Model to Evaluate the Aerodynamic Energy Requirements of Active Materials in Morphing Wings

Gregory W. Pettit

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

Master of Science
in
Mechanical Engineering

Harry H. Robertshaw, Chair
Daniel J. Inman
Rakesh K. Kapania

December, 2001
Blacksburg, Virginia

Keywords: Morphing, Wings, Energy, Control, and Smart Materials

Abstract

Model to Evaluate the Aerodynamic Energy Requirements of Active Materials in Morphing Wings

Gregory W. Pettit

A computational model is presented which predicts the force, stroke, and energy needed to overcome aerodynamic loads encountered by morphing wings during aircraft maneuvers. This low-cost model generates wing section shapes needed to follow a desired flight path, computes the resulting aerodynamic forces using a unique combination of conformal mapping and the vortex panel method, computes the longitudinal motion of the simulated aircraft, and closes the loop with a zero-error control law. The aerodynamic force prediction method has been verified against two more expensive codes. This overall model will be used to predict the performance of morphing wings and the requirements for the active material actuators in the wings.

Contents

Dedication

This work is dedicated to my wife for her patience and the time we spent apart; and in memory of Dr. Charles E. Knight for it was his inspiration that convinced me to continue my education.

Acknowledgments

I would like to thank the staff, graduate students, and professors at the Center for Intelligent Material, Systems, and Structures (CIMSS) for all the help and support during the meaningful years of my academic experience. The atmosphere, environment, and attitudes found at CIMSS makes for a positive and productive environment. Many thanks to Dr. Frank Gern and Dr. William Mason for the many hours I spent in their office discussing aerodynamics and aircraft configurations. I would also like to thank my committee members Dr. Harry H. Robertshaw, Dr. Daniel J. Inman, and Dr. Rakesh K. Kapania.

This work would have not been possible if it wasn't for the support of the Air Force Office of Scientific Research (AFOSR) under grant number F49620-99-1-0294. We especially appreciate the advise of our contract monitor Brian Sanders.

List of Figures

Chapter 1

Introduction and Literature Review

1.1 Motivation

This work addresses the energy needs for active materials to be used to morph, to change the shape of, wings envisioned for future aircraft. Eliminating discrete control surfaces on airfoils in favor of subtly reshaping the airfoil to perform maneuvers is an attractive idea. This idea was proven beneficial when Scherer *et. al.* performed a series of wind tunnel test. In these tests they showed using shape memory alloy (SMA) actuators for twist, flap, and aileron deflections an improved roll and lift over conventional designs was found to be 17 and 15 percent, respectively [L. B. Scherer, et. al., 1999]. This increase in rolling performance was also shown in the work by Inman *et. al.*, where they showed the benefits of controlling the pressure distribution, Figure 1.1 [D. J. Inman, et. al., 2000]. They showed that when a flap was deflected, the suction peak was concentrated to one area aft of the elastic axis of the wing, Figure 1.1a. By subtly reshaping the airfoil, this pressure distribution could be smoothly placed over a region closer to this axis, Figure 1.1b. Control of this pressure distribution not only helps in performance, it also allows for a lighter and more flexible structure to be constructed.

Morphing airfoils would also allow for planform variations. These planform variations help to overcome the ever so common engineering trade-off by enabling each flight condition

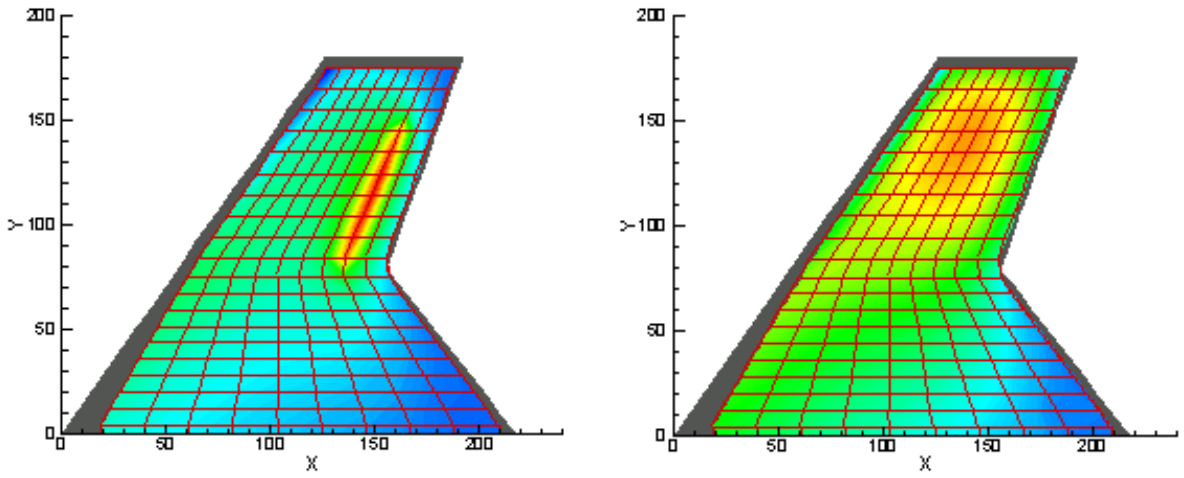


Figure 1.1: Pressure distribution comparing a flap to a morphed surface: (a) pressure peak from the deflection of a flap; and (b) smooth pressure distribution from a morphed surface.

to be optimized by changing configurations for cruise, takeoff, and landing [J. J. Spillman, 1992]. In addition, with a system of distributed actuators and sensors, the overall reliability could be improved. This reliability stems from the redundancy of actuators and the ability to detect damage while simultaneously controlling excessive vibration.

1.2 Previous Approaches

There are several approaches underway to explore methods of morphing. Scott *et. al.* showed that it was possible to maneuver a vehicle by inducing small perturbations on the surface of the wing. This work sounds promising since it does not involve bending or twisting a structure purposely designed rigid for flight. However, this is not the case with most approaches like that of the German Aerospace Center (DLR). Here they proposed a belt rib type construction that was capable of camber changes [F. Campanile, et. al., 2000]. They used a system of ribs connected to an outer belt that would have varying stiffnesses in the

joints. By applying a moment to the structure and varying these stiffnesses they could control the camber.

Another approach, like that evaluated by Khot *et. al.*, consisted of morphing a wing with axial load carrying cross elements in the structure. These cross elements were used as actuators by applying tensile and compressive loads on the wing structure to get the desired twist and camber changes necessary for a roll [N. S. Khot, et. al., 1998]. Yet another approach investigated, looked at changing camber by applying torques for and aft of the wing box structure. Here camber would be changed by bending the least rigid section of the wing. However, like the previous approaches discussed, it too would involve high strain energy.

In parallel to the ongoing investigations to determine a structure capable of morphing, there is a strong interest in developing active materials to perform such actuation. One such group is located within the Materials Division at NASA Langley Research Center. Here, Simpson *et. al.* are investigating innovative materials to conduct such morphing [J. O. Simpson, et. al., 1998]. With the need for large displacements and high load carrying capabilites required by aerospace structures Simpson *et. al.* have been focusing much of there attention to RAINBOW and THUNDER actuators. These actuators are capable of displacements as high as $1.7mm$ with no load decreasing to $1mm$ with a point load of $250g$.

1.3 Inverse Approach

This work is unique in that it takes an inverse approach to the morphing quest. By assuming that shape changes can be produced independent of structure, a tool can be developed that will allow us to predict the force, stroke, and energy required for a distributed system of active material actuators. Then using these requirements, actuators can be chosen and a structure designed.

While there are two important energy considerations, strain energy needed to morph the structure and energy needed to overcome the aerodynamic forces, this work is concerned

with the latter. With this inverse approach, we cannot predict the configuration of the active materials that will make up the morphing wing; therefore, we are not able to predict how much energy will be required to strain the wing structure. We can envision some designs that will have very little strain energy in the structure after morphing. Those designs could have variable length links or members that cannot be back driven by wing forces. We can also envision some designs that will have significant amounts of strain energy in the structure. These designs could have actuators straining the structure of a typical wing box. Therefore, we are addressing the aerodynamic loads only.

The model used for this approach is broken into five major parts: shape generation, aerodynamics, dynamics, control, and power, Figure 1.2. When the five parts are integrated together, the model becomes a versatile tool that allows for the input of a time based desired course with the output being time, power, and stroke required by the actuators to fly the course. The model is unique for three reasons. First, the shape generation allows for a vast array of shapes suitable for flight along with an easy method of changing this shape. Second, the aerodynamics uses the combination of two methods that will allow for the air loads to quickly and accurately be determined for a wing with a finite span. Third, the air loads can be found on the upper and lower surfaces not just on a mean camber line, which aids in the calculation of the energy required to change the shape.

1.4 Thesis Overview

The document devotes a chapter to the theory and assumptions behind each section of the model. Each chapter progressively builds on information developed in previous chapters. In addition, in discussing the individual sections, insight is given to how to extend the model to include more physical aspects of flight. While the code used to simulate the model is not present, a detailed description of how to use the code, available upon request, is presented as an Appendix.

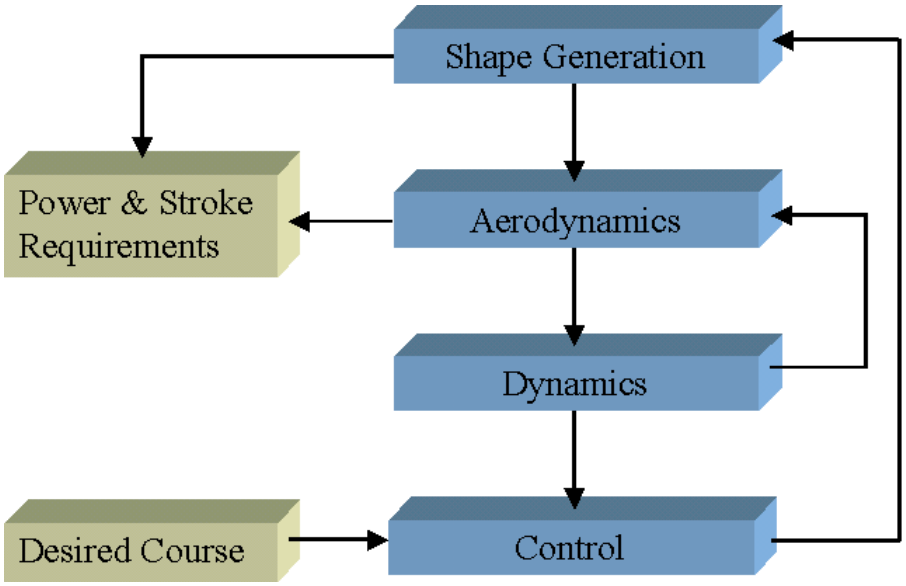


Figure 1.2: Block diagram demonstrating the flow of the model and the organization of this document.

Chapter 2

Formulation of Shape Generation Module

In order to meet the needs of a planform that can quickly and easily change shape, a method of conformal mapping was chosen. This method allows for a unit circle to be transformed into the shape of an airfoil section. While the basic mapping, known as Joukowski's transformation, has been around for many years, the mapping has been expanded to incorporate many NACA airfoils [Jones, 1990]. Since this extended transformation starts from a unit circle, it proves to be very useful when calculating the aerodynamics. Once we have the capability of generating airfoil sections, the method is performed in succession to generate the entire wing. This wing generation can be thought of as a bookshelf. Each book with its own shape and size is analogous to a wing section. Then when all the books are stacked side by side, they create a three-dimensional shape much like a cube. Although books can not fly, when all of the wing sections are stacked side by side, they create a three-dimensional wing.

2.1 Generation of Wing Section

The mapping from a circle to an airfoil section consists of four steps. These steps are done in succession with each step building on the previous. To gain a better understanding of the mapping, it is helpful to visualize each step. A graphical result from each of these steps is shown in Figure 2.1.

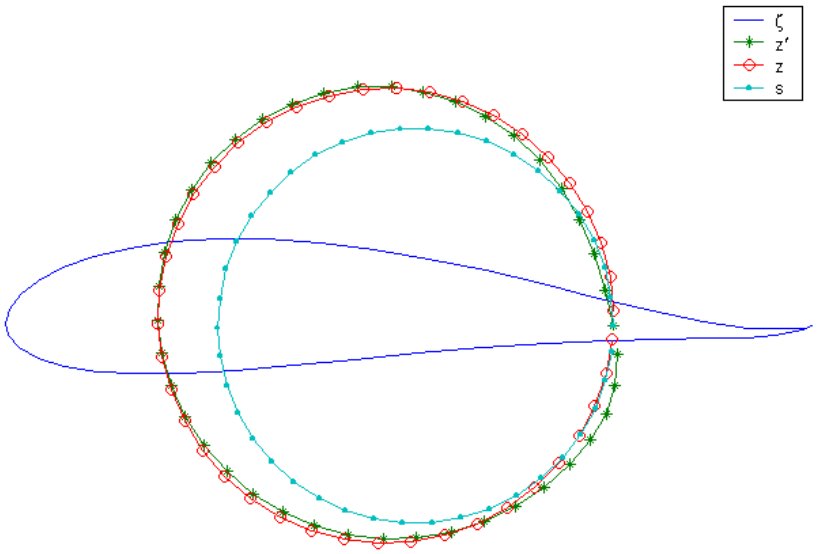


Figure 2.1: The four steps of the conformal mapping approach. Each step is a mapping that builds on the previous map. The mapping yields five shape parameters that enable many NACA airfoils to be generated such as the 6 series, which is suitable for flying wings.

The first step shown in equation 2.1 is to define the unit circle, s, in the complex plane:

$$s = \cos\theta + i\sin\theta \tag{2.1}$$

Second, to map the s plane into the z plane, the center of the unit circle is offset by the amount $x_c + iy_c$, and the point $1 + i0$ of the unit circle is mapped to the point, $x_t + iy_t$. Now that we have two points common to each plane, we can develop the mapping shown by equation 2.2:

$$z = [x_t - x_c + i * (y_t - y_c)]s + x_c + iy_c \tag{2.2}$$

This mapping introduces four shape parameters: x_c, y_c, x_t, and y_t. The effects that these parameters have on the shape of the airfoil along with suitable ranges for these parameters are shown in Table 2.1. Third, the offset circle, z, is mapped into the z' plane as shown in equation 2.3:

$$z' = z + \frac{(-1 + x_t + iy_t)(x_t + iy_t - \Delta)}{z + \Delta} \tag{2.3}$$

This mapping is a slight elongation of the offset circle, z, and introduces another shape parameter, Δ. The effects and suitable ranges for Δ are also shown in Table 2.1. This third step is an extension of Joukowski's transformation and allows for a greater variety of airfoils such as the NACA M6, which has a reflexed trailing edge and is well suited for tailless airplanes, [Jones, 1990]. The fourth and final step is to put the morphed circle through a mapping known as Joukowski's transformation shown in equation 2.4:

$$\zeta = z' + \frac{1}{z'} \tag{2.4}$$

Here, the elongated circle takes the shape of an airfoil section. The coordinates of this airfoil section are complex and denoted by ζ.

Table 2.1: Shape parameters and their effects on an airfoil section.

Parameter	Effect	Range
x_c	Thickness of airfoil	-.2 to 0
y_c	Camber towards leading edge	-.2 to .2
x_t	Thickness towards trailing edge	1 to 1.1
y_t	Camber towards trailing edge	-.1 to .1
Δ	Position of the effects of x_t	0 to .8

The mapping described above allows for an airfoil section to be created with a wide array of possible shapes. In addition, it allows for these shapes to easily be changed by adjusting the five shape parameters. These shape changes become important later when they are used for controlling the wing while in simulated flight.

