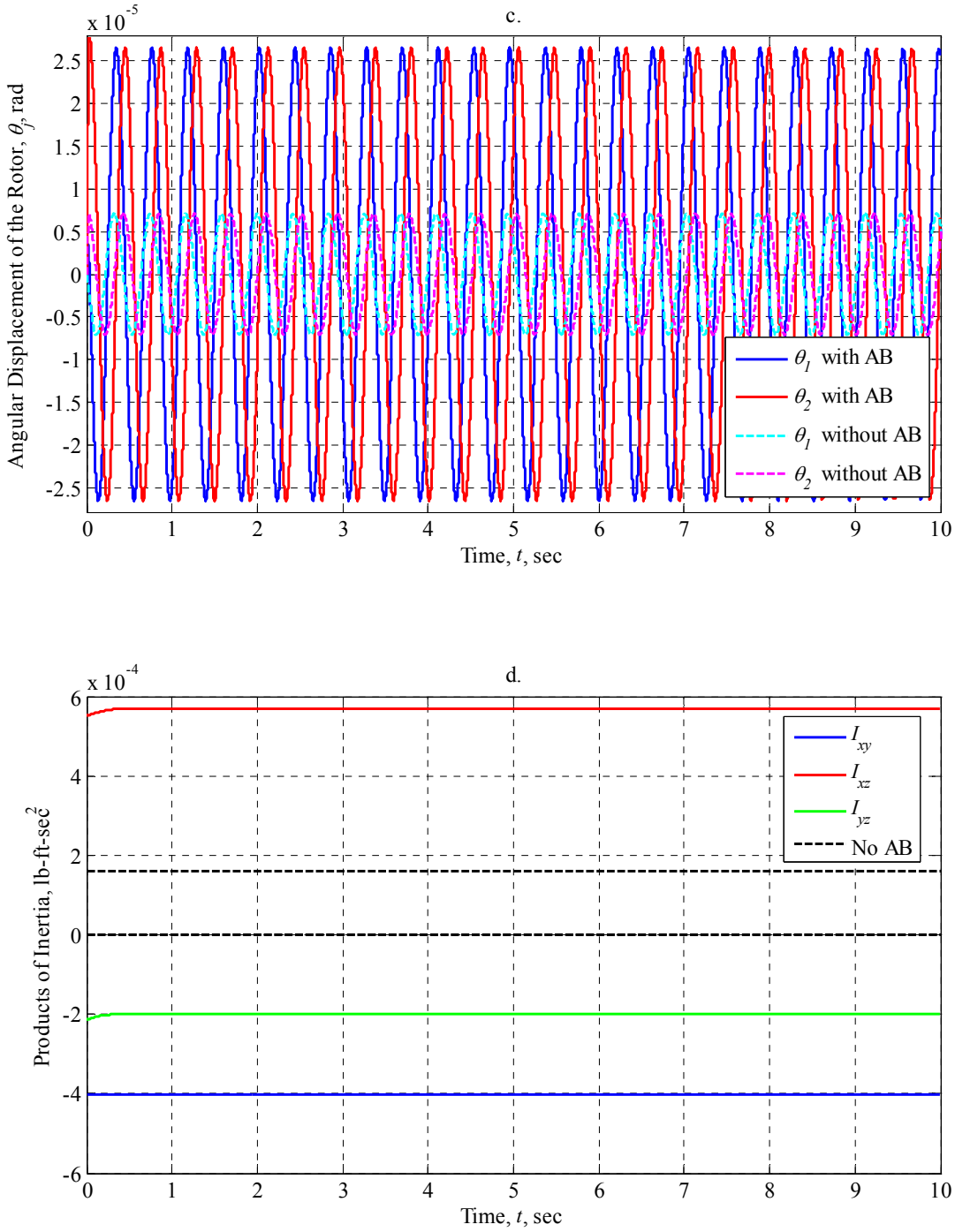

Figure 5.4.2.2-II: AB Performance at 15 rad/sec with μ_s = 0.001 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

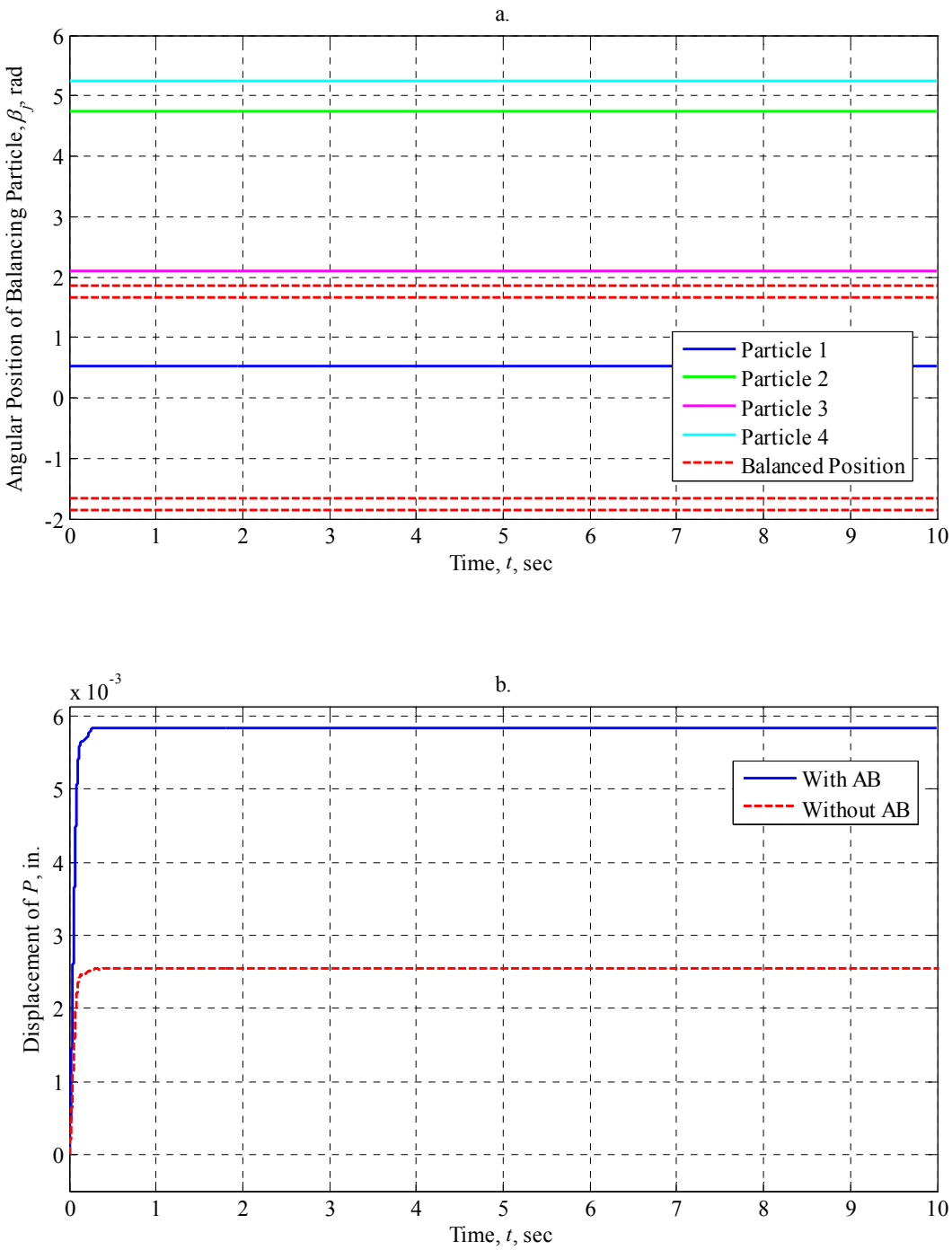

Figure 5.4.2.3-I: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

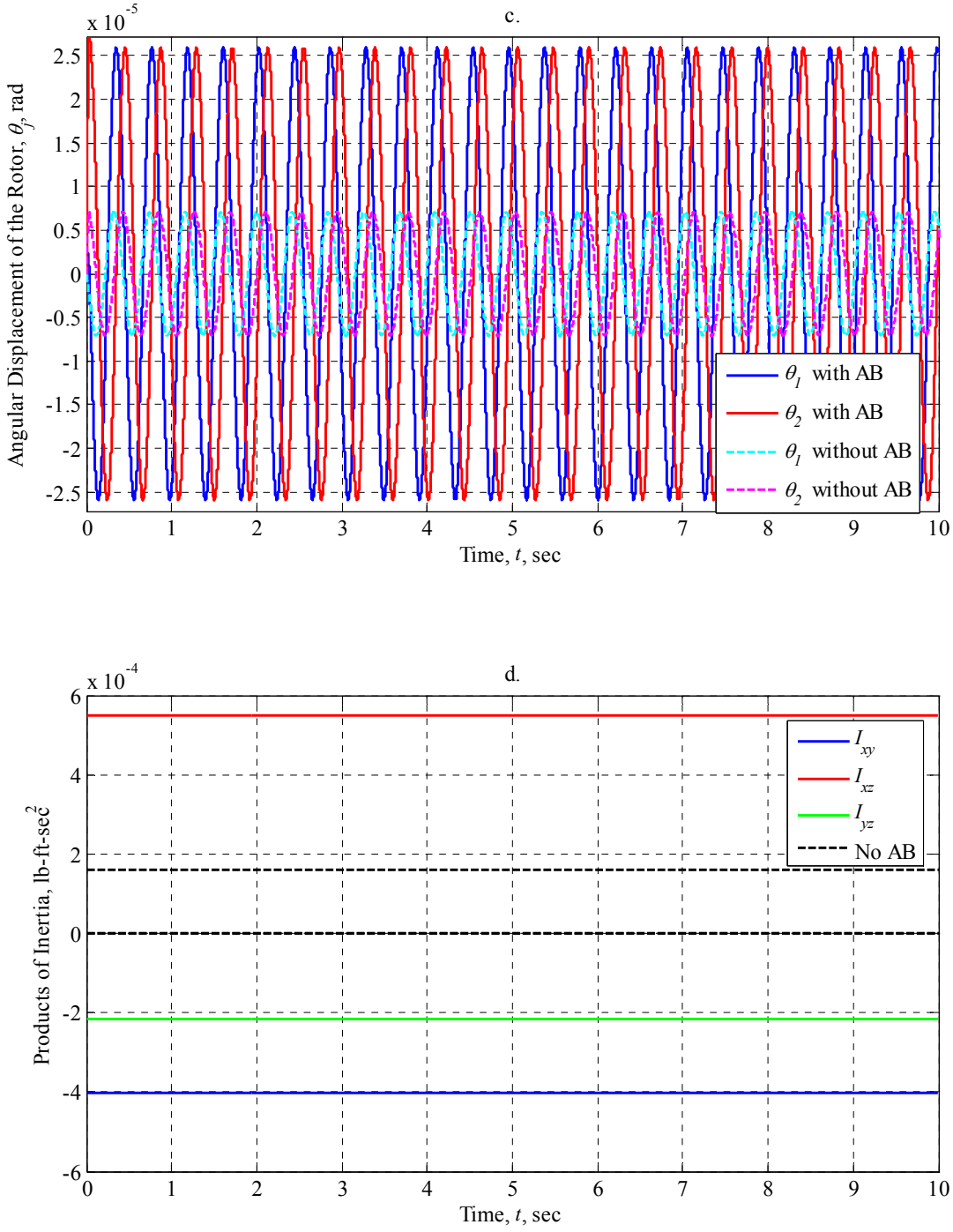

Figure 5.4.2.3-II: AB Performance at 15 rad/sec with μ_s = 0.1 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

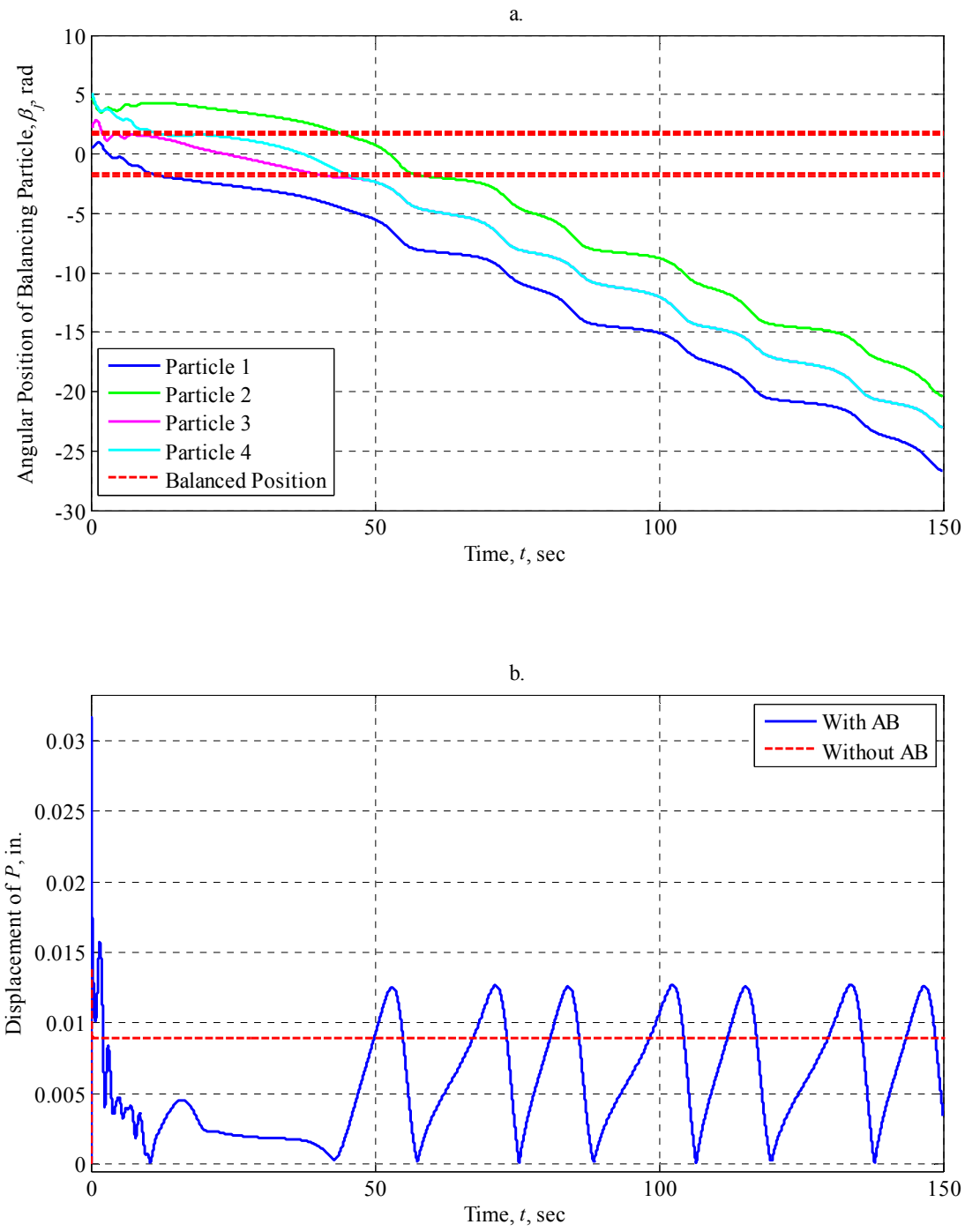

Figure 5.4.2.4-I: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

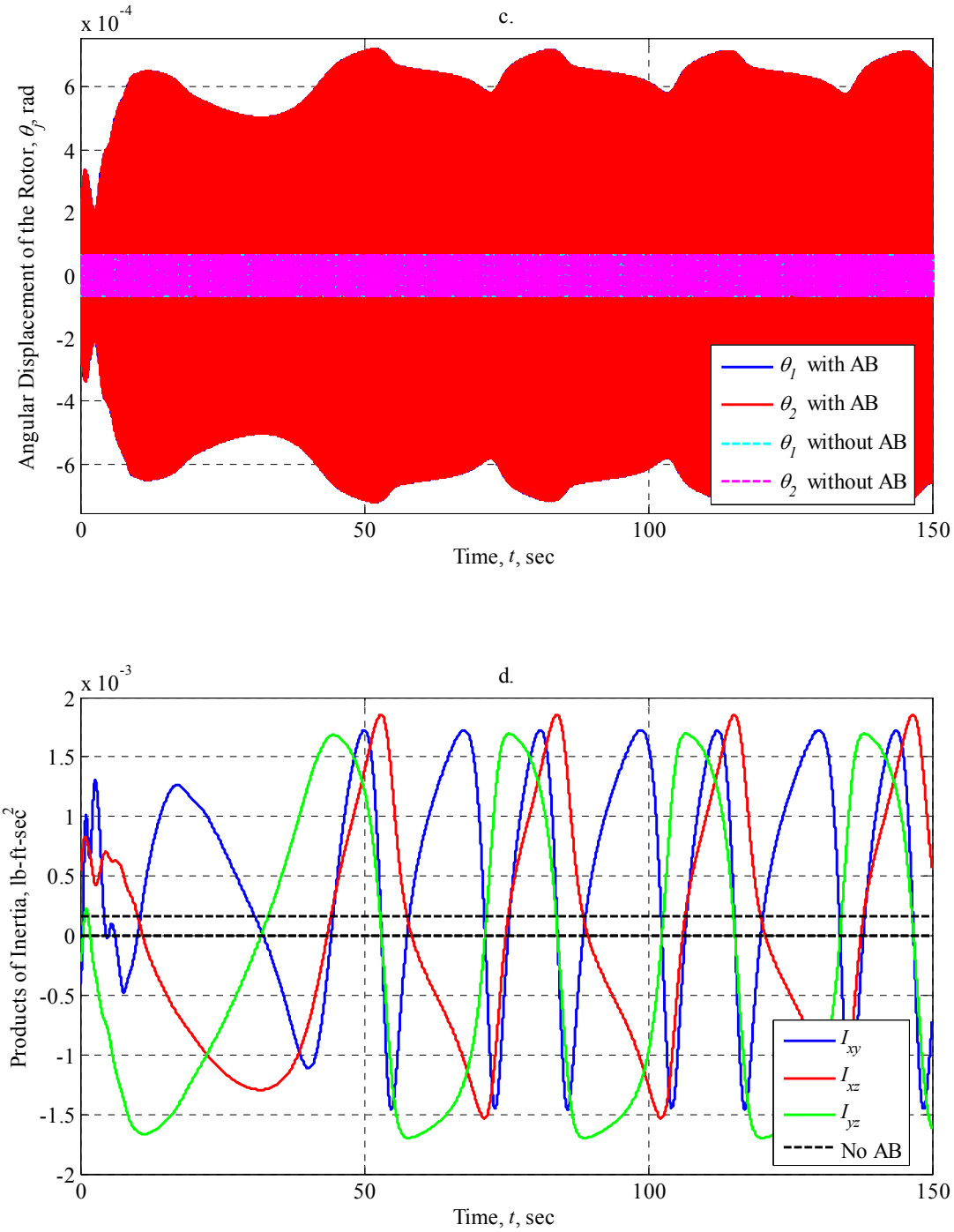

Figure 5.4.2.4-II: AB Performance at 50 rad/sec with $\mu_s = 0$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

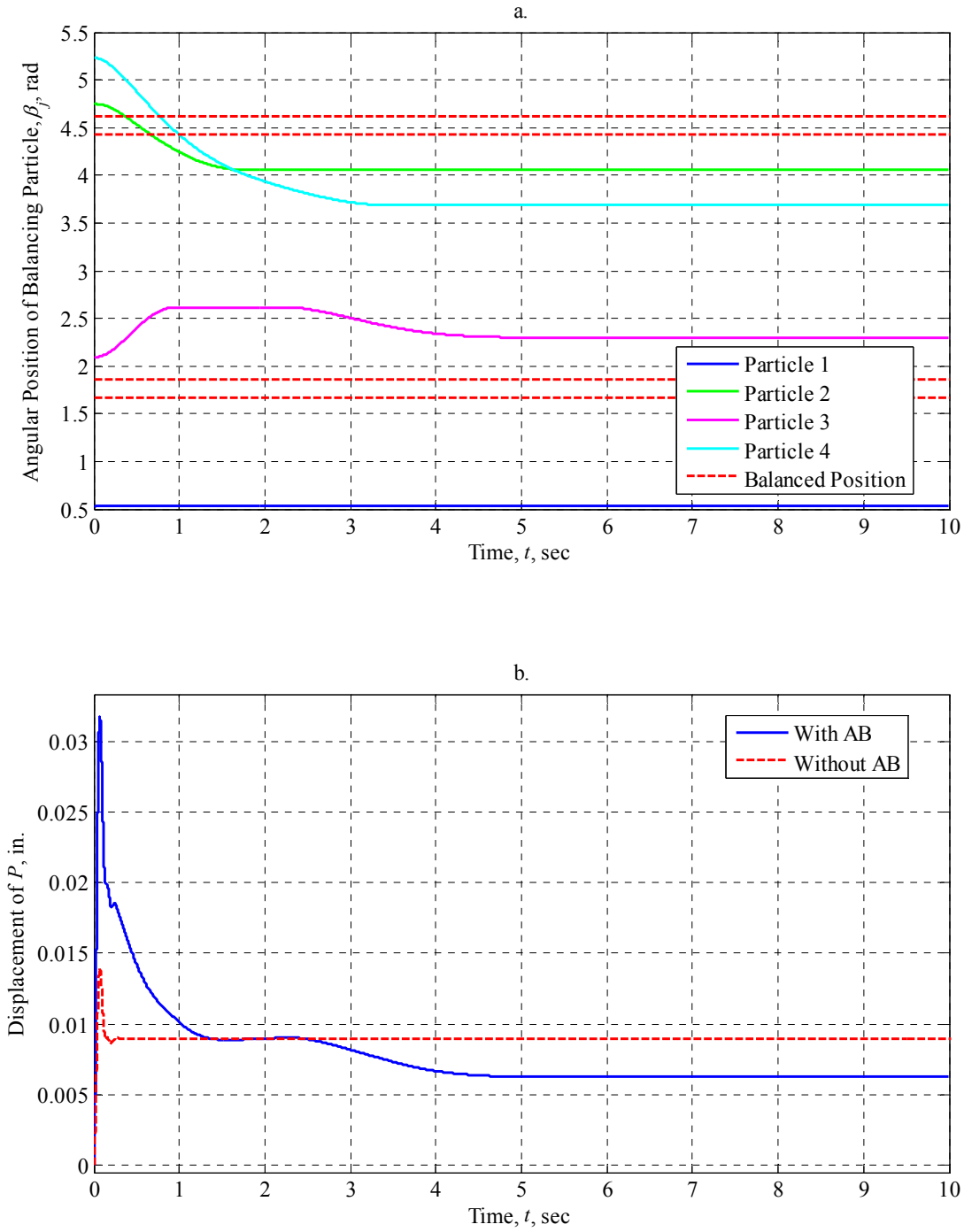

Figure 5.4.2.5-I: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

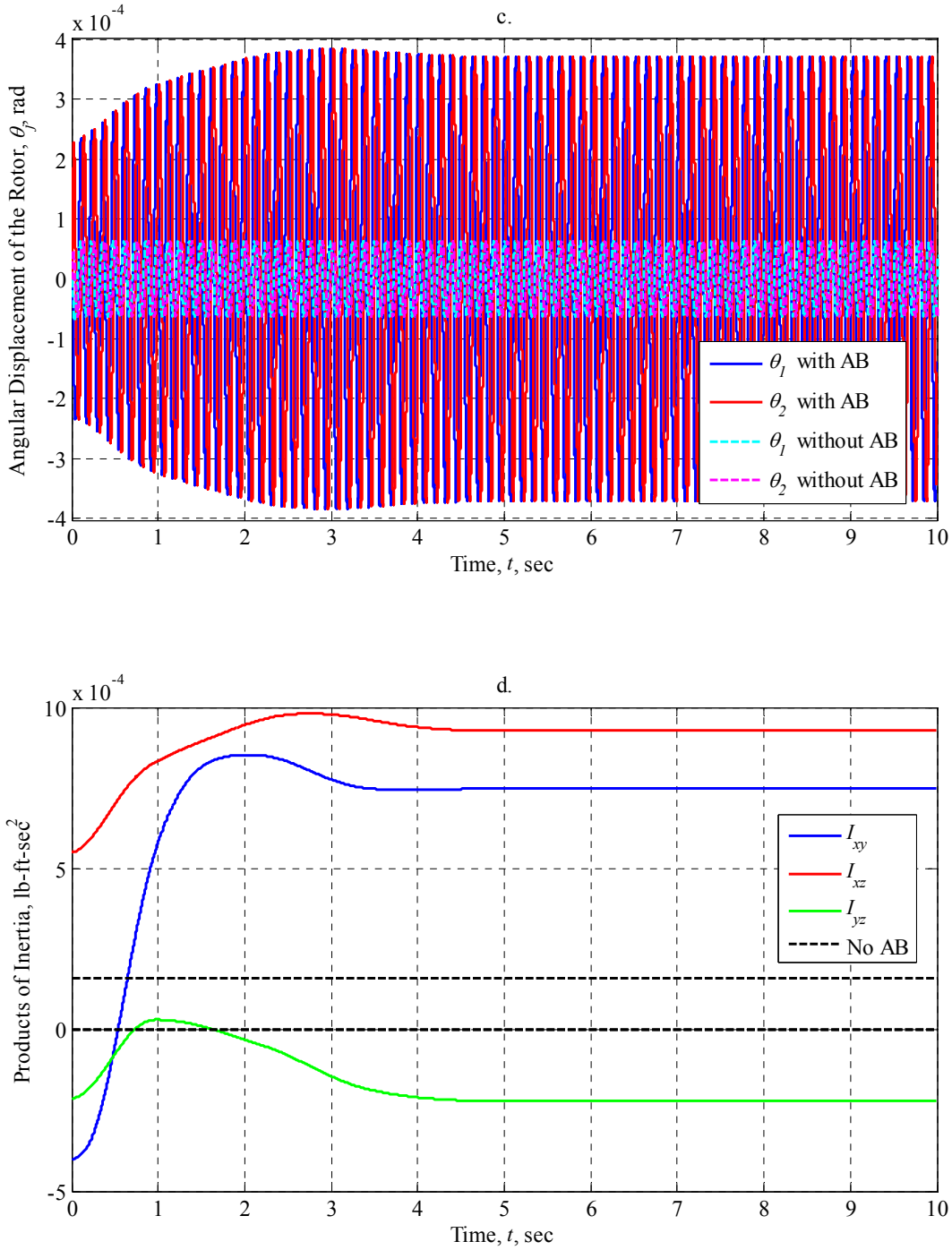

Figure 5.4.2.5-II: AB Performance at 50 rad/sec with μ_s = 0.001 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

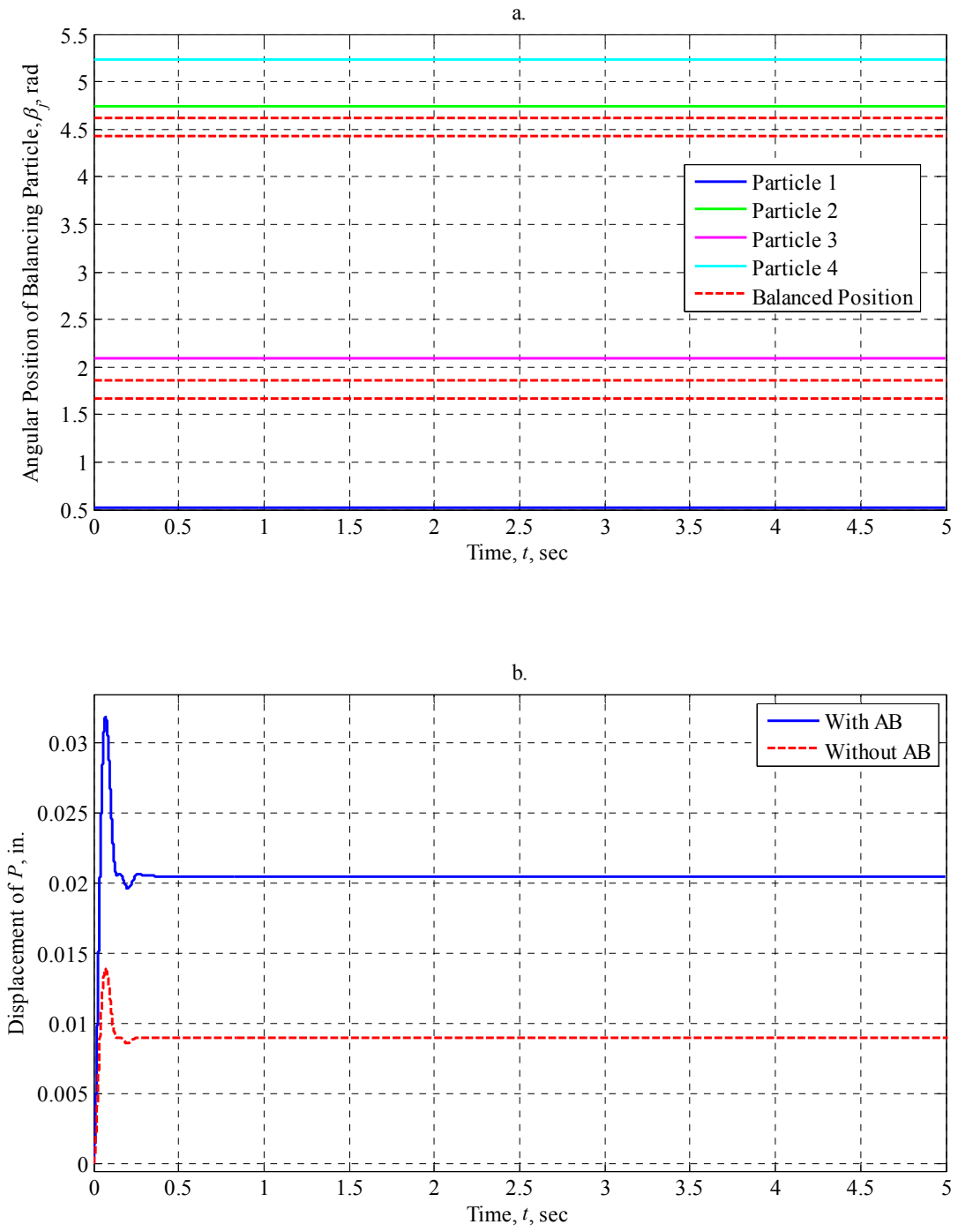

Figure 5.4.2.6-I: AB Performance at 50 rad/sec with $\mu_s = 0.1$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

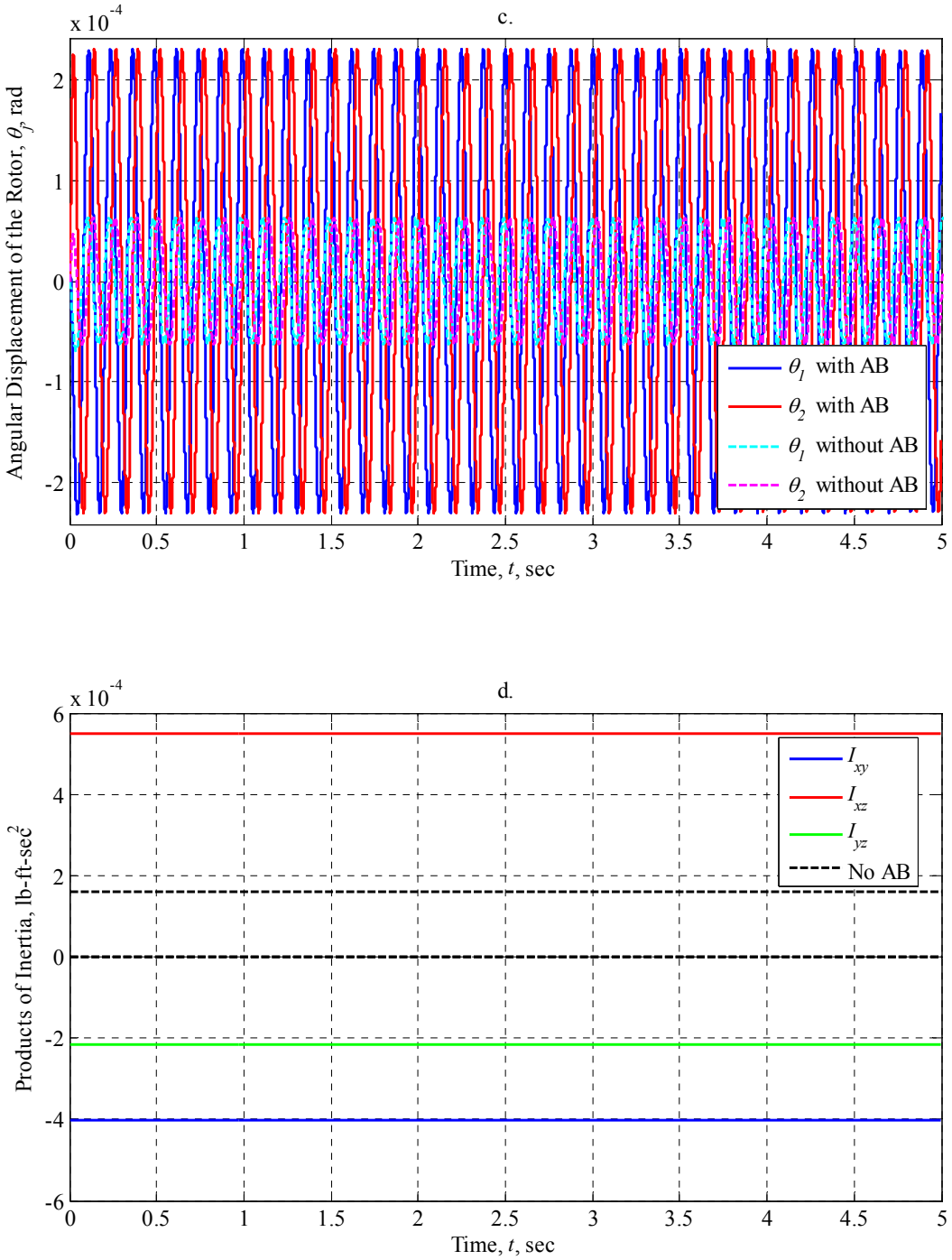

Figure 5.4.2.6-II: AB Performance at 50 rad/sec with μ_s = 0.1 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