2.2 Generation of Wing

To create the wing, the mapping technique described in the previous section is done in succession along each spanwise location of the wing, η. These successive transformations yield a set of complex coordinates, ζ, for each spanwise station. To better illustrate this point, Figure 2.2 shows an example of an entire wing composed of the different sections. In addition, this figure shows an airfoil section at the $\eta = -.5$ spanwise station. The real and imaginary values of this section, $\Re[\zeta(-.5)]$ and $\Im[\zeta(-.5)]$, become the $x_\zeta(-.5)$ and $z_\zeta(-.5)$ coordinates of the wing for the location $\eta = -.5$. The $y_\zeta(-.5)$ coordinate is then chosen according to the desired span location of the airfoil section. To achieve the coordinates of the entire wing, this successive mapping is repeated for all desired locations of η.

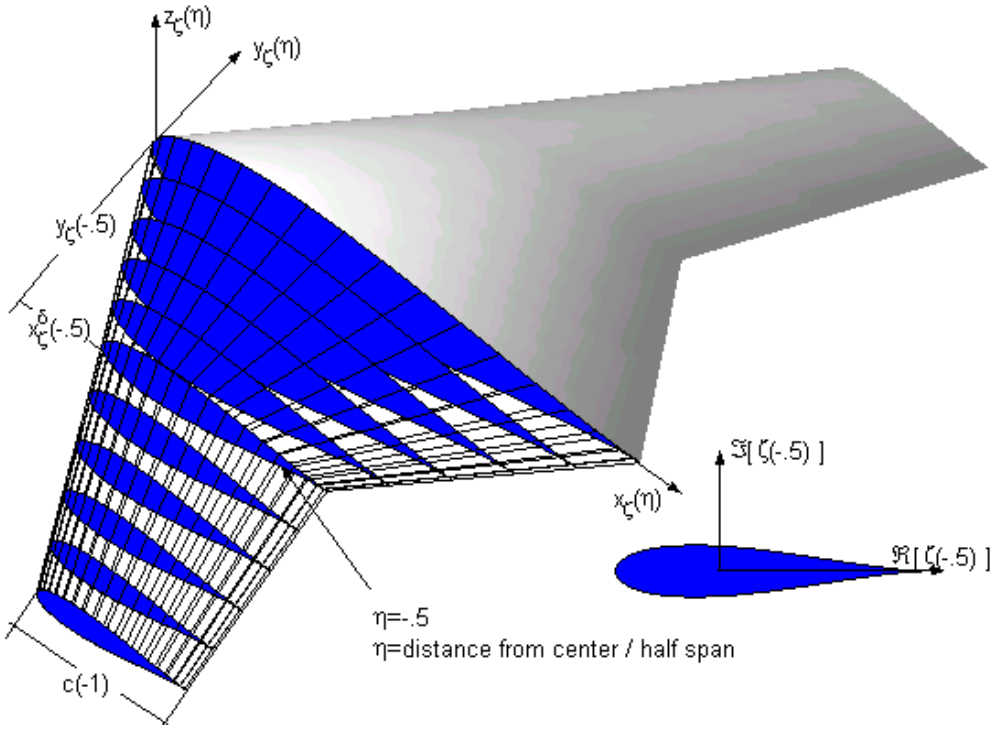


Figure 2.2: Wing generated using conformal mapping at different spanwise locations. Also shown is the airfoil section at the $\eta = -.5$ spanwise station. The entire wing is made up a series of these sections at desired locations of η.

In order to achieve a wing with taper, the chord at each spanwise station, $c(\eta)$, is

chosen. To achieve this chord, the values of $\zeta(\eta)$ are first non-dimensionalized by the length of the existing chord. These values of $\zeta(\eta)$ are then scaled to the desired chord. The wing can then be given sweep and dihedral by offsetting the values of $x_\zeta(\eta)$ and $z_\zeta(\eta)$ by $x_\zeta^\delta(\eta)$ and $z_\zeta^\delta(\eta)$, respectively. Another desirable trait of a wing is to have twist. This twist is obtained by rotating the values of $\zeta(\eta)$ by θ_t as shown in equation 2.5:

$$\zeta(\eta) = \zeta e^{-i\theta_t(\eta)} \tag{2.5}$$

These steps, which allow for much flexibility when generating wings with taper, sweep, and dihedral, are shown in Table 2.2

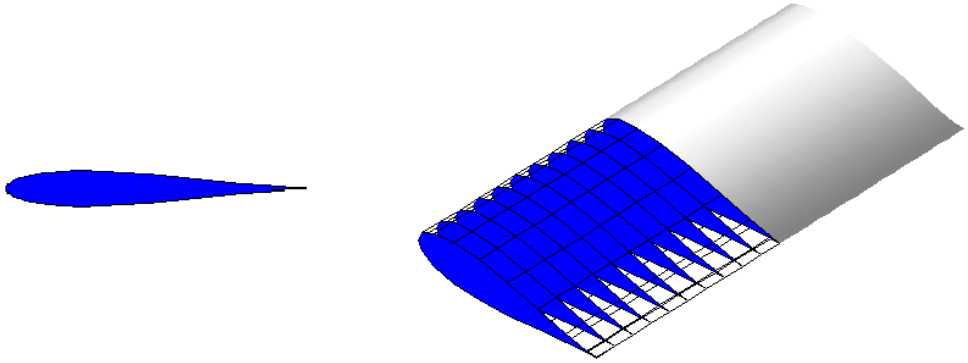
Table 2.2: Shape parameters and their effects on wing.

Parameter	Effect
η	Spanwise location
$y_\zeta(\eta)$	y coordinates of wing as a function of span
$x_\zeta(\eta)$	x coordinates of wing as a function of span
$z_\zeta(\eta)$	z coordinates of wing as a function of span
$x_\zeta^\delta(\eta)$	Sweep of wing as a function of span
$z_\zeta^\delta(\eta)$	Dihedral of wing as a function of span
$\theta_t(\eta)$	Twist of wing as a function of span
$c(\eta)$	Taper of wing as a function of span

2.3 Generation of Wing Example

This section is devoted to demonstrating the flexibility of the wing generation process described in the previous sections. The demonstration will consist of an example that begins with a wing section and transforms, through a series of steps, into the wing that will be used for the remainder of this document.

To begin, a generic wing section is created. For demonstration purposes this wing section will be symmetric with no camber or twist as shown in Figure 2.3a. This wing section is expanded on later in the example. This section is then placed at desired spanwise locations, η, as shown in Figure 2.3b. This process can be thought of as extruding the two-dimensional wing section into a three-dimensional shape.



(a) (b)

Figure 2.3: Wing created from placing wing sections at different spanwise locations: (a) wing section created from conformal mapping; and (b) wing section placed at different spanwise locations.

Next, it is desired for the wing to have a sweep angle of 35 degrees. This sweep angle is achieved by offsetting each of the spanwise locations by the amount x_ζ^δ. The y_ζ coordinates of the wing have been non-dimensionalized by the half span of the wing as shown in equation 2.6:

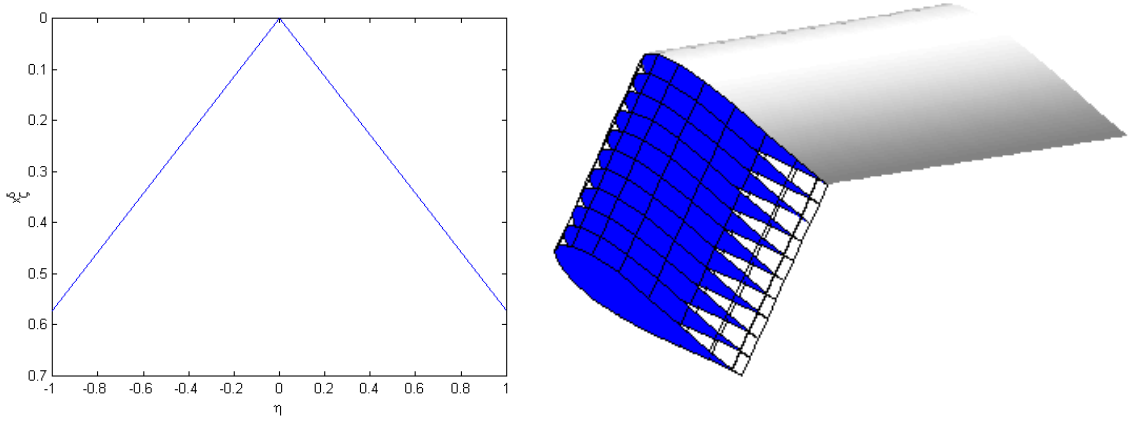
$$\eta = \frac{y_\zeta}{halfspan} \tag{2.6}$$

where η now represents the non-dimensionalized span of the wing and ranges from -1 to 1. Therefore, the desired offset, x_ζ^δ, can be represented by equation 2.7:

$$x_\zeta^\delta(\eta) = \eta * \sin 35 \tag{2.7}$$

Figure 2.4 shows a plot of desired offset and the effects that it has on the wing. Although wings generally have a linear sweep angle, as shown in this example, it is not necessary.

Representing this sweep as a function affords the possibilities of unique designs. This ability to easily change shape parameters makes the shape generation model a powerful tool.

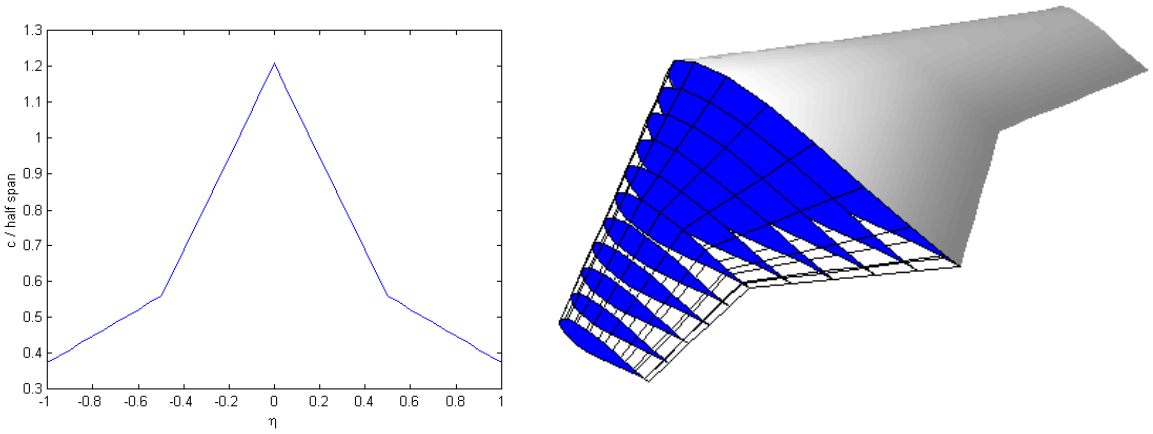


(a) (b)

Figure 2.4: Effects of offsetting the x_ζ coordinates on wing shape: (a) the x_ζ^δ parameter as a function of span; and (b) shifting x_ζ^δ allows for sweep in the wing.

Now that sweep has been added to the wing, we will take it a step further and add taper. This taper can be thought of as making the chord of the wing longer or shorter at different span locations. However, one should note that the wing sections still start from the x_ζ^δ locations. If this were not the case, then the sweep angle would be changed. For this example the chord as a function of η, as well as the results this scaling has on the wing, are shown in Figure 2.5.

The shape that has been created thus far will be used in the remainder of the document as the non-morphed planform. Since the end goal of this model is to determine the energy requirements of active materials in morphing wings, the wings will have to be morphed to maneuver in flight. This morphing is achieved by varying certain shape parameters, discussed in the previous sections. This change in shape parameters will morph the wing causing the aerodynamic loads to vary. Once the equations of motion are applied, this variation will then cause deviations of the flight path, thus enabling the possibilities for



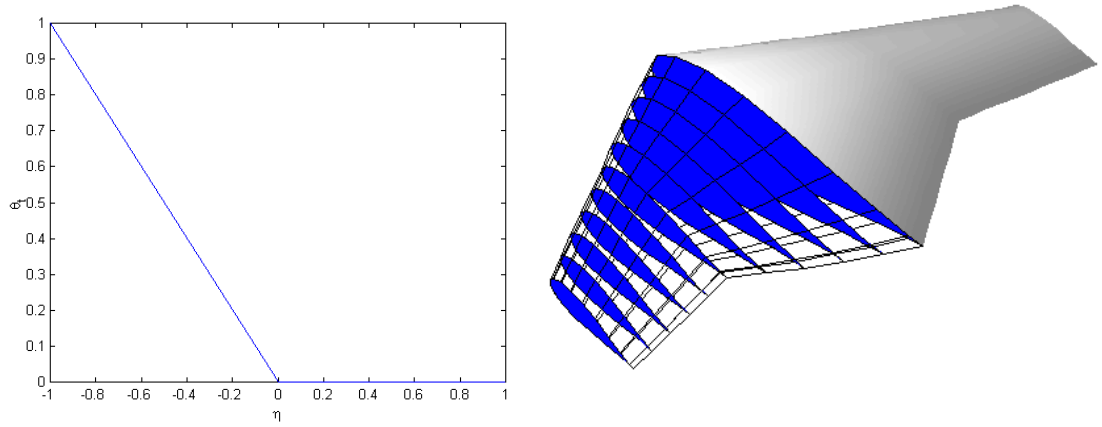
(a) (b)

Figure 2.5: Effects of scaling the chord on wing shape: (a) the nondimensionalized chord, $c/halfspan$, as a function of span; and (b) scaling the chord allows for taper in the wing.

control.

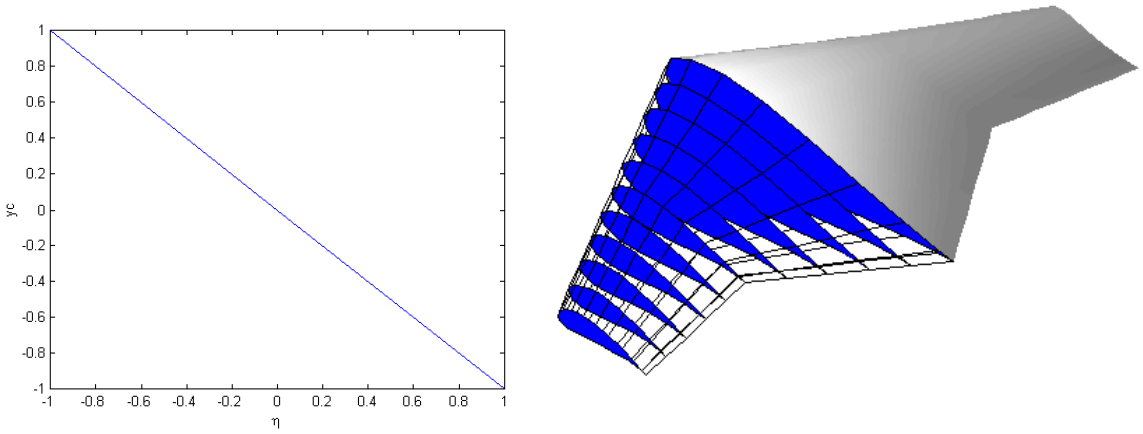
An example of this morphing would be to twist the wing. Setting the twist, θ_t, as a function of η will cause the lift to change on the wing. The example shown in Figure 2.6 shows twist being applied to the left half of the wing. For this case, the lift would increase on the left wing causing a rolling moment about the x_ζ axis.