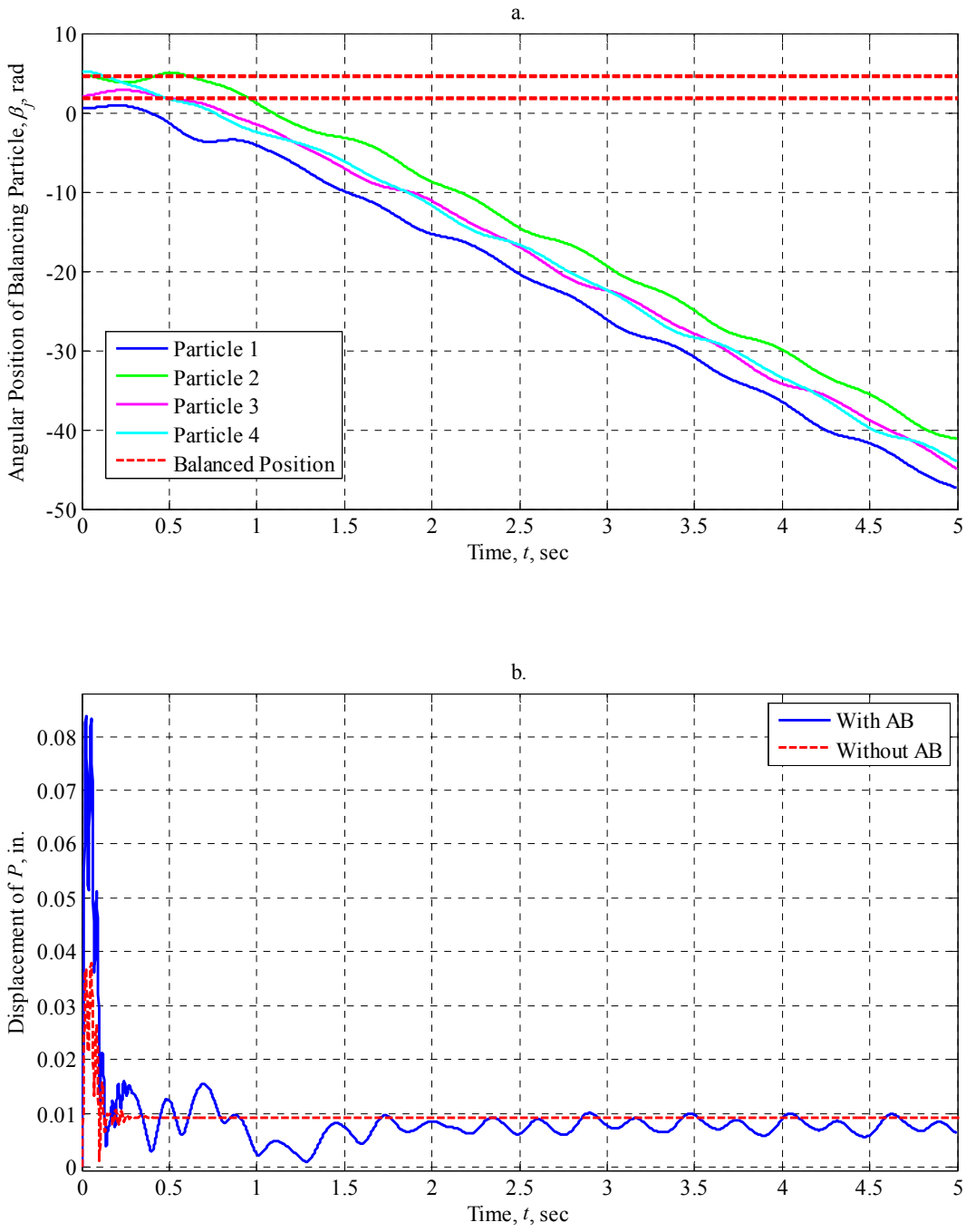

Figure 5.4.2.7-I: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

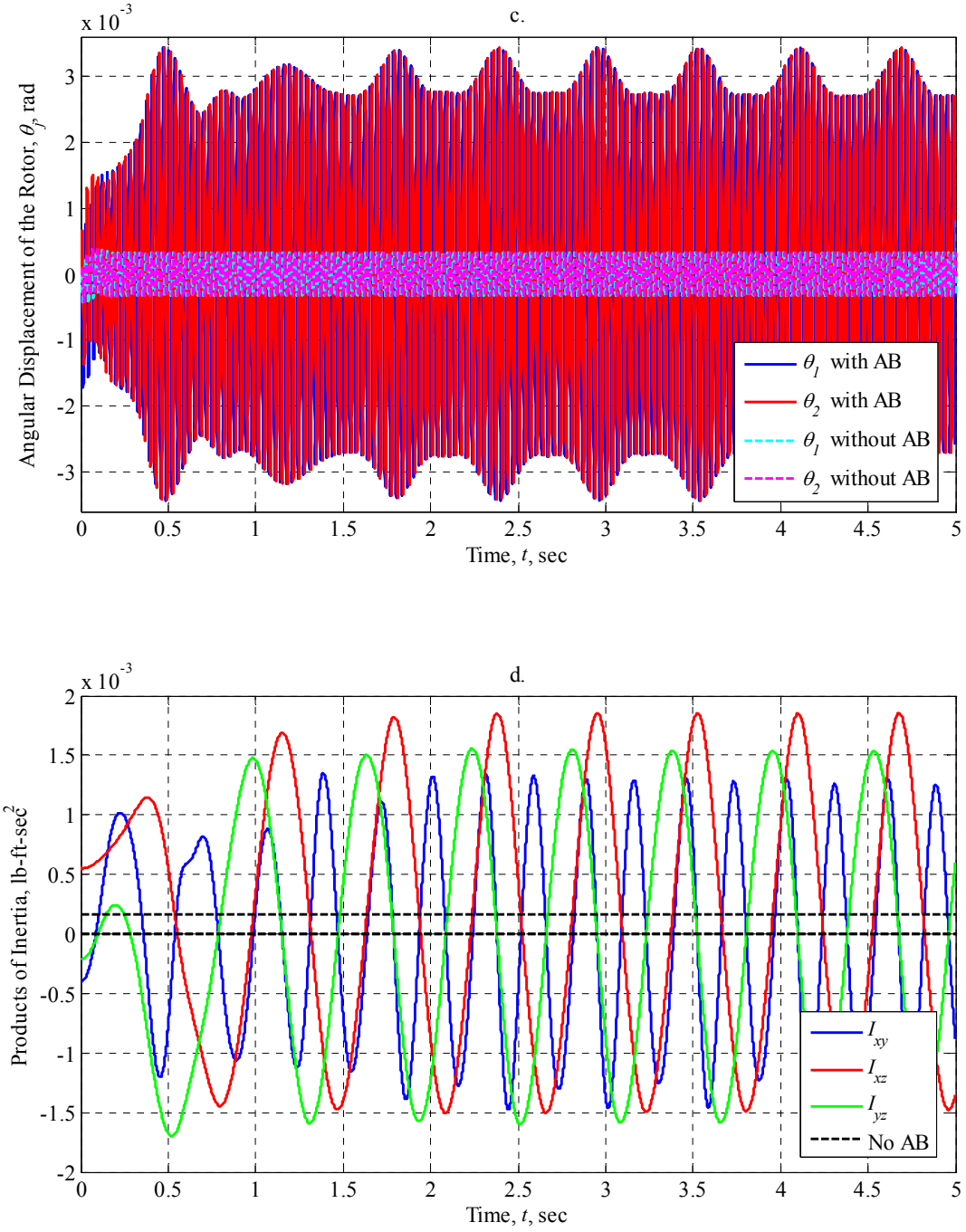

Figure 5.4.2.7-II: AB Performance at 200 rad/sec with $\mu_s = 0$ (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

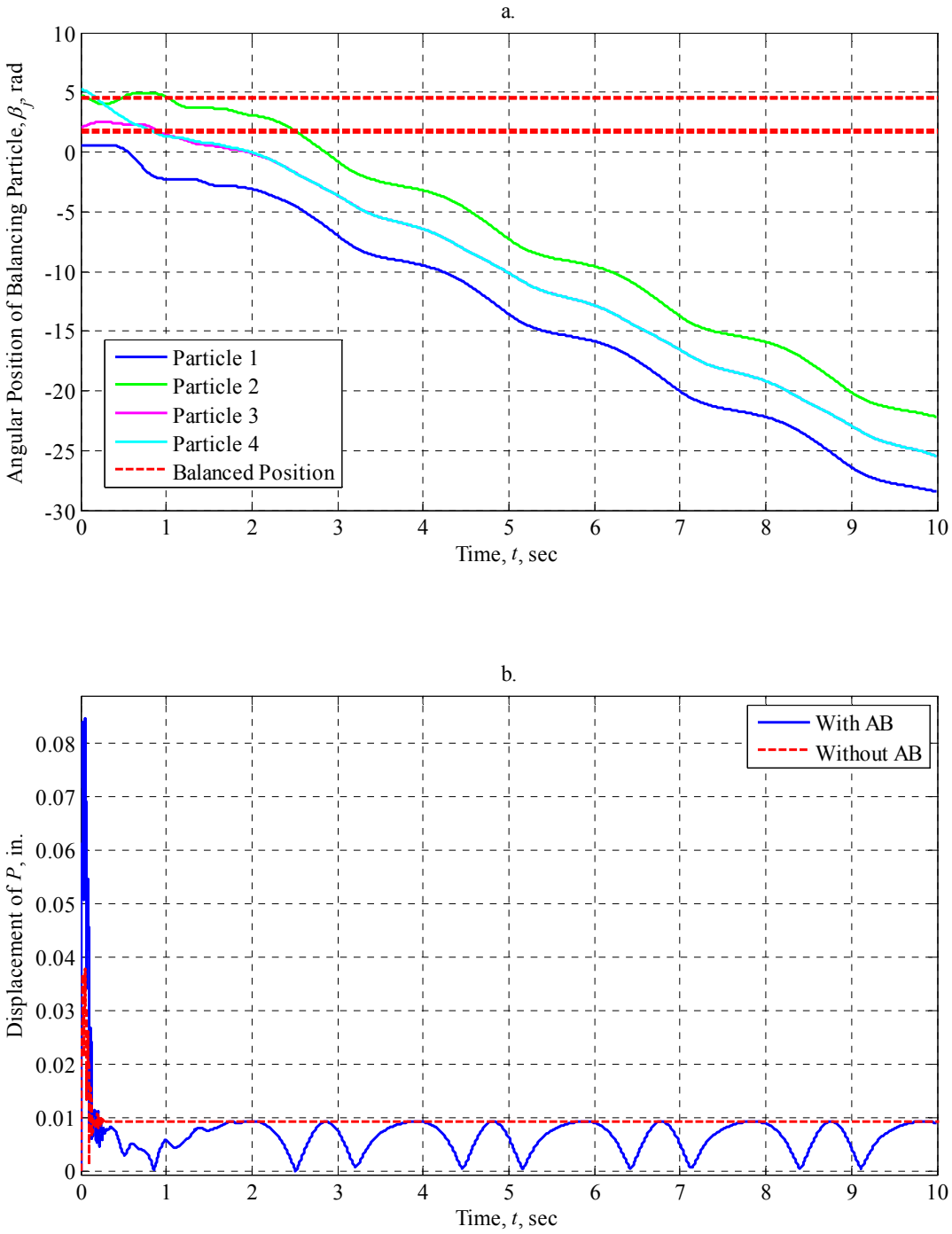

Figure 5.4.2.8-I: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

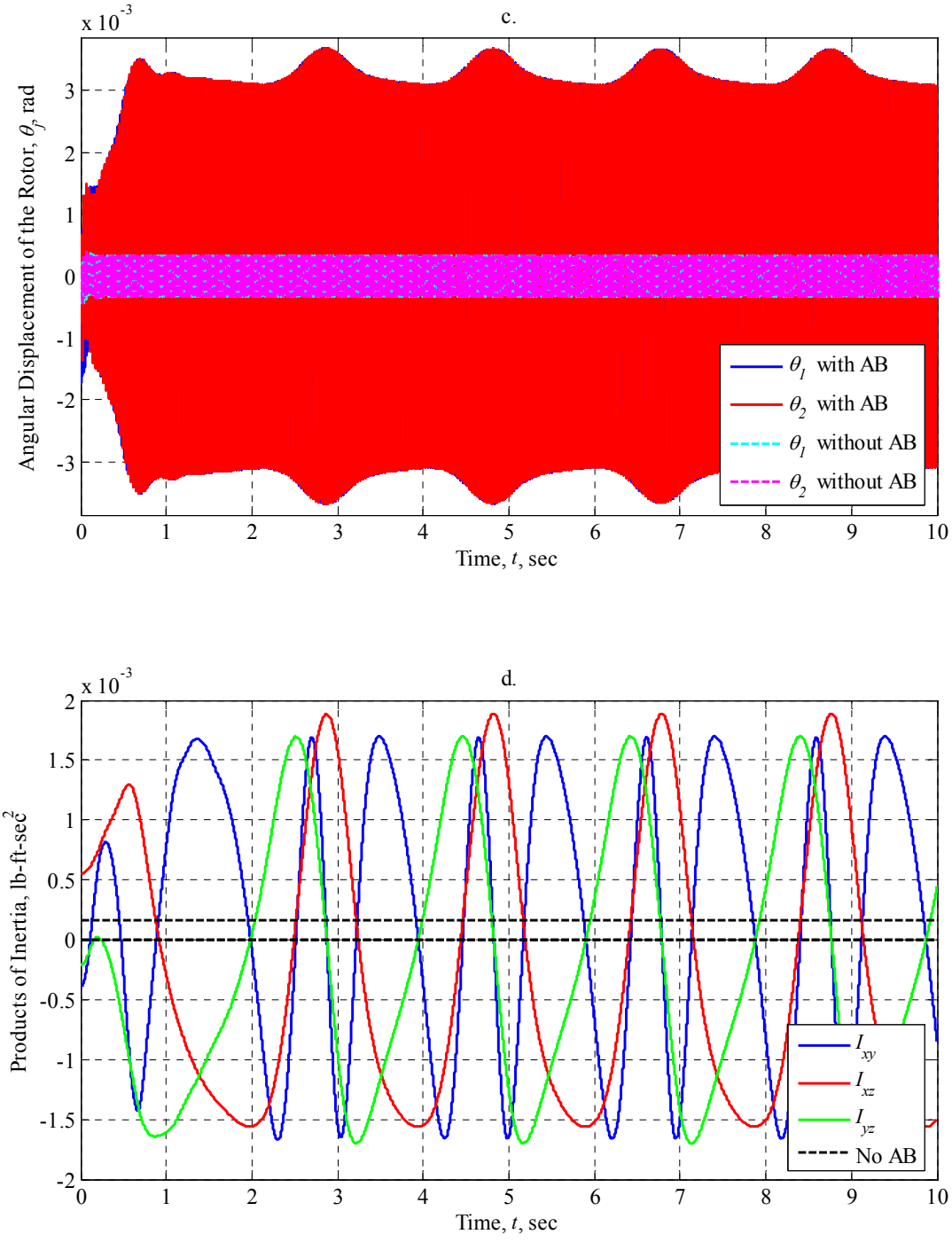

Figure 5.4.2.8-II: AB Performance at 200 rad/sec with μ_s = 0.001 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

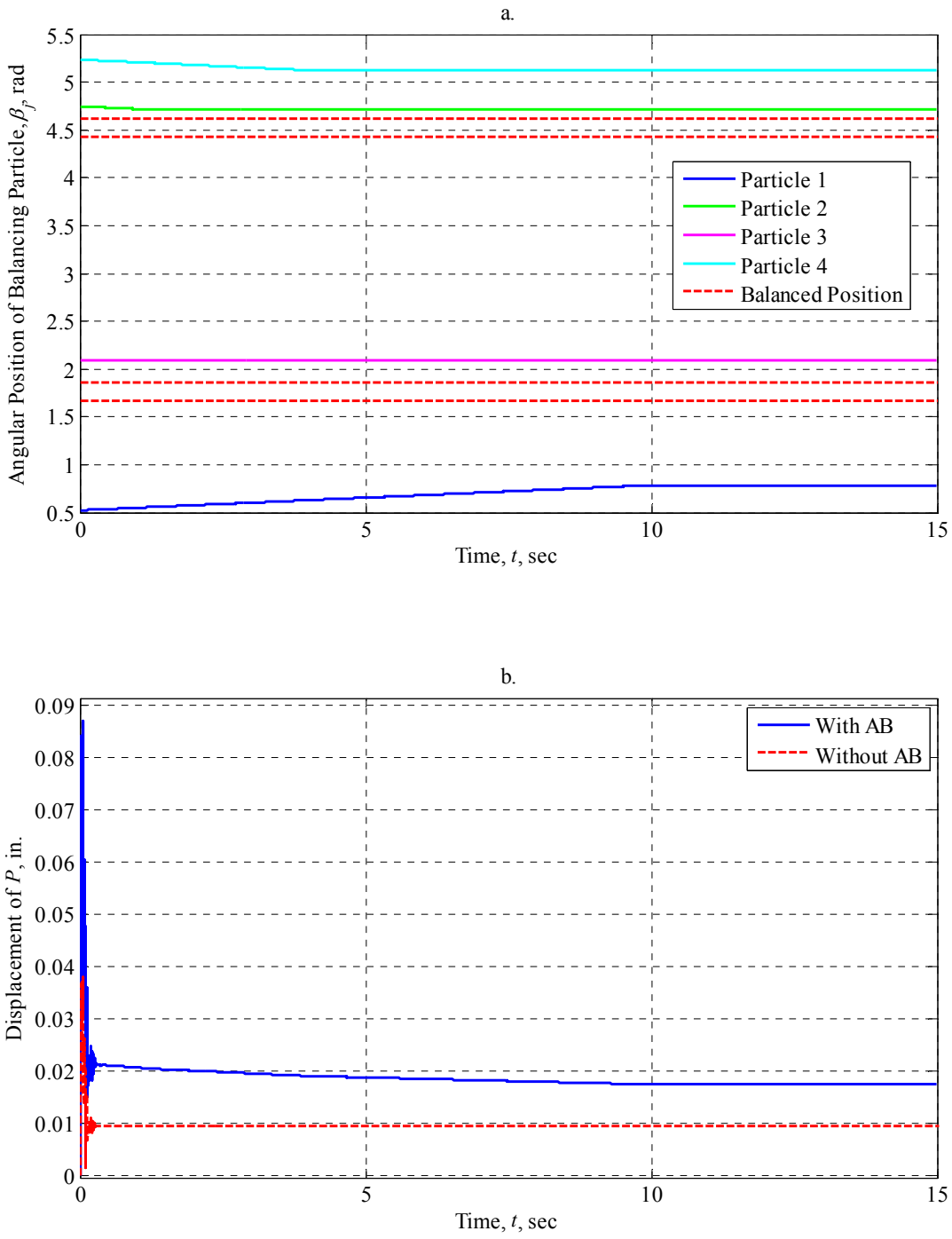

Figure 5.4.2.9-I: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

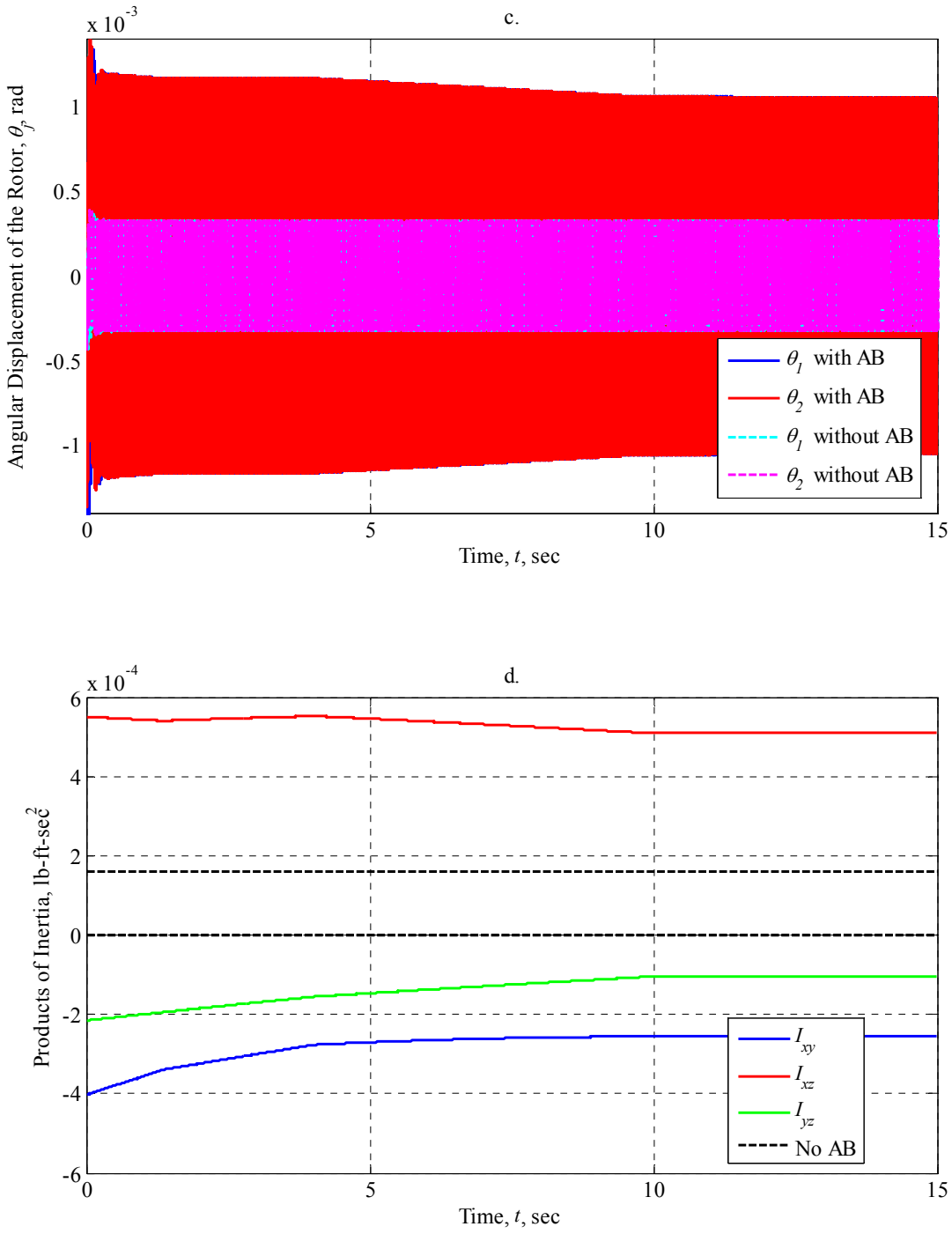

Figure 5.4.2.9-II: AB Performance at 200 rad/sec with μ_s = 0.1 (Vertically-Oriented Arbitrarily-Imbalanced Rotor, Dual Plane Balancing with Two Particles per Race)

6 Summary and Conclusions

6.1 Problem Description

A balancing system for a six-degree-of-freedom cylindrical rotor has been modeled, and extensive numerical studies have been conducted. The state of the system is described by three inertial translational displacements and 1-2-3 Euler angles for the rotor, and circumferential relative angular displacements for the race-constrained balancing masses, which are affected by friction. Due to the complexity of the slip/no slip possibilities of the balancing masses, the kinetics have been carried out via a Newtonian approach. The numerical simulations were conducted with a custom four-cycle Runge-Kutta integration routine for conditions associated with an automobile wheel.

The rotor is assumed to be relatively short along its spin axis of symmetry and has a single support point at its geometric center. This linear elastic support consists of three translational spring/damper units which resist motion in three inertial directions as well as three torsional spring/damper units which resist rotation about appropriate axes. The rotor itself has a general and undesired imbalance.

Affixed to the rotor are two races which are concentric with the rotor axis of symmetry. Each race contains one or two balancing masses (modeled as particles for most of this investigation), which are subjected to viscous friction forces resulting from lubrication with the race, as well as Coulomb friction forces resulting from contact with the race. Rolling friction associated with small spherical balancing masses can be treated by using an appropriately low value of the Coulomb coefficient of static friction.

At each time step in the simulation process, a determination must be made in regard to whether the balancing mass is sliding or stationary relative to the rotor. If the relative sliding velocity is greater than a set threshold, sliding is assumed to continue. If the relative sliding velocity is below this small threshold, the friction force compatible with no sliding is calculated. If this friction force is available, no slipping occurs for the time step. If the required friction force is not available, slipping occurs for that time step.

Because there are four total balancing particles, and each particle is able to stop relative to the rotor at different times, or begin moving at a later in the simulation process, a total of 16 different sets of equations of motion were developed to fully simulate all possible results. The number of equations in each set ranges from six to ten, where six represents a situation where all

four balancing particles come to rest relative to the rotor, and ten represents the a situation where all four particles move relative to the rotor. Since different combinations of particles are stopped relative to the rotor at different time steps, a significant amount of logic is exercised at each time step to ensure that the simulation is consistent with the motion properties of the balancing particles. The resulting logic structure features 81 possible completion paths at each time step.

Previous efforts usually limit the motion of the rotor geometric center to a horizontal plane and assume the balancing masses are not subjected to Coulomb friction forces in the race. This is done to simplify the equations of motion and eliminate the need for complex logic structures in programming. Additionally, most previous models ignore the effect which gravity has on the balancing particles since it is considered small in relation to the centripetal accelerations generated with high operating speeds. The model presented here is able to orient the rotor axis vertically, where gravitational forces lie perpendicular to the plane of the rotor, or horizontally, where gravitational forces lie in the plane of the rotor. As numerical results indicate, the inclusion of gravitational and Coulomb friction forces can have significant influence on balancing particle behavior.

6.2 Summary of Results

Numerous simulations were run featuring different types of mass imbalances for the rotor and different numbers of balancing particles. The first imbalance explored was a pure static imbalance that results in pure translational motions of the rotor geometric center from its balanced position. The rotor featured a mass imbalance at its mid-plane that was placed on the inner rotor surface. Single-plane balancing was employed for these simulations and featured a single race (positioned in the midplane of the rotor) equipped with one or two balancing particles. The race radius was equivalent to the radial position of the rotor imbalance. Simulations were conducted for various operating speeds and Coulomb friction levels. In all cases, the masses of both the imbalance and an individual balancing particle were identical.

When the operating speed was less than the translational natural frequency of the rotor, the balancing particle(s) were unable to locate positions within the race to counteract the effect of the rotor imbalance. Rotor performance was worsened for most Coulomb friction levels, and any slight, and temporary, improvements could only be attributed to the chosen initial conditions for the balancing particles. If the operating speed was above the system translational natural

frequency, the particle(s) improved rotor performance for low Coulomb friction levels. In the absence of Coulomb friction, the balancing particle(s) were able to exactly cancel the imbalance of the rotor, and do so in a timely fashion. Increasing the Coulomb friction to values commonly found for the rolling resistance of steel on steel showed mixed performance gains. Some operating speeds showed improved performance while others had degraded performance. It is interesting to note that higher operating speeds and non-zero Coulomb friction levels improved rotor performance in some cases. Once the Coulomb friction was increased to sliding friction values for steel on steel, the rotor performance was degraded in all cases.

When gravitational effects were included for the static imbalance simulations, the geometric center was unable to reach a constant steady-state amplitude. Instead, the geometric center had an oscillatory motion owing to the addition of mass to the system and an altered static equilibrium point. When the operating speed was above the translational natural frequency, the balancing particle(s) improved rotor performance. In some cases, this improvement was better than for the vertical-axis rotor that neglected gravitational effects.

The next type of simulation that was conducted featured a pure dynamic imbalance of a vertically-oriented rotor. The rotor featured two imbalance masses diametrically opposed about the rotor geometric center which results in purely rotational displacements about intermediate coordinate axes, but no translational displacements of the rotor geometric center. Dual-plane balancing was employed for these simulations and featured one or two balancing particles per race, with the races being positioned in the same transverse planes as the imbalance masses. The race radius was equivalent to the radial position of the rotor imbalances and the masses of all imbalances and balancing particles were identical. For both one and two balancing particles per race, there is no improvement in the angular displacements of the rotor caused by the dynamic imbalance. While it is theoretically possible to eliminate both translational and rotational displacements with the supplied conditions, the balancing particles are only able to eliminate translational vibrations caused by their introduction into the rotor/imbalance system. The particles worsened the rotor performance when operated below the translational natural frequency of the support. In these simulations, a synchronous motion was observed where the balancing particles converged to the same angular position within the races to function as a single particle creating large steady-state translational displacements of the geometric center. At operating speeds above the translational natural frequency of the rotor support, the particles

again exhibit a synchronous type motion by occupying opposite angular positions within the plane of the rotor which effectively eliminates their added imbalance. This did not eliminate products of inertia for the rotor/imbalance system, and in most cases, made them larger. When the friction level was increased slightly, the particles stopped before reaching positions that would eliminate balancing-particle induced vibrations. This left a residual displacement of the geometric center and worsened the products of inertia for the rotor.

The final type of simulation conducted involved an arbitrarily positioned mass imbalance that left the rotor statically and dynamically imbalanced. Dual-plane balancing was employed by introducing four balancing particles, two balancing particles per race, at symmetric locations on either side of the rotor midplane and outboard of the arbitrarily placed imbalance mass. While theoretically possible to eliminate all products of inertia and the rotor/imbalance center-of-mass coordinates in the plane of the rotor, the addition of the balancing particles did nothing to eliminate products of inertia or improve the rotational motions of the rotor. When the operating speed was above the translational natural frequency of the rotor support, there was a tendency for the particles to eliminate the imbalance caused by their addition to the system, but whether this had any positive effect on rotor performance is strongly dictated by the choice of initial conditions for the balancing particles as well as the Coulomb friction level in the race. If the Coulomb friction level is set high, the particles are unable to find their balanced positions and rotor performance is degraded.

One last comment centers on the time required for the balancing particles to come to rest relative to the rotor in the presence of increasing Coulomb friction and increasing operating speeds. Simulations show that simultaneous increases in Coulomb friction and operating speed can allow the particles to move relative to the rotor for longer periods of time. The particles typically moved for much longer periods of time as the Coulomb friction was increased from a rolling-resistance mode to a sliding mode. Additionally, the particles were able to move for longer periods of time at the same Coulomb friction level when the operating speed was increased from 50 rad/sec to 200 rad/sec. These results do not appear to be the result of numerical integration because they were observed for all simulation types and for numerous time step sizes.

6.3 Concluding Remarks and Future Work

While there is a mechanism in place to drive particles to positions that eliminate translational displacements of the rotor, there is no such mechanism readily apparent to accomplish the same task in regards to rotational displacements. To be fully balanced, a rotor must have all three center-of-mass coordinates and all three products of inertia about a centroidal set of coordinate axes eliminated. The simulations for the model studied in this paper show this situation is not readily achieved when the operating conditions are less than ideal and the Coulomb friction level between the balancing particles and the race is greater than zero.