Another possibility, instead of adding twist, would be to add camber. Like twist, camber is also used to vary the load distribution on the wing. Camber is added to the wing by choosing the shape parameter, y_c, as a function of span, η. One such example is shown in Figure 2.7. In this case, positive camber is added to the left half of the wing, while negative camber is added to the right. This change in camber will add positive lift to the left wing and conversely cause the right wing to lose lift. This change in lift will again cause a rolling moment about the x_ζ axis.



(a) (b)

Figure 2.6: Effects of adding twist on wing shape: (a) the twist, θ_t, as a function of span; and (b) changing the twist allows for the lift distribution to change on the wing.



(a) (b)

Figure 2.7: Effects of adding camber on wing shape: (a) the parameter, y_c, as a function of span; and (b) changing the camber allows for the lift distribution to change on the wing.

While the morphing schemes described above are very simple, when used in parallel, they offer a variety of complex shape changes. These shape changes can be quickly and easily implemented, providing a powerful tool when used in conjunction with the remaining parts of the model, which will be described in the following chapters.

Chapter 3

Formulation of Aerodynamic Module

There are several methods to model the flow around a wing. Typically, each method does one thing well, but lacks in other areas. For example, the vortex lattice method captures the three dimensional flow effects. However, this lattice method is based on thin airfoil theory. This thin airfoil theory means the results are generally based on a mean camber line and not on the outer surface of the wing. Since the end goal of this model is to predict the amount of work necessary to change the shape of the wing, we should look at localized areas on the skin of the wing in which actuators are located. In addition, the vortex lattice method requires large amounts of computation time compared to methods like conformal mapping. Since our model will be used for dynamic analysis, computation time is an important issue. In contrast, conformal mapping techniques do not capture three-dimensional flow effects, but requires very little computation time.

For these reasons we have chosen to use a combination of two methods. First, conformal mapping and potential flow will be used to describe the flow around the outer surface of the wing. Knowing the flow around the outer surface is important because it will allow for the work to be determined at localized areas on the wing surface. However, conformal mapping can not capture the flow effects on a wing with a finite span. Second, a modified version of the vortex lattice method, also based on potential flow, was used. This modified vortex lattice is known as a vortex panel that consists of one vortex positioned at each span-

wise panel of the wing as opposed to a lattice of vortices. The vortex panel method runs quickly compared with the lattice method and allows for the aerodynamics of a wing with a finite span to be calculated. However, this method's shortcoming stems from its ability to predict a load at only one point at each spanwise panel. Third, these methods will be combined to allow for the finite span aerodynamics to be calculated along the entire surface of the wing.

3.1 Aerodynamics of Wing Section Using Conformal Mapping and Potential Flow

From the conservation of mass, we know that the mass of a system must remain constant. If we assume our system to be a fixed control volume, cv, the conservation of mass can be written as:

$$\frac{\partial}{\partial t}\int_{cv}\rho d\vee + \int_{cs}\rho V\cdot ndA = 0 \tag{3.1}$$

where $\frac{\partial}{\partial t}\int_{cv}\rho d\vee$ represents the change in mass of the system, and $\int_{cs}\rho V\cdot ndA$ represents the mass exchange across the surface of the volume. Equation 3.1 is commonly known as the continuity equation and in Cartesian coordinates can be written as

$$\frac{\partial\rho}{\partial t}+\frac{\partial\rho u}{\partial x}+\frac{\partial\rho v}{\partial y}+\frac{\partial\rho w}{\partial z}=0 \tag{3.2}$$

where u, v, and w denote the velocity of the fluid in the x, y, and z directions. Furthermore, if we assume the flow to be steady then density is not a function of time and $\frac{\partial\rho}{\partial t}$ goes to zero [Munson et. al., 1998]. In addition, by assuming the flow is incompressible then the density is constant and the continuity equation can be simplified as shown in equation 3.3:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0 \tag{3.3}$$

Since the flow for the wing section is two dimensional, there will be no flow in the y direction, leaving equation 3.4:

$$\frac{\partial u}{\partial x}+\frac{\partial w}{\partial z}=0 \tag{3.4}$$

By defining a function known as the potential function, ϕ, we can describe the fluid's velocity at any point by differentiating with respect to the desired coordinate, as shown in equation 3.5:

$$\frac{\partial \phi}{\partial x} = u \quad ; \quad \frac{\partial \phi}{\partial z} = w \tag{3.5}$$

This function is called the velocity potential because it describes the potential of the fluid at any point in the flow field. Substitution of this function into the continuity equation yields the well known Laplace's equation shown in equation 3.6:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial z^2} = 0 \tag{3.6}$$

By assuming the fluid particles do not spin, then the fluid is said to be irrotational and can be described by equation 3.7:

$$\frac{\partial u}{\partial z} - \frac{\partial v}{\partial x} = 0 \tag{3.7}$$

We can define a function known as the stream function, ψ, which is perpendicular to the potential function and is tangent to the flow everywhere in the field. The stream function's derivative is related to the fluid velocities as given in equation 3.8:

$$\frac{\partial \psi}{\partial z} = u \quad , \quad \frac{\partial \psi}{\partial x} = -w \tag{3.8}$$

The stream function is helpful in describing the rotation of the fluid flow. When this function is substituted into equation 3.7, we again get Laplace's Equation:

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial z^2} = 0 \tag{3.9}$$

In order to solve for the flow around a wing section, we need the solution to Laplace's equation for the fluid flowing around the given section. Laplace's equation is a homogenous, second-order, partial differential equation with constant coefficients; and there are several techniques, such as separation of variables, available to solve this type of equation. However, one technique, using complex functions, lends itself nicely to solving the flow around a wing section. We can define the complex function, w, as the complex potential function shown in equation 3.10:

$$\mathrm{w} = \phi + i\psi \tag{3.10}$$

In addition, we can assume that the complex potential is differentiable everywhere in the complex plane that is defined by $s = x+iz$. We can now show that this function is a solution to Laplace's equation. Since this function is differentiable over the domain of s, then from Cauchy-Riemann's theorem, the relations shown in equations 3.11 and 3.12 must hold:

$$\frac{\partial \phi}{\partial x} = \frac{\partial \psi}{\partial z} = 0 \tag{3.11}$$

$$\frac{\partial \phi}{\partial z} = -\frac{\partial \psi}{\partial x} = 0 \tag{3.12}$$

Substitution of the relations in equation 3.5 and equation 3.8 shows that these relations do indeed hold. Since the complex potential function is differentiable everywhere, then the complex function's derivative is the same regardless from which direction we differentiate, x or z. Differentiating equation 3.11 with respect to x and differentiating equation 3.12 with respect to z yields the relationships shown in equations 3.13 and 3.14:

$$\frac{\partial^2 \phi}{\partial x^2} = \frac{\partial^2 \psi}{\partial x \partial z} = 0 \tag{3.13}$$

$$\frac{\partial^2 \phi}{\partial z^2} = -\frac{\partial^2 \psi}{\partial x \partial z} = 0 \tag{3.14}$$

Furthermore, since Laplace's equation is a linear differential equation, then the superposition of multiple solutions is also a solution. The summation of equations 3.13 and 3.14 equals Laplace's equation as shown in equation 3.15, thus proving that the complex potential function is indeed a solution to Laplace's Equation:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial z^2} = \frac{\partial^2 \psi}{\partial x \partial z} - \frac{\partial^2 \psi}{\partial x \partial z} = 0 \tag{3.15}$$

Since the complex potential function is a solution to Laplace's Equation we can use this function to describe the flow of simple fluid fields. Then using the principle of superposition, we can add these solutions together to describe the fluid flow of more complex fields. The first complex potential function of concern is the function that represents the flow of a uniform stream with a velocity, V, inclined at an angle, α, as shown in Figure 3.1a and is represented by equation 3.16:

$$\mathrm{w}(s) = Vse^{-i\alpha} \tag{3.16}$$

Two other functions commonly used are sources and sinks. Figure 3.1b shows the effect that a source and sink have on the flow field and can be represented by equation 3.17:

$$\mathrm{w}(s) = \frac{m}{2\pi} ln(s) \tag{3.17}$$

where m is the rate per unit length that the fluid is being created due to a source or destroyed due to a sink. When a source and a sink of equal strength are placed apart at some distance, a, they are referred to as a dipole. Equation 3.18 shows the complex potential function that represents the flow of a dipole inclined at an angle α:

$$\mathrm{w}(s) = \frac{m}{2\pi} ln\left(\frac{s+a}{s-a}\right) e^{-i\alpha} \tag{3.18}$$

where m is the rate per unit length at which the fluid is being created and destroyed and a is the distance between them. If we take the limit as a goes to zero, provided that m goes to infinity, we get a doublet shown in Figure 3.1e as represented by equation 3.19:

$$\mathrm{w}(s) = \frac{2\mu}{2\pi s} e^{-i\alpha} \tag{3.19}$$

where $\mu = 2ma$. Lastly, a vortex with circulation strength Γ is shown in Figure 3.1c and represents the velocity of fluid in a circular path around its center. This velocity is inversely proportional to the distance from its center. The complex potential function representing this vortex is shown in equation 3.20:

$$\mathrm{w}(s) = \frac{i\Gamma}{2\pi} ln(s) \tag{3.20}$$

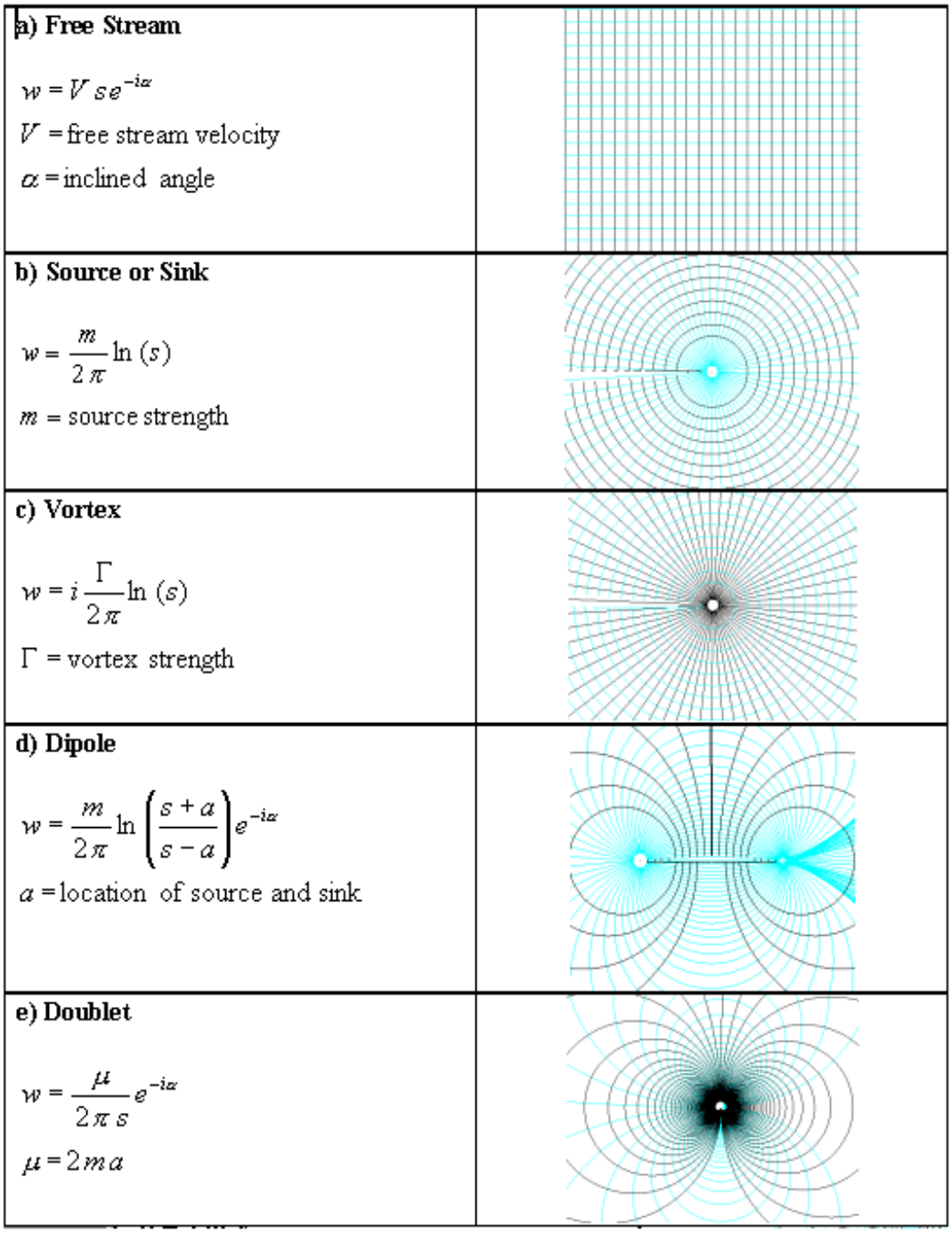


Figure 3.1: Simple flow fields that when summed together can represent a more complex flow: (a) uniform flow inclined at an angle, α, with respect to the x-axis; (b) source or sink, represented by fluid being generated or destroyed at a point; (c) vortex, flow in a circular motion with velocity, V, inversely proportional to the distance from the origin; (d) dipole, source and sink placed apart at a distance $2a$ inclined at an angle α; and (e) doublet, is a dipole as m goes to infinity and a goes to zero;

As stated previously, we can superimpose these complex potential functions to represent complex flow fields. By superimposing a free stream velocity with a doublet, both of which are inclined at an angle α, we can obtain the new potential function as shown in equation 3.21:

$$\mathrm{w}(s) = \left(Vs + \frac{2\mu}{2\pi s}\right) e^{-i\alpha} \tag{3.21}$$

If the doublet strength μ is chosen such that $\mu = r^2 2\pi V$ this potential function can be used to describe the flow around a circle of radius r. Substituting the value for μ and differentiating with respect to s the flow around a circle can be described by equation 3.22:

$$\frac{d\mathrm{w}}{ds} = Ve^{-i\alpha}\left(1 - \frac{r^2}{s^2}\right) \tag{3.22}$$

Figure 3.2 represents an example of the flow around such a circle. For this case, the circle was the unit circle and the free stream velocity was inclined to an angle of 15 degrees. This figure shows the velocity vectors around the circle along with the stream lines. The velocity vectors are equal in magnitude on opposite sides of the circle, which means the pressure is also equal. With the pressure being equal on both sides of the circle, there will be no net force implying the lift is zero.

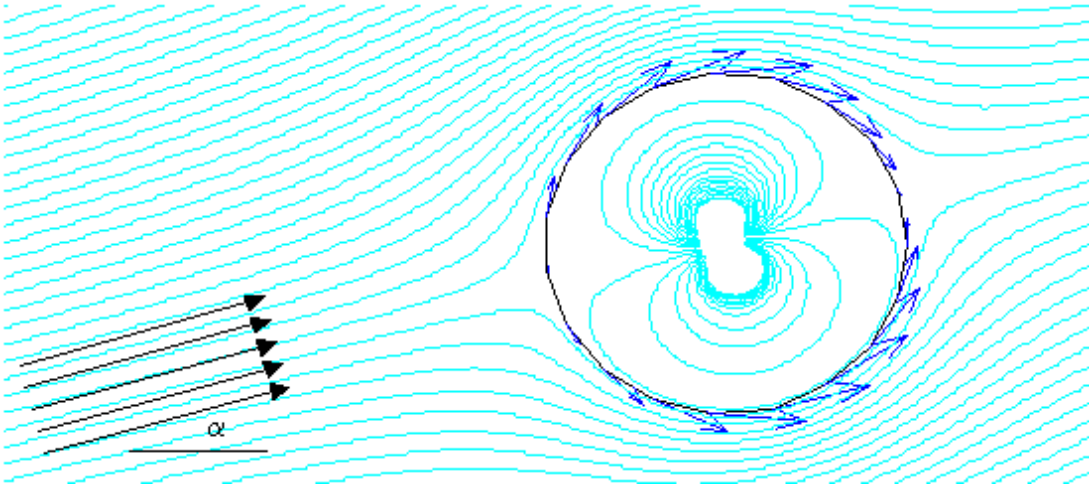


Figure 3.2: Flow around a unit circle, which is represented by the superposition of a uniform stream and a doublet.