It would be of great interest to explore the effect of allowing the balancing particles to move in the axial direction of the rotor to see if rotational displacements caused by a dynamic or arbitrary imbalance could be countered with balancing particle motion having two degrees of freedom. It would also be more realistic to model the balancing particles as spheres and include rolling effects and deformations of the contact point between the balancing masses and the race. Since the presence of Coulomb friction strongly affects the steady-state position of the balancing particles, the transient effects of the rotor spin up should be examined. It is obvious that significant care must be taken when choosing initial conditions for the particles if a balanced position is going to be guaranteed. Additionally, having an adaptive algorithm that allows a user-defined number of balancing particles would provide the ability for enhanced simulations that utilize a powder-like balancer. These balancer types are common in practice, and the relative motion characteristics of granules would feature prominent stiction effects resulting from Coulomb friction and require significant statistical considerations.

Overall, the problem of interest is exceedingly complex with highly nonlinear equations of motion. This work offers a modest step forward by focusing on the effects of Coulomb friction which act at the interface between the balancing particles and the race. Coulomb friction can never be eliminated, and its presence introduces a wide range of behaviors that differ strongly from idealized cases.

REFERENCES

1. Meirovitch, L., *Fundamentals of Vibrations*. 2001, New York: McGraw Hill.
2. Zhou, S. and J. Shi, *Active Balancing and Vibration Control of Rotating Machinery: A Survey.* Shock and Vibration Digest, 2001. **33**(4): p. 361-371.
3. Gosiewski, Z., *Automatic Balancing of Flexible Rotors, Part I: Theoretical Background.* Journal of Sound and Vibration, 1985. **100**(4): p. 551-567.
4. Gosiewski, Z., *Automatic Balancing of Flexible Rotors, Part II: Synthesis of System.* Journal of Sound and Vibration, 1987. **114**(1): p. 103-119.
5. Bleuler, H., et al., *Application of digital signal processors for industrial magnetic bearings.* Control Systems Technology, IEEE Transactions on, 1994. **2**(4): p. 280-289.
6. Thearle, E.L., *A New Type of Dynamic-Balancing Machine.* Transactions of ASME (Applied Mechanics), 1932. **APM-54-12**: p. 131-141.
7. Thearle, E.L., *Automatic Dynamic Balancers Part 1 - The Leblanc Balancer.* Machine Design, 1950. **22**(September): p. 119-124.
8. Thearle, E.L., *Automatic Dynamic Balancers Part 2 - Ring, Pendulum and Ball Balancers.* Machine Design, 1950. **22**(September): p. 103-106, 152.
9. Ernst, H., *Automatic Precision Balancing.* Machine Design, 1951. **23**(January): p. 107-114.
10. Alexander, J.D., *An Automatic Dynamic Balancer.* Proceedings for the Second Southeastern Conference, 1964. **2**: p. 415-426.
11. Cade, J.W., *Self-Compensating Balancing in Rotating Mechanisms*, in *Design news*. 1965. p. 234-239.
12. Sharp, R.S., *An Analysis of a Self-Balancing System for Rigid Rotors.* Journal of Mechanical Engineering Science, 1975. **17**(4): p. 186-189.
13. Hedaya, M.T. and R.S. Sharp, *An Analysis of a New Type of Automatic Balancer.* Journal of Mechanical Engineering Science, 1977. **19**(5): p. 221-226.
14. Bövik, P. and C. Högfors, *Autobalancing of Rotors.* Journal of Sound and Vibration, 1986. **111**(3): p. 429-440.
15. Kubo, S., et al., *Automatic Balancer - (Pendulum Balancer).* Bulletin of the Jsme-Japan Society of Mechanical Engineers, 1986. **29**(249): p. 924-928.
16. Tadeusz, M., *Position Error Occurrence in Self Balancers Used on Rigid Rotors of Rotating Machinery.* Mechanism and Machine Theory, 1988. **23**(1): p. 71-78.
17. Lee, J.K. and W.K. VanMoorhem, *Analytical and Experimental Analysis of a Self-Compensating Dynamic Balancer in a Rotating Mechanism.* Journal of Dynamic Systems Measurement and Control-Transactions of the Asme, 1996. **118**(3): p. 468-475.
18. Rajalingham, C., R.B. Bhat, and S. Rakheja, *Automatic Balancing of Flexible Vertical Rotors Using a Guided Ball.* International Journal of Mechanical Sciences, 1998. **40**(9): p. 825-834.
19. Rajalingham, C. and S. Rakheja, *Whirl Suppression In Hand-Held Power Tool Rotors Using Guided Rolling Balancers.* Journal of Sound and Vibration, 1998. **217**(3): p. 453-466.
20. Rajalingham, C. and R.B. Bhat, *Complete Balancing of a Disk Mounted on a Vertical Cantilever Shaft Using a Two-Ball Automatic Balancer.* Journal of Sound and Vibration, 2006. **290**(1-2): p. 169-191.

21. Chung, J. and D.S. Ro, *Dynamic Analysis Of An Automatic Dynamic Balancer For Rotating Mechanisms.* Journal of Sound and Vibration, 1999. **228**(5): p. 1035-1056.
22. Hwang, C.H. and J. Chung, *Dynamic Analysis of an Automatic Ball Balancer with Double Races.* Jsme International Journal Series C-Mechanical Systems Machine Elements and Manufacturing, 1999. **42**(2): p. 265-272.
23. Kim, W. and J. Chung, *Performance of Automatic Ball Balancers on Optical Disc Drives.* Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 2002. **216**(11): p. 1071-1080.
24. Chung, J. and I. Jang, *Dynamic Response And Stability Analysis Of An Automatic Ball Balancer For A Flexible Rotor.* Journal of Sound and Vibration, 2003. **259**(1): p. 31-43.
25. Chung, J., *Effect of Gravity and Angular Velocity on an Automatic Ball Balancer.* Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 2005. **219**(1): p. 43-51.
26. Kim, W., D.-J. Lee, and J. Chung, *Three-Dimensional Modelling and Dynamic Analysis of an Automatic Ball Balancer in an Optical Disk Drive.* Journal of Sound and Vibration, 2005. **285**(3): p. 547-569.
27. Sperling, L., F. Merten, and H. Duckstein, *Self-Synchronization and Automatic Balancing in Rotor Dynamics.* Journal of Rotating Machinery, 2000. **6**(4): p. 275-285.
28. Sperling, L., *Selbstsynchronisation statisch und dynamisch unwuchtiger Vibratoren Teil I: Grundlagen.* Technische Mechanik, 1994. **14**(1): p. 61-76.
29. Sperling, L., *Selbstsynchronisation statisch und dynamisch unwuchtiger Vibratoren Teil I: Ausfuhrung und Beispiele.* Technische Mechanik, 1994. **14**(2): p. 85-96.
30. Sperling, L., F. Merten, and H. Duckstein, *Rotation and Vibration in Beispielen zur Methode der direkten Bewegungsteilung.* Technische Mechanik, 1997. **17**(3): p. 231-243.
31. Sperling, L., et al., *Simulation of Two-Plane Automatic Balancing of a Rigid Rotor.* Math. Comput. Simul., 2002. **58**(4-6): p. 351-365.
32. Ryzhik, B., L. Sperling, and H. Duckstein, *Sommerfeld-Effect in Automatic Balancing*, in *XXI International Conference on Theoretical and Applied Mechanics (ICTAM)*. 2004: Warsaw, Poland.
33. Ryzhik, B., H. Duckstein, and L. Sperling, *Partial Compensation of Unbalance by One- and Two-Plane Automatic Balancing Devices.* International Journal of Rotating Machinery, 2004. **10**(3): p. 193-201.
34. Ryzhik, B., H. Duckstein, and L. Sperling, *Automatic Balancing of the Unsymmetrical Rigid Rotor.* PAMM, 2003. **2**(1): p. 70-71.
35. Sperling, L., B. Ryzhik, and H. Duckstein, *Single-Plane Auto-Balancing of Rigid Rotors.* Technische Mechanik, 2004. **24**(1): p. 1-24.
36. Ryzhik, B., L. Sperling, and H. Duckstein, *Non-Synchronous Motions Near Critical Speeds in a Single-Plane Auto-Balancing Device.* Technische Mechanik, 2004. **24**(1): p. 25-36.
37. Ryzhik, B., L. Sperling, and H. Duckstein, *Auto-Balancing of Anisotropically Supported Rigid Rotors.* Technische Mechanik, 2004. **24**(1): p. 37-50.
38. Wettergren, H.L. *Auto-Balance Anisotropic Mounted Rotors*. in *ASME 2001 Design Engineering Technical Conference and Computers and Information in Engineering Conference*. 2001. Pittsburgh, PA.
39. Wettergren, H.L., *Using Guided Balls to Auto-Balance Rotors.* Journal of Engineering for Gas Turbines and Power, 2002. **124**(4): p. 971-975.

40. Kang, J.-R., et al., *The Dynamics of a Ball-Type Balancer System Equipped with a Pair of Free-Moving Balancing Masses.* Journal of Vibration and Acoustics, 2001. **123**(4): p. 456-465.
41. Huang, W.Y., et al., *The Application Of Ball-Type Balancers For Radial Vibration Reduction Of High-Speed Optic Disk Drives.* Journal of Sound and Vibration, 2002. **250**(3): p. 415-430.
42. Chao, P.C.P., Y.-D. Huang, and C.-K. Sung, *Non-Planar Dynamic Modeling for the Optical Disk Drive Spindles Equipped with an Automatic Balancer.* Mechanism and Machine Theory, 2003. **38**(11): p. 1289-1305.
43. Sung, C.-K., P.C.P. Chao, and B.-C. Yo, *Effects of Nonlinear Damping Washers on the Automatic Ball Balancer for Optical Disk Drives.* ASME Conference Proceedings, 2005. **2005**(47381): p. 1083-1090.
44. Chao, P.C.P., C.-K. Sung, and H.-C. Leu, *Effects of Rolling Friction of the Balancing Balls on the Automatic Ball Balancer for Optical Disk Drives.* Journal of Tribology, 2005. **127**(4): p. 845-856.
45. Chao, P.C.P., C.-K. Sung, and C.-C. Wang, *Dynamic Analysis of the Optical Disk Drives Equipped with an Automatic Ball Balancer with Consideration of Torsional Motions.* Journal of Applied Mechanics, 2005. **72**(6): p. 826-842.
46. Chao, P., et al., *Non-Planar Modeling and Experimental Validation of a Spindle–Disk System Equipped with an Automatic Balancer System in Optical Disk Drives.* Microsystem Technologies, 2007. **13**(8): p. 1227-1239.
47. Chao, P.C.P., C.W. Chiu, and K.T. Shih, *A Novel Low-Torque Ball Re-Positioning Scheme Based on a Sliding-Mode Ball Observer for an Automatic Balancer System.* Shock and Vibration, 2008. **15**(2): p. 101-126.
48. Olsson, K.-O., *Limits for the Use of Auto-Balancing.* International Journal of Rotating Machinery, 2004. **10**(3): p. 221-226.
49. van de Wouw, N., et al., *Performance of an Automatic Ball Balancer with Dry Friction.* International Journal of Bifurcation and Chaos, 2005. **15**(1): p. 65-82.
50. van de Wouw, N. and R.I. Leine, *Attractivity of Equilibrium Sets of Systems with Dry Friction.* Nonlinear Dynamics, 2004. **35**(1): p. 19-39.
51. van de Wouw, N. and R.I. Leine. *Stability of Stationary Sets in Nonlinear Systems with Set-Valued Friction*. in *Decision and Control, 2006 45th IEEE Conference on*. 2006.
52. Leine, R. and N. van de Wouw, *Stability Properties of Equilibrium Sets of Non-Linear Mechanical Systems with Dry Friction and Impact.* Nonlinear Dynamics, 2008. **51**(4): p. 551-583.
53. Meraz, M.A., et al., *Self Balancing System for Rotating Mechanisms.* Revista Facultad de Ingenieria - Universidad de Tarapaca, 2005. **13**(2): p. 59-64.
54. Yang, Q., et al., *Study on the Influence of Friction in an Automatic Ball Balancing System.* Journal of Sound and Vibration, 2005. **285**(1-2): p. 73-99.
55. Horvath, R., G.T. Flowers, and J. Fausz, *Influence of Nonideaities on the Performance of a Self-Balancing Rotor System.* ASME Conference Proceedings, 2005. **2005**(42126): p. 233-242.
56. Horvath, R., G.T. Flowers, and J. Fausz, *Passive Balancing of Rotor Systems Using Pendulum Balancers.* Journal of Vibration and Acoustics, 2008. **130**(4): p. 041011-11.

57. Green, K., A.R. Champneys, and N.J. Lieven, *Bifurcation Analysis of an Automatic Dynamic Balancing Mechanism for Eccentric Rotors.* Journal of Sound and Vibration, 2006. **291**(3-5): p. 861-881.
58. Green, K., A.R. Champneys, and M.I. Friswell, *Analysis of the Transient Response of an Automatic Dynamic Balancer for Eccentric Rotors.* International Journal of Mechanical Sciences, 2006. **48**(3): p. 274-293.
59. Green, K., et al., *The Stability of Automatic Ball Balancers*, in *IFToMM Seventh International Conference on Rotor Dynamics* 2006: Vienna, Austria.
60. Rodrigues, D.J., et al., *Automatic Balancing of a Rigid Rotor with Misaligned Shaft.* Applied Mechanics and Materials (Volumes 5-6), 2006. **Modern Practice in Stress and Vibration Analysis VI**: p. 231-236.
61. Rodrigues, D.J., et al., *Automatic Two-Plane Balancing for Rigid Rotors.* International Journal of Non-Linear Mechanics, 2008. **43**(6): p. 527-541.
62. Rodrigues, D.J., et al., *Device Asymmetries and the Effect of the Rotor Run-Up in a Two-Plane Automatic Ball Balancing System*. 2008, Bristol Centre for Applied Nonlinear Mathematics. p. 9.
63. Rodrigues, D.J., et al., *A Consideration of Support Asymmetry in an Automatic Ball Balancing System*, in *Sixth Euromech Nonlinear Oscillations Conference*. 2008: St. Petersburg, Russia.
64. Green, K., et al., *Investigation of a Multi-Ball, Automatic Dynamic Balancing Mechanism for Eccentric Rotors.* Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 2008. **366**(1866): p. 705-728.
65. Lu, C.-J., *Stability Analysis of a Single-Ball Automatic Balancer.* Journal of Vibration and Acoustics, 2006. **128**(1): p. 122-125.
66. Lu, C.-J. and C.-H. Hung, *Stability Analysis of a Three-Ball Automatic Balancer.* Journal of Vibration and Acoustics, 2008. **130**(5): p. 051008-7.
67. Lu, C.-J., M.-C. Wang, and S.-H. Huang, *Analytical Study of the Stability of a Two-Ball Automatic Balancer.* Mechanical Systems and Signal Processing, 2009. **23**(3): p. 884-896.
68. Majewski, T. and R. Sokołowska, *Flexible Rotor with the System of Automatic Compensation of Dynamic Forces*. 2007. p. 464-469.
69. Filimonikhina, I. and G. Filimonikhin, *Conditions for Ralancing a Rotating Body in an Isolated System with Automatic Balancers.* International Applied Mechanics, 2007. **43**(11): p. 1276-1282.
70. Cheng, C.C., et al., *Design and Analysis of Auto-Balancer of an Optical Disk Drive Using Speed-Dependent Vibration Absorbers.* Journal of Sound and Vibration, 2008. **311**(1-2): p. 200-211.
71. Ehyaei, J. and M.M. Moghaddam, *Dynamic Response and Stability Analysis of an Unbalanced Flexible Rotating Shaft Equipped with n Automatic Ball-Balancers.* Journal of Sound and Vibration, 2009. **321**(3-5): p. 554-571.
72. Liu, J. and Y. Ishida, *Vibration Suppression of Rotating Machinery Utilizing an Automatic Ball Balancer and Discontinuous Spring Characteristics.* Journal of Vibration and Acoustics, 2009. **131**(4): p. 041004-7.
73. DeSmidt, H.A., *Imbalance Vibration Suppression of a Supercritical Shaft via an Automatic Balancing Device.* Journal of Vibration and Acoustics, 2009. **131**(4): p. 041001-13.

74. Greenwood, D.T., *Principles of Dynamics*. 2 ed. 1988, Englewood Cliffs: Prentice-Hall, Inc.
75. Kraige, L.G. and J.L. Meriam, *Engineering Mechanics: Dynamics*. 6 ed. Vol. 2. 2007, New York: Wiley.

APPENDIX A: LOGIC DECISIONS FOR NUMERICAL INTEGRATION

Table A.1: Logic Decisions for Numerical Integration

Case 1 $\lvert\dot{\beta}_1\rvert \le tol,\ \lvert\dot{\beta}_2\rvert \le tol,$ $\lvert\dot{\beta}_3\rvert \le tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_1$, $\dot{\beta}_2$, $\dot{\beta}_3$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_0 Compute interaction forces using Function $F_{int,0}$	if $F_{f_1} \le F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_0 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,0}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_1 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_2 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,2}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_3 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,3}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_4 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,4}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{12} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,12}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{13} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,13}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{14} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,14}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{23} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,23}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{24} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,24}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} \le F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_3$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{34} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,34}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} > F_{\max_2}$,	Reset $\dot{\beta}_1$, $\dot{\beta}_2$, and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{123}

	$F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$, $\dot{\beta}_2$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$, $\dot{\beta}_3$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_2$, $\dot{\beta}_3$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max_1}$, $F_{f_2} > F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$, $\dot{\beta}_2$, $\dot{\beta}_3$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 2 $\lvert\dot{\beta}_1\rvert > tol$, $\lvert\dot{\beta}_2\rvert \le tol$, $\lvert\dot{\beta}_3\rvert \le tol$, and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_2$, $\dot{\beta}_3$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_1 Compute interaction forces using Function $F_{int,1}$	if $F_{f_2} \le F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_1 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1}$ Proceed to next integration step
	else if $F_{f_2} > F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{12} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,12}$ Proceed to next integration step
	else if $F_{f_2} \le F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{13} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,13}$ Proceed to next integration step
	else if $F_{f_2} \le F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{14} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,14}$ Proceed to next integration step
	else if $F_{f_2} > F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{123} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_2} > F_{\max_2}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else if $F_{f_2} \le F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_3$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else $F_{f_2} > F_{\max_2}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_2$, $\dot{\beta}_3$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$

		Proceed to next integration step
Case 3 $\lvert\dot{\beta}_1\rvert \le tol,\ \lvert\dot{\beta}_2\rvert > tol,$ $\lvert\dot{\beta}_3\rvert \le tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_1$, $\dot{\beta}_3$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_2 Compute interaction forces using Function $F_{int,2}$	if $F_{f_1} \le F_{\max 1}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_2 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,2}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{12} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,12}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{23} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,23}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{24} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,24}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{123} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_3} \le F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_3$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max 1}$, $F_{f_3} > F_{\max_3}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$, $\dot{\beta}_3$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 4 $\lvert\dot{\beta}_1\rvert \le tol,\ \lvert\dot{\beta}_2\rvert \le tol,$ $\lvert\dot{\beta}_3\rvert > tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_1$, $\dot{\beta}_2$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_3 Compute interaction forces using Function $F_{int,3}$	if $F_{f_1} \le F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_3 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,3}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{13} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,13}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{23} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,23}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{34} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,34}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_2} > F_{\max_2}$,	Reset $\dot{\beta}_1$ and $\dot{\beta}_2$ in y_k

	and $F_{f_4} \le F_{\max_4}$	Compute k_1, k_2, k_3, and k_4 using Function F_{123} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$, $\dot{\beta}_2$, and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 5 $\lvert\dot{\beta}_1\rvert \le tol$, $\lvert\dot{\beta}_2\rvert \le tol$, $\lvert\dot{\beta}_3\rvert \le tol$, and $\lvert\dot{\beta}_4\rvert > tol$ Set $\dot{\beta}_1$, $\dot{\beta}_2$, and $\dot{\beta}_3$ equal to zero in y_k Compute k_1 using Function F_4 Compute interaction forces using Function $F_{int,4}$	if $F_{f_1} \le F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_3} \le F_{\max_3}$	Compute k_2, k_3, and k_4 using Function F_4 Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,4}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_3} \le F_{\max_3}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{14} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,14}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_3} \le F_{\max_3}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{24} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,24}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{34} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,34}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_3} \le F_{\max_3}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max 1}$, $F_{f_2} \le F_{\max_2}$, and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max 1}$, $F_{f_2} > F_{\max_2}$, and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_1$, $\dot{\beta}_2$, and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 6	if $F_{f_3} \le F_{\max_3}$ and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_{12} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,12}$

$\lvert\dot{\beta}_1\rvert > tol,\ \lvert\dot{\beta}_2\rvert > tol,$ $\lvert\dot{\beta}_3\rvert \le tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_3$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_{12} Compute interaction forces using Function $F_{int,12}$		Proceed to next integration step
	else if $F_{f_3} > F_{\max_3}$ and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{123} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_3} \le F_{\max_3}$ and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else $F_{f_3} > F_{\max_3}$ and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_3$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 7 $\lvert\dot{\beta}_1\rvert > tol,\ \lvert\dot{\beta}_2\rvert \le tol,$ $\lvert\dot{\beta}_3\rvert > tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_2$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_{13} Compute interaction forces using Function $F_{int,13}$	if $F_{f_2} \le F_{\max_2}$ and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_{13} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,13}$ Proceed to next integration step
	else if $F_{f_2} > F_{\max_2}$ and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{123} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_2} \le F_{\max_2}$ and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else $F_{f_2} > F_{\max_2}$ and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 8 $\lvert\dot{\beta}_1\rvert > tol,\ \lvert\dot{\beta}_2\rvert \le tol,$ $\lvert\dot{\beta}_3\rvert \le tol,$ and $\lvert\dot{\beta}_4\rvert > tol$ Set $\dot{\beta}_2$, and $\dot{\beta}_3$ equal to zero in y_k Compute k_1 using Function F_{14} Compute interaction forces using Function $F_{int,14}$	if $F_{f_2} \le F_{\max_2}$ and $F_{f_3} \le F_{\max_3}$	Compute k_2, k_3, and k_4 using Function F_{14} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,14}$ Proceed to next integration step
	else if $F_{f_2} > F_{\max_2}$ and $F_{f_3} \le F_{\max_3}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else if $F_{f_2} \le F_{\max_2}$ and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else $F_{f_2} > F_{\max_2}$ and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_2$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 9 $\lvert\dot{\beta}_1\rvert \le tol,\ \lvert\dot{\beta}_2\rvert > tol,$ $\lvert\dot{\beta}_3\rvert > tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$	if $F_{f_1} \le F_{\max_1}$ and $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_{23} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,23}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$ and $F_{f_4} \le F_{\max_4}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{123}

Set $\dot{\beta}_1$, and $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_{23} Compute interaction forces using Function $F_{int,23}$		Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$ and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max_1}$ and $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 10 $\lvert\dot{\beta}_1\rvert \le tol,\ \lvert\dot{\beta}_2\rvert > tol,$ $\lvert\dot{\beta}_3\rvert \le tol,$ and $\lvert\dot{\beta}_4\rvert > tol$ Set $\dot{\beta}_1$, and $\dot{\beta}_3$ equal to zero in y_k Compute k_1 using Function F_{24} Compute interaction forces using Function $F_{int,24}$	if $F_{f_1} \le F_{\max_1}$ and $F_{f_3} \le F_{\max_3}$	Compute k_2, k_3, and k_4 using Function F_{24} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,24}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$ and $F_{f_3} \le F_{\max_3}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$ and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max_1}$ and $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 11 $\lvert\dot{\beta}_1\rvert \le tol,\ \lvert\dot{\beta}_2\rvert \le tol,$ $\lvert\dot{\beta}_3\rvert > tol,$ and $\lvert\dot{\beta}_4\rvert > tol$ Set $\dot{\beta}_1$, and $\dot{\beta}_2$ equal to zero in y_k Compute k_1 using Function F_{34} Compute interaction forces using Function $F_{int,34}$	if $F_{f_1} \le F_{\max_1}$ and $F_{f_2} \le F_{\max_2}$	Compute k_2, k_3, and k_4 using Function F_{34} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,34}$ Proceed to next integration step
	else if $F_{f_1} > F_{\max_1}$ and $F_{f_2} \le F_{\max_2}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else if $F_{f_1} \le F_{\max_1}$ and $F_{f_2} > F_{\max_2}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max_1}$ and $F_{f_2} > F_{\max_2}$	Reset $\dot{\beta}_1$ and $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 12 $\lvert\dot{\beta}_1\rvert > tol,\ \lvert\dot{\beta}_2\rvert > tol,$ $\lvert\dot{\beta}_3\rvert > tol,$ and $\lvert\dot{\beta}_4\rvert \le tol$ Set $\dot{\beta}_4$ equal to zero in y_k Compute k_1 using Function F_{123} Compute interaction forces using Function $F_{int,123}$	if $F_{f_4} \le F_{\max_4}$	Compute k_2, k_3, and k_4 using Function F_{123} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,123}$ Proceed to next integration step
	else $F_{f_4} > F_{\max_4}$	Reset $\dot{\beta}_4$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step

Case 13 $\left\|\dot{\beta}_1\right\| > tol,\ \left\|\dot{\beta}_2\right\| > tol,$ $\left\|\dot{\beta}_3\right\| \le tol,$ and $\left\|\dot{\beta}_4\right\| > tol$ Set $\dot{\beta}_3$ equal to zero in y_k Compute k_1 using Function F_{124} Compute interaction forces using Function $F_{int,124}$	if $F_{f_3} \le F_{\max_3}$	Compute k_2, k_3, and k_4 using Function F_{124} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,124}$ Proceed to next integration step
	else $F_{f_3} > F_{\max_3}$	Reset $\dot{\beta}_3$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 14 $\left\|\dot{\beta}_1\right\| > tol,\ \left\|\dot{\beta}_2\right\| \le tol,$ $\left\|\dot{\beta}_3\right\| > tol,$ and $\left\|\dot{\beta}_4\right\| > tol$ Set $\dot{\beta}_2$ equal to zero in y_k Compute k_1 using Function F_{134} Compute interaction forces using Function $F_{int,134}$	if $F_{f_2} \le F_{\max_2}$	Compute k_2, k_3, and k_4 using Function F_{134} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,134}$ Proceed to next integration step
	else $F_{f_2} > F_{\max_2}$	Reset $\dot{\beta}_2$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 15 $\left\|\dot{\beta}_1\right\| \le tol,\ \left\|\dot{\beta}_2\right\| > tol,$ $\left\|\dot{\beta}_3\right\| > tol,$ and $\left\|\dot{\beta}_4\right\| > tol$ Set $\dot{\beta}_1$ equal to zero in y_k Compute k_1 using Function F_{234} Compute interaction forces using Function $F_{int,234}$	if $F_{f_1} \le F_{\max_1}$	Compute k_2, k_3, and k_4 using Function F_{234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,234}$ Proceed to next integration step
	else $F_{f_1} > F_{\max_1}$	Reset $\dot{\beta}_1$ in y_k Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute y_{k+1} using Eq. (3.13) Compute interaction forces using $F_{int,1234}$ Proceed to next integration step
Case 16 $\left\|\dot{\beta}_1\right\| > tol,\ \left\|\dot{\beta}_2\right\| > tol,$ $\left\|\dot{\beta}_3\right\| > tol,$ and $\left\|\dot{\beta}_4\right\| > tol$	Compute k_1, k_2, k_3, and k_4 using Function F_{1234} Compute interaction forces using $F_{\text{int},1234}$ Compute y_{k+1} using Eq. (3.13) Proceed to next integration step	

APPENDIX B: PRIMARY MATLAB SOURCE CODE

The primary MatLab source code controls the logic path selected at every integration step. It utilizes separate function files, a sample of which is shown in Appendices C and D, to compute the motion variables of interest and the interaction forces between the balancing particles and the rotor at each time step. It utilizes a fourth-order Runge-Kutta routine to predict the system behavior at the next time step. While the entire source code for performing a simulation is shown below, only files corresponding to Case 7, where particles 1 and 3 are in motion relative to the rotor and particles 2 and 4 are stationary, will be shown in the following appendices.

```
% File: Wheel_Simulation_with_Marbles_and_Coulomb_and_Viscous_Friction.m
% Author: Jeff Bolton
% Creation Date: October 20, 2009

% Description: This script file solves the equations of motion for a six
% degree-of-freedom rigid wheel having an off-centered mass-imbalance. The
% wheel is allowed to translate in the x-y-z directions and allowed to
% rotate through a 1-2-3 Euler angle set about the geometric center of the
% wheel.  Additionally, the wheel has four point masses constrained to move
% in two races fixed to the wheel which lie in two planes perpendicular to
% the axis of the wheel.  The marbles are subjected to both a viscous and
% dry friction force created by a fluid contained in the race and through
% contact with the rough sides of the race, respectively.

tic;

% Clear all variables, close all figures, and clear the command window.
clear all;
close all;
clc;

% Define the global system parameters that can be passed between the
% script-file and the associated function-files.
global g M m m1 m2 m3 m4 r ri ro r1 r2 r3 r4 d dm d1 d2 d3 d4 bm kx ky kz kth1 kth2 kth3 cx cy cz
cth1 cth2 cth3 w3 muc mus muk Ixx Iyy Izz Ixy Iyx Ixz Izx Iyz Izy

% Define the numerical values for the above system parameters.
g = 32.2;  %acceleration due to gravity
M = 40/g;  %mass of the wheel in slugs
m = 1/16/g;  %mass of the wheel's imbalance
m1 = m;  %mass of the first marble
m2 = m1;  %mass of the second marble
m3 = m;  %mass of the third marble
m4 = m3;  %mass of the fourth marble
ro = 9/12;  %outer radius of the wheel
ri = 8/12;  %inner radius of the wheel
r = 6/12;  %radial position of the imbalance mass
r1 = ri;  %radial position of the first marble
r2 = r1;  %radial position of the second marble
r3 = ri;  %radial position of the third marble
r4 = r3;  %radial position of the fourth marble
bm = 0; %initial angular position of the imbalance mass in the body frame
d = 8/12;  %depth of the wheel
dm = 2/12;  %axial position of the imbalance mass
d1 = d/2;  %axial position of the first marble
d2 = d1;  %axial position of the second marble
d3 = -d/2;  %axial position of the third marble
d4 = d3;  %axial position of the fourth marble
```

```
%Compute the Inertia Tensor for the Wheel and Imbalance Alone
Ixx = 1/12*M*d^2 + 1/4*M*(ro^2 + ri^2) + m*((r*sin(bm))^2 + dm^2);
Iyy = 1/12*M*d^2 + 1/4*M*(ro^2 + ri^2) + m*((r*cos(bm))^2 + dm^2);
Izz = 1/2*M*(ro^2 + ri^2) + m*r^2;
Ixy = m*r^2*cos(bm)*sin(bm);
Iyx = Ixy;
Ixz = m*r*dm*cos(bm);
Izx = Ixz;
Iyz = m*r*dm*sin(bm);
Izy = Iyz;

%List of Remaining Wheel Parameters
kx = 100;  %spring stiffness in the x-direction (vertical)
ky = 100;  %spring stiffness in the y-direction (horizontal)
kz = 1e5;  %spring stiffness in the z-direction (axial)
zetax = 0.7;  %damping ratio in the x-direction
zetay = 0.7;  %damping ratio in the y-direction
kth1 = 5e3;  %torsional spring stiffness about the inertial X-axis
kth2 = 5e3;  %torsional spring stiffness about the intermediate y2-axis
kth3 = 0;  %torsional spring stiffness about the body-fixed z-axis
zetath1 = 0.7;  %torsional damping ratio around the inertial X-axis
zetath2 = 0.7;  %torsional damping ratio around the intermediate y2-axis
cx = 2*zetax*sqrt(kx*M);  %viscous damping coefficient in the inertial X-direction
cy = 2*zetay*sqrt(kx*M);  %viscous damping coefficient in the inertial Y-direction
cz = 500;  %viscous damping coefficient in the inertial Z-direction
cth1 = 2*zetath1*sqrt(kth1*Ixx);  %torsional viscous damping coefficient about the inertial X-
axis
cth2 = 2*zetath2*sqrt(kth2*Iyy);  %torsional viscous damping coefficient about the y2-axis
cth3 = 0;  %torsional viscous damping coefficient about the body-fixed z-axis
w3 = 50;  %initial angular velocity about the body-fixed z-axis
muc = 0.005;  %viscous damping coefficient for the fluid in the marble races
mus = 0.1;  %coefficient of static friction for steel on steel
muk = 0.75*mus;  %coefficient of kinetic friction for steel on steel
%g = 0; %ignore gravity to consider a vertically oriented rotor

% Define the initial conditions for the system
x = 0; y = 0; z = 0;  %initial position of the geometric center of the wheel
th1 = 0; th2 = 0; th3 = 0;  %initial angular displacement of the body-fixed axes
b1 =  30*pi/180; b2 = 272*pi/180; b3 = 120*pi/180; b4 = 300*pi/180;  %initial angular
displacements of the marbles relative to the body-fixed axes
xd = 0; yd = 0; zd = 0;  %initial velocity of the geometric center of the wheel
th1d = 0; th2d = 0; th3d = w3;   %initial angular velocity of the body-fixed axes
b1d = 0.15; b2d = 0.15; b3d = 0; b4d = -0.15;  %initial angular velocity of the marbles relative
to the body-fixed axes

% Form the State Vector Y
Y = [x y z th1 th2 th3 b1 b2 b3 b4 xd yd zd th1d th2d th3d b1d b2d b3d b4d];


% Define the integration loop for the solution
j = 1; %initialize counter variable j
time = 300;  %total time length of simulation
dt = 0.001;  %size of time steps
t = 0:dt:time;  %set array of time increments

%Define the tolerance to check if the marble velocity goes to zero
tol = 1e-6;

data = zeros(length(t),30);  %Initialize the matrix of output data (positions, velocities, and
accelerations)
force = zeros(length(t),12);  %Initialize the matrix of interaction forces
choice = zeros(length(t),1);  %Initialize the vector which will track the integration path
selected at each time step

%Initialize the Mass Moments of Inertia and Products of Inertia Vectors for the entire System
Ixxp = zeros(length(t),1);
Iyyp = zeros(length(t),1);
Izzp = zeros(length(t),1);
Ixyp = zeros(length(t),1);
Ixzp = zeros(length(t),1);
```

```
Iyzp = zeros(length(t),1);

for j = 1:length(t)
    data(j,1:20) = Y;  %Populate the data matrix with the positions and velocities at the current
time step

    beta1dot = abs(Y(17));  %magnitude of marble 1 angular velocity
    beta2dot = abs(Y(18));  %magnitude of marble 2 angular velocity
    beta3dot = abs(Y(19));  %magnitude of marble 3 angular velocity
    beta4dot = abs(Y(20));  %magnitude of marble 4 angular velocity

    %All of the marbles slip
    if beta1dot >= tol && beta2dot >= tol && beta3dot >= tol && beta4dot >= tol

        %Compute the Runge-Kutta Slopes for the current time step
        k1 = Fun1234(t(j),Y);
        k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
        k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
        k4 = Fun1234(t(j)+dt,Y+k3*dt);

        force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current time
step
        Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
        data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the current
time step
        choice(j) = 1234;  %Mark the path taken by noting which marbles slip

    %Marbles 1, 2, and 3 slip
    elseif beta1dot >= tol && beta2dot >= tol && beta3dot >= tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marble 4 slips or sticks for the
current time step
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun123(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint123(t(j),Ynew,k1);
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff4 <= F4max  %Assumption of no-slip is valid

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid and marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
```

```
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Marbles 1, 2, and 4 slip
    elseif beta1dot >= tol && beta2dot >= tol && beta3dot < tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marble 3 slips or sticks for the
current time step
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun124(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint124(t(j),Ynew,k1);
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;

        if Ff3 <= F3max  %Assumption of no-slip is valid

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid and marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Marbles 1, 3, and 4 slip
    elseif beta1dot >= tol && beta2dot < tol && beta3dot >= tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marble 4 slips or sticks for the
current time step
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun134(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint134(t(j),Ynew,k1);
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;

        if Ff2 <= F2max  %Assumption of no-slip is valid

            %Compute the remaining Runge-Kutta Slopes
```

```
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid and marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Marbles 2, 3, and 4 slip
    elseif beta1dot < tol && beta2dot >= tol && beta3dot >= tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marble 1 slips or sticks for the
current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun234(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint234(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;

        if Ff1 <= F1max  %Assumption of no-slip is valid

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid and marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
```

```
        end

    %Marbles 1 and 2 slip
    elseif beta1dot >= tol && beta2dot >= tol && beta3dot < tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 3 and 4 slip or stick for the
current time step
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun12(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint12(t(j),Ynew,k1);
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip is valid for both marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun12(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun12(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun12(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 12;  %Mark the path taken by noting which marbles slip

        elseif Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip is valid for marble 3, but
marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        elseif Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip is valid for marble 4, but
marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun123(t(j),Ynew);
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
```

```
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        else %Assumption of no slip is invalid for both marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end


    %Marbles 1 and 3 slip
    elseif beta1dot >= tol && beta2dot < tol && beta3dot >= tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 2 and 4 slip or stick for the
current time step
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun13(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint13(t(j),Ynew,k1);
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff2 <= F2max && Ff4 <= F4max  %Assumption of no-slip is valid for both marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun13(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun13(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun13(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 13;  %Mark the path taken by noting which marbles slip

        elseif Ff2 <= F2max && Ff4 > F4max  %Assumption of no-slip is valid for marble 2, but
marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
```

```
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        elseif Ff2 > F2max && Ff4 <= F4max  %Assumption of no-slip is valid for marble 4, but
marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun123(t(j),Ynew);
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        else %Assumption of no slip is invalid for both marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Marbles 1 and 4 slip
    elseif beta1dot >= tol && beta2dot < tol && beta3dot < tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marbles 2 and 3 slip or stick for the
current time step
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun14(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint14(t(j),Ynew,k1);
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;

        if Ff2 <= F2max && Ff3 <= F3max  %Assumption of no-slip is valid for both marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun14(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun14(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun14(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
```

```
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 14;  %Mark the path taken by noting which marbles slip

        elseif Ff2 <= F2max && Ff3 > F3max  %Assumption of no-slip is valid for marble 2, but
marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        elseif Ff2 > F2max && Ff3 <= F3max  %Assumption of no-slip is valid for marble 3, but
marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        else %Assumption of no slip is invalid for both marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end


    %Marbles 2 and 3 slip
    elseif beta1dot < tol && beta2dot >= tol && beta3dot >= tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1 and 4 slip or stick for the
current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun23(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
```

```
        Forces = Fint23(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff1 <= F1max && Ff4 <= F4max  %Assumption of no-slip is valid for both marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun23(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun23(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun23(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 23;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff4 > F4max  %Assumption of no-slip is valid for marble 1, but
marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff4 <= F4max  %Assumption of no-slip is valid for marble 4, but
marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 1
            Ynew(17) = Y(17);  %Reset beta1dot

            k1 = Fun123(t(j),Ynew);
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        else %Assumption of no slip is invalid for both marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
```

```
        end

    %Marbles 2 and 4 slip
    elseif beta1dot < tol && beta2dot >= tol && beta3dot < tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1 and 3 slip or stick for the
current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun24(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint24(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;

        if Ff1 <= F1max && Ff3 <= F3max  %Assumption of no-slip is valid for both marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun24(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun24(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun24(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 24;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff3 > F3max  %Assumption of no-slip is valid for marble 1, but
marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff3 <= F3max  %Assumption of no-slip is valid for marble 3, but
marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(17) = Y(17);  %Reset beta1dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
```

```
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        else %Assumption of no slip is invalid for both marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Marbles 3 and 4 slip
    elseif beta1dot < tol && beta2dot < tol && beta3dot >= tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1 and 2 slip or stick for the
current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun34(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint34(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;

        if Ff1 <= F1max && Ff2 <= F2max  %Assumption of no-slip is valid for both marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun34(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun34(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun34(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 34;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max  %Assumption of no-slip is valid for marble 1, but
marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
```

```
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max  %Assumption of no-slip is valid for marble 2, but
marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 1
            Ynew(17) = Y(17);  %Reset beta1dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        else %Assumption of no slip is invalid for both marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector
            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Only Marble 1 slips
    elseif beta1dot >= tol && beta2dot < tol && beta3dot < tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 2, 3, and 4 slip or stick for
the current time step
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun1(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint1(t(j),Ynew,k1);
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff2 <= F2max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip is valid for all
marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun1(t(j)+1/2*dt,Ynew+1/2*k1*dt);
```

```
            k3 = Fun1(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun1(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 1;  %Mark the path taken by noting which marbles slip

        elseif Ff2 > F2max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip is valid for
marbles 3 and 4, but marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun12(t(j),Ynew);
            k2 = Fun12(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun12(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun12(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint12(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 12;  %Mark the path taken by noting which marbles slip

        elseif Ff2 <= F2max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip is valid for
marbles 2 and 4, but marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun13(t(j),Ynew);
            k2 = Fun13(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun13(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun13(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint13(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 13;  %Mark the path taken by noting which marbles slip

        elseif Ff2 <= F2max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip is valid for
marbles 2 and 3, but marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun14(t(j),Ynew);
            k2 = Fun14(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun14(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun14(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint14(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 14;  %Mark the path taken by noting which marbles slip

        elseif Ff2 > F2max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip is valid for
marble 4, but marbles 2 and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2 and 3
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun123(t(j),Ynew);
```

```
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        elseif Ff2 > F2max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip is valid for
marble 3, but marbles 2 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2 and 4
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        elseif Ff2 <= F2max && Ff3 > F3max && Ff4 > F4max  %Assumption of no-slip is valid for
marble 2, but marbles 3 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 3 and 4
            Ynew(19) = Y(19);  %Reset beta3dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid for all marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector

            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Only Marble 2 slips
    elseif beta1dot < tol && beta2dot >= tol && beta3dot < tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1, 3, and 4 slip or stick for
the current time step
```

```
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun2(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint2(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff1 <= F1max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip is valid for all
marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun2(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun2(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun2(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 2;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip is valid for
marbles 3 and 4, but marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 1
            Ynew(17) = Y(17);  %Reset beta1dot

            k1 = Fun12(t(j),Ynew);
            k2 = Fun12(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun12(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun12(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint12(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 12;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip is valid for
marbles 1 and 4, but marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun23(t(j),Ynew);
            k2 = Fun23(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun23(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun23(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint23(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
```

```
            choice(j) = 23;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip is valid for
marbles 1 and 3, but marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun24(t(j),Ynew);
            k2 = Fun24(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun24(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun24(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint24(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 24;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip is valid for
marble 4, but marbles 1 and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 3
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun123(t(j),Ynew);
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip is valid for
marble 3, but marbles 1 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 4
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff3 > F3max && Ff4 > F4max  %Assumption of no-slip is valid for
marble 1, but marbles 3 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 3 and 4
            Ynew(19) = Y(19);  %Reset beta3dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);
```

```
            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid for all marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector

            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Only Marble 3 slips
    elseif beta1dot < tol && beta2dot < tol && beta3dot >= tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1, 2, and 4 slip or stick for
the current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun3(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint3(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff1 <= F1max && Ff2 <= F2max && Ff4 <= F4max  %Assumption of no-slip is valid for all
marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun3(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun3(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun3(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 3;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff4 <= F4max  %Assumption of no-slip is valid for
marbles 2 and 4, but marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 1
            Ynew(17) = Y(17);  %Reset beta1dot
```

```
            k1 = Fun13(t(j),Ynew);
            k2 = Fun13(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun13(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun13(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint13(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 13;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff4 <= F4max  %Assumption of no-slip is valid for
marbles 1 and 4, but marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun23(t(j),Ynew);
            k2 = Fun23(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun23(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun23(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint23(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 23;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 <= F2max && Ff4 > F4max  %Assumption of no-slip is valid for
marbles 1 and 2, but marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun34(t(j),Ynew);
            k2 = Fun34(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun34(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun34(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint34(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 34;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 > F2max && Ff4 <= F4max  %Assumption of no-slip is valid for
marble 4, but marbles 1 and 2 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 2
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun123(t(j),Ynew);
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff4 > F4max  %Assumption of no-slip is valid for
marble 2, but marbles 1 and 4 slip
```

```
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 4
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff4 > F4max  %Assumption of no-slip is valid for
marble 1, but marbles 2 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2 and 4
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid for all marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector

            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %Only Marble 4 slips
    elseif beta1dot < tol && beta2dot < tol && beta3dot < tol && beta4dot >= tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1, 2, and 3 slip or stick for
the current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun4(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint4(t(j),Ynew,k1);
```

```
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;

        if Ff1 <= F1max && Ff2 <= F2max && Ff3 <= F3max  %Assumption of no-slip is valid for all
marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun4(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun4(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun4(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 4;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff3 <= F3max  %Assumption of no-slip is valid for
marbles 2 and 3, but marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 1
            Ynew(17) = Y(17);  %Reset beta1dot

            k1 = Fun14(t(j),Ynew);
            k2 = Fun14(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun14(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun14(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint14(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 14;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff3 <= F3max  %Assumption of no-slip is valid for
marbles 1 and 3, but marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun24(t(j),Ynew);
            k2 = Fun24(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun24(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun24(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint24(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 24;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 <= F2max && Ff3 > F3max  %Assumption of no-slip is valid for
marbles 1 and 2, but marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun34(t(j),Ynew);
            k2 = Fun34(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun34(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun34(t(j)+dt,Ynew+k3*dt);
```

```
            force(j,:) = Fint34(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 34;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 > F2max && Ff3 <= F4max  %Assumption of no-slip is valid for
marble 3, but marbles 1 and 2 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 2
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff3 > F3max  %Assumption of no-slip is valid for
marble 2, but marbles 1 and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 3
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff3 > F3max  %Assumption of no-slip is valid for
marble 1, but marbles 2 and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2 and 3
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid for all marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector

            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
```

```
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end

    %No marbles slip
    elseif beta1dot < tol && beta2dot < tol && beta3dot < tol && beta4dot < tol

        Ynew = Y;  %Form a new state vector to determine if marbles 1, 2, 3, and 4 slip or stick
for the current time step
        Ynew(17) = 0;  %Set beta1dot equal to zero for the current time step to assume
equilibrium
        Ynew(18) = 0;  %Set beta2dot equal to zero for the current time step to assume
equilibrium
        Ynew(19) = 0;  %Set beta3dot equal to zero for the current time step to assume
equilibrium
        Ynew(20) = 0;  %Set beta4dot equal to zero for the current time step to assume
equilibrium

        %Compute the Runge-Kutta Slopes at the beginning of the current time step
        k1 = Fun0(t(j),Ynew);

        %Examine the normal force, equilibrium friction force, and  maximum allowable friction
force to check assumption of no-slip
        Forces = Fint0(t(j),Ynew,k1);
        Fxy1 = abs(Forces(1));
        Ff1 = abs(Forces(3));
        F1max = mus*Fxy1;
        Fxy2 = abs(Forces(4));
        Ff2 = abs(Forces(6));
        F2max = mus*Fxy2;
        Fxy3 = abs(Forces(7));
        Ff3 = abs(Forces(9));
        F3max = mus*Fxy3;
        Fxy4 = abs(Forces(10));
        Ff4 = abs(Forces(12));
        F4max = mus*Fxy4;

        if Ff1 <= F1max && Ff2 <= F2max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip
is valid for all marbles

            %Compute the remaining Runge-Kutta Slopes
            k2 = Fun0(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun0(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun0(t(j)+dt,Ynew+k3*dt);

            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            force(j,:) = Forces;  %Compute the interaction forces at the current time step
            choice(j) = 0;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-
slip is valid for marbles 2, 3, and 4, but marble 1 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 1
            Ynew(17) = Y(17);  %Reset beta1dot

            k1 = Fun1(t(j),Ynew);
            k2 = Fun1(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun1(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun1(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint1(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
```

```
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-
slip is valid for marbles 1, 3, and 4, but marble 2 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 2
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun2(t(j),Ynew);
            k2 = Fun2(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun2(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun2(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint2(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 2;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 <= F2max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-
slip is valid for marbles 1, 2, and 4, but marble 3 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 3
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun3(t(j),Ynew);
            k2 = Fun3(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun3(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun3(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint3(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 3;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 <= F2max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-
slip is valid for marbles 1, 2, and 3, but marble 4 slips
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marble 4
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun4(t(j),Ynew);
            k2 = Fun4(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun4(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun4(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint4(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 4;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 > F2max && Ff3 <= F3max && Ff4 <= F4max  %Assumption of no-slip
is valid for marbles 3 and 4, but marbles 1 and 2 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 2
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(18) = Y(18);  %Reset beta2dot

            k1 = Fun12(t(j),Ynew);
            k2 = Fun12(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun12(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun12(t(j)+dt,Ynew+k3*dt);
```

```
            force(j,:) = Fint12(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 12;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip
is valid for marbles 2 and 4, but marbles 1 and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 3
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun13(t(j),Ynew);
            k2 = Fun13(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun13(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun13(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint13(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 13;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 <= F2max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip
is valid for marbles 2 and 3, but marbles 1 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1 and 4
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun14(t(j),Ynew);
            k2 = Fun14(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun14(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun14(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint14(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 14;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip
is valid for marbles 1 and 4, but marbles 2 and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2 and 3
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun23(t(j),Ynew);
            k2 = Fun23(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun23(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun23(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint23(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 23;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip
is valid for marbles 1 and 3, but marbles 2 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2 and 4
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(20) = Y(20);  %Reset beta4dot
```

```
            k1 = Fun24(t(j),Ynew);
            k2 = Fun24(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun24(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun24(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint24(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 24;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 <= F2max && Ff3 > F3max && Ff4 > F4max  %Assumption of no-slip
is valid for marbles 1 and 2, but marbles 3 and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 3 and 4
            Ynew(19) = Y(19);  %Reset beta3dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun34(t(j),Ynew);
            k2 = Fun34(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun34(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun34(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint34(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 34;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 > F2max && Ff3 > F3max && Ff4 <= F4max  %Assumption of no-slip
is valid for marble 4, but marbles 1, 2, and 3 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1, 2, and 3
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(19) = Y(19);  %Reset beta3dot

            k1 = Fun123(t(j),Ynew);
            k2 = Fun123(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun123(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun123(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint123(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 123;  %Mark the path taken by noting which marbles slip

        elseif Ff1 > F1max && Ff2 > F2max && Ff3 <= F3max && Ff4 > F4max  %Assumption of no-slip
is valid for marble 3, but marbles 1, 2, and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1, 2, and 4
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun124(t(j),Ynew);
            k2 = Fun124(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun124(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun124(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint124(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 124;  %Mark the path taken by noting which marbles slip
```

```
        elseif Ff1 > F1max && Ff2 <= F2max && Ff3 > F3max && Ff4 > F4max  %Assumption of no-slip
is valid for marble 2, but marbles 1, 3, and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 1, 3, and 4
            Ynew(17) = Y(17);  %Reset beta1dot
            Ynew(19) = Y(19);  %Reset beta3dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun134(t(j),Ynew);
            k2 = Fun134(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun134(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun134(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint134(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 134;  %Mark the path taken by noting which marbles slip

        elseif Ff1 <= F1max && Ff2 > F2max && Ff3 > F3max && Ff4 > F4max  %Assumption of no-slip
is valid for marble 1, but marbles 2, 3, and 4 slip
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector values for marbles 2, 3, and 4
            Ynew(18) = Y(18);  %Reset beta2dot
            Ynew(19) = Y(19);  %Reset beta3dot
            Ynew(20) = Y(20);  %Reset beta4dot

            k1 = Fun234(t(j),Ynew);
            k2 = Fun234(t(j)+1/2*dt,Ynew+1/2*k1*dt);
            k3 = Fun234(t(j)+1/2*dt,Ynew+1/2*k2*dt);
            k4 = Fun234(t(j)+dt,Ynew+k3*dt);

            force(j,:) = Fint234(t(j),Ynew,k1);  %Compute the interaction forces at the current
time step
            Y = Ynew+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 234;  %Mark the path taken by noting which marbles slip

        else %Assumption of no-slip is invalid for all marbles
            %Compute the Runge-Kutta Slopes for the current time step using the original state
vector

            k1 = Fun1234(t(j),Y);
            k2 = Fun1234(t(j)+1/2*dt,Y+1/2*k1*dt);
            k3 = Fun1234(t(j)+1/2*dt,Y+1/2*k2*dt);
            k4 = Fun1234(t(j)+dt,Y+k3*dt);

            force(j,:) = Fint1234(t(j),Y,k1);  %Compute the interaction forces at the current
time step
            Y = Y+1/6*(k1+2*k2+2*k3+k4)*dt;  %Compute the state vector for the next time step
            data(j,21:30) = k1(11:20);  %Append the accelerations to the data matrix for the
current time step
            choice(j) = 1234;  %Mark the path taken by noting which marbles slip
        end
    end

    %Compute the Inertia Tensor for the system at the current time step
    Ixxp(j) = Ixx + m1*((r1*sin(data(j,7)))^2 + d1^2) + m2*((r2*sin(data(j,8)))^2 + d2^2) +
m3*((r3*sin(data(j,9)))^2 + d3^2) + m4*((r4*sin(data(j,10)))^2 + d4^2);
    Iyyp(j) = Iyy + m1*((r1*cos(data(j,7)))^2 + d1^2) + m2*((r2*cos(data(j,8)))^2 + d2^2) +
m3*((r3*cos(data(j,9)))^2 + d3^2) + m4*((r4*cos(data(j,10)))^2 + d4^2);
    Izzp(j) = Izz + m1*((r1*sin(data(j,7)))^2 + (r1*cos(data(j,7)))^2) +
m2*((r2*sin(data(j,8)))^2 + (r2*cos(data(j,8)))^2) + m3*((r3*sin(data(j,9)))^2 +
(r3*cos(data(j,9)))^2) + m4*((r4*sin(data(j,10)))^2 + (r4*cos(data(j,10)))^2);
    Ixyp(j) = -(Ixy + m1*r1*cos(data(j,7))*r1*sin(data(j,7)) +
m2*r2*cos(data(j,8))*r2*sin(data(j,8)) + m3*r3*cos(data(j,9))*r3*sin(data(j,9)) +
m4*r4*cos(data(j,10))*r4*sin(data(j,10)));
    Ixzp(j) = -(Ixz + m1*r1*cos(data(j,7))*d1 + m2*r2*cos(data(j,8))*d2 + m3*r3*cos(data(j,9))*d3
+ m4*r4*cos(data(j,10))*d4);
```

```
    Iyzp(j) = -(Iyz + m1*d1*r1*sin(data(j,7)) + m2*d2*r2*sin(data(j,8)) + m3*d3*r3*sin(data(j,9))
+ m4*d4*r4*sin(data(j,10)));

end

seconds = toc;  %Simulation time in seconds
minutes = seconds/60;  %Simulation time in minutes
hours = minutes/60;  %Simulation time in hours

fprintf('The simulation took %0.2f seconds, or %0.2f minutes, or %0.2f hours to run.\n', seconds,
minutes, hours)
```

APPENDIX C: SAMPLE SOURCE CODE FOR EQUATIONS OF MOTION

Due to the complexity and length of the MatLab program utilized in solving the simulations for this research, Appendix C will only contain an example of one of the sixteen function files used to solve the equations of motion for the six rotor motion variables and said balancing particles. Shown below is Case 7 from Table 3.1 where particles 1 and 3 move relative to the cylinder while particles 2 and 4 are stationary.