Next, by adding a vortex located at the center of the circle, with fluid flowing in a clockwise direction, we can change flow field. The new complex potential function is shown

in equation 3.23:

$$\frac{d\mathrm{w}}{ds} = Ve^{-i\alpha}\left(1 - \frac{r^2}{s^2}\right) + \frac{i\Gamma}{2\pi s} \tag{3.23}$$

The effects of this vortex can be visualized by observing the differences between Figures 3.2 and 3.3. By adding flow tangent to the circle in a clockwise direction, the flow velocity is increased along the top of the circle and decreased along the bottom. Now the velocity tangent to the circle is faster along the top causing a low pressure and thus resulting in a lifting force. However, how much circulation should we add with the vortex?

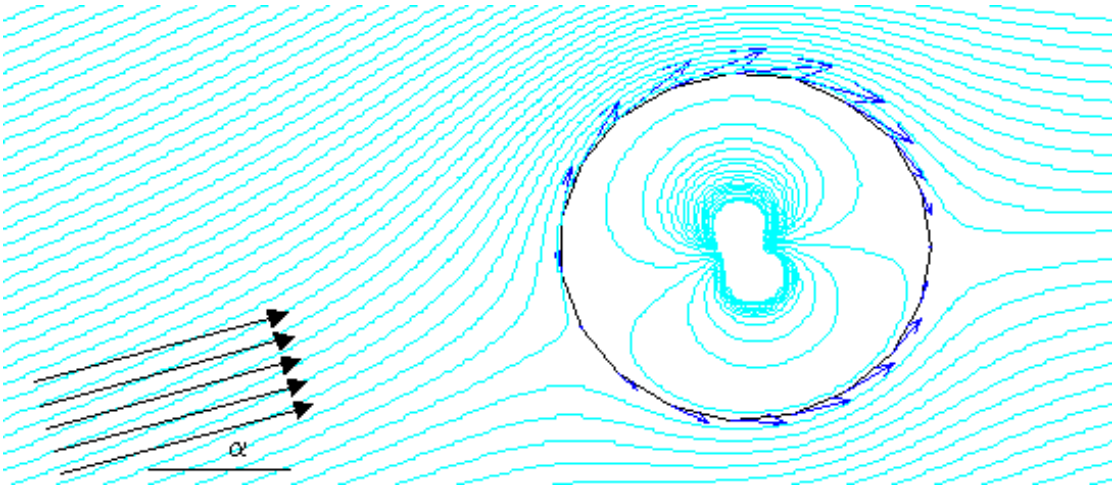


Figure 3.3: Flow around a unit circle with circulation, which is represented by the superposition of a uniform stream, doublet, and a vortex. The uniform stream inclined at an angle α does not appear to line up with the stream lines. This misalignment is from the vortex adding circulation to the flow. However, one should note that at infinity the uniform flow field and the stream lines match.

In the previous chapter on shape generation it was shown that a unit circle could be transformed into an airfoil section. Therefore, if we can calculate the flow around this unit circle, then it is possible to map the flow from the s plane to the ζ plane yielding the flow around an airfoil section as shown in Figure 3.4. This mapping can be achieved since the conformal mapping process left us with three functions: (1) s as a function of z; (2) z as a function of z'; and (3) z' as a function of ζ; In addition from potential flow we have a function relating w as a function of s. Therefore by applying the chain rule as shown in

equation 3.24 we can solve for the flow around the airfoil:

$$\frac{d\mathrm{w}}{d\zeta} = \frac{d\mathrm{w}}{ds}\frac{ds}{dz}\frac{dz}{dz'}\frac{dz'}{d\zeta} \tag{3.24}$$

Furthermore, it was shown that the point at $s = 1 + i0$ on the unit circle transforms into the trailing edge of the wing. In order for the flow to leave the wing smoothly, there must be a stagnation point, zero velocity, at the trailing edge. Therefore, the vortex strength, Γ, can be calculated by substituting the point $s = 1 + i0$ into equation 3.23 and setting it equal to zero. As shown in equation 3.25, Γ is a function of the fluid velocity, V, and angle of attack, α:

$$\Gamma = 4\pi V \sin\alpha \tag{3.25}$$

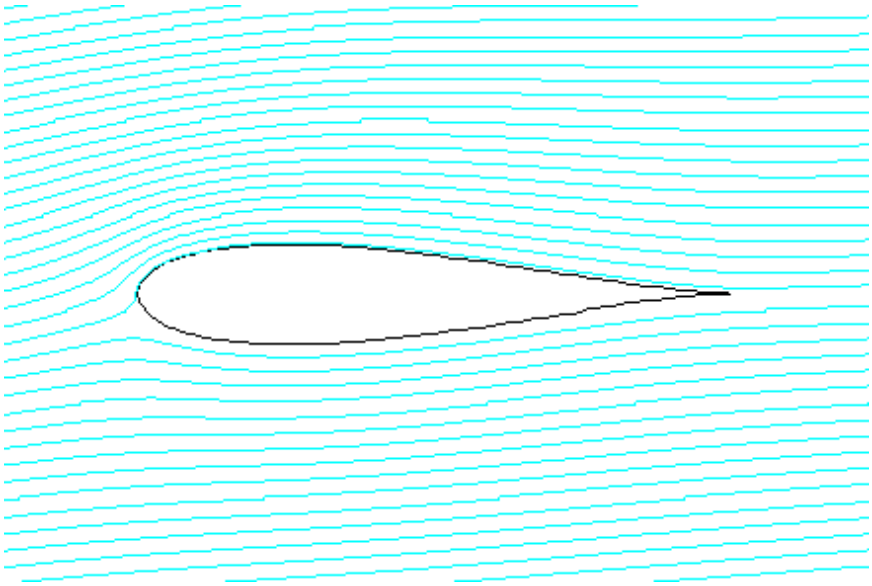


Figure 3.4: Flow around an airfoil section with α equal to 5 degrees.

Once the velocity profile is known surrounding the airfoil section, we can apply Bernoulli's equation to get the pressure distribution. This pressure distribution can be non-dimensionalized by the dynamic pressure allowing for the evaluation of the pressure distribution for a wing section over an entire subsonic flight regime. This non-dimensionalized

pressure is called the coefficient of pressure and is shown in equation 3.26:

$$c_p = 1 - \left(\frac{V}{V_\infty}\right)^2 \tag{3.26}$$

The results of transforming the velocity profile found in Figure 3.4 into a pressure distribution is shown Figure 3.5. The negative pressure coefficients represent a negative pressure, thus resulting in lift.

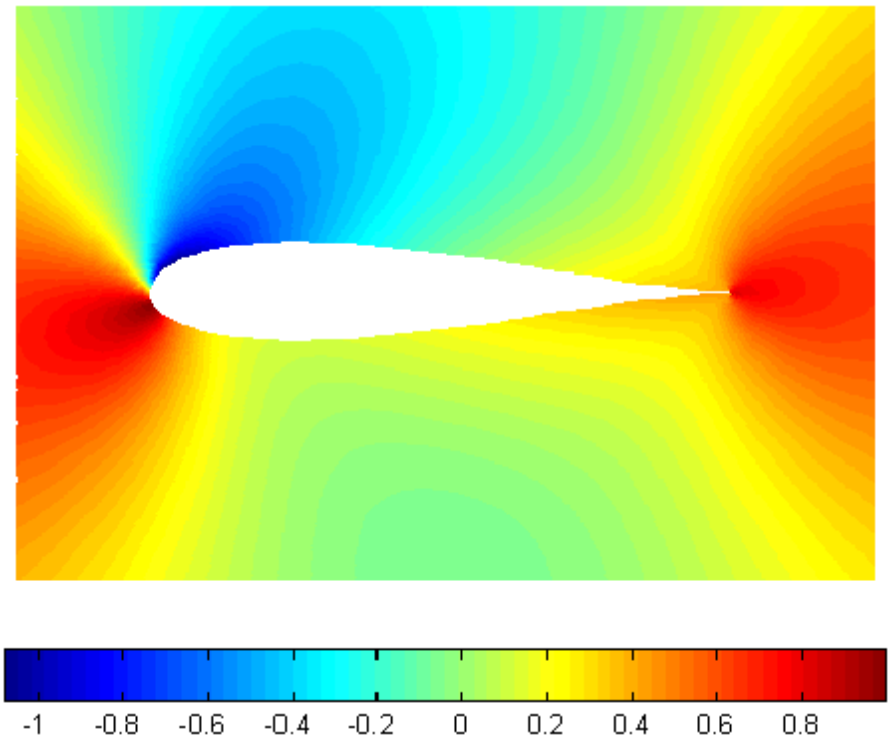


Figure 3.5: Pressure distribution surrounding an airfoil section with α equal to 5 degrees.

3.2 Aerodynamics of Wing Using the Vortex Panel Method

The previous solution was for a wing section and for assumed two-dimensional flow. However, this two-dimensional assumption is valid only for wings that can be modeled as having an infinite span. An infinite span is when the span of the wing is much larger than the chord. An example of a such a wing would be that of a glider. Since gliders have long slender wings, most of the flow is two dimensional with no flow along the wing. Due to the lack of flow along the wing there is less induced drag making these wings ideal for gliding. However, when maneuvering is important, such as with our case, the wing should have a much lower aspect ratio. The lower aspect ratio reduces the bending moment at the root of the wing.

When a wing has finite span, otherwise known as low aspect ratio, three dimensional flow effects become important. This importance stems from the high pressure on the bottom of the wing trying to equalize with the low pressure on the top. This equalization occurs at the wing tips and is the cause of what is known as wing tip vortices. These tip vortices cause a downwash of velocity on the wing, resulting in loss of lift and an increase in induced drag. Therefore, in order to solve for the aerodynamics of wing with a finite span, we must choose a method capable of capturing these physics.

To capture the physics of a finite span, a vortex panel method will be used. This panel method is a simplified version of the vortex lattice method, which is an industry standard. The major difference is that the vortex panel method consists of panels only in the spanwise direction, while the lattice method has panels in both the spanwise and chordwise directions. Therefore, the vortex panel method may have 10 panels to model the flow over the wing, while the lattice method may have 100. This means that only a 10x10 matrix has to be inverted for the solution to a vortex panel method, while a 100x100 matrix has to be inverted for a vortex lattice method. By reducing the number of panels, the time needed for a solution is greatly reduced and, as mentioned previously, this reduction becomes important in a dynamic simulation.

The main idea behind these vortex methods is to solve for the circulation strength of

each vortex, Γ_n, located on each panel. The lift of each panel is directly proportional to this circulation strength through equation 3.27:

$$l_n = \rho_\infty U_\infty \Gamma_n \tag{3.27}$$

The velocity induced from a vortex is related to the strength of the vortex, Γ, by the law of Biot and Savart that yields the relationship shown in equation 3.28:

$$\vec{dv_m} = \frac{\Gamma_n(\vec{dl} \times \vec{r})}{4\pi r^3} \tag{3.28}$$

where $\vec{dl}$ is the vector representing a vortex filament and $\vec{r}$ is the vector relating the normal distance from the vortex filament in which the velocity is desired.

In order to implement this method, a vortex filament is placed on each panel. This filament starts at positive infinity and goes to a point located at the left hand side $\frac{1}{4}$ chord point of the panel [Bertin and Smith, 1998]. It then travels along the $\frac{1}{4}$ chord point to the right hand side of the panel. Finally the vortex filament terminates at positive infinity as shown in Figure 3.6. In addition to the vortices, there is a control point centered on each panel at the $\frac{3}{4}$ chord point as shown in figure 3.6.

To solve for the circulation strength of a vortex, a boundary condition is applied at the control point. The applied boundary condition states that the flow must remain tangent to the wing everywhere. Therefore, the flow normal to the wing must be zero. If we had one vortex and one control point we could apply equation 3.28 with the boundary condition and solve for the unknown circulation strength, Γ. This method can be extended to multiple panels through the principle of superposition. If we have N panels, then there will be N horseshoe vortices. Therefore we can develop an equation with N unknowns. For example, consider one control point m. The velocity induced at this control point becomes the combined effect of each vortex with circulation strength Γ_n. Then, applying the boundary condition that this velocity, V_m, normal to the wing at the control point is zero, we end up with an equation with N unknowns as shown in equation 3.29:

$$\vec{dV_m} = \sum_{n=1}^{N} \vec{C_{m,n}} \Gamma_n \tag{3.29}$$

By choosing the number of control points equal to the number of panels we are left with a square matrix, which can be inverted to solve for the circulation of each vortex, Γ_n.

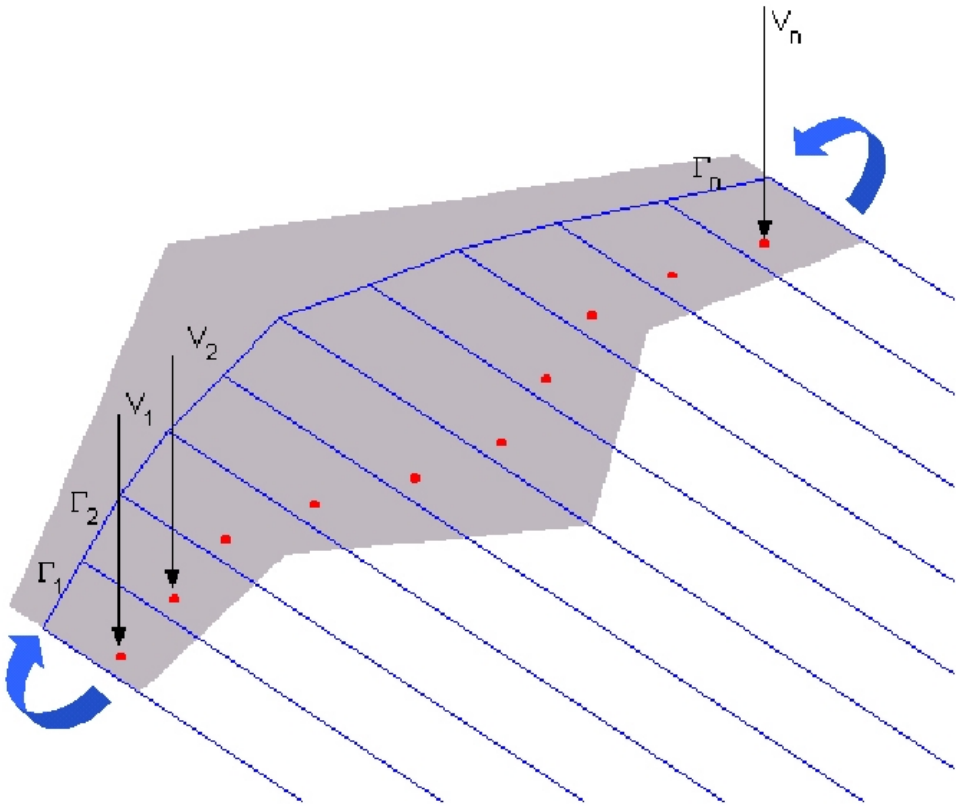


Figure 3.6: The layout of the horseshoe vortices located at the $\frac{1}{4}$ chord point of the wing along with the control points located at the $\frac{3}{4}$ chord point.

3.3 Aerodynamics of Wing Combining Conformal Mapping and Vortex Panel Method

We can obtain the lift of the wing from the vortex panel method and then approximate the distribution of this lift using conformal mapping. This combination is done by calculating the potential flow at each spanwise station and substituting the circulation Γ found from the vortex panel method, thus giving the flow around a circle at each spanwise station. Since each wing section was generated from a circle, the flow around the wing can be calculated through a geometric relation, as described in the conformal mapping section.

Since the potential flow theory used is based on a two-dimensional analysis and the

vortex panel method considers three dimensional effects, two steps must be taken to integrate these methods. First, the Kutta condition, which states that the flow from the upper and lower surface must leave the wing smoothly, must still be satisfied. By choosing s in equation 3.23 such that it corresponds to the trailing edge of the wing, we can solve for the angle of attack, α, that will ensure the velocity at the trailing edge is zero. Again the point $s = 1$ is used since it corresponds to the trailing edge yielding equation 3.30:

$$\alpha(\eta) = \sin^{-1}\left(\frac{\Gamma(\eta)}{4\pi V}\right) = \frac{\Gamma(\eta)}{4\pi V} \tag{3.30}$$

This result is one that we would expect. Since the flow is circulating around the wing tips due to the high pressure on the bottom of the wing and the low pressure on top, there is some downwash. This downwash will change the angle of attack of the free stream velocity relative to the panel.

The second factor to consider is that we must ensure that the total lift of each panel is equal to the lift found from equation 3.27. Therefore, a direct comparison is made and the free stream velocity is scaled slightly at each panel to insure the match. Again this result is one that we would expect. Because the vortex panel method is a three-dimensional analysis, there will be some velocity flowing along the span of the wing. This flow along the span of the wing will cause the incident velocity flowing over the wing to be less.

The aerodynamic code was validated against two other codes: NASTRAN and a Vortex Lattice Code. The results of these comparisons in Figure 3.7 show the code does an excellent job of predicting the loads. In addition, this method allows for calculations to be made in less than a quarter of a second on a desktop PC with the accuracy of the results shown [Gern, et. al., 2001]. Other benefits include that this combined method does not assume symmetry and the locations of the vortices and control points are calculated directly from the shape generation in the previous chapter. Therefore, it will have the flexibility to be used in a six degrees of freedom simulation.

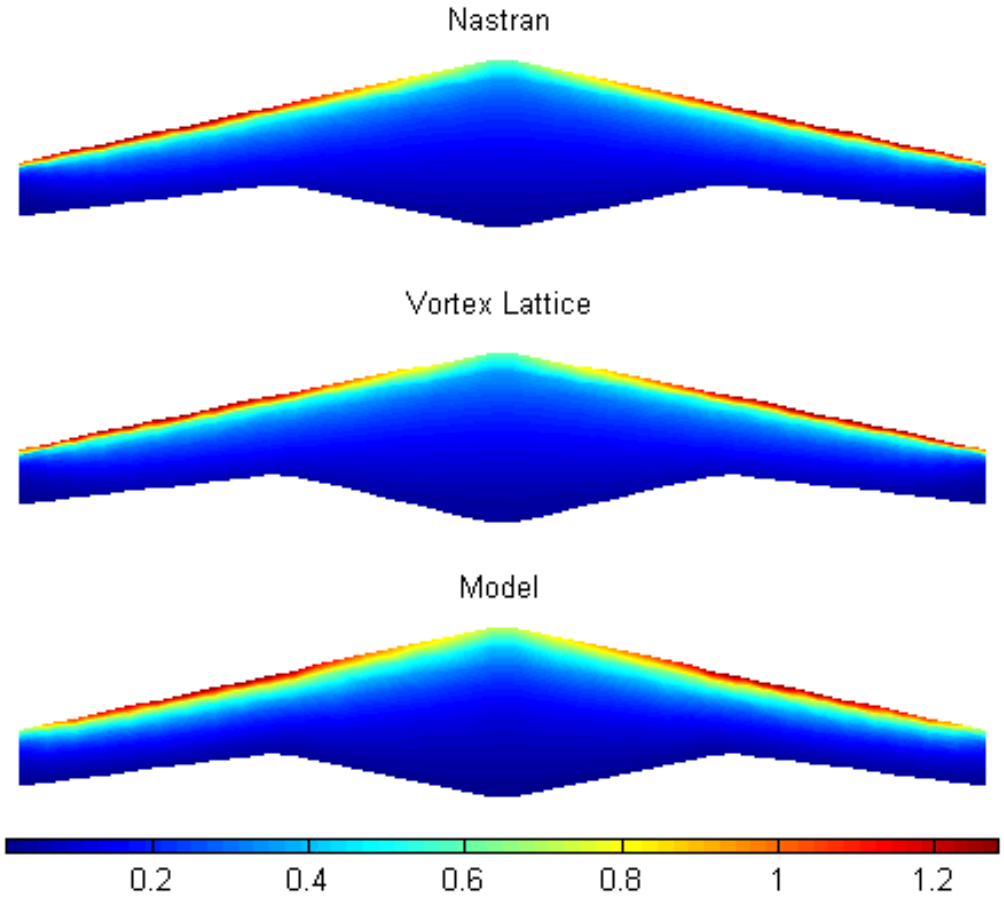


Figure 3.7: Planform evaluated using three models: NASTRAN, vortex lattice method, and combined method. The shown results are the ΔCp assuming a rigid wing with no twist at 5 degrees angle of attack and a dynamic pressure of 2.5 *psi*.

Chapter 4

Formulation of Dynamics Module

Using the steady aerodynamic's presented in the previous chapter the forces over the entire wing can be calculated. By representing these forces as a force couple system located at the center of mass, the equations of motion can be applied and the rigid body motion of the vehicle can be determined. While these forces are calculated from steady aerodynamics, they can be used for dynamic analysis since the motions occur at a relatively low frequency. This chapter is devoted to developing these equations of motion and noting the assumptions made to derive them. While the final equations are included in many sources, their derivations are included for completeness [Bryson, 1994, Ashley, 1992].

4.1 Coordinate System

Previously when describing the shape and aerodynamics of the wing, a fixed right handed coordinate system with the y axes pointing towards the port wing has been used. However, when describing the rigid body dynamics of such a wing it is helpful to have two coordinate systems: one fixed to the wing and the other fixed to the earth. In addition to the extra coordinate system, the coordinate system fixed to the wing is still right handed, except the y axes now points towards the starboard wing. While confusing, this change in the coordinate system is used to remain true to convention.

The dynamics used to calculate rigid body motion will be relative to the non-inertial axes fixed to the body, while position will be described relative to the coordinate system fixed to the earth. For example, the equations of motion will include velocities that are relative to the axes system fixed to the vehicle, while a time history of these velocities will be used to describe the vehicles position relative to the earth.

The rotating coordinate system fixed to the wing brings rise to some additional equations. To understand these additional equations, we will begin by calculating the velocity of the mass shown in Figure 4.1a. The velocity of the mass, $\dot{\vec{r}}_i$, must be written relative to an inertial axes. Using vector addition this derivative can be re-written as follows:

$$\dot{\vec{r}}_i = \dot{\vec{r}}_c + \dot{\vec{\rho}}_i \tag{4.1}$$

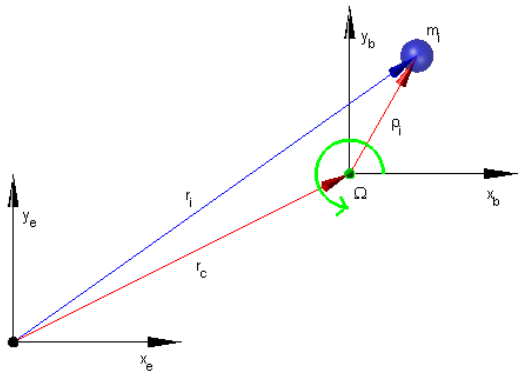


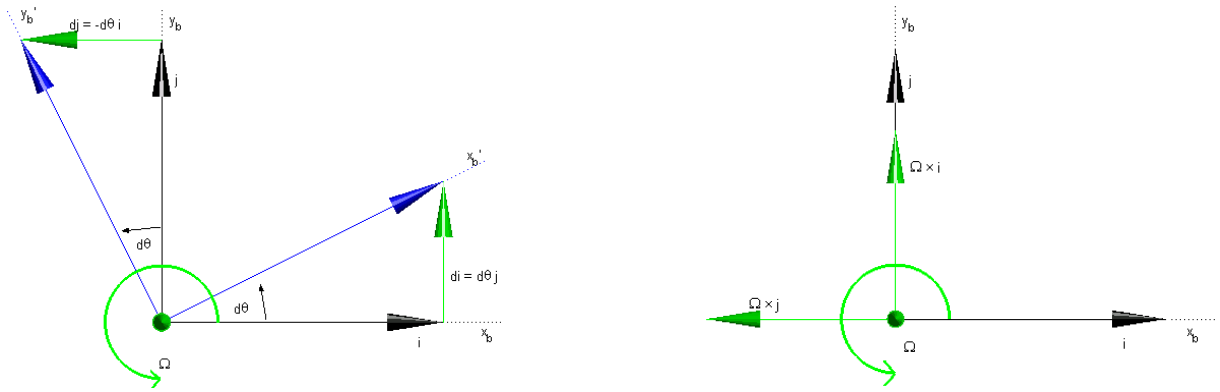
Figure 4.1: Coordinate system showing both the inertial axes fixed to the earth and the rotating system attached to the body.

Since $\vec{r_c}$ is relative to the inertial axes, its derivative is simply the velocity of point c, the center of mass, thus it can be denoted by V_c. However, $\vec{\rho_i}$ is relative to a rotating axes as shown in Figure 4.1. This rotation means that not only is it possible for the vector to change, but the axes can change as well. Therefore, the chain rule must be applied. For example, let $\vec{\rho}_i$ be the vector represented by equation 4.2:

$$\vec{\rho}_i = x\hat{\imath} + y\hat{\jmath} + z\hat{k} \tag{4.2}$$

Then its derivative becomes:

$$\dot{\vec{\rho}}_i = \dot{x}\hat{\imath} + \dot{y}\hat{\jmath} + \dot{z}\hat{k} + x\dot{\hat{\imath}} + y\dot{\hat{\jmath}} + z\dot{\hat{k}} \tag{4.3}$$



(a) (b)

Figure 4.2: The following figures shows the rotating coordinate system: (a) the small angle approximation; and (b) the derivatives represented in vector form.

The first part of the derivative, $\dot{x}\hat{\imath} + \dot{y}\hat{\jmath} + \dot{z}\hat{k}$, is the velocity of the mass relative to point c and will be denoted by V_{rel} as shown below:

$$\dot{\vec{\rho}}_i = V_{rel} + x\dot{\hat{\imath}} + y\dot{\hat{\jmath}} + z\dot{\hat{k}} \tag{4.4}$$

However, the second part of the derivative is more complicated. In order to determine an expression for $x\dot{\hat{\imath}} + y\dot{\hat{\jmath}} + z\dot{\hat{k}}$, it is helpful to visualize the rotation of the axes and what is happening to the unit vectors as shown in Figure 4.2. In Figure 4.2a an expression for $d\hat{\imath}$ and $d\hat{\jmath}$ can be determined. By assuming $d\theta$ is a small angle then $d\hat{\imath}$ and $d\hat{\jmath}$ can be approximated by the following geometric relationships given by equation 4.5:

$$d\hat{\imath} = d\theta\hat{\jmath} \quad ; \quad d\hat{\jmath} = -d\theta\hat{\imath} \tag{4.5}$$

Then dividing both sides of equation 4.5 by dt the equation reduces to the following:

$$\begin{aligned} \frac{d\hat{\imath}}{dt} = \frac{d\theta}{dt}\hat{\jmath} \quad ; \quad \frac{d\hat{\jmath}}{dt} = -\frac{d\theta}{dt}\hat{\imath} \\ \dot{\hat{\imath}} = \Omega\hat{\jmath} \quad ; \quad \dot{\hat{\jmath}} = -\Omega\hat{\imath} \end{aligned} \tag{4.6}$$

Plotting the vectors represented in equation 4.6 as shown in Figure 4.2b we can see that the derivative of a unit vector is the cross product between the vector of rotation, Ω, and the unit vector itself as shown in equation 4.7

$$\begin{aligned} \dot{\hat{\imath}} &= \Omega \times \hat{\imath} \\ \dot{\hat{\jmath}} &= \Omega \times \hat{\jmath} \\ \dot{\hat{k}} &= \Omega \times \hat{k} \end{aligned} \tag{4.7}$$

Thus the final expression for $\dot{\vec{\rho}_i}$ becomes:

$$\dot{\vec{\rho}_i} = V_{rel} + \Omega \times \vec{\rho}_i \tag{4.8}$$

It should be noted that the expression shown in equation 4.8 is true for any vector described in a rotating coordinate system and will be used throughout the derivation below.

4.2 Non-Linear Equations of Motion

4.2.1 Translational Equations

We begin with Newton's second law of motion, which states that the sum of the forces is equal to mass times its acceleration as shown in equation 4.9:

$$\sum \vec{F} = m\vec{a} \tag{4.9}$$

This law cannot be proven analytically and is instead validated only from empirical data. Furthermore, this law can be manipulated to many forms, which can be chosen according to the particular solution methods being used. By replacing the acceleration vector in equation 4.9 with the change in velocity, equation 4.9 becomes:

$$\sum \vec{F} = \frac{md(\vec{V})}{dt} \tag{4.10}$$

If the mass is constant with respect to time the mass can be moved inside the derivative and the sum of the forces becomes equal to the change in linear momentum as shown in

equation 4.11:

$$\sum \vec{F} = \frac{d(m\vec{V})}{dt} = \dot{\vec{G}} \tag{4.11}$$

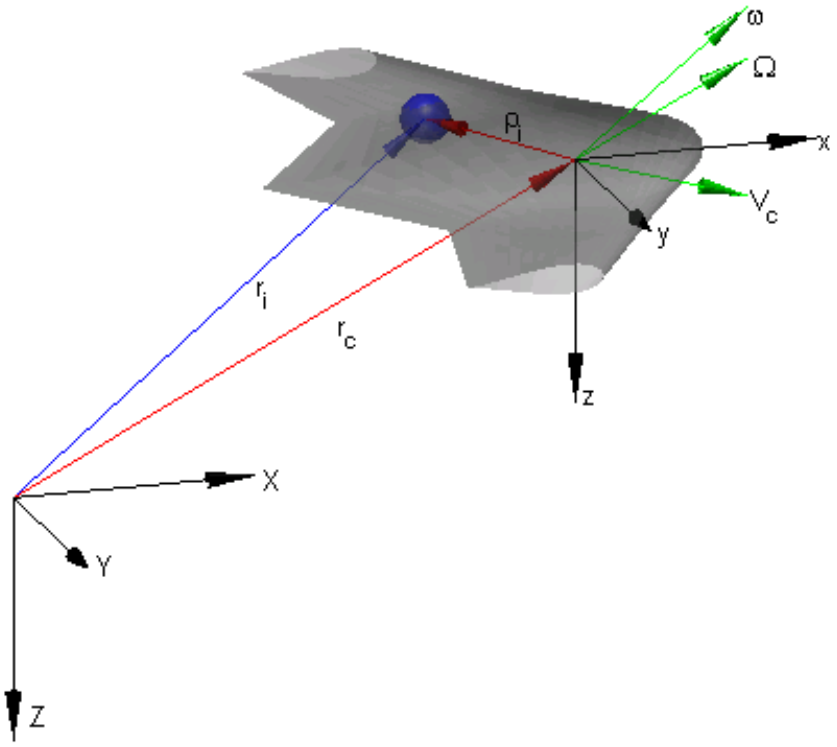


Figure 4.3: Diagram of the system to be simulated, where Ω is the rotation of the body axes located at the center of mass and ω is the rotation of the body of mass. The inertial axes system is represented by the subscript i.