Case 7: Particles 1 and 3 Move Relative to the Cylinder

```
function dY = Fun13(t,Y)

% File: Fun13.m
% Author: Jeff Bolton
% Creation Date: October 18, 2009

% Description: This function-m file solves the equations of motion for a six
% degree-of-freedom rigid wheel having an off-centered mass-imbalance and four
% balancing marbles that are subjected to viscous and dry friction while traveling
% along fixed races within the wheel. The wheel is allowed to translate in the
% x-y-z directions and allowed to rotate through a 1-2-3 Euler angle set about
% the geometric center of the wheel.  Y is the state vector for the wheel,
% and dY is the derivative of the state vector.
%
% In this function, marbles 1 and 3 slip.
%
% Y = [x y z th1 th2 th3 b1 b2 b3 b4 xd yd zd th1d th2d th3d b1d b2d b3d b4d]
% dY = [xd yd zd th1d th2d th3d b1d b2d b3d b4d xdd ydd zdd th1dd th2dd th3dd b1dd b2dd b3dd
b4dd]

global g M m m1 m2 m3 m4 r ri ro r1 r2 r3 r4 d dm d1 d2 d3 d4 bm kx ky kz kth1 kth2 kth3 cx cy cz
cth1 cth2 cth3 w3 muc muk Ixx Iyy Izz Ixy Iyx Ixz Izx Iyz Izy

% Define the variables of the equation in terms of the input vector Y
x = Y(1);
y = Y(2);
z = Y(3);
th1 = Y(4);
th2 = Y(5);
th3 = Y(6);
b1 = Y(7);
b2 = Y(8);
b3 = Y(9);
b4 = Y(10);
xd = Y(11);
yd = Y(12);
zd = Y(13);
th1d = Y(14);
th2d = Y(15);
th3d = Y(16);
b1d = Y(17);
b2d = Y(18);
b3d = Y(19);
b4d = Y(20);

MM13 = zeros(8,8);  %Initialize the Mass Matrix M[t,y]

f13 = zeros(8,1);  %Initialize the Force Vector f(t,y)

%Populate the Various Elements of the Mass Matrix M[t,y]
```

```
MM13(1,1) = -m - M - m2*cos(b2)*cos(th2)^2*cos(th3)*cos(b2 + th3) -
m4*cos(b4)*cos(th2)^2*cos(th3)*cos(b4 + th3) + (m1*cos(b1)*cos(th2)^2*cos(th3)^2)/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (m3*cos(b3)*cos(th2)^2*cos(th3)^2)/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
m1*sin(th2)^2 - m2*sin(th2)^2 - m3*sin(th2)^2 - m4*sin(th2)^2 +
(m1*cos(th2)^2*cos(th3)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*cos(th2)^2*cos(b2 + th3)*sin(b2)*sin(th3) + (m3*cos(th2)^2*cos(th3)*sin(b3)*sin(th3))/(cos(b3)
- muk*sign(b3d)*sin(b3)) + m4*cos(th2)^2*cos(b4 + th3)*sin(b4)*sin(th3) -
(m1*muk*cos(th2)^2*cos(th3)*sign(b1d)*sin(b1 + th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) -
m2*cos(th2)^2*cos(th3)*sin(b2)*sin(b2 + th3) - m2*cos(b2)*cos(th2)^2*sin(th3)*sin(b2 + th3) -
(m3*muk*cos(th2)^2*cos(th3)*sign(b3d)*sin(b3 + th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
m4*cos(th2)^2*cos(th3)*sin(b4)*sin(b4 + th3) - m4*cos(b4)*cos(th2)^2*sin(th3)*sin(b4 + th3);
MM13(1,2) = cos(th2)*(m1*sin(th1)*sin(th2) + m2*sin(th1)*sin(th2) + m3*sin(th1)*sin(th2) +
m4*sin(th1)*sin(th2) - m2*cos(b2)^2*cos(th3)^2*sin(th1)*sin(th2) -
m4*cos(b4)^2*cos(th3)^2*sin(th1)*sin(th2) + (m1*cos(b1)*cos(th3)^2*sin(th1)*sin(th2))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) - m2*cos(th3)^2*sin(b2)^2*sin(th1)*sin(th2) +
(m3*cos(b3)*cos(th3)^2*sin(th1)*sin(th2))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
m4*cos(th3)^2*sin(b4)^2*sin(th1)*sin(th2) - (m1*cos(b1)*cos(th1)*cos(th3)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (m3*cos(b3)*cos(th1)*cos(th3)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m1*cos(th3)*sin(b1)*sin(th1)*sin(th2)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m3*cos(th3)*sin(b3)*sin(th1)*sin(th2)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m1*cos(th1)*sin(b1)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*sin(b3)*sin(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) -
m2*cos(b2)^2*sin(th1)*sin(th2)*sin(th3)^2 - m4*cos(b4)^2*sin(th1)*sin(th2)*sin(th3)^2 -
m2*sin(b2)^2*sin(th1)*sin(th2)*sin(th3)^2 - m4*sin(b4)^2*sin(th1)*sin(th2)*sin(th3)^2 -
(m1*muk*sign(b1d)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))*sin(b1 + th3))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) - (m3*muk*sign(b3d)*(cos(th3)*sin(th1)*sin(th2) +
cos(th1)*sin(th3))*sin(b3 + th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)));
MM13(1,3) = cos(th2)*(-(m1*cos(th1)*sin(th2)) - m2*cos(th1)*sin(th2) - m3*cos(th1)*sin(th2) -
m4*cos(th1)*sin(th2) + m2*cos(b2)^2*cos(th1)*cos(th3)^2*sin(th2) +
m4*cos(b4)^2*cos(th1)*cos(th3)^2*sin(th2) + (m1*cos(b1)*cos(th1)*cos(th3)^2*sin(th2))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + m2*cos(th1)*cos(th3)^2*sin(b2)^2*sin(th2) +
(m3*cos(b3)*cos(th1)*cos(th3)^2*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*cos(th1)*cos(th3)^2*sin(b4)^2*sin(th2) - (m1*cos(b1)*cos(th3)*sin(th1)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (m3*cos(b3)*cos(th3)*sin(th1)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m1*cos(th1)*cos(th3)*sin(b1)*sin(th2)*sin(th3))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (m3*cos(th1)*cos(th3)*sin(b3)*sin(th2)*sin(th3))/(-cos(b3) +
muk*sign(b3d)*sin(b3)) + (m1*sin(b1)*sin(th1)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*sin(b3)*sin(th1)*sin(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m2*cos(b2)^2*cos(th1)*sin(th2)*sin(th3)^2 + m4*cos(b4)^2*cos(th1)*sin(th2)*sin(th3)^2 +
m2*cos(th1)*sin(b2)^2*sin(th2)*sin(th3)^2 + m4*cos(th1)*sin(b4)^2*sin(th2)*sin(th3)^2 +
(m1*muk*sign(b1d)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b1 + th3))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (m3*muk*sign(b3d)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3))*sin(b3 + th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)));
MM13(1,4) = (cos(th2)*sin(th3)*(-(d1*m1*cos(b1)*cos(b3)*cos(th2)*cos(th3)) -
d3*m3*cos(b1)*cos(b3)*cos(th2)*cos(th3) + m3*r3*cos(b1)*sin(th2) + m1*r1*cos(b3)*sin(th2) +
d1*m1*cos(b3)*cos(th2)*sin(b1)*sin(th3) + d3*m3*cos(b1)*cos(th2)*sin(b3)*sin(th3) +
muk*sign(b3d)*(cos(b1)*cos(th2)*((d1*m1 + d3*m3)*cos(th3)*sin(b3) + d3*m3*cos(b3)*sin(th3)) -
m1*sin(b3)*(r1*sin(th2) + d1*cos(th2)*sin(b1)*sin(th3))) - muk*sign(b1d)*(-
(cos(b3)*cos(th2)*((d1*m1 + d3*m3)*cos(th3)*sin(b1) + d1*m1*cos(b1)*sin(th3))) +
m3*sin(b1)*(r3*sin(th2) + d3*cos(th2)*sin(b3)*sin(th3)) + muk*cos(th2)*sign(b3d)*((d1*m1 +
d3*m3)*cos(th3)*sin(b1)*sin(b3) + (d3*m3*cos(b3)*sin(b1) +
d1*m1*cos(b1)*sin(b3))*sin(th3)))))/((cos(b1) - muk*sign(b1d)*sin(b1))*(-cos(b3) +
muk*sign(b3d)*sin(b3)));
MM13(1,5) = -(dm*m*cos(th2)) - d2*m2*cos(b2)*cos(th2)*cos(th3)*cos(b2 + th3) -
d4*m4*cos(b4)*cos(th2)*cos(th3)*cos(b4 + th3) + (d1*m1*cos(b1)*cos(th2)*cos(th3)^2)/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (d3*m3*cos(b3)*cos(th2)*cos(th3)^2)/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
m*r*cos(bm)*cos(th3)*sin(th2) + m1*r1*cos(b1 + th3)*sin(th2) + m2*r2*cos(b2 + th3)*sin(th2) +
m3*r3*cos(b3 + th3)*sin(th2) + m4*r4*cos(b4 + th3)*sin(th2) +
(d1*m1*cos(th2)*cos(th3)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
d2*m2*cos(th2)*cos(b2 + th3)*sin(b2)*sin(th3) +
(d3*m3*cos(th2)*cos(th3)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
d4*m4*cos(th2)*cos(b4 + th3)*sin(b4)*sin(th3) - m*r*sin(bm)*sin(th2)*sin(th3) -
(d1*m1*muk*cos(th2)*cos(th3)*sign(b1d)*sin(b1 + th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) -
d2*m2*cos(th2)*cos(th3)*sin(b2)*sin(b2 + th3) - d2*m2*cos(b2)*cos(th2)*sin(th3)*sin(b2 + th3) -
(d3*m3*muk*cos(th2)*cos(th3)*sign(b3d)*sin(b3 + th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
d4*m4*cos(th2)*cos(th3)*sin(b4)*sin(b4 + th3) - d4*m4*cos(b4)*cos(th2)*sin(th3)*sin(b4 + th3);
MM13(1,6) = cos(th2)*((m1*r1*cos(b1)*cos(th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*r2*cos(th3)*sin(b2) + (m3*r3*cos(b3)*cos(th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*r4*cos(th3)*sin(b4) + m*r*cos(th3)*sin(bm) + m2*r2*cos(b2)*sin(th3) + m4*r4*cos(b4)*sin(th3) +
m*r*cos(bm)*sin(th3) + (m1*r1*sin(b1)^2*sin(th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*r3*sin(b3)^2*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
```

```
(m1*muk*r1*sign(b1d)*sin(b1)*sin(b1 + th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*muk*r3*sign(b3d)*sin(b3)*sin(b3 + th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)));
MM13(1,7) = (m1*r1*cos(th2)*sin(b1)*(-cos(b1 + th3) + muk*sign(b1d)*sin(b1 + th3)))/(-cos(b1) +
muk*sign(b1d)*sin(b1));
MM13(1,8) = (m3*r3*cos(th2)*sin(b3)*(-cos(b3 + th3) + muk*sign(b3d)*sin(b3 + th3)))/(-cos(b3) +
muk*sign(b3d)*sin(b3));

MM13(2,1) = -((cos(th2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3))*(-
(m1*cos(b3)*cos(th3)*sin(b1)) - m3*cos(b1)*cos(th3)*sin(b3) - m1*cos(b1)*cos(b3)*sin(th3) -
m3*cos(b1)*cos(b3)*sin(th3) + muk*sign(b3d)*(m1*cos(th3)*sin(b1)*sin(b3) + cos(b1)*(-
(m3*cos(b3)*cos(th3)) + (m1 + m3)*sin(b3)*sin(th3))) + muk*sign(b1d)*(-
(m1*cos(b1)*cos(b3)*cos(th3)) + sin(b1)*(m3*cos(th3)*sin(b3) + (m1 + m3)*cos(b3)*sin(th3)) +
muk*sign(b3d)*(m3*cos(b3)*cos(th3)*sin(b1) + sin(b3)*(m1*cos(b1)*cos(th3) - (m1 +
m3)*sin(b1)*sin(th3))))))/((cos(b1) - muk*sign(b1d)*sin(b1))*(-cos(b3) +
muk*sign(b3d)*sin(b3))));
MM13(2,2) = -m - M - m2*cos(b2)*cos(th1)^2*cos(th3)*cos(b2 + th3) -
m4*cos(b4)*cos(th1)^2*cos(th3)*cos(b4 + th3) - m1*cos(th2)^2*sin(th1)^2 -
m2*cos(th2)^2*sin(th1)^2 - m3*cos(th2)^2*sin(th1)^2 - m4*cos(th2)^2*sin(th1)^2 +
(m1*cos(th1)*cos(th3)^2*sin(b1)*sin(th1)*sin(th2))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*cos(th3)^2*sin(b3)*sin(th1)*sin(th2))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
m2*cos(b2)*cos(th3)*cos(b2 + th3)*sin(th1)^2*sin(th2)^2 - m4*cos(b4)*cos(th3)*cos(b4 +
th3)*sin(th1)^2*sin(th2)^2 + (m1*cos(b1)*cos(th3)^2*sin(th1)^2*sin(th2)^2)/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (m3*cos(b3)*cos(th3)^2*sin(th1)^2*sin(th2)^2)/(-cos(b3) +
muk*sign(b3d)*sin(b3)) + (m1*cos(th1)^2*cos(th3)*sin(b1)*sin(th3))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + m2*cos(th1)^2*cos(b2 + th3)*sin(b2)*sin(th3) +
(m3*cos(th1)^2*cos(th3)*sin(b3)*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
m4*cos(th1)^2*cos(b4 + th3)*sin(b4)*sin(th3) -
(2*m1*cos(b1)*cos(th1)*cos(th3)*sin(th1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*m3*cos(b3)*cos(th1)*cos(th3)*sin(th1)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m1*cos(th3)*sin(b1)*sin(th1)^2*sin(th2)^2*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*cos(b2 + th3)*sin(b2)*sin(th1)^2*sin(th2)^2*sin(th3) +
(m3*cos(th3)*sin(b3)*sin(th1)^2*sin(th2)^2*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*cos(b4 + th3)*sin(b4)*sin(th1)^2*sin(th2)^2*sin(th3) + (m1*cos(b1)*cos(th1)^2*sin(th3)^2)/(-
cos(b1) + muk*sign(b1d)*sin(b1)) + (m3*cos(b3)*cos(th1)^2*sin(th3)^2)/(-cos(b3) +
muk*sign(b3d)*sin(b3)) + (m1*cos(th1)*sin(b1)*sin(th1)*sin(th2)*sin(th3)^2)/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m3*cos(th1)*sin(b3)*sin(th1)*sin(th2)*sin(th3)^2)/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (m3*muk*sign(b3d)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))*(-
(cos(b3)*cos(th1)*cos(th3)) + cos(th3)*sin(b3)*sin(th1)*sin(th2) + cos(th1)*sin(b3)*sin(th3) +
cos(b3)*sin(th1)*sin(th2)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
(m1*muk*sign(b1d)*(cos(th3)*sin(th1)*sin(th2) +
cos(th1)*sin(th3))*(sin(b1)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) + cos(b1)*(-
(cos(th1)*cos(th3)) + sin(th1)*sin(th2)*sin(th3))))/(-cos(b1) + muk*sign(b1d)*sin(b1)) -
m2*cos(th1)^2*cos(th3)*sin(b2)*sin(b2 + th3) - m2*cos(th3)*sin(b2)*sin(th1)^2*sin(th2)^2*sin(b2 +
th3) - m2*cos(b2)*cos(th1)^2*sin(th3)*sin(b2 + th3) -
m2*cos(b2)*sin(th1)^2*sin(th2)^2*sin(th3)*sin(b2 + th3) - m4*cos(th1)^2*cos(th3)*sin(b4)*sin(b4 +
th3) - m4*cos(th3)*sin(b4)*sin(th1)^2*sin(th2)^2*sin(b4 + th3) -
m4*cos(b4)*cos(th1)^2*sin(th3)*sin(b4 + th3) - m4*cos(b4)*sin(th1)^2*sin(th2)^2*sin(th3)*sin(b4 +
th3);
MM13(2,3) = m1*cos(th1)*cos(th2)^2*sin(th1) + m2*cos(th1)*cos(th2)^2*sin(th1) +
m3*cos(th1)*cos(th2)^2*sin(th1) + m4*cos(th1)*cos(th2)^2*sin(th1) -
m2*cos(b2)^2*cos(th1)*cos(th3)^2*sin(th1) - m4*cos(b4)^2*cos(th1)*cos(th3)^2*sin(th1) -
m2*cos(th1)*cos(th3)^2*sin(b2)^2*sin(th1) - m4*cos(th1)*cos(th3)^2*sin(b4)^2*sin(th1) +
(m1*cos(th1)^2*cos(th3)^2*sin(b1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)^2*cos(th3)^2*sin(b3)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m2*cos(b2)^2*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2 +
m4*cos(b4)^2*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2 +
(m1*cos(b1)*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*cos(th1)*cos(th3)^2*sin(b2)^2*sin(th1)*sin(th2)^2 +
(m3*cos(b3)*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*cos(th1)*cos(th3)^2*sin(b4)^2*sin(th1)*sin(th2)^2 +
(m1*cos(th1)*cos(th3)*sin(b1)*sin(th1)*sin(th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*cos(th3)*sin(b3)*sin(th1)*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(m1*cos(b1)*cos(th1)^2*cos(th3)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*cos(b3)*cos(th1)^2*cos(th3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m1*cos(b1)*cos(th3)*sin(th1)^2*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m3*cos(b3)*cos(th3)*sin(th1)^2*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m1*cos(th1)*cos(th3)*sin(b1)*sin(th1)*sin(th2)^2*sin(th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*cos(th3)*sin(b3)*sin(th1)*sin(th2)^2*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
m2*cos(b2)^2*cos(th1)*sin(th1)*sin(th3)^2 - m4*cos(b4)^2*cos(th1)*sin(th1)*sin(th3)^2 -
(m1*cos(b1)*cos(th1)*sin(th1)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
m2*cos(th1)*sin(b2)^2*sin(th1)*sin(th3)^2 - (m3*cos(b3)*cos(th1)*sin(th1)*sin(th3)^2)/(cos(b3) -
```

```
muk*sign(b3d)*sin(b3)) - m4*cos(th1)*sin(b4)^2*sin(th1)*sin(th3)^2 +
(m1*sin(b1)*sin(th1)^2*sin(th2)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*sin(b3)*sin(th1)^2*sin(th2)*sin(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m2*cos(b2)^2*cos(th1)*sin(th1)*sin(th2)^2*sin(th3)^2 +
m4*cos(b4)^2*cos(th1)*sin(th1)*sin(th2)^2*sin(th3)^2 +
m2*cos(th1)*sin(b2)^2*sin(th1)*sin(th2)^2*sin(th3)^2 +
m4*cos(th1)*sin(b4)^2*sin(th1)*sin(th2)^2*sin(th3)^2 -
(m1*muk*sign(b1d)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*(-
(sin(b1)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))) + cos(b1)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3))))/(-cos(b1) + muk*sign(b1d)*sin(b1)) -
(m3*muk*sign(b3d)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*(-
(sin(b3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))) + cos(b3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3))))/(-cos(b3) + muk*sign(b3d)*sin(b3));
MM13(2,4) = dm*m*cos(th1)*cos(th2) + d2*m2*cos(b2)^2*cos(th1)*cos(th2)*cos(th3)^2 +
d4*m4*cos(b4)^2*cos(th1)*cos(th2)*cos(th3)^2 + d2*m2*cos(th1)*cos(th2)*cos(th3)^2*sin(b2)^2 +
d4*m4*cos(th1)*cos(th2)*cos(th3)^2*sin(b4)^2 + m1*r1*cos(th2)^2*cos(th3)*sin(b1)*sin(th1) +
m2*r2*cos(th2)^2*cos(th3)*sin(b2)*sin(th1) + m3*r3*cos(th2)^2*cos(th3)*sin(b3)*sin(th1) +
m4*r4*cos(th2)^2*cos(th3)*sin(b4)*sin(th1) + m*r*cos(th3)*sin(bm)*sin(th1) -
m2*r2*cos(b2)*cos(th1)*cos(th3)*sin(th2) - m4*r4*cos(b4)*cos(th1)*cos(th3)*sin(th2) -
m*r*cos(bm)*cos(th1)*cos(th3)*sin(th2) + (m1*r1*cos(th1)*cos(th3)*sin(b1)^2*sin(th2))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m3*r3*cos(th1)*cos(th3)*sin(b3)^2*sin(th2))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m1*r1*cos(b1)*cos(th3)*sin(b1)*sin(th1)*sin(th2)^2)/(cos(b1) -
muk*sign(b1d)*sin(b1)) + m2*r2*cos(th3)*sin(b2)*sin(th1)*sin(th2)^2 +
(m3*r3*cos(b3)*cos(th3)*sin(b3)*sin(th1)*sin(th2)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*r4*cos(th3)*sin(b4)*sin(th1)*sin(th2)^2 +
(d1*m1*cos(th1)*cos(th2)*cos(th3)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(th1)*cos(th2)*cos(th3)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m*r*cos(bm)*sin(th1)*sin(th3) + m1*r1*cos(b1)*cos(th2)^2*sin(th1)*sin(th3) +
m2*r2*cos(b2)*cos(th2)^2*sin(th1)*sin(th3) + m3*r3*cos(b3)*cos(th2)^2*sin(th1)*sin(th3) +
m4*r4*cos(b4)*cos(th2)^2*sin(th1)*sin(th3) +
(m1*r1*cos(b1)*cos(th1)*sin(b1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*r2*cos(th1)*sin(b2)*sin(th2)*sin(th3) +
(m3*r3*cos(b3)*cos(th1)*sin(b3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*r4*cos(th1)*sin(b4)*sin(th2)*sin(th3) + m*r*cos(th1)*sin(bm)*sin(th2)*sin(th3) +
(d1*m1*cos(b1)*cos(th2)*cos(th3)*sin(th1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(b3)*cos(th2)*cos(th3)*sin(th1)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m2*r2*cos(b2)*sin(th1)*sin(th2)^2*sin(th3) + m4*r4*cos(b4)*sin(th1)*sin(th2)^2*sin(th3) +
(m1*r1*sin(b1)^2*sin(th1)*sin(th2)^2*sin(th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*r3*sin(b3)^2*sin(th1)*sin(th2)^2*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
d2*m2*cos(b2)^2*cos(th1)*cos(th2)*sin(th3)^2 + d4*m4*cos(b4)^2*cos(th1)*cos(th2)*sin(th3)^2 +
(d1*m1*cos(b1)*cos(th1)*cos(th2)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
d2*m2*cos(th1)*cos(th2)*sin(b2)^2*sin(th3)^2 +
(d3*m3*cos(b3)*cos(th1)*cos(th2)*sin(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
d4*m4*cos(th1)*cos(th2)*sin(b4)^2*sin(th3)^2 +
(d1*m1*cos(th2)*sin(b1)*sin(th1)*sin(th2)*sin(th3)^2)/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(th2)*sin(b3)*sin(th1)*sin(th2)*sin(th3)^2)/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(m3*muk*sign(b3d)*(r3*sin(b3)*sin(th2) + d3*cos(th2)*sin(th3))*(-(cos(b3)*cos(th1)*cos(th3)) +
cos(th3)*sin(b3)*sin(th1)*sin(th2) + cos(th1)*sin(b3)*sin(th3) +
cos(b3)*sin(th1)*sin(th2)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(m1*muk*sign(b1d)*(r1*sin(b1)*sin(th2) +
d1*cos(th2)*sin(th3))*(sin(b1)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) + cos(b1)*(-
(cos(th1)*cos(th3)) + sin(th1)*sin(th2)*sin(th3))))/(-cos(b1) + muk*sign(b1d)*sin(b1));
MM13(2,5) = -(m1*r1*cos(th2)*cos(b1 + th3)*sin(th1)) - m2*r2*cos(th2)*cos(b2 + th3)*sin(th1) -
m3*r3*cos(th2)*cos(b3 + th3)*sin(th1) - m4*r4*cos(th2)*cos(b4 + th3)*sin(th1) -
m*r*cos(th2)*cos(bm + th3)*sin(th1) - dm*m*sin(th1)*sin(th2) - d2*m2*cos(b2)*cos(b2 +
th3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) - d4*m4*cos(b4)*cos(b4 +
th3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) -
(d1*m1*cos(b1)*cos(th3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (d1*m1*muk*cos(th3)*sign(b1d)*sin(b1)*(cos(th3)*sin(th1)*sin(th2) +
cos(th1)*sin(th3)))/(-cos(b1) + muk*sign(b1d)*sin(b1)) -
(d3*m3*cos(b3)*cos(th3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (d3*m3*muk*cos(th3)*sign(b3d)*sin(b3)*(cos(th3)*sin(th1)*sin(th2) +
cos(th1)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(d1*m1*muk*cos(b1)*cos(th3)*sign(b1d)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(-cos(b1)
+ muk*sign(b1d)*sin(b1)) + (d1*m1*cos(th3)*sin(b1)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(-cos(b1) + muk*sign(b1d)*sin(b1)) - d2*m2*cos(b2 +
th3)*sin(b2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)) +
(d3*m3*muk*cos(b3)*cos(th3)*sign(b3d)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(-cos(b3)
+ muk*sign(b3d)*sin(b3)) + (d3*m3*cos(th3)*sin(b3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) - d4*m4*cos(b4 +
th3)*sin(b4)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)) -
```

```
d2*m2*sin(b2)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))*sin(b2 + th3) +
d2*m2*cos(b2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3))*sin(b2 + th3) -
d4*m4*sin(b4)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))*sin(b4 + th3) +
d4*m4*cos(b4)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3))*sin(b4 + th3);
MM13(2,6) = -(m*r*cos(th1)*cos(bm + th3)) + (m1*r1*cos(b1)*sin(b1)*(cos(th3)*sin(th1)*sin(th2) +
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*sign(b1d)*sin(b1)^2*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + m2*r2*sin(b2)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) +
(m3*r3*cos(b3)*sin(b3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*muk*r3*sign(b3d)*sin(b3)^2*(cos(th3)*sin(th1)*sin(th2) +
cos(th1)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
m4*r4*sin(b4)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) - m2*r2*cos(b2)*(cos(th1)*cos(th3)
- sin(th1)*sin(th2)*sin(th3)) - m4*r4*cos(b4)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)) +
(m1*muk*r1*cos(b1)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*r1*sin(b1)^2*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*muk*r3*cos(b3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*r3*sin(b3)^2*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + m*r*sin(th1)*sin(th2)*sin(bm +
th3);
MM13(2,7) = (m1*r1*sin(b1)*(cos(b1 + th3)*sin(th1)*sin(th2) -
muk*sign(b1d)*(sin(b1)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) + cos(b1)*(-
(cos(th1)*cos(th3)) + sin(th1)*sin(th2)*sin(th3))) + cos(th1)*sin(b1 + th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1));
MM13(2,8) = (m3*r3*sin(b3)*(cos(b3 + th3)*sin(th1)*sin(th2) -
muk*sign(b3d)*(sin(b3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) + cos(b3)*(-
(cos(th1)*cos(th3)) + sin(th1)*sin(th2)*sin(th3))) + cos(th1)*sin(b3 + th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3));

MM13(3,1) = -((cos(th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*(-
(m1*cos(b3)*cos(th3)*sin(b1)) - m3*cos(b1)*cos(th3)*sin(b3) - m1*cos(b1)*cos(b3)*sin(th3) -
m3*cos(b1)*cos(b3)*sin(th3) + muk*sign(b3d)*(m1*cos(th3)*sin(b1)*sin(b3) + cos(b1)*(-
(m3*cos(b3)*cos(th3)) + (m1 + m3)*sin(b3)*sin(th3))) + muk*sign(b1d)*(-
(m1*cos(b1)*cos(b3)*cos(th3)) + sin(b1)*(m3*cos(th3)*sin(b3) + (m1 + m3)*cos(b3)*sin(th3)) +
muk*sign(b3d)*(m3*cos(b3)*cos(th3)*sin(b1) + sin(b3)*(m1*cos(b1)*cos(th3) - (m1 +
m3)*sin(b1)*sin(th3))))))/((cos(b1) - muk*sign(b1d)*sin(b1))*(-cos(b3) +
muk*sign(b3d)*sin(b3))));
MM13(3,2) = m1*cos(th1)*cos(th2)^2*sin(th1) + m2*cos(th1)*cos(th2)^2*sin(th1) +
m3*cos(th1)*cos(th2)^2*sin(th1) + m4*cos(th1)*cos(th2)^2*sin(th1) -
m2*cos(b2)^2*cos(th1)*cos(th3)^2*sin(th1) - m4*cos(b4)^2*cos(th1)*cos(th3)^2*sin(th1) -
m2*cos(th1)*cos(th3)^2*sin(b2)^2*sin(th1) - m4*cos(th1)*cos(th3)^2*sin(b4)^2*sin(th1) +
(m1*cos(th3)^2*sin(b1)*sin(th1)^2*sin(th2))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th3)^2*sin(b3)*sin(th1)^2*sin(th2))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
m2*cos(b2)^2*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2 +
m4*cos(b4)^2*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2 +
(m1*cos(b1)*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*cos(th1)*cos(th3)^2*sin(b2)^2*sin(th1)*sin(th2)^2 +
(m3*cos(b3)*cos(th1)*cos(th3)^2*sin(th1)*sin(th2)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*cos(th1)*cos(th3)^2*sin(b4)^2*sin(th1)*sin(th2)^2 +
(m1*cos(th1)*cos(th3)*sin(b1)*sin(th1)*sin(th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*cos(th3)*sin(b3)*sin(th1)*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(m1*cos(b1)*cos(th1)^2*cos(th3)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*cos(b3)*cos(th1)^2*cos(th3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m1*cos(b1)*cos(th3)*sin(th1)^2*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m3*cos(b3)*cos(th3)*sin(th1)^2*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m1*cos(th1)*cos(th3)*sin(b1)*sin(th1)*sin(th2)^2*sin(th3))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*cos(th3)*sin(b3)*sin(th1)*sin(th2)^2*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
m2*cos(b2)^2*cos(th1)*sin(th1)*sin(th3)^2 - m4*cos(b4)^2*cos(th1)*sin(th1)*sin(th3)^2 -
(m1*cos(b1)*cos(th1)*sin(th1)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
m2*cos(th1)*sin(b2)^2*sin(th1)*sin(th3)^2 - (m3*cos(b3)*cos(th1)*sin(th1)*sin(th3)^2)/(cos(b3) -
muk*sign(b3d)*sin(b3)) - m4*cos(th1)*sin(b4)^2*sin(th1)*sin(th3)^2 +
(m1*cos(th1)^2*sin(b1)*sin(th2)*sin(th3)^2)/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)^2*sin(b3)*sin(th2)*sin(th3)^2)/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
m2*cos(b2)^2*cos(th1)*sin(th1)*sin(th2)^2*sin(th3)^2 +
m4*cos(b4)^2*cos(th1)*sin(th1)*sin(th2)^2*sin(th3)^2 +
m2*cos(th1)*sin(b2)^2*sin(th1)*sin(th2)^2*sin(th3)^2 +
m4*cos(th1)*sin(b4)^2*sin(th1)*sin(th2)^2*sin(th3)^2 +
(m3*muk*sign(b3d)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3))*(cos(b3)*cos(th3)*sin(th1) +
cos(th1)*cos(th3)*sin(b3)*sin(th2) - sin(b3)*sin(th1)*sin(th3) +
cos(b3)*cos(th1)*sin(th2)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(m1*muk*sign(b1d)*(cos(th3)*sin(th1)*sin(th2) +
```

```
cos(th1)*sin(th3))*(sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
cos(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))))/(-cos(b1) + muk*sign(b1d)*sin(b1));
MM13(3,3) = -m - M - m1*cos(th1)^2*cos(th2)^2 - m2*cos(th1)^2*cos(th2)^2 -
m3*cos(th1)^2*cos(th2)^2 - m4*cos(th1)^2*cos(th2)^2 - m2*cos(b2)*cos(th3)*cos(b2 +
th3)*sin(th1)^2 - m4*cos(b4)*cos(th3)*cos(b4 + th3)*sin(th1)^2 +
(m1*cos(th1)*cos(th3)^2*sin(b1)*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*cos(th1)*cos(th3)^2*sin(b3)*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
m2*cos(b2)*cos(th1)^2*cos(th3)*cos(b2 + th3)*sin(th2)^2 - m4*cos(b4)*cos(th1)^2*cos(th3)*cos(b4 +
th3)*sin(th2)^2 + (m1*cos(b1)*cos(th1)^2*cos(th3)^2*sin(th2)^2)/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (m3*cos(b3)*cos(th1)^2*cos(th3)^2*sin(th2)^2)/(-cos(b3) +
muk*sign(b3d)*sin(b3)) + (m1*cos(th3)*sin(b1)*sin(th1)^2*sin(th3))/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + m2*cos(b2 + th3)*sin(b2)*sin(th1)^2*sin(th3) +
(m3*cos(th3)*sin(b3)*sin(th1)^2*sin(th3))/(-cos(b3) + muk*sign(b3d)*sin(b3)) + m4*cos(b4 +
th3)*sin(b4)*sin(th1)^2*sin(th3) +
(2*m1*cos(b1)*cos(th1)*cos(th3)*sin(th1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*m3*cos(b3)*cos(th1)*cos(th3)*sin(th1)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m1*cos(th1)^2*cos(th3)*sin(b1)*sin(th2)^2*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*cos(th1)^2*cos(b2 + th3)*sin(b2)*sin(th2)^2*sin(th3) +
(m3*cos(th1)^2*cos(th3)*sin(b3)*sin(th2)^2*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*cos(th1)^2*cos(b4 + th3)*sin(b4)*sin(th2)^2*sin(th3) + (m1*cos(b1)*sin(th1)^2*sin(th3)^2)/(-
cos(b1) + muk*sign(b1d)*sin(b1)) + (m3*cos(b3)*sin(th1)^2*sin(th3)^2)/(-cos(b3) +
muk*sign(b3d)*sin(b3)) + (m1*cos(th1)*sin(b1)*sin(th1)*sin(th2)*sin(th3)^2)/(-cos(b1) +
muk*sign(b1d)*sin(b1)) + (m3*cos(th1)*sin(b3)*sin(th1)*sin(th2)*sin(th3)^2)/(-cos(b3) +
muk*sign(b3d)*sin(b3)) - (m3*muk*sign(b3d)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3))*(cos(b3)*cos(th3)*sin(th1) + cos(th1)*cos(th3)*sin(b3)*sin(th2) -
sin(b3)*sin(th1)*sin(th3) + cos(b3)*cos(th1)*sin(th2)*sin(th3)))/(-cos(b3) +
muk*sign(b3d)*sin(b3)) - (m1*muk*sign(b1d)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3))*(sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
cos(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))))/(-cos(b1) + muk*sign(b1d)*sin(b1)) -
m2*cos(th3)*sin(b2)*sin(th1)^2*sin(b2 + th3) - m2*cos(th1)^2*cos(th3)*sin(b2)*sin(th2)^2*sin(b2 +
th3) - m2*cos(b2)*sin(th1)^2*sin(th3)*sin(b2 + th3) -
m2*cos(b2)*cos(th1)^2*sin(th2)^2*sin(th3)*sin(b2 + th3) - m4*cos(th3)*sin(b4)*sin(th1)^2*sin(b4 +
th3) - m4*cos(th1)^2*cos(th3)*sin(b4)*sin(th2)^2*sin(b4 + th3) -
m4*cos(b4)*sin(th1)^2*sin(th3)*sin(b4 + th3) - m4*cos(b4)*cos(th1)^2*sin(th2)^2*sin(th3)*sin(b4 +
th3);
MM13(3,4) = -(m1*r1*cos(th1)*cos(th2)^2*cos(th3)*sin(b1)) -
m2*r2*cos(th1)*cos(th2)^2*cos(th3)*sin(b2) - m3*r3*cos(th1)*cos(th2)^2*cos(th3)*sin(b3) -
m4*r4*cos(th1)*cos(th2)^2*cos(th3)*sin(b4) - m*r*cos(th1)*cos(th3)*sin(bm) +
dm*m*cos(th2)*sin(th1) + d2*m2*cos(b2)^2*cos(th2)*cos(th3)^2*sin(th1) +
d4*m4*cos(b4)^2*cos(th2)*cos(th3)^2*sin(th1) + d2*m2*cos(th2)*cos(th3)^2*sin(b2)^2*sin(th1) +
d4*m4*cos(th2)*cos(th3)^2*sin(b4)^2*sin(th1) - m2*r2*cos(b2)*cos(th3)*sin(th1)*sin(th2) -
m4*r4*cos(b4)*cos(th3)*sin(th1)*sin(th2) - m*r*cos(bm)*cos(th3)*sin(th1)*sin(th2) +
(m1*r1*cos(th3)*sin(b1)^2*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*r3*cos(th3)*sin(b3)^2*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m1*r1*cos(b1)*cos(th1)*cos(th3)*sin(b1)*sin(th2)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
m2*r2*cos(th1)*cos(th3)*sin(b2)*sin(th2)^2 -
(m3*r3*cos(b3)*cos(th1)*cos(th3)*sin(b3)*sin(th2)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) -
m4*r4*cos(th1)*cos(th3)*sin(b4)*sin(th2)^2 - m*r*cos(bm)*cos(th1)*sin(th3) -
m1*r1*cos(b1)*cos(th1)*cos(th2)^2*sin(th3) - m2*r2*cos(b2)*cos(th1)*cos(th2)^2*sin(th3) -
m3*r3*cos(b3)*cos(th1)*cos(th2)^2*sin(th3) - m4*r4*cos(b4)*cos(th1)*cos(th2)^2*sin(th3) +
(d1*m1*cos(th2)*cos(th3)*sin(b1)*sin(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(th2)*cos(th3)*sin(b3)*sin(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d1*m1*cos(b1)*cos(th1)*cos(th2)*cos(th3)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*cos(b3)*cos(th1)*cos(th2)*cos(th3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m1*r1*cos(b1)*sin(b1)*sin(th1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
m2*r2*sin(b2)*sin(th1)*sin(th2)*sin(th3) +
(m3*r3*cos(b3)*sin(b3)*sin(th1)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m4*r4*sin(b4)*sin(th1)*sin(th2)*sin(th3) + m*r*sin(bm)*sin(th1)*sin(th2)*sin(th3) -
m2*r2*cos(b2)*cos(th1)*sin(th2)^2*sin(th3) - m4*r4*cos(b4)*cos(th1)*sin(th2)^2*sin(th3) +
(m1*r1*cos(th1)*sin(b1)^2*sin(th2)^2*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*r3*cos(th1)*sin(b3)^2*sin(th2)^2*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
d2*m2*cos(b2)^2*cos(th2)*sin(th1)*sin(th3)^2 + d4*m4*cos(b4)^2*cos(th2)*sin(th1)*sin(th3)^2 +
(d1*m1*cos(b1)*cos(th2)*sin(th1)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
d2*m2*cos(th2)*sin(b2)^2*sin(th1)*sin(th3)^2 +
(d3*m3*cos(b3)*cos(th2)*sin(th1)*sin(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) +
d4*m4*cos(th2)*sin(b4)^2*sin(th1)*sin(th3)^2 +
(d1*m1*cos(th1)*cos(th2)*sin(b1)*sin(th2)*sin(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(th1)*cos(th2)*sin(b3)*sin(th2)*sin(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*sign(b3d)*(r3*sin(b3)*sin(th2) + d3*cos(th2)*sin(th3))*(cos(b3)*cos(th3)*sin(th1) +
cos(th1)*cos(th3)*sin(b3)*sin(th2) - sin(b3)*sin(th1)*sin(th3) +
cos(b3)*cos(th1)*sin(th2)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) -
```

```
(m1*muk*sign(b1d)*(r1*sin(b1)*sin(th2) +
d1*cos(th2)*sin(th3))*(sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
cos(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))))/(-cos(b1) + muk*sign(b1d)*sin(b1));
MM13(3,5) = m1*r1*cos(th1)*cos(th2)*cos(b1 + th3) + m2*r2*cos(th1)*cos(th2)*cos(b2 + th3) +
m3*r3*cos(th1)*cos(th2)*cos(b3 + th3) + m4*r4*cos(th1)*cos(th2)*cos(b4 + th3) +
m*r*cos(th1)*cos(th2)*cos(bm + th3) + dm*m*cos(th1)*sin(th2) + d2*m2*cos(b2)*cos(b2 +
th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) + d4*m4*cos(b4)*cos(b4 +
th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
(d1*m1*cos(b1)*cos(th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*muk*cos(th3)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(-cos(b1) + muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(b3)*cos(th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d3*m3*muk*cos(th3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) +
(d1*m1*muk*cos(b1)*cos(th3)*sign(b1d)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(-cos(b1)
+ muk*sign(b1d)*sin(b1)) + (d1*m1*cos(th3)*sin(b1)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(-cos(b1) + muk*sign(b1d)*sin(b1)) - d2*m2*cos(b2 +
th3)*sin(b2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
(d3*m3*muk*cos(b3)*cos(th3)*sign(b3d)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(-cos(b3)
+ muk*sign(b3d)*sin(b3)) + (d3*m3*cos(th3)*sin(b3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(-cos(b3) + muk*sign(b3d)*sin(b3)) - d4*m4*cos(b4 +
th3)*sin(b4)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
d2*m2*sin(b2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b2 + th3) +
d2*m2*cos(b2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b2 + th3) +
d4*m4*sin(b4)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b4 + th3) +
d4*m4*cos(b4)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b4 + th3);
MM13(3,6) = -(m*r*cos(bm + th3)*sin(th1)) - (m1*r1*cos(b1)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*sign(b1d)*sin(b1)^2*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - m2*r2*sin(b2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) -
(m3*r3*cos(b3)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*muk*r3*sign(b3d)*sin(b3)^2*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - m4*r4*sin(b4)*(cos(th1)*cos(th3)*sin(th2)
- sin(th1)*sin(th3)) - m2*r2*cos(b2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) -
m4*r4*cos(b4)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
(m1*muk*r1*cos(b1)*sign(b1d)*sin(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*r1*sin(b1)^2*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*muk*r3*cos(b3)*sign(b3d)*sin(b3)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*r3*sin(b3)^2*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - m*r*cos(th1)*sin(th2)*sin(bm +
th3);
MM13(3,7) = (m1*r1*sin(b1)*(cos(th3)*(sin(b1)*sin(th1) - cos(b1)*cos(th1)*sin(th2)) +
(cos(b1)*sin(th1) + cos(th1)*sin(b1)*sin(th2))*sin(th3) +
muk*sign(b1d)*(sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
cos(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))))/(cos(b1) - muk*sign(b1d)*sin(b1));
MM13(3,8) = (m3*r3*sin(b3)*(cos(th3)*(sin(b3)*sin(th1) - cos(b3)*cos(th1)*sin(th2)) +
(cos(b3)*sin(th1) + cos(th1)*sin(b3)*sin(th2))*sin(th3) +
muk*sign(b3d)*(sin(b3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
cos(b3)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))))/(cos(b3) - muk*sign(b3d)*sin(b3));

MM13(4,1) = (d1*m1*muk*cos(b1)*cos(th2)*cos(th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*cos(th2)*cos(th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) + d2*m2*cos(th2)*cos(b2 +
th3)*sin(b2) + (d3*m3*muk*cos(b3)*cos(th2)*cos(th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3))
+ (d3*m3*cos(th2)*cos(th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) + d4*m4*cos(th2)*cos(b4 +
th3)*sin(b4) - m1*r1*sin(b1)*sin(th2) - m2*r2*sin(b2)*sin(th2) - m3*r3*sin(b3)*sin(th2) -
m4*r4*sin(b4)*sin(th2) - m*r*sin(bm)*sin(th2) - dm*m*cos(th2)*sin(th3) -
d2*m2*cos(b2)*cos(th2)*sin(b2 + th3) - d4*m4*cos(b4)*cos(th2)*sin(b4 + th3);
MM13(4,2) = dm*m*cos(th1)*cos(th3) + m1*r1*cos(th2)*sin(b1)*sin(th1) +
m2*r2*cos(th2)*sin(b2)*sin(th1) + m3*r3*cos(th2)*sin(b3)*sin(th1) +
m4*r4*cos(th2)*sin(b4)*sin(th1) + m*r*cos(th2)*sin(bm)*sin(th1) +
(d1*m1*muk*cos(b1)*cos(th3)*sign(b1d)*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*cos(th3)*sin(b1)*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) + d2*m2*cos(b2 +
th3)*sin(b2)*sin(th1)*sin(th2) +
(d3*m3*muk*cos(b3)*cos(th3)*sign(b3d)*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*cos(th3)*sin(b3)*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) + d4*m4*cos(b4 +
th3)*sin(b4)*sin(th1)*sin(th2) + (d1*m1*muk*cos(b1)*cos(th1)*sign(b1d)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*cos(th1)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*muk*cos(b3)*cos(th1)*sign(b3d)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*cos(th1)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
dm*m*sin(th1)*sin(th2)*sin(th3) + d2*m2*cos(th1)*sin(b2)*sin(b2 + th3) - (d2*m2*cos(b2)*(-
```

```
2*cos(th1)*cos(b2 + th3) + 2*sin(th1)*sin(th2)*sin(b2 + th3)))/2 + d4*m4*cos(th1)*sin(b4)*sin(b4