Figure 4.3 represents the desired system for our simulation. Applying Newton's second law, equation 4.11, to the elemental masses of Figure 4.3 we get the following equation of motion:

$$\sum \vec{F} = \frac{d\left(\sum m_i \dot{\vec{r}_i}\right)}{dt} \tag{4.12}$$

Using vector addition $\vec{r_i}$ can be replaced by $\vec{r_c} + \vec{\rho_i}$ where the subscript c represents the center of mass and the subscript i represents the vector connecting the ith mass element to the center of mass. Making this substitution, the new equation of motion can be represented

by equation 4.13:

$$\sum \vec{F} = \frac{d\left(\sum m_i(\dot{\vec{r_c}} + \dot{\vec{\rho_i}})\right)}{dt} \tag{4.13}$$

The expression for $\dot{\vec{r_c}}$ becomes the velocity of the center of mass, V_c, and since $\dot{\vec{\rho_i}}$ is relative to a rotating coordinate system it is replaced with the expression found in equation 4.8. The new equation of motion is shown below:

$$\sum \vec{F} = \frac{d\left(\sum m_i(V_c + V_{rel} + \Omega \times \rho_i)\right)}{dt} \tag{4.14}$$

We will assume the body is rigid in order to simplify this equation. By making this assumption we are implying that the mass will not move relative to the center of mass, c, which makes V_{rel} equal to zero. In contrast, the goal of this model is to determine the amount of work necessary to move this mass against its external forces. However, we can make this assumption because the inertial effects of these motions will be relatively small compared to the overall dynamics of the vehicle. Another simplification can be made since point c is the center of mass. With this assumption $\sum m_i\rho_i$ becomes equal to zero by definition. These simplifications yield a new equation of motion shown below:

$$\sum \vec{F} = \frac{d\left(mV_c\right)}{dt} \tag{4.15}$$

The velocity of the body's center of mass, V_c, is relative to the earths axes while most of the forces on the body have been calculated relative to the rotating axes. For these reasons it is helpful to describe the dynamics relative to the moving coordinate system. This transformation can be done using the transformation of a time derivative, which transforms the derivate from one coordinate system to another as shown in equation 4.16:

$$\left(\frac{d\vec{V_c}}{dt}\right)_e = \left(\frac{d\vec{V_c}}{dt}\right)_b + \vec{\Omega} \times \vec{V_c} \tag{4.16}$$

In addition, if we fix the rotating coordinate system to the body, Ω becomes equal to ω reducing the equation of motion to the following:

$$\sum \vec{F} = m\left[\left(\frac{d\vec{V_c}}{dt}\right)_b + \vec{\omega} \times \vec{V_c}\right] \tag{4.17}$$

where

$$\left(\vec{V}_c\right)_b = u\hat{\imath} + v\hat{\jmath} + w\hat{k} \tag{4.18}$$

and

$$\vec{\omega} = p\hat{\imath} + q\hat{\jmath} + r\hat{k} \tag{4.19}$$

By separating equation 4.17 into component form we get the following first order equations used to describe translational motion:

$$\begin{aligned} \dot{u} &= \frac{F_x}{m} - qw + rv \\ \dot{v} &= \frac{F_y}{m} - ru + pw \\ \dot{w} &= \frac{F_z}{m} - pv + qu \end{aligned} \tag{4.20}$$

where the forces, F, are relative to the rotating body axes. These forces can be broken down into three groups aerodynamic, propulsive, and gravitational. The aerodynamic and propulsive forces are calculated relative to this rotating axes. However, the gravitational forces of the vehicle is relative to the earth.

4.2.2 Rotational Equations

While the above equation of motion is based on linear momentum and used to describe translational motion, there exists a set of parallel equations based on angular momentum and are used to describe rotational motion as shown in equation 4.21:

$$\sum \vec{M} = \frac{d\left(\sum \left(\rho_i \times m_i \dot{r}_i\right)\right)}{dt} = \dot{\vec{H}}_c \tag{4.21}$$

Again, the derivatives are with respect to the inertial axes and the moments are determined relative to the rotating axes. Therefore, we use the transformation of a time derivative to transfer the time derivative to the coordinate system relative to the rotating body. This transformation is shown below in equation 4.22:

$$\sum \vec{M} = \left[\left(\frac{d\vec{H}_c}{dt}\right)_b + \vec{\omega} \times \vec{H}_c\right] \tag{4.22}$$

Referring to Figure 4.3 the angular momentum H_c can be re-written and as shown below:

$$\begin{aligned} \vec{H}_c &= \sum (\rho_i \times m_i \dot{r}_i) \\ &= \sum \left[\rho_i \times m_i \left(\dot{\vec{r}_c} + \dot{\vec{\rho}_i}\right)\right] \\ &= \sum \left[\rho_i \times m_i \left(\vec{V_c} + \vec{V_{rel}} + \omega \times \rho_i\right)\right] \end{aligned} \tag{4.23}$$

The term for the expression of $\vec{H}_c$ containing $\vec{V_c}$ can be re-written as $\sum \rho_i \times m_i \vec{V_c} = -\vec{V_c} \times \sum m_i \rho_i$. Since point c is the center of mass then $\sum m_i \rho_i = 0$ by definition. The second term $\vec{V_{rel}}$ is zero from the assumption that the vehicle is a rigid body. That leaves the following expression for $\vec{H}_c$ as shown in equation 4.24:

$$\begin{aligned} \vec{H}_c &= \sum \left[\rho_i \times m_i \left(\omega \times \rho_i\right)\right] \\ &= \sum \left(p m_i \left(\rho_{yi}^2 + \rho_{zi}^2\right) - q m_i \rho_{xi} \rho_{yi} - r m_i \rho_{xi} \rho_{zi}\right) \hat{\imath} \\ &+ \left(q m_i \left(\rho_{zi}^2 + \rho_{xi}^2\right) - r m_i \rho_{yi} \rho_{zi} - p m_i \rho_{xi} \rho_{yi}\right) \hat{\jmath} \\ &+ \left(r m_i \left(\rho_{xi}^2 + \rho_{yi}^2\right) - p m_i \rho_{xi} \rho_{zi} - q m_i \rho_{yi} \rho_{zi}\right) \hat{k} \end{aligned} \tag{4.24}$$

Furthermore, since we have chosen point c as the center of mass, equation 4.23 becomes:

$$\begin{aligned} \vec{H}_c &= \left(p I_{xx} - q I_{xy} - r I_{xz}\right) \hat{\imath} \\ &+ \left(q I_{yy} - r I_{yz} - p I_{xy}\right) \hat{\jmath} \\ &+ \left(r I_{zz} - p I_{xz} - q I_{yz}\right) \hat{k} \end{aligned} \tag{4.25}$$

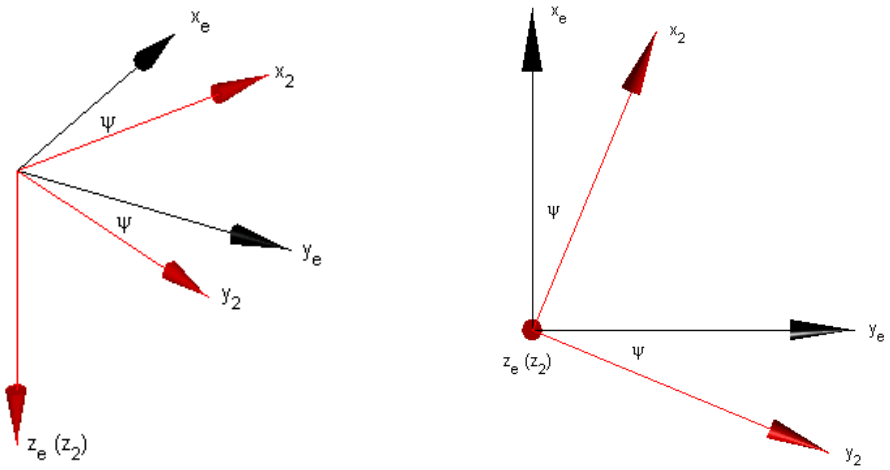
Since the planform is symmetric about one axes in both the xy and yz planes both I_{xy} and I_{yz} become zero. Substitution of equation 4.25 into equation 4.22 and separating into component form we get the following three first order equations used to describe rotational motion:

$$\begin{aligned} \dot{p} &= \frac{-I_{zz} L - I_{xz} \left(I_{xx} - I_{yy} + I_{zz}\right) pq + \left(I_{xz}^2 - I_{yy} I_{zz} + I_{zz}^2\right) qr - I_{xz} N}{I_{xz}^2 - I_{xx} I_{zz}} \\ \dot{q} &= \frac{M - I_{xz} p^2 + \left(I_{zz} - I_{xx}\right) pr + I_{xz} r^2}{I_{yy}} \\ \dot{r} &= \frac{-L I_{xz} - N I_{xx} + \left(I_{xx} I_{yy} - I_{xx}^2 - I_{xz}^2\right) pq + I_{xz} \left(I_{xx} + I_{zz} - I_{yy}\right) qr}{I_{xx} I_{zz} - I_{xz}^2} \end{aligned} \tag{4.26}$$

4.2.3 Flight Path Equations

It is not only important to know the vehicle's performance as a rigid body, but it is equally important to know its flight path and angle with respect to the inertial axes. This flight path becomes important when navigating locations or determining what relationship the vehicle has with respect to the earth. For example, the vehicle cannot take a picture of the ground if it is pointed or oriented in the wrong direction. Relating the vehicle back to the inertial axes can be achieved through a couple of different transformations. The one shown below and most commonly used is a transformation using Euler-angles. This transformation begins lined up with the inertial axes. Then the axes is yawed about the z-axis, ψ, pitched about the y-axis, θ, and finally it is rolled about the x-axis, ϕ. The order of operations is important. However, the transformation can be reversed by multiplying by the transformation matrix's transpose. The following figures and equations represent each step of the transformation.

First the yaw as shown in Figure 4.4 and represented by equation 4.27:



(a) (b)

Figure 4.4: The first Euler-angle rotation is a yaw about the z-axes: (a) the z-axes remains stationary; while (b) a positive rotation occurs about the z-axes.

$$\begin{Bmatrix} x_2 \\ y_2 \\ z_2 \end{Bmatrix} = \underbrace{\begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{T_{2e}} \begin{Bmatrix} x_e \\ y_e \\ z_e \end{Bmatrix} \tag{4.27}$$

or

$$\begin{Bmatrix} x_e \\ y_e \\ z_e \end{Bmatrix} = \underbrace{\begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{T_{e2}} \begin{Bmatrix} x_2 \\ y_2 \\ z_2 \end{Bmatrix} \tag{4.28}$$

Second the pitch as shown in Figure 4.4 and represented by equation 4.29:

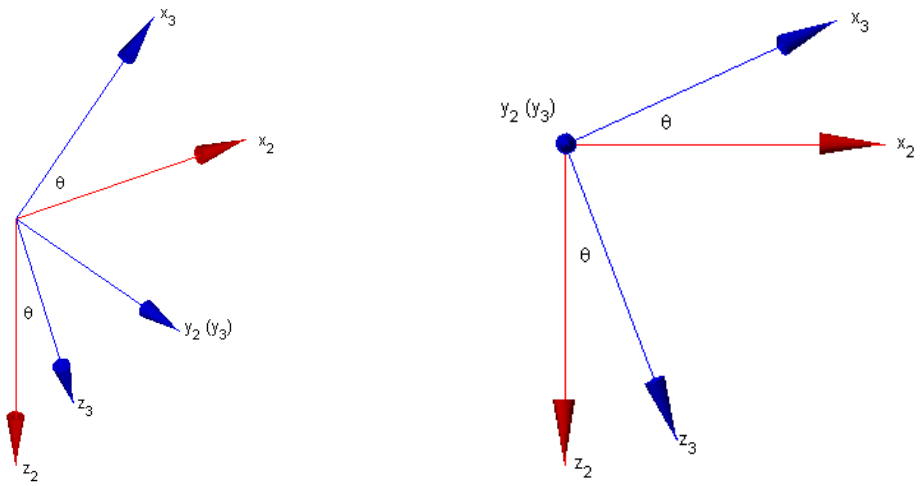


Figure 4.5: The second Euler-angle rotation is a pitch about the y-axes: (a) the y-axes remains stationary; while (b) a positive rotation occurs about the y-axes.

$$\begin{Bmatrix} x_3 \\ y_3 \\ z_3 \end{Bmatrix} = \underbrace{\begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}}_{T_{32}} \begin{Bmatrix} x_2 \\ y_2 \\ z_2 \end{Bmatrix} \tag{4.29}$$

or

$$\begin{Bmatrix} x_2 \\ y_2 \\ z_2 \end{Bmatrix} = \underbrace{\begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}}_{T_{23}} \begin{Bmatrix} x_3 \\ y_3 \\ z_3 \end{Bmatrix} \tag{4.30}$$

Third, the roll as shown in Figure 4.6 and represented by equation 4.31:

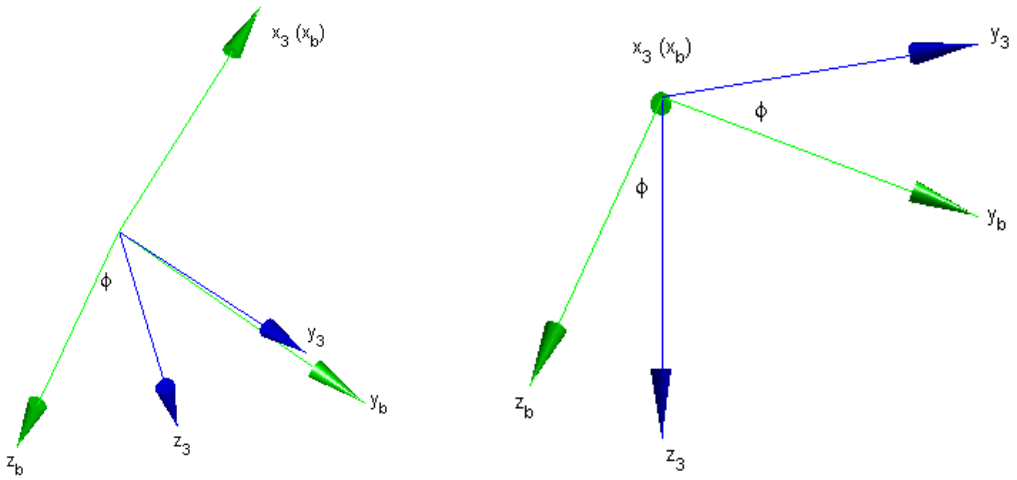


Figure 4.6: The third Euler-angle rotation is a roll about the x-axes: (a) the x-axes remains stationary; while (b) a positive rotation occurs about the x-axes.

$$\begin{Bmatrix} x_b \\ y_b \\ z_b \end{Bmatrix} = \overbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}}^{T_{b3}} \begin{Bmatrix} x_3 \\ y_3 \\ z_3 \end{Bmatrix} \tag{4.31}$$

or

$$\begin{Bmatrix} x_3 \\ y_3 \\ z_3 \end{Bmatrix} = \overbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}}^{T_{3b}} \begin{Bmatrix} x_b \\ y_b \\ z_b \end{Bmatrix} \tag{4.32}$$

With the three transformations known, it is possible to transfer between the two coordinate systems, body fixed versus earth fixed. As described earlier it is important to represent the flight path equations relative to the inertial reference coordinate system fixed to the earth. This transformation is shown below in equation 4.33.

$$\begin{Bmatrix} \dot{x_e} \\ \dot{y_e} \\ \dot{z_e} \end{Bmatrix} = T_{e2}T_{23}T_{3b} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \tag{4.33}$$

$$\begin{aligned} \dot{x_e} &= u\cos\theta\cos\psi + v(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) && (4.34)\\ &+ w(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\ \dot{y_e} &= u\cos\theta\sin\psi + v(\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi) && (4.35)\\ &+ w(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \\ \dot{z_e} &= -u\sin\theta + v\sin\phi\cos\theta + w\cos\phi\cos\theta && (4.36) \end{aligned}$$

With the addition of these flight path equations, comes three new unknowns, which are the flight path angles relative to the inertial axes. These flight path angles ϕ, θ, and ψ can be represented as states to the system. In order to determine a representation of these new states, we must represent them as a function of pre-existing states. To begin, the angular velocity of the body, ω, can be represented in two ways as shown in equations 4.37 and 4.38, [Ashley, 1992].

$$\begin{aligned} \omega &= p\hat{\imath}_b + q\hat{\jmath}_b + r\hat{k}_b && (4.37)\\ &= \dot{\phi}\hat{\imath}_b + \dot{\theta}\hat{\jmath}_3 + \dot{\psi}\hat{k}_2 && (4.38) \end{aligned}$$

Next, using the transformations defined above all of the unit vectors must be transformed to the body fixed axis as shown in equations 4.39 and 4.40.

$$\begin{aligned} \hat{\jmath}_3 &= \hat{\jmath}_b\cos\phi - \hat{k}_b\sin\phi && (4.39)\\ \hat{k}_2 &= \cos\theta k_3 - \sin\theta\hat{\imath}_3 && (4.40)\\ &= \cos\phi\cos\theta\hat{k}_b + \sin\phi\cos\theta\hat{\jmath}_b - \sin\theta\hat{\imath}_b \end{aligned}$$

Finally, substitution of equations 4.39 and 4.40 into equation 4.38 and equating like terms we can write an expression for the flight path angles in terms of existing states.

$$\dot{\phi} = p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \tag{4.41}$$

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{4.42}$$

$$\dot{\psi} = q\sin\phi\sec\theta + r\cos\phi\sec\theta \tag{4.43}$$

In summary, the equations derived above are necessary to simulate the flight of a wing with six degrees of freedom. These 12 equations are strongly coupled and highly non-linear and are again shown below:

$$\begin{aligned}
\dot{u} &= \frac{F_x}{m} - qw + rv \\
\dot{v} &= \frac{F_y}{m} - ru + pw \\
\dot{w} &= \frac{F_z}{m} - pv + qu \\
\dot{p} &= \frac{-I_{zz}L - I_{xz}\left(I_{xx} - I_{yy} + I_{zz}\right)pq + \left(I_{xz}^2 - I_{yy}I_{zz} + I_{zz}^2\right)qr - I_{xz}N}{I_{xz}^2 - I_{xx}I_{zz}} \\
\dot{q} &= \frac{M - I_{xz}p^2 + \left(I_{zz} - I_{xx}\right)pr + I_{xz}r^2}{I_{yy}} \\
\dot{r} &= \frac{-LI_{xz} - NI_{xx} + \left(I_{xx}I_{yy} - I_{xz}^2 - I_{xz}^2\right)pq + I_{xz}\left(I_{xx} + I_{zz} - I_{yy}\right)qr}{I_{xx}I_{zz} - I_{xz}^2} \\
\dot{x_e} &= u\cos\theta\cos\psi + v(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) \\
&+ w(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\
\dot{y_e} &= u\cos\theta\sin\psi + v(\sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi) \\
&+ w(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \\
\dot{z_e} &= -u\sin\theta + v\sin\phi\cos\theta + w\cos\phi\cos\theta \\
\dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\
\dot{\theta} &= q\cos\phi - r\sin\phi \\
\dot{\psi} &= q\sin\phi\sec\theta + r\cos\phi\sec\theta
\end{aligned}$$

4.3 Decoupled Equations of Motion

It is sometimes helpful to decouple the longitudinal from the lateral dynamics. In order to accomplish this task, we must make some assumptions. To begin, we must assume that the aircraft is symmetric about the x-z plane, i.e. left wing is the same as the right. This assumption will allows for $I_{yz} = I_{yx} = 0$ and will insure that u, w, q, and θ will not produce any Fy, L and N; and v, p, r, ϕ and ψ will not produce any Fx, Fz and M, [Bryson, 1994]. Next, we must assume that we begin with steady rectangular flight, where steady rectangular flight is achieved when $\dot{u} = \dot{v} = \dot{w} = \dot{p} = \dot{q} = \dot{r} = 0$ and the wings are level, $\phi = 0$. By assuming that v, p, and r are initially zero then they will remain zero and the longitudinal equations can be decoupled as shown below, [Ashley, 1992]:

$$\dot{u} = \frac{F_x}{m} - qw \tag{4.44}$$

$$\dot{w} = \frac{F_z}{m} + qu \tag{4.45}$$

$$\dot{q} = \frac{M}{I_{yy}} \tag{4.46}$$

$$\dot{\theta} = q \tag{4.47}$$

$$\dot{x_e} = u\cos\theta + w\sin\theta \tag{4.48}$$

$$\dot{z_e} = -u\sin\theta + w\cos\theta \tag{4.49}$$

Furthermore, by assuming that u, w, and q are initially zero then they too will remain zero and lateral equations can be decoupled as shown below:

$$\dot{v} = \frac{F_y}{m} \tag{4.50}$$

$$\dot{p} = \frac{-I_{zz}L - I_{xz}N}{I_{xz}^2 - I_{xx}I_{zz}} \tag{4.51}$$

$$\dot{r} = \frac{-LI_{xz} - NI_{xx}}{I_{xx}I_{zz} - I_{xz}^2} \tag{4.52}$$

$$\dot{y_e} = v\cos\phi\cos\psi \tag{4.53}$$

$$\dot{\phi} = p \tag{4.54}$$

$$\dot{\psi} = r\cos\phi \tag{4.55}$$

With the system split into the longitudinal and lateral equations of motion, the aircrafts natural modes of oscillation can be observed and individually compensated.

Chapter 5

Formulation of Control Module

In review, we have discussed three of the four modules that make up the simulation. First, the shape generation module generates a wing shape based on certain shape functions and shape parameters. Second, the aerodynamics module calculates the forces on the generated wing shape based on input from the dynamics and the shape generation modules. The third module then uses rigid body dynamics to calculate the trajectory of the system. By combining these three modules: Shape Generation, Aerodynamics, and Dynamics, shape parameters can be specified with an output of trajectory as shown in Figure 5.1a. However, using ones intuition to determine what shape parameters are necessary to achieve a given trajectory can become quite challenging even for the sharpest minds. Therefore, by linearizing the input output relationship shown in Figure 5.1b we can implement a zero error linear controller to achieve the desired dynamics, Figure 5.1c.

Since the system is observable and controllable we decided to use full state feed back for control. This method of control can be simplified further by feeding back all of the states and not using an observer. With the end goal of this research concerned with the energy necessary to morph a wing in flight and not to design a practical controller, the absence of an observer is acceptable. With this said, we investigate two possible methods of achieving a zero error tracking with full state feedback. The first method used full state feedback with a feed forward term that would allow for the steady state response to have zero error.

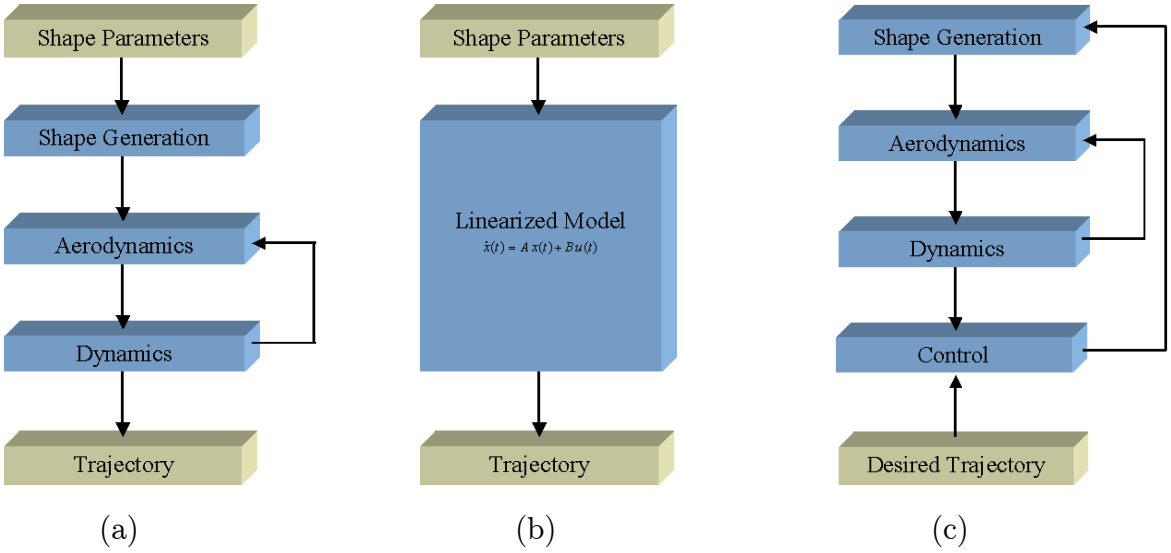


Figure 5.1: Block diagrams describing the control methodology: (a) program allows for shape parameters to be entered while ouputing the trajectory; (b) input output relationship is linearized to develop a controller; and (c) controller is implemented to track a desired trajectory.

This method shifts the steady state value based on the linear equations, which means that non-linearities could result in error. The second method augments the state matrix with the integral of the error in the states we wish to track. Therefore, slight variations in linearity will not cause any problems.

5.1 Open Loop State Equations

To begin the design of the compensator, the system shown in Figure 5.1a , which is represented by equations 5.1- 5.10, was first linearized by calculating the Jacobian:

$$\dot{u} = \frac{F_x(u, w, q, camber_x, reflex_x, twist_x)}{m} - qw + \frac{thrust_x}{m} \tag{5.1}$$

$$\dot{w} = \frac{F_z(u, w, q, camber_x, reflex_x, twist_x)}{m} + qu \tag{5.2}$$

$$\dot{q} = \frac{M(u, w, q, camber_x, reflex_x, twist_x)}{I_{yy}} \tag{5.3}$$

$$\dot{\theta} = q \tag{5.4}$$

$$\dot{x_e} = u\cos\theta + w\sin\theta \tag{5.5}$$

$$\dot{z_e} = -u\sin\theta + w\cos\theta \tag{5.6}$$

$$\dot{camber_x} = \frac{camber_u - camber_x}{\tau_{camber}} \tag{5.7}$$

$$\dot{reflex_x} = \frac{reflex_u - reflex_x}{\tau_{reflex}} \tag{5.8}$$

$$\dot{twist_x} = \frac{twist_u - twist_x}{\tau_{twist}} \tag{5.9}$$

$$\dot{thrust_x} = \frac{thrust_u - thrust_x}{\tau_{thrust}} \tag{5.10}$$

To avoid inaccuracy in the numerical simulation, first order dynamics have been added to the inputs of the shape parameters. One should note, while these equations represent the longitudinal dynamics and only three shape parameters were chosen as inputs, the model is not limited to these. For example, if sweep was desired as a possible shape parameter, it would be necessary to make a modification to the code. However, planforms, initial conditions, and certain control parameters can easily be changed in the input file to the simulation.

The Jacobian about the desired trimmed condition is calculated as shown in equations 5.11 and 5.12:

$$A = \frac{\partial f}{\partial x} \mid_{x^o,u^o} \tag{5.11}$$

$$B = \frac{\partial f}{\partial u} \mid_{x^o,u^o} \tag{5.12}$$

Upon completion of this linearization, the system in Figure 5.1b can be represented by equation 5.13:

$$\dot{x}(t) = Ax(t) + BU(t) \tag{5.13}$$

where $x(t)$ represent the longitudinal states and $U(t)$ represent the inputs to the system as

shown below:

$$x(t) = \left\{ \begin{array}{c} u \\ w \\ q \\ \theta \\ x_e \\ z_e \\ camber_x \\ reflex_x \\ twist_x \\ thrust_x \end{array} \right\}; U(t) = \left\{ \begin{array}{c} camber_u \\ reflex_u \\ twist_u \\ thrust_u \end{array} \right\} \tag{5.14}$$

With the plant now modeled as a linear time invariant system as shown in Figure 5.2, we can represent the system in algebraic form so that the new augmented state matrix can be determined. Figure 5.2 shows the addition of the integral of the states we wish to track, θ, x_e, and z_e. Adding these states to the system will prevent small non-linearities from causing problems in the steady state tracking. Again referring to Figure 5.2, the open loop state equations can be written as:

$$\dot{\xi}(t) = C_I \xi(t) - N_I R(t) \tag{5.15}$$

$$\dot{x}(t) = Ax(t) + BU(t) \tag{5.16}$$

where

$$\xi(t) = \left\{ \begin{array}{c} \int(\theta \text{ error}) \\ \int(x_e \text{ error}) \\ \int(z_e \text{ error}) \end{array} \right\}; R(t) = \left\{ \begin{array}{c} \theta(t) \\ x_e(t) \\ z_e(t) \end{array} \right\} \tag{5.17}$$

When these open loop state equations are augmented they become:

$$\left\{ \begin{array}{c} \dot{\xi}(t) \\ \dot{x}(t) \end{array} \right\} = \begin{bmatrix} 0 & C_I \\ 0 & A \end{bmatrix} \left\{ \begin{array}{c} \xi(t) \\ x(t) \end{array} \right\} + \begin{bmatrix} 0 \\ B \end{bmatrix} u(t) - \begin{bmatrix} N_I \\ 0 \end{bmatrix} R(t) \tag{5.18}$$

At this point, if we looked at the eigenvalues of the augmented A matrix the following poles would appear: The system would have five poles at zero, two of which represent the rigid body motion of x_e and z_e. The other three would be from the three integrators added to

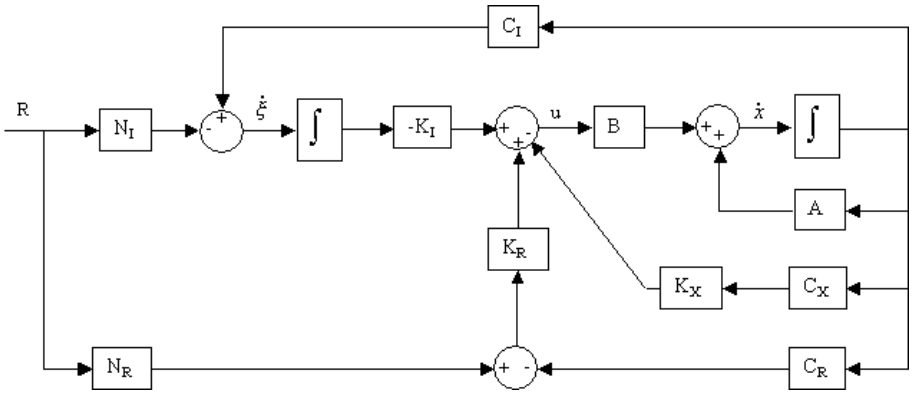


Figure 5.2: Block diagram of the full state and integral error feedback controller.

the system to ensure zero steady state error. Next there would be four real poles representing the first order time constants place on the inputs to the system. The remaining four complex poles would represent the short and long period of the rigid body. The long period, known as the phugoidal pitching mode, is highly undamped and depending on the location of the center of mass and planform configuration can be unstable. The short period generally dies out very fast; however, if flexibility is added to the structure, one must be careful not to allow the first bending mode to couple with the short period.