+ th3) - (d4*m4*cos(b4)*(-2*cos(th1)*cos(b4 + th3) + 2*sin(th1)*sin(th2)*sin(b4 + th3)))/2;
MM13(4,3) = -(m1*r1*cos(th1)*cos(th2)*sin(b1)) - m2*r2*cos(th1)*cos(th2)*sin(b2) -
m3*r3*cos(th1)*cos(th2)*sin(b3) - m4*r4*cos(th1)*cos(th2)*sin(b4) - m*r*cos(th1)*cos(th2)*sin(bm)
+ dm*m*cos(th3)*sin(th1) - (d1*m1*muk*cos(b1)*cos(th1)*cos(th3)*sign(b1d)*sin(th2))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (d1*m1*cos(th1)*cos(th3)*sin(b1)*sin(th2))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - d2*m2*cos(th1)*cos(b2 + th3)*sin(b2)*sin(th2) -
(d3*m3*muk*cos(b3)*cos(th1)*cos(th3)*sign(b3d)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*cos(th1)*cos(th3)*sin(b3)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
d4*m4*cos(th1)*cos(b4 + th3)*sin(b4)*sin(th2) +
(d1*m1*muk*cos(b1)*sign(b1d)*sin(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*sin(b1)*sin(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*muk*cos(b3)*sign(b3d)*sin(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*sin(b3)*sin(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
dm*m*cos(th1)*sin(th2)*sin(th3) + d2*m2*sin(b2)*sin(th1)*sin(b2 + th3) + d2*m2*cos(b2)*(cos(b2 +
th3)*sin(th1) + cos(th1)*sin(th2)*sin(b2 + th3)) + d4*m4*sin(b4)*sin(th1)*sin(b4 + th3) +
d4*m4*cos(b4)*(cos(b4 + th3)*sin(th1) + cos(th1)*sin(th2)*sin(b4 + th3));
MM13(4,4) = -(d2^2*m2*cos(b2)*cos(th2)*cos(b2 + th3)) - d4^2*m4*cos(b4)*cos(th2)*cos(b4 + th3) +
Ixz*sin(th2) + d2*m2*r2*cos(b2)*sin(th2) + d4*m4*r4*cos(b4)*sin(th2) -
(d1*m1*muk*r1*cos(b1)*sign(b1d)*sin(b1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*r1*sin(b1)^2*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*muk*r3*cos(b3)*sign(b3d)*sin(b3)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*r3*sin(b3)^2*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d1^2*m1*muk*cos(b1)*cos(th2)*sign(b1d)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1^2*m1*cos(th2)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3^2*m3*muk*cos(b3)*cos(th2)*sign(b3d)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3^2*m3*cos(th2)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) - cos(th2)*(Ixx*cos(th3) +
Ixy*sin(th3)) - m1*r1^2*cos(th2)*sin(b1)*sin(b1 + th3) - d2^2*m2*cos(th2)*sin(b2)*sin(b2 + th3) -
m2*r2^2*cos(th2)*sin(b2)*sin(b2 + th3) - m3*r3^2*cos(th2)*sin(b3)*sin(b3 + th3) -
d4^2*m4*cos(th2)*sin(b4)*sin(b4 + th3) - m4*r4^2*cos(th2)*sin(b4)*sin(b4 + th3);
MM13(4,5) = Ixy*cos(th3) + m1*r1^2*cos(b1 + th3)*sin(b1) +
(d1^2*m1*muk*cos(b1)*cos(th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1^2*m1*cos(th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) + d2^2*m2*cos(b2 + th3)*sin(b2) +
m2*r2^2*cos(b2 + th3)*sin(b2) + m3*r3^2*cos(b3 + th3)*sin(b3) +
(d3^2*m3*muk*cos(b3)*cos(th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3^2*m3*cos(th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) + d4^2*m4*cos(b4 + th3)*sin(b4) +
m4*r4^2*cos(b4 + th3)*sin(b4) - Ixx*sin(th3) - d2^2*m2*cos(b2)*sin(b2 + th3) -
d4^2*m4*cos(b4)*sin(b4 + th3);
MM13(4,6) = Ixz + d2*m2*r2*cos(b2) + d4*m4*r4*cos(b4) -
(d1*m1*muk*r1*cos(b1)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*r1*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*muk*r3*cos(b3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*r3*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3));
MM13(4,7) = -((d1*m1*muk*r1*cos(b1)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1))) -
(d1*m1*r1*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1));
MM13(4,8) = -((d3*m3*muk*r3*cos(b3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3))) -
(d3*m3*r3*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3));

MM13(5,1) = -(dm*m*cos(th2)*cos(th3)) - d2*m2*cos(b2)*cos(th2)*cos(b2 + th3) -
d4*m4*cos(b4)*cos(th2)*cos(b4 + th3) - (d1*m1*cos(b1)*cos(th2)*cos(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*muk*cos(th2)*cos(th3)*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (d3*m3*cos(b3)*cos(th2)*cos(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*muk*cos(th2)*cos(th3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m1*r1*cos(b1)*sin(th2) + m2*r2*cos(b2)*sin(th2) + m3*r3*cos(b3)*sin(th2) + m4*r4*cos(b4)*sin(th2)
+ m*r*cos(bm)*sin(th2) - d2*m2*cos(th2)*sin(b2)*sin(b2 + th3) - d4*m4*cos(th2)*sin(b4)*sin(b4 +
th3);
MM13(5,2) = -(m1*r1*cos(b1)*cos(th2)*sin(th1)) - m2*r2*cos(b2)*cos(th2)*sin(th1) -
m3*r3*cos(b3)*cos(th2)*sin(th1) - m4*r4*cos(b4)*cos(th2)*sin(th1) - m*r*cos(bm)*cos(th2)*sin(th1)
- dm*m*cos(th3)*sin(th1)*sin(th2) - d2*m2*cos(b2)*cos(b2 + th3)*sin(th1)*sin(th2) -
d4*m4*cos(b4)*cos(b4 + th3)*sin(th1)*sin(th2) -
(d1*m1*cos(b1)*cos(th3)*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*muk*cos(th3)*sign(b1d)*sin(b1)*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*cos(b3)*cos(th3)*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*muk*cos(th3)*sign(b3d)*sin(b3)*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
dm*m*cos(th1)*sin(th3) - (d1*m1*cos(b1)*cos(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*muk*cos(th1)*sign(b1d)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*cos(b3)*cos(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*muk*cos(th1)*sign(b3d)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
d2*m2*cos(b2)*cos(th1)*sin(b2 + th3) - (d2*m2*sin(b2)*(-2*cos(th1)*cos(b2 + th3) +
2*sin(th1)*sin(th2)*sin(b2 + th3)))/2 - d4*m4*cos(b4)*cos(th1)*sin(b4 + th3) - (d4*m4*sin(b4)*(-
2*cos(th1)*cos(b4 + th3) + 2*sin(th1)*sin(th2)*sin(b4 + th3)))/2;
```

```
MM13(5,3) = m1*r1*cos(b1)*cos(th1)*cos(th2) + m2*r2*cos(b2)*cos(th1)*cos(th2) +
m3*r3*cos(b3)*cos(th1)*cos(th2) + m4*r4*cos(b4)*cos(th1)*cos(th2) + d2*m2*cos(b2)*cos(th1)*cos(b2
+ th3)*sin(th2) + d4*m4*cos(b4)*cos(th1)*cos(b4 + th3)*sin(th2) +
(d1*m1*cos(b1)*cos(th1)*cos(th3)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*muk*cos(th1)*cos(th3)*sign(b1d)*sin(b1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d3*m3*cos(b3)*cos(th1)*cos(th3)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*muk*cos(th1)*cos(th3)*sign(b3d)*sin(b3)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d1*m1*cos(b1)*sin(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*muk*sign(b1d)*sin(b1)*sin(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*cos(b3)*sin(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*muk*sign(b3d)*sin(b3)*sin(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m*(r*cos(bm)*cos(th1)*cos(th2) + dm*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))) -
d2*m2*cos(b2)*sin(th1)*sin(b2 + th3) + d2*m2*sin(b2)*(cos(b2 + th3)*sin(th1) +
cos(th1)*sin(th2)*sin(b2 + th3)) - d4*m4*cos(b4)*sin(th1)*sin(b4 + th3) + d4*m4*sin(b4)*(cos(b4 +
th3)*sin(th1) + cos(th1)*sin(th2)*sin(b4 + th3));
MM13(5,4) = Iyx*cos(th2)*cos(th3) - d2^2*m2*cos(th2)*cos(b2 + th3)*sin(b2) -
d4^2*m4*cos(th2)*cos(b4 + th3)*sin(b4) + Iyz*sin(th2) +
(d1*m1*r1*cos(b1)*sin(b1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*muk*r1*sign(b1d)*sin(b1)^2*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
d2*m2*r2*sin(b2)*sin(th2) + (d3*m3*r3*cos(b3)*sin(b3)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3))
- (d3*m3*muk*r3*sign(b3d)*sin(b3)^2*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
d4*m4*r4*sin(b4)*sin(th2) + Iyy*cos(th2)*sin(th3) + (d1^2*m1*cos(b1)*cos(th2)*sin(th3))/(cos(b1)
- muk*sign(b1d)*sin(b1)) - (d1^2*m1*muk*cos(th2)*sign(b1d)*sin(b1)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d3^2*m3*cos(b3)*cos(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3^2*m3*muk*cos(th2)*sign(b3d)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
m1*r1^2*cos(b1)*cos(th2)*sin(b1 + th3) + d2^2*m2*cos(b2)*cos(th2)*sin(b2 + th3) +
m2*r2^2*cos(b2)*cos(th2)*sin(b2 + th3) + m3*r3^2*cos(b3)*cos(th2)*sin(b3 + th3) +
d4^2*m4*cos(b4)*cos(th2)*sin(b4 + th3) + m4*r4^2*cos(b4)*cos(th2)*sin(b4 + th3);
MM13(5,5) = -(Iyy*cos(th3)) - m1*r1^2*cos(b1)*cos(b1 + th3) - d2^2*m2*cos(b2)*cos(b2 + th3) -
m2*r2^2*cos(b2)*cos(b2 + th3) - m3*r3^2*cos(b3)*cos(b3 + th3) - d4^2*m4*cos(b4)*cos(b4 + th3) -
m4*r4^2*cos(b4)*cos(b4 + th3) - (d1^2*m1*cos(b1)*cos(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1^2*m1*muk*cos(th3)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3^2*m3*cos(b3)*cos(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3^2*m3*muk*cos(th3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) + Iyx*sin(th3) -
d2^2*m2*sin(b2)*sin(b2 + th3) - d4^2*m4*sin(b4)*sin(b4 + th3);
MM13(5,6) = Iyz + (d1*m1*r1*cos(b1)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*muk*r1*sign(b1d)*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) + d2*m2*r2*sin(b2) +
(d3*m3*r3*cos(b3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*muk*r3*sign(b3d)*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) + d4*m4*r4*sin(b4);
MM13(5,7) = (d1*m1*r1*cos(b1)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*muk*r1*sign(b1d)*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1));
MM13(5,8) = (d3*m3*r3*cos(b3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*muk*r3*sign(b3d)*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3));

MM13(6,1) = -((m1*muk*r1*cos(th2)*cos(th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1))) -
(m3*muk*r3*cos(th2)*cos(th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) + m2*r2*cos(th2)*sin(b2
+ th3) + m4*r4*cos(th2)*sin(b4 + th3) + m*r*cos(th2)*sin(bm + th3);
MM13(6,2) = -(m*r*cos(th1)*cos(bm + th3)) -
(m1*muk*r1*cos(th3)*sign(b1d)*sin(th1)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m3*muk*r3*cos(th3)*sign(b3d)*sin(th1)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m1*muk*r1*cos(th1)*sign(b1d)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m3*muk*r3*cos(th1)*sign(b3d)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (m2*r2*(-
2*cos(th1)*cos(b2 + th3) + 2*sin(th1)*sin(th2)*sin(b2 + th3)))/2 + (m4*r4*(-2*cos(th1)*cos(b4 +
th3) + 2*sin(th1)*sin(th2)*sin(b4 + th3)))/2 + m*r*sin(th1)*sin(th2)*sin(bm + th3);
MM13(6,3) = -(m*r*cos(bm + th3)*sin(th1)) +
(m1*muk*r1*cos(th1)*cos(th3)*sign(b1d)*sin(th2))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m3*muk*r3*cos(th1)*cos(th3)*sign(b3d)*sin(th2))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m1*muk*r1*sign(b1d)*sin(th1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m3*muk*r3*sign(b3d)*sin(th1)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) + m2*r2*(-(cos(b2 +
th3)*sin(th1)) - cos(th1)*sin(th2)*sin(b2 + th3)) + m4*r4*(-(cos(b4 + th3)*sin(th1)) -
cos(th1)*sin(th2)*sin(b4 + th3)) - m*r*cos(th1)*sin(th2)*sin(bm + th3);
MM13(6,4) = d2*m2*r2*cos(th2)*cos(b2 + th3) + d4*m4*r4*cos(th2)*cos(b4 + th3) - Izz*sin(th2) -
m2*r2^2*sin(th2) - m4*r4^2*sin(th2) + (m1*muk*r1^2*sign(b1d)*sin(b1)*sin(th2))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m3*muk*r3^2*sign(b3d)*sin(b3)*sin(th2))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d1*m1*muk*r1*cos(th2)*sign(b1d)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d3*m3*muk*r3*cos(th2)*sign(b3d)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + cos(th2)*(Izx*cos(th3) - Izy*sin(th3));
MM13(6,5) = Izy*cos(th3) - (d1*m1*muk*r1*cos(th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d3*m3*muk*r3*cos(th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) + Izx*sin(th3) +
d2*m2*r2*sin(b2 + th3) + d4*m4*r4*sin(b4 + th3);
```

```
MM13(6,6) = -Izz - m2*r2^2 - m4*r4^2 + (m1*muk*r1^2*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m3*muk*r3^2*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3));
MM13(6,7) = (m1*muk*r1^2*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1));
MM13(6,8) = (m3*muk*r3^2*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3));

MM13(7,1) = (m1*cos(th2)*(muk*cos(b1 + th3)*sign(b1d) + sin(b1 + th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1));
MM13(7,2) = (m1*(sin(b1)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) + cos(b1)*(-
(cos(th1)*cos(th3)) + sin(th1)*sin(th2)*sin(th3)) + muk*sign(b1d)*(cos(b1 +
th3)*sin(th1)*sin(th2) + cos(th1)*sin(b1 + th3))))/(cos(b1) - muk*sign(b1d)*sin(b1));
MM13(7,3) = (m1*(sin(b1)*(-(cos(th1)*cos(th3)*sin(th2)) + sin(th1)*sin(th3)) -
cos(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
muk*sign(b1d)*(cos(th3)*(sin(b1)*sin(th1) - cos(b1)*cos(th1)*sin(th2)) + (cos(b1)*sin(th1) +
cos(th1)*sin(b1)*sin(th2))*sin(th3))))/(cos(b1) - muk*sign(b1d)*sin(b1));
MM13(7,4) = (m1*(d1*cos(b1 - th2 + th3) + d1*cos(b1 + th2 + th3) - 2*r1*sin(th2) -
2*d1*muk*cos(th2)*sign(b1d)*sin(b1 + th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1)));
MM13(7,5) = (d1*m1*(muk*cos(b1 + th3)*sign(b1d) + sin(b1 + th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1));
MM13(7,6) = (m1*r1)/(-cos(b1) + muk*sign(b1d)*sin(b1));
MM13(7,7) = (m1*r1)/(-cos(b1) + muk*sign(b1d)*sin(b1));
MM13(7,8) = 0;

MM13(8,1) = (m3*cos(th2)*(muk*cos(b3 + th3)*sign(b3d) + sin(b3 + th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3));
MM13(8,2) = (m3*(sin(b3)*(cos(th3)*sin(th1)*sin(th2) + cos(th1)*sin(th3)) + cos(b3)*(-
(cos(th1)*cos(th3)) + sin(th1)*sin(th2)*sin(th3)) + muk*sign(b3d)*(cos(b3 +
th3)*sin(th1)*sin(th2) + cos(th1)*sin(b3 + th3))))/(cos(b3) - muk*sign(b3d)*sin(b3));
MM13(8,3) = (m3*(sin(b3)*(-(cos(th1)*cos(th3)*sin(th2)) + sin(th1)*sin(th3)) -
cos(b3)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
muk*sign(b3d)*(cos(th3)*(sin(b3)*sin(th1) - cos(b3)*cos(th1)*sin(th2)) + (cos(b3)*sin(th1) +
cos(th1)*sin(b3)*sin(th2))*sin(th3))))/(cos(b3) - muk*sign(b3d)*sin(b3));
MM13(8,4) = (m3*(d3*cos(b3 - th2 + th3) + d3*cos(b3 + th2 + th3) - 2*r3*sin(th2) -
2*d3*muk*cos(th2)*sign(b3d)*sin(b3 + th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3)));
MM13(8,5) = (d3*m3*(muk*cos(b3 + th3)*sign(b3d) + sin(b3 + th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3));
MM13(8,6) = (m3*r3)/(-cos(b3) + muk*sign(b3d)*sin(b3));
MM13(8,7) = 0;
MM13(8,8) = (m3*r3)/(-cos(b3) + muk*sign(b3d)*sin(b3));

%Populate the Various Elements of the Force Vector f(t,y)
f13(1) = kx*x + cx*xd + g*m2*cos(b2)*cos(th2)^2*cos(th3)*cos(b2 + th3) +
g*m4*cos(b4)*cos(th2)^2*cos(th3)*cos(b4 + th3) - m*r*th3d^2*cos(th2)*cos(bm + th3) -
(b1d^2*m1*r1*cos(b1)^2*cos(th2)*cos(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*r1*th3d*(2*b1d
+ th3d)*cos(b1)^2*cos(th2)*cos(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(g*m1*cos(b1)*cos(th2)^2*cos(th3)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*r1*th2d^2*cos(b1)*cos(th2)*cos(th3)^2*cos(b1 + th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*r1*th1d^2*cos(b1)*cos(th2)*cos(th3)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1
+ 2*th3)))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(b1d*muc*r1*cos(b1)*cos(th2)*cos(th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(b1d^2*m1*muk*r1*cos(b1)*cos(th2)*cos(th3)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*th3d*(2*b1d + th3d)*cos(b1)*cos(th2)*cos(th3)*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (g*m1*muk*cos(th2)^2*cos(th3)^2*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*muk*r1*th2d^2*cos(th2)*cos(th3)^2*cos(b1 +
th3)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*th1d^2*cos(th2)*cos(th3)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sign(b1d)*sin(b1))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d*muc*muk*r1*cos(th2)*cos(th3)*sign(b1d)*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
2*m2*r2*th1d*th2d*cos(th2)^2*cos(th3)*cos(b2 + th3)^2*sin(b2) -
(b3d^2*m3*r3*cos(b3)^2*cos(th2)*cos(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*r3*th3d*(2*b3d
+ th3d)*cos(b3)^2*cos(th2)*cos(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(g*m3*cos(b3)*cos(th2)^2*cos(th3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*r3*th2d^2*cos(b3)*cos(th2)*cos(th3)^2*cos(b3 + th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*r3*th1d^2*cos(b3)*cos(th2)*cos(th3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3
+ 2*th3)))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(b3d*muc*r3*cos(b3)*cos(th2)*cos(th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(b3d^2*m3*muk*r3*cos(b3)*cos(th2)*cos(th3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*muk*r3*th3d*(2*b3d + th3d)*cos(b3)*cos(th2)*cos(th3)*sign(b3d)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (g*m3*muk*cos(th2)^2*cos(th3)^2*sign(b3d)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*muk*r3*th2d^2*cos(th2)*cos(th3)^2*cos(b3 +
th3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*muk*r3*th1d^2*cos(th2)*cos(th3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
```

```
2*th3))*sign(b3d)*sin(b3))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d*muc*muk*r3*cos(th2)*cos(th3)*sign(b3d)*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) -
2*m4*r4*th1d*th2d*cos(th2)^2*cos(th3)*cos(b4 + th3)^2*sin(b4) +
muc*cos(th2)*cos(th3)*(b1d*r1*sin(b1) + b2d*r2*sin(b2) + b3d*r3*sin(b3) + b4d*r4*sin(b4)) -
d1*m1*(th2d^2 + th1d^2*cos(th2)^2)*sin(th2) - d2*m2*(th2d^2 + th1d^2*cos(th2)^2)*sin(th2) -
d3*m3*(th2d^2 + th1d^2*cos(th2)^2)*sin(th2) - d4*m4*(th2d^2 + th1d^2*cos(th2)^2)*sin(th2) -
2*d2*m2*th1d*th2d*cos(th2)*cos(th3)*cos(b2 + th3)*sin(b2)*sin(th2) -
2*d4*m4*th1d*th2d*cos(th2)*cos(th3)*cos(b4 + th3)*sin(b4)*sin(th2) + g*m1*sin(th2)^2 +
g*m2*sin(th2)^2 + g*m3*sin(th2)^2 + g*m4*sin(th2)^2 - m*th2d^2*(r*cos(th2)*cos(bm + th3) +
dm*sin(th2)) + m2*r2*th1d*cos(th2)*cos(b2 + th3)*sin(th2)*(2*th3d + th1d*sin(th2)) +
m4*r4*th1d*cos(th2)*cos(b4 + th3)*sin(th2)*(2*th3d + th1d*sin(th2)) + m1*r1*th1d*cos(th2)*cos(b1
+ th3)*sin(th2)*(2*(b1d + th3d) + th1d*sin(th2)) + m3*r3*th1d*cos(th2)*cos(b3 +
th3)*sin(th2)*(2*(b3d + th3d) + th1d*sin(th2)) + (d2*m2*th1d^2*cos(b2)*cos(th2)*cos(th3)*cos(b2 +
th3)*sin(2*th2))/2 + (d4*m4*th1d^2*cos(b4)*cos(th2)*cos(th3)*cos(b4 + th3)*sin(2*th2))/2 +
(d1*m1*th1d^2*cos(b1)*cos(th2)*cos(th3)^2*sin(2*th2))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(d1*m1*muk*th1d^2*cos(th2)*cos(th3)^2*sign(b1d)*sin(b1)*sin(2*th2))/(2*(cos(b1) -
muk*sign(b1d)*sin(b1))) + (d3*m3*th1d^2*cos(b3)*cos(th2)*cos(th3)^2*sin(2*th2))/(2*(cos(b3) -
muk*sign(b3d)*sin(b3))) -
(d3*m3*muk*th1d^2*cos(th2)*cos(th3)^2*sign(b3d)*sin(b3)*sin(2*th2))/(2*(cos(b3) -
muk*sign(b3d)*sin(b3))) + muc*(b1d*r1*cos(b1) + b2d*r2*cos(b2) + b3d*r3*cos(b3) +
b4d*r4*cos(b4))*cos(th2)*sin(th3) - 2*m2*r2*th1d*th2d*cos(b2)*cos(th2)^2*cos(b2 + th3)^2*sin(th3)
- 2*m4*r4*th1d*th2d*cos(b4)*cos(th2)^2*cos(b4 + th3)^2*sin(th3) +
(b1d^2*m1*muk*r1*cos(b1)^2*cos(th2)*sign(b1d)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*th3d*(2*b1d + th3d)*cos(b1)^2*cos(th2)*sign(b1d)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (g*m1*muk*cos(b1)*cos(th2)^2*cos(th3)*sign(b1d)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*muk*r1*th2d^2*cos(b1)*cos(th2)*cos(th3)*cos(b1 +
th3)*sign(b1d)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*th1d^2*cos(b1)*cos(th2)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sign(b1d)*sin(th3))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d^2*m1*r1*cos(b1)*cos(th2)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*r1*th3d*(2*b1d + th3d)*cos(b1)*cos(th2)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(g*m1*cos(th2)^2*cos(th3)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*r1*th2d^2*cos(th2)*cos(th3)*cos(b1 + th3)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1))
+ (m1*r1*th1d^2*cos(th2)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sin(b1)*sin(th3))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d*muc*muk*r1*cos(b1)*cos(th2)*sign(b1d)*sin(b1)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(b1d*muc*r1*cos(th2)*sin(b1)^2*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
g*m2*cos(th2)^2*cos(b2 + th3)*sin(b2)*sin(th3) +
(b3d^2*m3*muk*r3*cos(b3)^2*cos(th2)*sign(b3d)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*muk*r3*th3d*(2*b3d + th3d)*cos(b3)^2*cos(th2)*sign(b3d)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (g*m3*muk*cos(b3)*cos(th2)^2*cos(th3)*sign(b3d)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*muk*r3*th2d^2*cos(b3)*cos(th2)*cos(th3)*cos(b3 +
th3)*sign(b3d)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*muk*r3*th1d^2*cos(b3)*cos(th2)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sign(b3d)*sin(th3))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d^2*m3*r3*cos(b3)*cos(th2)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*r3*th3d*(2*b3d + th3d)*cos(b3)*cos(th2)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(g*m3*cos(th2)^2*cos(th3)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*r3*th2d^2*cos(th2)*cos(th3)*cos(b3 + th3)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3))
+ (m3*r3*th1d^2*cos(th2)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sin(b3)*sin(th3))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d*muc*muk*r3*cos(b3)*cos(th2)*sign(b3d)*sin(b3)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(b3d*muc*r3*cos(th2)*sin(b3)^2*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
g*m4*cos(th2)^2*cos(b4 + th3)*sin(b4)*sin(th3) - 2*d2*m2*th1d*th2d*cos(b2)*cos(th2)*cos(b2 +
th3)*sin(th2)*sin(th3) - 2*d4*m4*th1d*th2d*cos(b4)*cos(th2)*cos(b4 + th3)*sin(th2)*sin(th3) +
(2*d1*m1*th1d*th2d*cos(b1)*cos(th2)*cos(th3)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1))
- (2*d1*m1*muk*th1d*th2d*cos(th2)*cos(th3)*sign(b1d)*sin(b1)*sin(th2)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (2*d3*m3*th1d*th2d*cos(b3)*cos(th2)*cos(th3)*sin(th2)*sin(th3))/(cos(b3)
- muk*sign(b3d)*sin(b3)) -
(2*d3*m3*muk*th1d*th2d*cos(th2)*cos(th3)*sign(b3d)*sin(b3)*sin(th2)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) -
(d1*m1*muk*th1d^2*cos(b1)*cos(th2)*cos(th3)*sign(b1d)*sin(2*th2)*sin(th3))/(2*(cos(b1) -
muk*sign(b1d)*sin(b1))) -
(d1*m1*th1d^2*cos(th2)*cos(th3)*sin(b1)*sin(2*th2)*sin(th3))/(2*(cos(b1) -
muk*sign(b1d)*sin(b1))) - (d2*m2*th1d^2*cos(th2)*cos(b2 + th3)*sin(b2)*sin(2*th2)*sin(th3))/2 -
(d3*m3*muk*th1d^2*cos(b3)*cos(th2)*cos(th3)*sign(b3d)*sin(2*th2)*sin(th3))/(2*(cos(b3) -
muk*sign(b3d)*sin(b3))) -
(d3*m3*th1d^2*cos(th2)*cos(th3)*sin(b3)*sin(2*th2)*sin(th3))/(2*(cos(b3) -
muk*sign(b3d)*sin(b3))) - (d4*m4*th1d^2*cos(th2)*cos(b4 + th3)*sin(b4)*sin(2*th2)*sin(th3))/2 -
(2*d1*m1*muk*th1d*th2d*cos(b1)*cos(th2)*sign(b1d)*sin(th2)*sin(th3)^2)/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (2*d1*m1*th1d*th2d*cos(th2)*sin(b1)*sin(th2)*sin(th3)^2)/(cos(b1) -
```

```
muk*sign(b1d)*sin(b1)) -
(2*d3*m3*muk*th1d*th2d*cos(b3)*cos(th2)*sign(b3d)*sin(th2)*sin(th3)^2)/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (2*d3*m3*th1d*th2d*cos(th2)*sin(b3)*sin(th2)*sin(th3)^2)/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (2*m1*r1*th1d*cos(b1)*cos(th2)*cos(th3)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*m1*muk*r1*th1d*cos(th2)*cos(th3)*sign(b1d)*sin(b1)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*m1*muk*r1*th1d*cos(b1)*cos(th2)*sign(b1d)*sin(th3)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*m1*r1*th1d*cos(th2)*sin(b1)*sin(th3)*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (2*m3*r3*th1d*cos(b3)*cos(th2)*cos(th3)*((b3d
+ th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (2*m3*muk*r3*th1d*cos(th2)*cos(th3)*sign(b3d)*sin(b3)*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
+ (2*m3*muk*r3*th1d*cos(b3)*cos(th2)*sign(b3d)*sin(th3)*((b3d + th3d)*cos(b3)*sin(th2) -
th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*m3*r3*th1d*cos(th2)*sin(b3)*sin(th3)*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + 2*m1*r1*th2d*(b1d + th3d)*sin(th2)*sin(b1 +
th3) + g*m2*cos(th2)^2*cos(th3)*sin(b2)*sin(b2 + th3) + 2*m2*r2*th2d*th3d*sin(th2)*sin(b2 + th3)
+ 2*d2*m2*th1d*th2d*cos(b2)*cos(th2)*cos(th3)*sin(th2)*sin(b2 + th3) +
(d2*m2*th1d^2*cos(th2)*cos(th3)*sin(b2)*sin(2*th2)*sin(b2 + th3))/2 +
g*m2*cos(b2)*cos(th2)^2*sin(th3)*sin(b2 + th3) -
2*d2*m2*th1d*th2d*cos(th2)*sin(b2)*sin(th2)*sin(th3)*sin(b2 + th3) +
(d2*m2*th1d^2*cos(b2)*cos(th2)*sin(2*th2)*sin(th3)*sin(b2 + th3))/2 + (m2*r2*cos(th2)*(-th2d^2 +
th1d^2*cos(th2)^2)*cos(th3)*sin(b2)*sin(2*(b2 + th3)))/2 + (m2*r2*cos(b2)*cos(th2)*(-th2d^2 +
th1d^2*cos(th2)^2)*sin(th3)*sin(2*(b2 + th3)))/2 - (m2*r2*cos(b2)*cos(th2)*cos(th3)*(4*(th3d^2 +
th2d^2*cos(b2 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b2 + th3))) -
8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 +
(m2*r2*cos(th2)*sin(b2)*sin(th3)*(4*(th3d^2 + th2d^2*cos(b2 + th3)^2) - th1d*(th1d*(-3 +
cos(2*th2) + 2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b2 +
th3)))))/4 + 2*m3*r3*th2d*(b3d + th3d)*sin(th2)*sin(b3 + th3) +
g*m4*cos(th2)^2*cos(th3)*sin(b4)*sin(b4 + th3) + 2*m4*r4*th2d*th3d*sin(th2)*sin(b4 + th3) +
2*d4*m4*th1d*th2d*cos(b4)*cos(th2)*cos(th3)*sin(th2)*sin(b4 + th3) +
(d4*m4*th1d^2*cos(th2)*cos(th3)*sin(b4)*sin(2*th2)*sin(b4 + th3))/2 +
g*m4*cos(b4)*cos(th2)^2*sin(th3)*sin(b4 + th3) -
2*d4*m4*th1d*th2d*cos(th2)*sin(b4)*sin(th2)*sin(th3)*sin(b4 + th3) +
(d4*m4*th1d^2*cos(b4)*cos(th2)*sin(2*th2)*sin(th3)*sin(b4 + th3))/2 + (m4*r4*cos(th2)*(-th2d^2 +
th1d^2*cos(th2)^2)*cos(th3)*sin(b4)*sin(2*(b4 + th3)))/2 + (m4*r4*cos(b4)*cos(th2)*(-th2d^2 +
th1d^2*cos(th2)^2)*sin(th3)*sin(2*(b4 + th3)))/2 - (m4*r4*cos(b4)*cos(th2)*cos(th3)*(4*(th3d^2 +
th2d^2*cos(b4 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) -
8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4 +
(m4*r4*cos(th2)*sin(b4)*sin(th3)*(4*(th3d^2 + th2d^2*cos(b4 + th3)^2) - th1d*(th1d*(-3 +
cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b4 +
th3)))))/4 + 2*m*r*th2d*th3d*sin(th2)*sin(bm + th3);
f13(2) = ky*y + cy*yd + 2*m*r*th1d*th2d*cos(th1)*cos(th2)*cos(bm + th3) + dm*m*(th1d^2 +
th2d^2)*cos(th2)*sin(th1) + d1*m1*cos(th2)*(th2d^2 + th1d^2*cos(th2)^2)*sin(th1) +
d2*m2*cos(th2)*(th2d^2 + th1d^2*cos(th2)^2)*sin(th1) + d3*m3*cos(th2)*(th2d^2 +
th1d^2*cos(th2)^2)*sin(th1) + d4*m4*cos(th2)*(th2d^2 + th1d^2*cos(th2)^2)*sin(th1) -
2*m*r*th1d*th3d*cos(bm + th3)*sin(th1) + 2*dm*m*th1d*th2d*cos(th1)*sin(th2) -
g*m1*cos(th2)*sin(th1)*sin(th2) - g*m2*cos(th2)*sin(th1)*sin(th2) -
g*m3*cos(th2)*sin(th1)*sin(th2) - g*m4*cos(th2)*sin(th1)*sin(th2) - m*r*(th1d^2 + th2d^2 +
th3d^2)*cos(bm + th3)*sin(th1)*sin(th2) - m2*r2*th1d*cos(th2)^2*cos(b2 + th3)*sin(th1)*(2*th3d +
th1d*sin(th2)) - m4*r4*th1d*cos(th2)^2*cos(b4 + th3)*sin(th1)*(2*th3d + th1d*sin(th2)) -
m1*r1*th1d*cos(th2)^2*cos(b1 + th3)*sin(th1)*(2*(b1d + th3d) + th1d*sin(th2)) -
m3*r3*th1d*cos(th2)^2*cos(b3 + th3)*sin(th1)*(2*(b3d + th3d) + th1d*sin(th2)) -
g*m2*cos(b2)*cos(th2)*cos(b2 + th3)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)) -
g*m4*cos(b4)*cos(th2)*cos(b4 + th3)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)) +
(b1d^2*m1*r1*cos(b1)^2*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*r1*th3d*(2*b1d + th3d)*cos(b1)^2*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (g*m1*cos(b1)*cos(th2)*cos(th3)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (m1*r1*th1d^2*cos(b1)*(3*cos(b1) -
2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 + 2*th3))*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) + (b1d*muc*r1*cos(b1)*sin(b1)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(b1d^2*m1*muk*r1*cos(b1)*sign(b1d)*sin(b1)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th3d*(2*b1d +
th3d)*cos(b1)*sign(b1d)*sin(b1)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (g*m1*muk*cos(th2)*cos(th3)*sign(b1d)*sin(b1)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
```

```
(m1*muk*r1*th2d^2*cos(th3)*cos(b1 + th3)*sign(b1d)*sin(b1)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th1d^2*(3*cos(b1) -
2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 + 2*th3))*sign(b1d)*sin(b1)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(b1d*muc*muk*r1*sign(b1d)*sin(b1)^2*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b1)
- muk*sign(b1d)*sin(b1)) + 2*m2*r2*th1d*th2d*cos(th2)*cos(b2 + th3)^2*sin(b2)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)) + (b3d^2*m3*r3*cos(b3)^2*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*r3*th3d*(2*b3d + th3d)*cos(b3)^2*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (g*m3*cos(b3)*cos(th2)*cos(th3)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (m3*r3*th1d^2*cos(b3)*(3*cos(b3) -
2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 + 2*th3))*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) + (b3d*muc*r3*cos(b3)*sin(b3)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(b3d^2*m3*muk*r3*cos(b3)*sign(b3d)*sin(b3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th3d*(2*b3d +
th3d)*cos(b3)*sign(b3d)*sin(b3)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (g*m3*muk*cos(th2)*cos(th3)*sign(b3d)*sin(b3)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3*th2d^2*cos(th3)*cos(b3 + th3)*sign(b3d)*sin(b3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th1d^2*(3*cos(b3) -
2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 + 2*th3))*sign(b3d)*sin(b3)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(b3d*muc*muk*r3*sign(b3d)*sin(b3)^2*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/(cos(b3)
- muk*sign(b3d)*sin(b3)) + 2*m4*r4*th1d*th2d*cos(th2)*cos(b4 + th3)^2*sin(b4)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)) - muc*(b1d*r1*sin(b1) + b2d*r2*sin(b2) +
b3d*r3*sin(b3) + b4d*r4*sin(b4))*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)) +
2*d2*m2*th1d*th2d*cos(b2 + th3)*sin(b2)*sin(th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)) + 2*d4*m4*th1d*th2d*cos(b4 + th3)*sin(b4)*sin(th2)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)) - (d2*m2*th1d^2*cos(b2)*cos(b2 +
th3)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3)))/2 -
(d4*m4*th1d^2*cos(b4)*cos(b4 + th3)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/2 - (d1*m1*th1d^2*cos(b1)*cos(th3)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2))
- cos(th1)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(d1*m1*muk*th1d^2*cos(th3)*sign(b1d)*sin(b1)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(d3*m3*th1d^2*cos(b3)*cos(th3)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(d3*m3*muk*th1d^2*cos(th3)*sign(b3d)*sin(b3)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(2*d1*m1*th1d*th2d*cos(b1)*sin(th2)*sin(th3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d1*m1*muk*th1d*th2d*sign(b1d)*sin(b1)*sin(th2)*sin(th3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*d3*m3*th1d*th2d*cos(b3)*sin(th2)*sin(th3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*d3*m3*muk*th1d*th2d*sign(b3d)*sin(b3)*sin(th2)*sin(th3)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (2*m1*r1*th1d*cos(b1)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*m1*muk*r1*th1d*sign(b1d)*sin(b1)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*((b1d +
th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1))
+ (2*m3*r3*th1d*cos(b3)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
- (2*m3*muk*r3*th1d*sign(b3d)*sin(b3)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
- muc*(b1d*r1*cos(b1) + b2d*r2*cos(b2) + b3d*r3*cos(b3) + b4d*r4*cos(b4))*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)) + 2*m2*r2*th1d*th2d*cos(b2)*cos(th2)*cos(b2 +
th3)^2*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)) +
2*m4*r4*th1d*th2d*cos(b4)*cos(th2)*cos(b4 + th3)^2*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)) - (b1d^2*m1*muk*r1*cos(b1)^2*sign(b1d)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th3d*(2*b1d +
th3d)*cos(b1)^2*sign(b1d)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (g*m1*muk*cos(b1)*cos(th2)*cos(th3)*sign(b1d)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3)*sign(b1d)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1*th1d^2*cos(b1)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sign(b1d)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(4*(cos(b1) -
```

```
muk*sign(b1d)*sin(b1))) - (b1d^2*m1*r1*cos(b1)*sin(b1)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*r1*th3d*(2*b1d +
th3d)*cos(b1)*sin(b1)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (g*m1*cos(th2)*cos(th3)*sin(b1)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*r1*th2d^2*cos(th3)*cos(b1 +
th3)*sin(b1)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1))
- (m1*r1*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sin(b1)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(4*(cos(b1) -
muk*sign(b1d)*sin(b1))) - (b1d*muc*muk*r1*cos(b1)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(b1d*muc*r1*sin(b1)^2*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + g*m2*cos(th2)*cos(b2 + th3)*sin(b2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)) - (b3d^2*m3*muk*r3*cos(b3)^2*sign(b3d)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th3d*(2*b3d +
th3d)*cos(b3)^2*sign(b3d)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (g*m3*muk*cos(b3)*cos(th2)*cos(th3)*sign(b3d)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3)*sign(b3d)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3*th1d^2*cos(b3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sign(b3d)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(4*(cos(b3) -
muk*sign(b3d)*sin(b3))) - (b3d^2*m3*r3*cos(b3)*sin(b3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*r3*th3d*(2*b3d +
th3d)*cos(b3)*sin(b3)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (g*m3*cos(th2)*cos(th3)*sin(b3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*r3*th2d^2*cos(th3)*cos(b3 +
th3)*sin(b3)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
- (m3*r3*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sin(b3)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(4*(cos(b3) -
muk*sign(b3d)*sin(b3))) - (b3d*muc*muk*r3*cos(b3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(b3d*muc*r3*sin(b3)^2*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + g*m4*cos(th2)*cos(b4 + th3)*sin(b4)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)) + 2*d2*m2*th1d*th2d*cos(b2)*cos(b2 + th3)*sin(th2)*(cos(th1)*cos(th3)
- sin(th1)*sin(th2)*sin(th3)) + 2*d4*m4*th1d*th2d*cos(b4)*cos(b4 +
th3)*sin(th2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)) +
(d1*m1*muk*th1d^2*cos(b1)*cos(th3)*sign(b1d)*sin(2*th2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(d1*m1*th1d^2*cos(th3)*sin(b1)*sin(2*th2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) + (d2*m2*th1d^2*cos(b2 +
th3)*sin(b2)*sin(2*th2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/2 +
(d3*m3*muk*th1d^2*cos(b3)*cos(th3)*sign(b3d)*sin(2*th2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(d3*m3*th1d^2*cos(th3)*sin(b3)*sin(2*th2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) + (d4*m4*th1d^2*cos(b4 +
th3)*sin(b4)*sin(2*th2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/2 +
(2*d1*m1*muk*th1d*th2d*cos(b1)*sign(b1d)*sin(th2)*sin(th3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d1*m1*th1d*th2d*sin(b1)*sin(th2)*sin(th3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d3*m3*muk*th1d*th2d*cos(b3)*sign(b3d)*sin(th2)*sin(th3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*d3*m3*th1d*th2d*sin(b3)*sin(th2)*sin(th3)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(2*m1*muk*r1*th1d*cos(b1)*sign(b1d)*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3))*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (2*m1*r1*th1d*sin(b1)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3))*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b1)
- muk*sign(b1d)*sin(b1)) - (2*m3*muk*r3*th1d*cos(b3)*sign(b3d)*((b3d + th3d)*cos(b3)*sin(th2) -
th2d*cos(th2)*cos(b3 + th3)*sin(th3))*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3)
- muk*sign(b3d)*sin(b3)) - (2*m3*r3*th1d*sin(b3)*((b3d + th3d)*cos(b3)*sin(th2) -
th2d*cos(th2)*cos(b3 + th3)*sin(th3))*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3)))/(cos(b3)
- muk*sign(b3d)*sin(b3)) - 2*m1*r1*th2d*(b1d + th3d)*cos(th2)*sin(th1)*sin(b1 + th3) -
2*m2*r2*th2d*th3d*cos(th2)*sin(th1)*sin(b2 + th3) - g*m2*cos(th2)*sin(b2)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*sin(b2 + th3) -
2*d2*m2*th1d*th2d*cos(b2)*sin(th2)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*sin(b2 +
th3) - (d2*m2*th1d^2*sin(b2)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3))*sin(b2 + th3))/2 - g*m2*cos(b2)*cos(th2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3))*sin(b2 + th3) + 2*d2*m2*th1d*th2d*sin(b2)*sin(th2)*(cos(th1)*cos(th3)
- sin(th1)*sin(th2)*sin(th3))*sin(b2 + th3) - (d2*m2*th1d^2*cos(b2)*sin(2*th2)*(cos(th1)*cos(th3)
- sin(th1)*sin(th2)*sin(th3))*sin(b2 + th3))/2 - (m2*r2*(-th2d^2 + th1d^2*cos(th2)^2)*sin(b2)*(-
```

```
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*sin(2*(b2 + th3)))/2 - (m2*r2*cos(b2)*(-th2d^2
+ th1d^2*cos(th2)^2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3))*sin(2*(b2 + th3)))/2 +
(m2*r2*cos(b2)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b2 +
th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 - (m2*r2*sin(b2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b2 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) +
2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 -
2*m3*r3*th2d*(b3d + th3d)*cos(th2)*sin(th1)*sin(b3 + th3) -
2*m4*r4*th2d*th3d*cos(th2)*sin(th1)*sin(b4 + th3) - g*m4*cos(th2)*sin(b4)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*sin(b4 + th3) -
2*d4*m4*th1d*th2d*cos(b4)*sin(th2)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*sin(b4 +
th3) - (d4*m4*th1d^2*sin(b4)*sin(2*th2)*(-(cos(th3)*sin(th1)*sin(th2)) -
cos(th1)*sin(th3))*sin(b4 + th3))/2 - g*m4*cos(b4)*cos(th2)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3))*sin(b4 + th3) + 2*d4*m4*th1d*th2d*sin(b4)*sin(th2)*(cos(th1)*cos(th3)
- sin(th1)*sin(th2)*sin(th3))*sin(b4 + th3) - (d4*m4*th1d^2*cos(b4)*sin(2*th2)*(cos(th1)*cos(th3)
- sin(th1)*sin(th2)*sin(th3))*sin(b4 + th3))/2 - (m4*r4*(-th2d^2 + th1d^2*cos(th2)^2)*sin(b4)*(-
(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*sin(2*(b4 + th3)))/2 - (m4*r4*cos(b4)*(-th2d^2
+ th1d^2*cos(th2)^2)*(cos(th1)*cos(th3) - sin(th1)*sin(th2)*sin(th3))*sin(2*(b4 + th3)))/2 +
(m4*r4*cos(b4)*(-(cos(th3)*sin(th1)*sin(th2)) - cos(th1)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b4 +
th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4 - (m4*r4*sin(b4)*(cos(th1)*cos(th3) -
sin(th1)*sin(th2)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b4 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) +
2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4 -
m*r*(th1d^2 + th3d^2)*cos(th1)*sin(bm + th3) - 2*m*r*th2d*th3d*cos(th2)*sin(th1)*sin(bm + th3) -
2*m*r*th1d*th3d*cos(th1)*sin(th2)*sin(bm + th3);
f13(3) = kz*z + cz*zd - dm*m*(th1d^2 + th2d^2)*cos(th1)*cos(th2) -
d1*m1*cos(th1)*cos(th2)*(th2d^2 + th1d^2*cos(th2)^2) - d2*m2*cos(th1)*cos(th2)*(th2d^2 +
th1d^2*cos(th2)^2) - d3*m3*cos(th1)*cos(th2)*(th2d^2 + th1d^2*cos(th2)^2) -
d4*m4*cos(th1)*cos(th2)*(th2d^2 + th1d^2*cos(th2)^2) + 2*m*r*th1d*th3d*cos(th1)*cos(bm + th3) +
2*m*r*th1d*th2d*cos(th2)*cos(bm + th3)*sin(th1) + g*m1*cos(th1)*cos(th2)*sin(th2) +
g*m2*cos(th1)*cos(th2)*sin(th2) + g*m3*cos(th1)*cos(th2)*sin(th2) +
g*m4*cos(th1)*cos(th2)*sin(th2) + m*r*(th1d^2 + th2d^2 + th3d^2)*cos(th1)*cos(bm + th3)*sin(th2)
+ 2*dm*m*th1d*th2d*sin(th1)*sin(th2) + m2*r2*th1d*cos(th1)*cos(th2)^2*cos(b2 + th3)*(2*th3d +
th1d*sin(th2)) + m4*r4*th1d*cos(th1)*cos(th2)^2*cos(b4 + th3)*(2*th3d + th1d*sin(th2)) +
m1*r1*th1d*cos(th1)*cos(th2)^2*cos(b1 + th3)*(2*(b1d + th3d) + th1d*sin(th2)) +
m3*r3*th1d*cos(th1)*cos(th2)^2*cos(b3 + th3)*(2*(b3d + th3d) + th1d*sin(th2)) -
g*m2*cos(b2)*cos(th2)*cos(b2 + th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) -
g*m4*cos(b4)*cos(th2)*cos(b4 + th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
(b1d^2*m1*r1*cos(b1)^2*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*r1*th3d*(2*b1d + th3d)*cos(b1)^2*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(g*m1*cos(b1)*cos(th2)*cos(th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (m1*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3)*(cos(th1)*cos(th3)*sin(th2)
- sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (m1*r1*th1d^2*cos(b1)*(3*cos(b1) -
2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 + 2*th3))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d*muc*r1*cos(b1)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (b1d^2*m1*muk*r1*cos(b1)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th3d*(2*b1d +
th3d)*cos(b1)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) +
(g*m1*muk*cos(th2)*cos(th3)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th2d^2*cos(th3)*cos(b1 +
th3)*sign(b1d)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) -
cos(b1 + 2*th3))*sign(b1d)*sin(b1)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(4*(cos(b1)
- muk*sign(b1d)*sin(b1))) - (b1d*muc*muk*r1*sign(b1d)*sin(b1)^2*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) + 2*m2*r2*th1d*th2d*cos(th2)*cos(b2 +
th3)^2*sin(b2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) +
(b3d^2*m3*r3*cos(b3)^2*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*r3*th3d*(2*b3d + th3d)*cos(b3)^2*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(g*m3*cos(b3)*cos(th2)*cos(th3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (m3*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3)*(cos(th1)*cos(th3)*sin(th2)
- sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (m3*r3*th1d^2*cos(b3)*(3*cos(b3) -
2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 + 2*th3))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d*muc*r3*cos(b3)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (b3d^2*m3*muk*r3*cos(b3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th3d*(2*b3d +
th3d)*cos(b3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
```

```
muk*sign(b3d)*sin(b3)) +
(g*m3*muk*cos(th2)*cos(th3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th2d^2*cos(th3)*cos(b3 +
th3)*sign(b3d)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) -
cos(b3 + 2*th3))*sign(b3d)*sin(b3)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/(4*(cos(b3)
- muk*sign(b3d)*sin(b3))) - (b3d*muc*muk*r3*sign(b3d)*sin(b3)^2*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + 2*m4*r4*th1d*th2d*cos(th2)*cos(b4 +
th3)^2*sin(b4)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) - muc*(b1d*r1*sin(b1) +
b2d*r2*sin(b2) + b3d*r3*sin(b3) + b4d*r4*sin(b4))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)) + 2*d2*m2*th1d*th2d*cos(b2 + th3)*sin(b2)*sin(th2)*(cos(th1)*cos(th3)*sin(th2)
- sin(th1)*sin(th3)) + 2*d4*m4*th1d*th2d*cos(b4 +
th3)*sin(b4)*sin(th2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)) -
(d2*m2*th1d^2*cos(b2)*cos(b2 + th3)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/2 - (d4*m4*th1d^2*cos(b4)*cos(b4 +
th3)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3)))/2 -
(d1*m1*th1d^2*cos(b1)*cos(th3)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(d1*m1*muk*th1d^2*cos(th3)*sign(b1d)*sin(b1)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(d3*m3*th1d^2*cos(b3)*cos(th3)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(d3*m3*muk*th1d^2*cos(th3)*sign(b3d)*sin(b3)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(2*d1*m1*th1d*th2d*cos(b1)*sin(th2)*sin(th3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d1*m1*muk*th1d*th2d*sign(b1d)*sin(b1)*sin(th2)*sin(th3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*d3*m3*th1d*th2d*cos(b3)*sin(th2)*sin(th3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*d3*m3*muk*th1d*th2d*sign(b3d)*sin(b3)*sin(th2)*sin(th3)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (2*m1*r1*th1d*cos(b1)*((b1d +
th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 + th3)*sin(th3))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (2*m1*muk*r1*th1d*sign(b1d)*sin(b1)*((b1d
+ th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 + th3)*sin(th3))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (2*m3*r3*th1d*cos(b3)*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (2*m3*muk*r3*th1d*sign(b3d)*sin(b3)*((b3d
+ th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3))*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - muc*(b1d*r1*cos(b1) + b2d*r2*cos(b2) +
b3d*r3*cos(b3) + b4d*r4*cos(b4))*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
2*m2*r2*th1d*th2d*cos(b2)*cos(th2)*cos(b2 + th3)^2*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)) + 2*m4*r4*th1d*th2d*cos(b4)*cos(th2)*cos(b4 +
th3)^2*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) -
(b1d^2*m1*muk*r1*cos(b1)^2*sign(b1d)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (m1*muk*r1*th3d*(2*b1d + th3d)*cos(b1)^2*sign(b1d)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(g*m1*muk*cos(b1)*cos(th2)*cos(th3)*sign(b1d)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3)*sign(b1d)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1*th1d^2*cos(b1)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sign(b1d)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(4*(cos(b1) -
muk*sign(b1d)*sin(b1))) - (b1d^2*m1*r1*cos(b1)*sin(b1)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*r1*th3d*(2*b1d +
th3d)*cos(b1)*sin(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (g*m1*cos(th2)*cos(th3)*sin(b1)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (m1*r1*th2d^2*cos(th3)*cos(b1 +
th3)*sin(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1))
- (m1*r1*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sin(b1)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(4*(cos(b1) -
muk*sign(b1d)*sin(b1))) - (b1d*muc*muk*r1*cos(b1)*sign(b1d)*sin(b1)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(b1d*muc*r1*sin(b1)^2*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + g*m2*cos(th2)*cos(b2 + th3)*sin(b2)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)) - (b3d^2*m3*muk*r3*cos(b3)^2*sign(b3d)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*muk*r3*th3d*(2*b3d +
th3d)*cos(b3)^2*sign(b3d)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (g*m3*muk*cos(b3)*cos(th2)*cos(th3)*sign(b3d)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3)*sign(b3d)*(cos(th3)*sin(th1) +
```

```
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3*th1d^2*cos(b3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sign(b3d)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(4*(cos(b3) -
muk*sign(b3d)*sin(b3))) - (b3d^2*m3*r3*cos(b3)*sin(b3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*r3*th3d*(2*b3d +
th3d)*cos(b3)*sin(b3)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (g*m3*cos(th2)*cos(th3)*sin(b3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - (m3*r3*th2d^2*cos(th3)*cos(b3 +
th3)*sin(b3)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
- (m3*r3*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sin(b3)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(4*(cos(b3) -
muk*sign(b3d)*sin(b3))) - (b3d*muc*muk*r3*cos(b3)*sign(b3d)*sin(b3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(b3d*muc*r3*sin(b3)^2*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + g*m4*cos(th2)*cos(b4 + th3)*sin(b4)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)) + 2*d2*m2*th1d*th2d*cos(b2)*cos(b2 + th3)*sin(th2)*(cos(th3)*sin(th1)
+ cos(th1)*sin(th2)*sin(th3)) + 2*d4*m4*th1d*th2d*cos(b4)*cos(b4 +
th3)*sin(th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)) +
(d1*m1*muk*th1d^2*cos(b1)*cos(th3)*sign(b1d)*sin(2*th2)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(d1*m1*th1d^2*cos(th3)*sin(b1)*sin(2*th2)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) + (d2*m2*th1d^2*cos(b2 +
th3)*sin(b2)*sin(2*th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/2 +
(d3*m3*muk*th1d^2*cos(b3)*cos(th3)*sign(b3d)*sin(2*th2)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(d3*m3*th1d^2*cos(th3)*sin(b3)*sin(2*th2)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) + (d4*m4*th1d^2*cos(b4 +
th3)*sin(b4)*sin(2*th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/2 +
(2*d1*m1*muk*th1d*th2d*cos(b1)*sign(b1d)*sin(th2)*sin(th3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d1*m1*th1d*th2d*sin(b1)*sin(th2)*sin(th3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d3*m3*muk*th1d*th2d*cos(b3)*sign(b3d)*sin(th2)*sin(th3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*d3*m3*th1d*th2d*sin(b3)*sin(th2)*sin(th3)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(2*m1*muk*r1*th1d*cos(b1)*sign(b1d)*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3))*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (2*m1*r1*th1d*sin(b1)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3))*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b1)
- muk*sign(b1d)*sin(b1)) - (2*m3*muk*r3*th1d*cos(b3)*sign(b3d)*((b3d + th3d)*cos(b3)*sin(th2) -
th2d*cos(th2)*cos(b3 + th3)*sin(th3))*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3)
- muk*sign(b3d)*sin(b3)) - (2*m3*r3*th1d*sin(b3)*((b3d + th3d)*cos(b3)*sin(th2) -
th2d*cos(th2)*cos(b3 + th3)*sin(th3))*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3)))/(cos(b3)
- muk*sign(b3d)*sin(b3)) + 2*m1*r1*th2d*(b1d + th3d)*cos(th1)*cos(th2)*sin(b1 + th3) +
2*m2*r2*th2d*th3d*cos(th1)*cos(th2)*sin(b2 + th3) -
g*m2*cos(th2)*sin(b2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b2 + th3) -
2*d2*m2*th1d*th2d*cos(b2)*sin(th2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b2 + th3)
- (d2*m2*th1d^2*sin(b2)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b2 +
th3))/2 - g*m2*cos(b2)*cos(th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b2 + th3) +
2*d2*m2*th1d*th2d*sin(b2)*sin(th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b2 + th3)
- (d2*m2*th1d^2*cos(b2)*sin(2*th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b2 +
th3))/2 - (m2*r2*(-th2d^2 + th1d^2*cos(th2)^2)*sin(b2)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3))*sin(2*(b2 + th3)))/2 - (m2*r2*cos(b2)*(-th2d^2 +
th1d^2*cos(th2)^2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(2*(b2 + th3)))/2 +
(m2*r2*cos(b2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b2 +
th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 - (m2*r2*sin(b2)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b2 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) +
2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 +
2*m3*r3*th2d*(b3d + th3d)*cos(th1)*cos(th2)*sin(b3 + th3) +
2*m4*r4*th2d*th3d*cos(th1)*cos(th2)*sin(b4 + th3) -
g*m4*cos(th2)*sin(b4)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b4 + th3) -
2*d4*m4*th1d*th2d*cos(b4)*sin(th2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b4 + th3)
- (d4*m4*th1d^2*sin(b4)*sin(2*th2)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*sin(b4 +
th3))/2 - g*m4*cos(b4)*cos(th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b4 + th3) +
2*d4*m4*th1d*th2d*sin(b4)*sin(th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b4 + th3)
- (d4*m4*th1d^2*cos(b4)*sin(2*th2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(b4 +
th3))/2 - (m4*r4*(-th2d^2 + th1d^2*cos(th2)^2)*sin(b4)*(cos(th1)*cos(th3)*sin(th2) -
sin(th1)*sin(th3))*sin(2*(b4 + th3)))/2 - (m4*r4*cos(b4)*(-th2d^2 +
th1d^2*cos(th2)^2)*(cos(th3)*sin(th1) + cos(th1)*sin(th2)*sin(th3))*sin(2*(b4 + th3)))/2 +
```

```
(m4*r4*cos(b4)*(cos(th1)*cos(th3)*sin(th2) - sin(th1)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b4 +
th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4 - (m4*r4*sin(b4)*(cos(th3)*sin(th1) +
cos(th1)*sin(th2)*sin(th3))*(4*(th3d^2 + th2d^2*cos(b4 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) +
2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4 +
2*m*r*th2d*th3d*cos(th1)*cos(th2)*sin(bm + th3) - m*r*sin(th1)*(th1d^2 + th3d^2 +
2*th1d*th3d*sin(th2))*sin(bm + th3);
f13(4) = Iyz*th3d^2 + b1d*d1*muc*r1*cos(b1) + b2d*d2*muc*r2*cos(b2) + muc*(b3d*d3*r3*cos(b3) +
b4d*d4*r4*cos(b4)) + kth1*th1*cos(th3) + Ixx*th2d*th3d*cos(th3) - Iyy*th2d*th3d*cos(th3) +
Izz*th2d*th3d*cos(th3) - Izy*th2d^2*cos(th3)^2 - 2*d2*m2*r2*th1d*th2d*cos(b2)*cos(th2)*cos(b2 +
th3)^2 - 2*d4*m4*r4*th1d*th2d*cos(b4)*cos(th2)*cos(b4 + th3)^2 - d1*m1*r1*(th2d^2 +
th1d^2*cos(th2)^2)*sin(b1) + (b1d^2*d1*m1*muk*r1*cos(b1)^2*sign(b1d))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*muk*r1*th3d*(2*b1d + th3d)*cos(b1)^2*sign(b1d))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (d1*g*m1*muk*cos(b1)*cos(th2)*cos(th3)*sign(b1d))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*muk*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3)*sign(b1d))/(cos(b1)
- muk*sign(b1d)*sin(b1)) + (d1*m1*muk*r1*th1d^2*cos(b1)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1
+ th3) - cos(b1 + 2*th3))*sign(b1d))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d^2*d1*m1*r1*cos(b1)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (d1*m1*r1*th3d*(2*b1d +
th3d)*cos(b1)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*g*m1*cos(th2)*cos(th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*r1*th2d^2*cos(th3)*cos(b1 + th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*m1*r1*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sin(b1))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d*d1*muc*muk*r1*cos(b1)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(b1d*d1*muc*r1*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) - d2*m2*r2*(th2d^2 +
th1d^2*cos(th2)^2)*sin(b2) - d2*g*m2*cos(th2)*cos(b2 + th3)*sin(b2) - d3*m3*r3*(th2d^2 +
th1d^2*cos(th2)^2)*sin(b3) + (b3d^2*d3*m3*muk*r3*cos(b3)^2*sign(b3d))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d3*m3*muk*r3*th3d*(2*b3d + th3d)*cos(b3)^2*sign(b3d))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (d3*g*m3*muk*cos(b3)*cos(th2)*cos(th3)*sign(b3d))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d3*m3*muk*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3)*sign(b3d))/(cos(b3)
- muk*sign(b3d)*sin(b3)) + (d3*m3*muk*r3*th1d^2*cos(b3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3
+ th3) - cos(b3 + 2*th3))*sign(b3d))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d^2*d3*m3*r3*cos(b3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (d3*m3*r3*th3d*(2*b3d +
th3d)*cos(b3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*g*m3*cos(th2)*cos(th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*r3*th2d^2*cos(th3)*cos(b3 + th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(d3*m3*r3*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sin(b3))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d*d3*muc*muk*r3*cos(b3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(b3d*d3*muc*r3*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) - d4*m4*r4*(th2d^2 +
th1d^2*cos(th2)^2)*sin(b4) - d4*g*m4*cos(th2)*cos(b4 + th3)*sin(b4) -
2*d2^2*m2*th1d*th2d*cos(b2)*cos(b2 + th3)*sin(th2) - 2*d4^2*m4*th1d*th2d*cos(b4)*cos(b4 +
th3)*sin(th2) + g*m1*r1*sin(b1)*sin(th2) + g*m2*r2*sin(b2)*sin(th2) + g*m3*r3*sin(b3)*sin(th2) +
g*m4*r4*sin(b4)*sin(th2) + m2*r2^2*th1d*cos(th2)*cos(b2 + th3)*sin(b2)*(2*th3d + th1d*sin(th2)) +
m4*r4^2*th1d*cos(th2)*cos(b4 + th3)*sin(b4)*(2*th3d + th1d*sin(th2)) +
m1*r1^2*th1d*cos(th2)*cos(b1 + th3)*sin(b1)*(2*(b1d + th3d) + th1d*sin(th2)) +
m3*r3^2*th1d*cos(th2)*cos(b3 + th3)*sin(b3)*(2*(b3d + th3d) + th1d*sin(th2)) -
(d1^2*m1*muk*th1d^2*cos(b1)*cos(th3)*sign(b1d)*sin(2*th2))/(2*(cos(b1) - muk*sign(b1d)*sin(b1)))
- (d1^2*m1*th1d^2*cos(th3)*sin(b1)*sin(2*th2))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(d2^2*m2*th1d^2*cos(b2 + th3)*sin(b2)*sin(2*th2))/2 -
(d3^2*m3*muk*th1d^2*cos(b3)*cos(th3)*sign(b3d)*sin(2*th2))/(2*(cos(b3) - muk*sign(b3d)*sin(b3)))
- (d3^2*m3*th1d^2*cos(th3)*sin(b3)*sin(2*th2))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(d4^2*m4*th1d^2*cos(b4 + th3)*sin(b4)*sin(2*th2))/2 + kth2*th2*sin(th3) + cth2*th2d*sin(th3) +
Ixy*th2d*th3d*sin(th3) + Iyx*th2d*th3d*sin(th3) - Izx*th2d^2*cos(th3)*sin(th3) -
(2*d1^2*m1*muk*th1d*th2d*cos(b1)*sign(b1d)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*d1^2*m1*th1d*th2d*sin(b1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*d3^2*m3*muk*th1d*th2d*cos(b3)*sign(b3d)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(2*d3^2*m3*th1d*th2d*sin(b3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*d1*m1*muk*r1*th1d*cos(b1)*sign(b1d)*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (2*d1*m1*r1*th1d*sin(b1)*((b1d +
th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1))
+ (2*d3*m3*muk*r3*th1d*cos(b3)*sign(b3d)*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (2*d3*m3*r3*th1d*sin(b3)*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
+ th1d^2*(Iyz*sin(th2)^2 + cos(th2)^2*sin(th3)*(Izx*cos(th3) - Izy*sin(th3)) +
cos(th2)*sin(th2)*(Iyx*cos(th3) + (Iyy - Izz)*sin(th3))) - th1d*(-(cth1*cos(th3)) + th3d*(-
2*Iyz*sin(th2) + cos(th2)*((-Ixy - Iyx)*cos(th3) + (Ixx - Iyy + Izz)*sin(th3))) +
th2d*(sin(th2)*((Ixx + Iyy - Izz)*cos(th3) + (Ixy - Iyx)*sin(th3)) + cos(th2)*(Ixz +
Izx*cos(2*th3) - Izy*sin(2*th3)))) + 2*m1*r1^2*th2d*(b1d + th3d)*sin(b1)*sin(b1 + th3) +
d2*g*m2*cos(b2)*cos(th2)*sin(b2 + th3) + 2*m2*r2^2*th2d*th3d*sin(b2)*sin(b2 + th3) -
2*d2^2*m2*th1d*th2d*sin(b2)*sin(th2)*sin(b2 + th3) + (d2^2*m2*th1d^2*cos(b2)*sin(2*th2)*sin(b2 +
```

```
th3))/2 + (d2*m2*r2*cos(b2)*(-th2d^2 + th1d^2*cos(th2)^2)*sin(2*(b2 + th3)))/2 +
(d2*m2*r2*sin(b2)*(4*(th3d^2 + th2d^2*cos(b2 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) +
2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 +
2*m3*r3^2*th2d*(b3d + th3d)*sin(b3)*sin(b3 + th3) + d4*g*m4*cos(b4)*cos(th2)*sin(b4 + th3) +
2*m4*r4^2*th2d*th3d*sin(b4)*sin(b4 + th3) - 2*d4^2*m4*th1d*th2d*sin(b4)*sin(th2)*sin(b4 + th3) +
(d4^2*m4*th1d^2*cos(b4)*sin(2*th2)*sin(b4 + th3))/2 + (d4*m4*r4*cos(b4)*(-th2d^2 +
th1d^2*cos(th2)^2)*sin(2*(b4 + th3)))/2 + (d4*m4*r4*sin(b4)*(4*(th3d^2 + th2d^2*cos(b4 + th3)^2)
- th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4;
f13(5) = -(Ixz*th3d^2) + d1*m1*r1*cos(b1)*(th2d^2 + th1d^2*cos(th2)^2) + d2*m2*r2*cos(b2)*(th2d^2
+ th1d^2*cos(th2)^2) + d3*m3*r3*cos(b3)*(th2d^2 + th1d^2*cos(th2)^2) + d4*m4*r4*cos(b4)*(th2d^2 +
th1d^2*cos(th2)^2) + kth2*th2*cos(th3) + cth2*th2d*cos(th3) - Ixy*th2d*th3d*cos(th3) -
Iyx*th2d*th3d*cos(th3) + d2*g*m2*cos(b2)*cos(th2)*cos(b2 + th3) + d4*g*m4*cos(b4)*cos(th2)*cos(b4
+ th3) + b1d*d1*muc*r1*sin(b1) - (b1d^2*d1*m1*r1*cos(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*r1*th3d*(2*b1d + th3d)*cos(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(d1*g*m1*cos(b1)*cos(th2)*cos(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(d1*m1*r1*th1d^2*cos(b1)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3)))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) - (b1d*d1*muc*r1*cos(b1)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (b1d^2*d1*m1*muk*r1*cos(b1)*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*muk*r1*th3d*(2*b1d + th3d)*cos(b1)*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (d1*g*m1*muk*cos(th2)*cos(th3)*sign(b1d)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (d1*m1*muk*r1*th2d^2*cos(th3)*cos(b1 + th3)*sign(b1d)*sin(b1))/(cos(b1)
- muk*sign(b1d)*sin(b1)) + (d1*m1*muk*r1*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3)
- cos(b1 + 2*th3))*sign(b1d)*sin(b1))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d*d1*muc*muk*r1*sign(b1d)*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) + b2d*d2*muc*r2*sin(b2)
- 2*d2*m2*r2*th1d*th2d*cos(th2)*cos(b2 + th3)^2*sin(b2) - (b3d^2*d3*m3*r3*cos(b3)^2)/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (d3*m3*r3*th3d*(2*b3d + th3d)*cos(b3)^2)/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d3*g*m3*cos(b3)*cos(th2)*cos(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(d3*m3*r3*th1d^2*cos(b3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3)))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) - (b3d*d3*muc*r3*cos(b3)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (b3d^2*d3*m3*muk*r3*cos(b3)*sign(b3d)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d3*m3*muk*r3*th3d*(2*b3d + th3d)*cos(b3)*sign(b3d)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (d3*g*m3*muk*cos(th2)*cos(th3)*sign(b3d)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) + (d3*m3*muk*r3*th2d^2*cos(th3)*cos(b3 + th3)*sign(b3d)*sin(b3))/(cos(b3)
- muk*sign(b3d)*sin(b3)) + (d3*m3*muk*r3*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3)
- cos(b3 + 2*th3))*sign(b3d)*sin(b3))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d*d3*muc*muk*r3*sign(b3d)*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) -
2*d4*m4*r4*th1d*th2d*cos(th2)*cos(b4 + th3)^2*sin(b4) + muc*(b3d*d3*r3*sin(b3) +
b4d*d4*r4*sin(b4)) - g*m1*r1*cos(b1)*sin(th2) - g*m2*r2*cos(b2)*sin(th2) -
g*m3*r3*cos(b3)*sin(th2) - g*m4*r4*cos(b4)*sin(th2) - 2*d2^2*m2*th1d*th2d*cos(b2 +
th3)*sin(b2)*sin(th2) - 2*d4^2*m4*th1d*th2d*cos(b4 + th3)*sin(b4)*sin(th2) -
m2*r2^2*th1d*cos(b2)*cos(th2)*cos(b2 + th3)*(2*th3d + th1d*sin(th2)) -
m4*r4^2*th1d*cos(b4)*cos(th2)*cos(b4 + th3)*(2*th3d + th1d*sin(th2)) -
m1*r1^2*th1d*cos(b1)*cos(th2)*cos(b1 + th3)*(2*(b1d + th3d) + th1d*sin(th2)) -
m3*r3^2*th1d*cos(b3)*cos(th2)*cos(b3 + th3)*(2*(b3d + th3d) + th1d*sin(th2)) +
(d2^2*m2*th1d^2*cos(b2)*cos(b2 + th3)*sin(2*th2))/2 + (d4^2*m4*th1d^2*cos(b4)*cos(b4 +
th3)*sin(2*th2))/2 + (d1^2*m1*th1d^2*cos(b1)*cos(th3)*sin(2*th2))/(2*(cos(b1) -
muk*sign(b1d)*sin(b1))) - (d1^2*m1*muk*th1d^2*cos(th3)*sign(b1d)*sin(b1)*sin(2*th2))/(2*(cos(b1)
- muk*sign(b1d)*sin(b1))) + (d3^2*m3*th1d^2*cos(b3)*cos(th3)*sin(2*th2))/(2*(cos(b3) -
muk*sign(b3d)*sin(b3))) - (d3^2*m3*muk*th1d^2*cos(th3)*sign(b3d)*sin(b3)*sin(2*th2))/(2*(cos(b3)
- muk*sign(b3d)*sin(b3))) - kth1*th1*sin(th3) + Ixx*th2d*th3d*sin(th3) - Iyy*th2d*th3d*sin(th3) -
Izz*th2d*th3d*sin(th3) + Izy*th2d^2*cos(th3)*sin(th3) +
(2*d1^2*m1*th1d*th2d*cos(b1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*d1^2*m1*muk*th1d*th2d*sign(b1d)*sin(b1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d3^2*m3*th1d*th2d*cos(b3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(2*d3^2*m3*muk*th1d*th2d*sign(b3d)*sin(b3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
Izx*th2d^2*sin(th3)^2 - (2*d1*m1*r1*th1d*cos(b1)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*d1*m1*muk*r1*th1d*sign(b1d)*sin(b1)*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) - (2*d3*m3*r3*th1d*cos(b3)*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
+ (2*d3*m3*muk*r3*th1d*sign(b3d)*sin(b3)*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (th1d^2*(-2*Ixz*sin(th2)^2 + sin(2*th2)*((Ixx
- Izz)*cos(th3) + Ixy*sin(th3)) + 2*cos(th2)^2*cos(th3)*(Izx*cos(th3) - Izy*sin(th3))))/2 -
th1d*(cth1*sin(th3) + th3d*(2*Ixz*sin(th2) + cos(th2)*((-Ixx + Iyy + Izz)*cos(th3) - (Ixy +
Iyx)*sin(th3))) + th2d*(sin(th2)*((Ixy - Iyx)*cos(th3) - (Ixx + Iyy - Izz)*sin(th3)) +
cos(th2)*(Iyz - Izy*cos(2*th3) - Izx*sin(2*th3)))) - 2*m1*r1^2*th2d*(b1d + th3d)*cos(b1)*sin(b1 +
th3) - 2*m2*r2^2*th2d*th3d*cos(b2)*sin(b2 + th3) + d2*g*m2*cos(th2)*sin(b2)*sin(b2 + th3) +
2*d2^2*m2*th1d*th2d*cos(b2)*sin(th2)*sin(b2 + th3) + (d2^2*m2*th1d^2*sin(b2)*sin(2*th2)*sin(b2 +
```

```
th3))/2 + (d2*m2*r2*(-th2d^2 + th1d^2*cos(th2)^2)*sin(b2)*sin(2*(b2 + th3)))/2 -
(d2*m2*r2*cos(b2)*(4*(th3d^2 + th2d^2*cos(b2 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) +
2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b2 + th3)))))/4 -
2*m3*r3^2*th2d*(b3d + th3d)*cos(b3)*sin(b3 + th3) - 2*m4*r4^2*th2d*th3d*cos(b4)*sin(b4 + th3) +
d4*g*m4*cos(th2)*sin(b4)*sin(b4 + th3) + 2*d4^2*m4*th1d*th2d*cos(b4)*sin(th2)*sin(b4 + th3) +
(d4^2*m4*th1d^2*sin(b4)*sin(2*th2)*sin(b4 + th3))/2 + (d4*m4*r4*(-th2d^2 +
th1d^2*cos(th2)^2)*sin(b4)*sin(2*(b4 + th3)))/2 - (d4*m4*r4*cos(b4)*(4*(th3d^2 + th2d^2*cos(b4 +
th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b4 + th3)))))/4;
f13(6) = -(muc*(b1d*r1^2 + b2d*r2^2 + b3d*r3^2 + b4d*r4^2)) + kth3*th3 + cth3*th3d - (-Ixz +
Izx)*th2d*th3d*cos(th3) + (th2d^2*(Ixy - Iyx + (Ixy + Iyx)*cos(2*th3)))/2 +
2*m2*r2^2*th1d*th2d*cos(th2)*cos(b2 + th3)^2 + 2*m4*r4^2*th1d*th2d*cos(th2)*cos(b4 + th3)^2 -
(b1d^2*m1*muk*r1^2*cos(b1)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1^2*th3d*(2*b1d + th3d)*cos(b1)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(g*m1*muk*r1*cos(th2)*cos(th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1^2*th2d^2*cos(th3)*cos(b1 + th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(m1*muk*r1^2*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sign(b1d))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(b1d*muc*muk*r1^2*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(b3d^2*m3*muk*r3^2*cos(b3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3^2*th3d*(2*b3d + th3d)*cos(b3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(g*m3*muk*r3*cos(th2)*cos(th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3^2*th2d^2*cos(th3)*cos(b3 + th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(m3*muk*r3^2*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sign(b3d))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(b3d*muc*muk*r3^2*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
2*d2*m2*r2*th1d*th2d*cos(b2 + th3)*sin(th2) + 2*d4*m4*r4*th1d*th2d*cos(b4 + th3)*sin(th2) +
(d1*m1*muk*r1*th1d^2*cos(th3)*sign(b1d)*sin(2*th2))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(d3*m3*muk*r3*th1d^2*cos(th3)*sign(b3d)*sin(2*th2))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) - (Iyz
- Izy)*th2d*th3d*sin(th3) + (2*d1*m1*muk*r1*th1d*th2d*sign(b1d)*sin(th2)*sin(th3))/(cos(b1) -
muk*sign(b1d)*sin(b1)) + (2*d3*m3*muk*r3*th1d*th2d*sign(b3d)*sin(th2)*sin(th3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (2*m1*muk*r1^2*th1d*sign(b1d)*((b1d + th3d)*cos(b1)*sin(th2) -
th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*m3*muk*r3^2*th1d*sign(b3d)*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) - ((Ixx - Iyy)*th2d^2*sin(2*th3))/2 -
(th1d^2*(sin(2*th2)*(Iyz*cos(th3) + Ixz*sin(th3)) + cos(th2)^2*(-Ixy + Iyx + (Ixy +
Iyx)*cos(2*th3) + (-Ixx + Iyy)*sin(2*th3))))/2 - th1d*(th3d*cos(th2)*((Iyz - Izy)*cos(th3) + (Ixz
- Izx)*sin(th3)) + th2d*(sin(th2)*((-Ixz - Izx)*cos(th3) + (Iyz + Izy)*sin(th3)) + cos(th2)*(-Izz
+ (Ixx - Iyy)*cos(2*th3) + (Ixy + Iyx)*sin(2*th3)))) - g*m2*r2*cos(th2)*sin(b2 + th3) -
(d2*m2*r2*th1d^2*sin(2*th2)*sin(b2 + th3))/2 - (m2*r2^2*(-th2d^2 + th1d^2*cos(th2)^2)*sin(2*(b2 +
th3)))/2 - g*m4*r4*cos(th2)*sin(b4 + th3) - (d4*m4*r4*th1d^2*sin(2*th2)*sin(b4 + th3))/2 -
(m4*r4^2*(-th2d^2 + th1d^2*cos(th2)^2)*sin(2*(b4 + th3)))/2;
f13(7) = b1d*muc*r1*cos(b1) + 2*m1*r1*th1d*th2d*cos(th2)*cos(th3)*cos(b1 + th3) +
(b1d^2*m1*muk*r1*cos(b1)^2*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (m1*muk*r1*th3d*(2*b1d
+ th3d)*cos(b1)^2*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(g*m1*muk*cos(b1)*cos(th2)*cos(th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*th2d^2*cos(b1)*cos(th3)*cos(b1 + th3)*sign(b1d))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*muk*r1*th1d^2*cos(b1)*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sign(b1d))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) + (b1d^2*m1*r1*cos(b1)*sin(b1))/(cos(b1)
- muk*sign(b1d)*sin(b1)) + (m1*r1*th3d*(2*b1d + th3d)*cos(b1)*sin(b1))/(cos(b1) -
muk*sign(b1d)*sin(b1)) - (g*m1*cos(th2)*cos(th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*r1*th2d^2*cos(th3)*cos(b1 + th3)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(m1*r1*th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) - cos(b1 +
2*th3))*sin(b1))/(4*(cos(b1) - muk*sign(b1d)*sin(b1))) +
(b1d*muc*muk*r1*cos(b1)*sign(b1d)*sin(b1))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(b1d*muc*r1*sin(b1)^2)/(cos(b1) - muk*sign(b1d)*sin(b1)) + 2*d1*m1*th1d*th2d*cos(th3)*sin(th2) -
m1*r1*sin(b1)*(b1d + th3d + th1d*sin(th2))^2 -
(d1*m1*muk*th1d^2*cos(b1)*cos(th3)*sign(b1d)*sin(2*th2))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) -
(d1*m1*th1d^2*cos(th3)*sin(b1)*sin(2*th2))/(2*(cos(b1) - muk*sign(b1d)*sin(b1))) -
g*m1*cos(th2)*sin(th3) + m1*r1*th2d^2*cos(b1 + th3)*sin(th3) -
d1*m1*th1d^2*cos(th2)*sin(th2)*sin(th3) -
(2*d1*m1*muk*th1d*th2d*cos(b1)*sign(b1d)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) -
(2*d1*m1*th1d*th2d*sin(b1)*sin(th2)*sin(th3))/(cos(b1) - muk*sign(b1d)*sin(b1)) +
(2*m1*muk*r1*th1d*cos(b1)*sign(b1d)*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1)) + (2*m1*r1*th1d*sin(b1)*((b1d +
th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 + th3)*sin(th3)))/(cos(b1) - muk*sign(b1d)*sin(b1))
- m1*r1*th1d^2*cos(th2)^2*cos(th3)*sin(b1 + th3);
f13(8) = b3d*muc*r3*cos(b3) + 2*m3*r3*th1d*th2d*cos(th2)*cos(th3)*cos(b3 + th3) +
(b3d^2*m3*muk*r3*cos(b3)^2*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (m3*muk*r3*th3d*(2*b3d
+ th3d)*cos(b3)^2*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(g*m3*muk*cos(b3)*cos(th2)*cos(th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
```

```
(m3*muk*r3*th2d^2*cos(b3)*cos(th3)*cos(b3 + th3)*sign(b3d))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*muk*r3*th1d^2*cos(b3)*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sign(b3d))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) + (b3d^2*m3*r3*cos(b3)*sin(b3))/(cos(b3)
- muk*sign(b3d)*sin(b3)) + (m3*r3*th3d*(2*b3d + th3d)*cos(b3)*sin(b3))/(cos(b3) -
muk*sign(b3d)*sin(b3)) - (g*m3*cos(th2)*cos(th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*r3*th2d^2*cos(th3)*cos(b3 + th3)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(m3*r3*th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) - cos(b3 +
2*th3))*sin(b3))/(4*(cos(b3) - muk*sign(b3d)*sin(b3))) +
(b3d*muc*muk*r3*cos(b3)*sign(b3d)*sin(b3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(b3d*muc*r3*sin(b3)^2)/(cos(b3) - muk*sign(b3d)*sin(b3)) + 2*d3*m3*th1d*th2d*cos(th3)*sin(th2) -
m3*r3*sin(b3)*(b3d + th3d + th1d*sin(th2))^2 -
(d3*m3*muk*th1d^2*cos(b3)*cos(th3)*sign(b3d)*sin(2*th2))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) -
(d3*m3*th1d^2*cos(th3)*sin(b3)*sin(2*th2))/(2*(cos(b3) - muk*sign(b3d)*sin(b3))) -
g*m3*cos(th2)*sin(th3) + m3*r3*th2d^2*cos(b3 + th3)*sin(th3) -
d3*m3*th1d^2*cos(th2)*sin(th2)*sin(th3) -
(2*d3*m3*muk*th1d*th2d*cos(b3)*sign(b3d)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) -
(2*d3*m3*th1d*th2d*sin(b3)*sin(th2)*sin(th3))/(cos(b3) - muk*sign(b3d)*sin(b3)) +
(2*m3*muk*r3*th1d*cos(b3)*sign(b3d)*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3)) + (2*m3*r3*th1d*sin(b3)*((b3d +
th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 + th3)*sin(th3)))/(cos(b3) - muk*sign(b3d)*sin(b3))
- m3*r3*th1d^2*cos(th2)^2*cos(th3)*sin(b3 + th3);

%Form the Equations of Motion for the Rigid Wheel-Imbalance System
doubledots = MM13\f13;

%Define the Output Vector of Slopes
dY = [Y(11:20) doubledots(1:7)' 0 doubledots(8) 0];
```

APPENDIX D: SAMPLE SOURCE CODE FOR INTERACTION FORCE CALCULATIONS

Similar to Appendix C, Appendix D will only list the source code for one interaction force calculation. The interaction force calculation is used to determine whether the balancing particles stay at rest or continue moving as defined by the logic assumptions made in the primary source code. Case 7 is shown below and features particles 1 and 3 in motion, subjected to kinetic friction forces, while particles 2 and 4 are stationary and subjected to static friction forces provided the interaction force components returned by the fuction remain below the maximum allowable static friction force at the current time step.

Case 7: Particles 1 and 3 Move Relative to the Cylinder

```
function F13 = Fint13(t,Y,dY)

% File: Fint13.m
% Author: Jeff Bolton
% Creation Date: October 18, 2009

% Description: This function-m file solves for the interaction force
% components between a six degree-of-freedom wheel and four marbles
% constrained to move along fixed races on the inner surface of the wheel.
%
% In this function, marbles 1 and 3 slip.
%
% Y = [x y z th1 th2 th3 b1 b2 b3 b4 xd yd zd th1d th2d th3d b1d b2d b3d b4d]
% dY = [xd yd zd th1d th2d th3d b1d b2d b3d b4d xdd ydd zdd th1dd th2dd th3dd b1dd b2dd b3dd
b4dd]

global g M m m1 m2 m3 m4 r ri ro r1 r2 r3 r4 d dm d1 d2 d3 d4 bm kx ky kz kth1 kth2 kth3 cx cy cz
cth1 cth2 cth3 w3 muc muk Ixx Iyy Izz Ixy Iyx Ixz Izx Iyz Izy

% Define the equation variables in terms of the input vectors Y and dY.
x = Y(1);
y = Y(2);
z = Y(3);
th1 = Y(4);
th2 = Y(5);
th3 = Y(6);
b1 = Y(7);
b2 = Y(8);
b3 = Y(9);
b4 = Y(10);
xd = Y(11);
yd = Y(12);
zd = Y(13);
th1d = Y(14);
th2d = Y(15);
th3d = Y(16);
b1d = Y(17);
b2d = Y(18);
b3d = Y(19);
b4d = Y(20);
xdd = dY(11);
ydd = dY(12);
zdd = dY(13);
th1dd = dY(14);
```

```
th2dd = dY(15);
th3dd = dY(16);
b1dd = dY(17);
b2dd = dY(18);
b3dd = dY(19);
b4dd = dY(20);

F13 = zeros(1,12);  %Initialize the interaction force vector

%Populate the Various Elements of the Force Vector F13(t,y,y')
F13(1) = (4*b1d*muc*r1*sin(b1) + m1*(4*b1d^2*r1*cos(b1) - 4*g*cos(th2)*cos(th3) -
4*cos(th3)*(xdd*cos(th2) + (-(zdd*cos(th1)) + ydd*sin(th1))*sin(th2)) - 4*(ydd*cos(th1) +
zdd*sin(th1))*sin(th3) - 2*d1*(cos(th3)*(2*th2dd + th1d^2*sin(2*th2)) - 2*(th1dd*cos(th2) -
2*th1d*th2d*sin(th2))*sin(th3)) + r1*(th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) -
cos(b1 + 2*th3)) + 4*(th3d*(2*b1d + th3d)*cos(b1) + th2d^2*cos(th3)*cos(b1 + th3) + sin(b1)*(b1dd
+ th3dd + th1dd*sin(th2))) + 8*th1d*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3)))))/(4*(cos(b1) - muk*sign(b1d)*sin(b1)));
F13(2) = m1*(zdd*cos(th1)*cos(th2) - d1*(th2d^2 + th1d^2*cos(th2)^2) - ydd*cos(th2)*sin(th1) + (g
+ xdd)*sin(th2) + r1*(cos(b1 + th3)*(-th2dd + th1d*cos(th2)*(2*(b1d + th3d) + th1d*sin(th2))) +
(2*th2d*(b1d + th3d) + th1dd*cos(th2))*sin(b1 + th3)));
F13(3) = (muk*sign(b1d)*(4*b1d*muc*r1*sin(b1) + m1*(4*b1d^2*r1*cos(b1) - 4*g*cos(th2)*cos(th3) -
4*cos(th3)*(xdd*cos(th2) + (-(zdd*cos(th1)) + ydd*sin(th1))*sin(th2)) - 4*(ydd*cos(th1) +
zdd*sin(th1))*sin(th3) - 2*d1*(cos(th3)*(2*th2dd + th1d^2*sin(2*th2)) - 2*(th1dd*cos(th2) -
2*th1d*th2d*sin(th2))*sin(th3)) + r1*(th1d^2*(3*cos(b1) - 2*cos(2*th2)*cos(th3)*cos(b1 + th3) -
cos(b1 + 2*th3)) + 4*(th3d*(2*b1d + th3d)*cos(b1) + th2d^2*cos(th3)*cos(b1 + th3) + sin(b1)*(b1dd
+ th3dd + th1dd*sin(th2))) + 8*th1d*((b1d + th3d)*cos(b1)*sin(th2) - th2d*cos(th2)*cos(b1 +
th3)*sin(th3))))))/(4*(cos(b1) - muk*sign(b1d)*sin(b1)));
F13(4) = (m2*(-4*cos(b2 + th3)*((g + xdd)*cos(th2) + (-(zdd*cos(th1)) + ydd*sin(th1))*sin(th2)) -
4*(ydd*cos(th1) + zdd*sin(th1))*sin(b2 + th3) - 2*d2*(cos(b2 + th3)*(2*th2dd + th1d^2*sin(2*th2))
- 2*(th1dd*cos(th2) - 2*th1d*th2d*sin(th2))*sin(b2 + th3)) + r2*(4*(th3d^2 + th2d^2*cos(b2 +
th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b2 + th3))) - 8*th3d*sin(th2) +
4*th2d*cos(th2)*sin(2*(b2 + th3))))))/4;
F13(5) = -(m2*(-(zdd*cos(th1)*cos(th2)) + d2*(th2d^2 + th1d^2*cos(th2)^2) + ydd*cos(th2)*sin(th1)
- (g + xdd)*sin(th2) - r2*(cos(b2 + th3)*(-th2dd + th1d*cos(th2)*(2*th3d + th1d*sin(th2))) +
(2*th2d*th3d + th1dd*cos(th2))*sin(b2 + th3))));
F13(6) = (m2*(2*(g + xdd)*cos(th2)*sin(b2 + th3) - 2*zdd*(cos(b2 + th3)*sin(th1) +
cos(th1)*sin(th2)*sin(b2 + th3)) + ydd*(-2*cos(th1)*cos(b2 + th3) + 2*sin(th1)*sin(th2)*sin(b2 +
th3)) + d2*(2*cos(b2 + th3)*(th1dd*cos(th2) - 2*th1d*th2d*sin(th2)) + (2*th2dd +
th1d^2*sin(2*th2))*sin(b2 + th3)) + r2*(-2*(th3dd + 2*th1d*th2d*cos(th2)*cos(b2 + th3)^2 +
th1dd*sin(th2)) + (-th2d^2 + th1d^2*cos(th2)^2)*sin(2*(b2 + th3)))))/2;
F13(7) = (4*b3d*muc*r3*sin(b3) + m3*(4*b3d^2*r3*cos(b3) - 4*g*cos(th2)*cos(th3) -
4*cos(th3)*(xdd*cos(th2) + (-(zdd*cos(th1)) + ydd*sin(th1))*sin(th2)) - 4*(ydd*cos(th1) +
zdd*sin(th1))*sin(th3) - 2*d3*(cos(th3)*(2*th2dd + th1d^2*sin(2*th2)) - 2*(th1dd*cos(th2) -
2*th1d*th2d*sin(th2))*sin(th3)) + r3*(th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) -
cos(b3 + 2*th3)) + 4*(th3d*(2*b3d + th3d)*cos(b3) + th2d^2*cos(th3)*cos(b3 + th3) + sin(b3)*(b3dd
+ th3dd + th1dd*sin(th2))) + 8*th1d*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3)))))/(4*(cos(b3) - muk*sign(b3d)*sin(b3)));
F13(8) = m3*(zdd*cos(th1)*cos(th2) - d3*(th2d^2 + th1d^2*cos(th2)^2) - ydd*cos(th2)*sin(th1) + (g
+ xdd)*sin(th2) + r3*(cos(b3 + th3)*(-th2dd + th1d*cos(th2)*(2*(b3d + th3d) + th1d*sin(th2))) +
(2*th2d*(b3d + th3d) + th1dd*cos(th2))*sin(b3 + th3)));
F13(9) = (muk*sign(b3d)*(4*b3d*muc*r3*sin(b3) + m3*(4*b3d^2*r3*cos(b3) - 4*g*cos(th2)*cos(th3) -
4*cos(th3)*(xdd*cos(th2) + (-(zdd*cos(th1)) + ydd*sin(th1))*sin(th2)) - 4*(ydd*cos(th1) +
zdd*sin(th1))*sin(th3) - 2*d3*(cos(th3)*(2*th2dd + th1d^2*sin(2*th2)) - 2*(th1dd*cos(th2) -
2*th1d*th2d*sin(th2))*sin(th3)) + r3*(th1d^2*(3*cos(b3) - 2*cos(2*th2)*cos(th3)*cos(b3 + th3) -
cos(b3 + 2*th3)) + 4*(th3d*(2*b3d + th3d)*cos(b3) + th2d^2*cos(th3)*cos(b3 + th3) + sin(b3)*(b3dd
+ th3dd + th1dd*sin(th2))) + 8*th1d*((b3d + th3d)*cos(b3)*sin(th2) - th2d*cos(th2)*cos(b3 +
th3)*sin(th3))))))/(4*(cos(b3) - muk*sign(b3d)*sin(b3)));
F13(10) = (m4*(-4*cos(b4 + th3)*((g + xdd)*cos(th2) + (-(zdd*cos(th1)) + ydd*sin(th1))*sin(th2))
- 4*(ydd*cos(th1) + zdd*sin(th1))*sin(b4 + th3) - 2*d4*(cos(b4 + th3)*(2*th2dd +
th1d^2*sin(2*th2)) - 2*(th1dd*cos(th2) - 2*th1d*th2d*sin(th2))*sin(b4 + th3)) + r4*(4*(th3d^2 +
th2d^2*cos(b4 + th3)^2) - th1d*(th1d*(-3 + cos(2*th2) + 2*cos(th2)^2*cos(2*(b4 + th3))) -
8*th3d*sin(th2) + 4*th2d*cos(th2)*sin(2*(b4 + th3))))))/4;
F13(11) = -(m4*(-(zdd*cos(th1)*cos(th2)) + d4*(th2d^2 + th1d^2*cos(th2)^2) +
ydd*cos(th2)*sin(th1) - (g + xdd)*sin(th2) - r4*(cos(b4 + th3)*(-th2dd + th1d*cos(th2)*(2*th3d +
th1d*sin(th2))) + (2*th2d*th3d + th1dd*cos(th2))*sin(b4 + th3))));
F13(12) = (m4*(2*(g + xdd)*cos(th2)*sin(b4 + th3) - 2*zdd*(cos(b4 + th3)*sin(th1) +
cos(th1)*sin(th2)*sin(b4 + th3)) + ydd*(-2*cos(th1)*cos(b4 + th3) + 2*sin(th1)*sin(th2)*sin(b4 +
th3)) + d4*(2*cos(b4 + th3)*(th1dd*cos(th2) - 2*th1d*th2d*sin(th2)) + (2*th2dd +
th1d^2*sin(2*th2))*sin(b4 + th3)) + r4*(-2*(th3dd + 2*th1d*th2d*cos(th2)*cos(b4 + th3)^2 +
th1dd*sin(th2)) + (-th2d^2 + th1d^2*cos(th2)^2)*sin(2*(b4 + th3)))))/2;
```