5.2 Closed Loop State Equations

Referring to Figure 5.2, we have chosen to track θ, x_e, and z_e, yielding the following values for C_I and C_R:

$$C_I = C_R = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.19}$$

The states u, w, q, $camber_x$, $reflex_x$, $twist_x$, and $thrust_x$ are compared to their initial conditions yielding the following for C_x:

$$C_x = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.20}$$

Next we chose to use a linear quadratic regulator, LQR, algorithm to choose the control gains. While this method does not guarantee the best tracking versus actuator power, it does serve as a good starting point. The LQR algorithm is based on optimization of a quadratic cost function, J, as shown in equation 5.21:

$$J = \int_0^\infty \left[x^T(t) Q_1 x(t) + U^T(t) Q_2 U(t) \right] dt \tag{5.21}$$

This algorithm allows for the user to trade off importance between control effort and regulation of the error in the states. If more weight is given to the control effort, the gains will be chosen in such a way to lower the control effort.

For Q_1 we have chosen to minimize the integral of the error in the states we wish to track, $\xi(t)$ as shown below:

$$Q_1 = C^T I(3) C \tag{5.22}$$

where,

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.23}$$

If the integral of the error in the tracking states goes to zero, then we can be assured that the system is indeed tracking. Equation 5.24 shows Q_2, which has been chosen as the identity matrix giving equal weighting to each control input.

$$Q_2 = I(4) \tag{5.24}$$

In order to get the desired output it may be necessary to change the weighting on some states or control inputs. For example, if pitch, θ, is not responding as fast as altitude, z_e, more weight could be given to this state by changing C:

$$C = \begin{bmatrix} 50 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.25}$$

Upon completion of the control gain calculations the closed loop eigenvalues can be determined from the closed loop state matrix. From Figure 5.2, $U(t)$ can be written as follows:

$$U(t) = -K_I \xi(t) + K_R \left[N_R R(t) - C_R x(t) \right] - K_x C_x x(t) \tag{5.26}$$

Substitution of equation 5.26 into equation 5.18 and setting $R(t) = 0$, the closed loop state matrix becomes:

$$A_{cl} = \begin{bmatrix} 0 & C_I \\ -BK_I & A - BK_R C_R - BK_x C_x \end{bmatrix} \tag{5.27}$$

Chapter 6

Energy Calculations

While there are two important energy considerations, strain energy needed to morph the structure and energy needed to overcome the aerodynamic forces, this work is concerned with the latter. We cannot predict the configuration of the active materials that will make up the morphing wing; therefore, we are not able to predict how much energy will be required to strain the wing structure. We can envision some designs that will have very little strain energy in the structure after morphing. Those designs could have variable length links or members that cannot be back driven by wing forces. We can also envision some designs that will have significant amounts of strain energy in the structure. These designs could have actuators straining the structure of a typical wing box. Therefore, we are addressing the aerodynamic loads only. To calculate this energy, we first determine the power necessary to morph the wing, and then this power is integrated to determine the energy.

6.1 Power Calculations

To calculate the power, information is fed from both the aerodynamics module and the shape module as shown in Figure 6.1. Three separate power values are of concern. First referring to Figure 6.2, the power necessary to twist each section, P_η, is calculated from the moment about the twist axis times the angular velocity of the particular section as shown

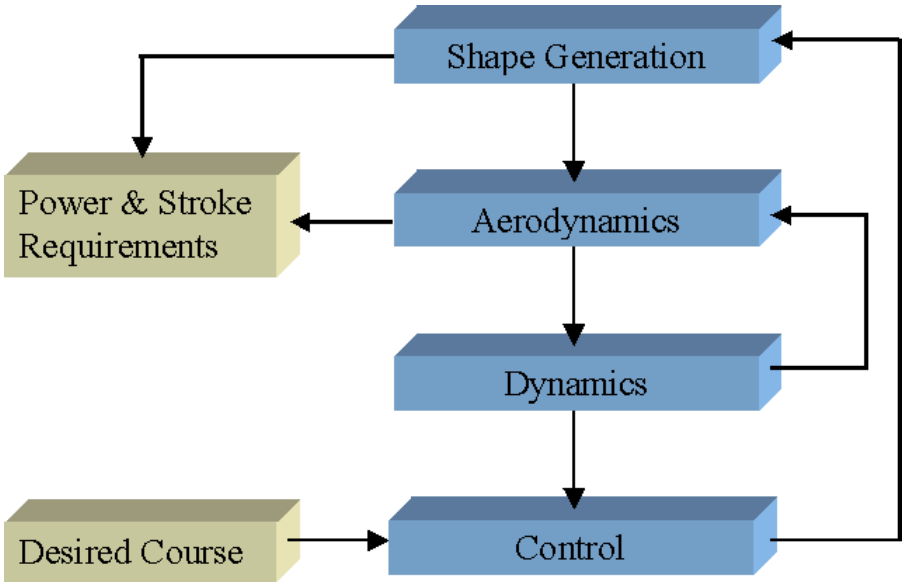


Figure 6.1: Block diagram showing where the power is calculated.

in equation 6.1:

$$P_\eta = \sum_n (r_n \times F_n)\, d\theta \tag{6.1}$$

Second, power to morph any location, k, on the wing is calculated by the dot product between the force and velocity of the given section as shown in equation 6.2:

$$P_k = F_k \bullet dr_k \tag{6.2}$$

The velocity of point k is determined relative to a plane that runs from the leading edge to the trailing edge of each wing section as shown in Figure 6.2. The third power is the total power assuming reversibility. This power shown in equation 6.3 is used as a metric that will be helpful in comparing different control schemes:

$$P_{total} = \sum_k P_k + \sum_\eta P_\eta \tag{6.3}$$

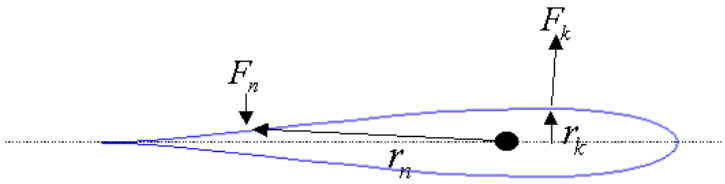


Figure 6.2: Model used to calculate the power requirements at each section.

6.2 Energy Calculated from the Power

Once the power has been calculated, we can proceed to determine the necessary energy requirements. We are concerned with two energy values that will help serve as a metric, one assumes reversibility and the other does not. In addition to these energy requirements, we want to know the force displacement characteristics of the most demanding actuator. This plot will help in determining what type of actuators can be used for the job.

Based on the power definitions defined above, the power is negative when we had to put energy into the actuator. In contrast, the power is positive when the surface was moved by the free stream velocity. With that said, in order to obtain the total reversible energy for the simulation, we must perform the integration shown below in equation 6.4:

$$E_{rev} = \int_0^{t_\infty} P_{total} dt \tag{6.4}$$

To determine the irreversible energy, we only integrate the negative portion of this power as shown in equation 6.5:

$$E = \int_0^{t_\infty} P_{total} dt \mid (P_{total} < 0) + 0 \mid (P_{total} \geq 0) \tag{6.5}$$

The last value of concern is the force displacement plot of the most demanding actuator. To create this plot, the most demanding actuator is first determined by scanning P_k for all values of time. The actuator with the most negative value is then flagged and the time history of the force versus displacement can be evaluated.

Chapter 7

Conclusions and Future Enhancements

7.1 Conclusion

The theory behind a computational model has been presented. This model is capable of predicting the force, stroke, and energy needs necessary to overcome the aerodynamic forces encountered while in flight. This model is unique in that it allows for a shape to be easily determined and changed. In addition, the model incorporates a time-efficient aerodynamic model, validated against more time-expensive codes, that can calculate the air-loads on the present shape. While the dynamics only include a three degree of freedom system, the theory has been presented to extend the model into the full six degrees. Finally, a zero error compensator is presented that allows the user to input a desired course to the model, which in turn will output the stroke, power, and energy requirements.

In addition to the theory, an overview of the code used to simulate the model is also presented. This overview includes a step by step discussion of the simulation and is supported with an actual example. The example includes the input file to the simulation followed by the output and a description of the these files. This example will serve as a starting point to further exercise and develop the model.

7.2 Future Enhancements

There are many enhancements that would serve well if added to the model. First and foremost would be the addition of a genetic algorithm. Since there are numerous input parameters that can be varied, it is difficult for the human mind to observe changes specific to each parameter. With the addition of such an algorithm, which could be added to the front end of the simulation, it would allow the inputs to the simulation to be varied in a systematic way, while minimizing the energy requirements of the actuators.

Another possible enhancement would be the inclusion of induced drag effects. Induced drag is the drag that is caused by the shedding of vortices, which in turn is proportional to the lift. When a vortex is shed there is an accompanying downwash in velocity. This downwash in velocity lowers the angle of attack and produces a corresponding force in the x-direction of the vehicle, which is known as induced drag. The addition of this drag would allow not only for efficiencies between different planforms to be determined, but would also allow for yaw control. With a method of yawing the aircraft, the extension of the model into six degrees of freedom would become a possible feat.

One final enhancement would be to add structural dynamics to the system. Upon minimizing the energy input to the system, an actual prototype could be designed. This prototype would have real actuators and some type of structure that would allow for the above modeled shape changes. The dynamics of this smart structure could feasibly be added as another module to the pre-existing code. The shape would be generated, the loads calculated, and the structure would bend accordingly. Then the process would repeat with the new shape until the system converged. With the addition of the structure would come new control strategies such as combining twist and camber to control the center of pressure relative to the elastic axis.

The model presented serves as a starting point to greatly explore the possibility of flight with morphing surfaces. By simulating the model in a batch style environment allows for many unattended runs to be calculated allowing for a vast array of shapes and control laws to be tested. In addition, this model lends itself nicely to be expanded to incorporate

many features.

Appendix A

Simulation of Formulated Model

A simulation was built based on the model discussed in the previous chapters. The simulation is a collection of MATLAB m-files and Simulink mdl-files that are integrated in such a way that allows for un-attended runs. The user can put a series of configuration files in a directory, which will be read and simulated.

A.1 Simulation Configuration

To begin, the simulation file structure is shown in Figure A.1a. At the top of the hierarchy is the main directory called *wing_long*, which stands for a wing with longitudinal dynamics. In this directory there are four sub-directories. The first is the *build* directory, which contains all of the files used in the simulation. The second, is the *input* directory that contains the input files, which will be discussed later. Third, the output directory contains the simulation output. The code creates a directory in the *output* directory. This directory carries the same name as the input file. If a directory already exists with the name of the input file, then the code will continue to add a number to the end of the directory till there is no match. The fourth and last sub-directory is the complete directory. Upon completion of the simulation the input file is moved to the complete directory.

In order to better understand the code, we will take a closer look at the simulation

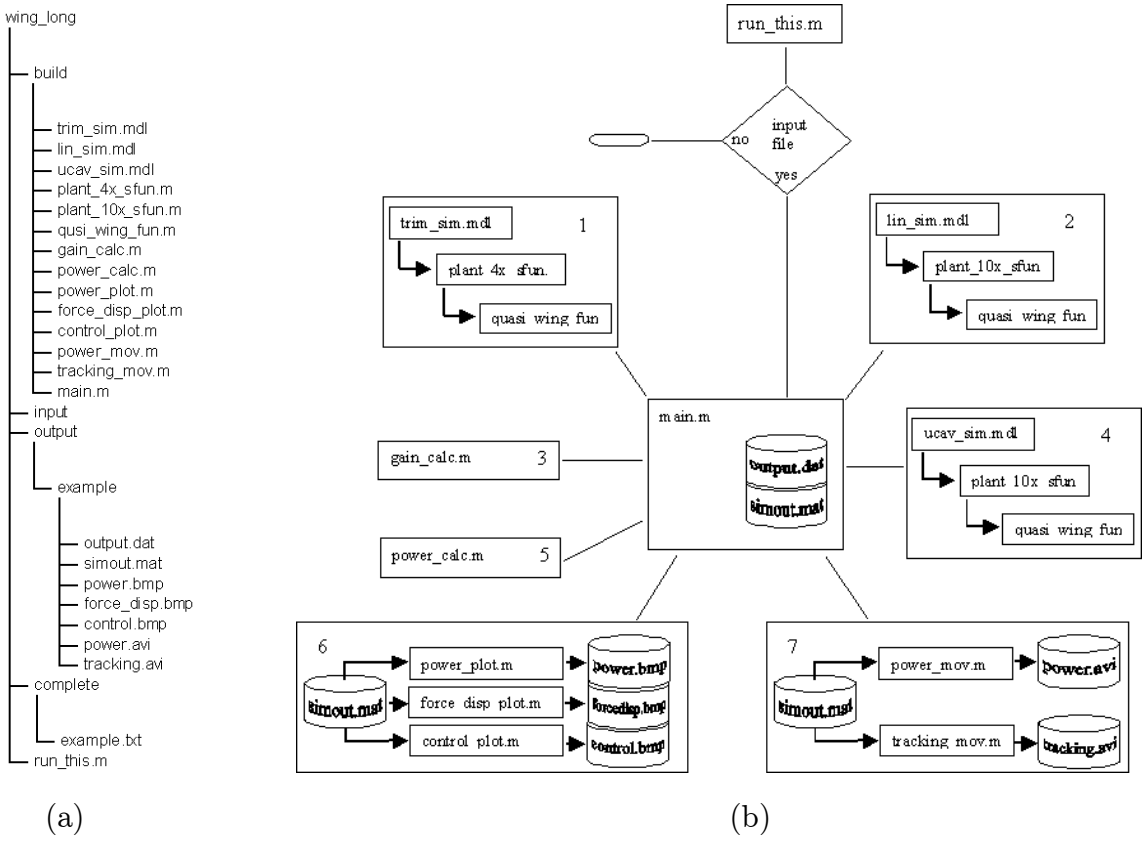


Figure A.1: Simulation and file structure diagram: (a) File structure of the code used to simulate the system; and (b) Block diagram of the code used to simulate the system.

block diagram as shown in Figure A.1b. All of the files with the exception of *run_this.m* are located in the build directory. To begin the simulation, the user starts MATLAB and sets the path to the *wing_long* directory. Then by typing *run_this.m* the code begins to execute. If there is a file in the input directory, *run_this.m* will then call *main.m*. The *main.m* file then reads the input file, creates a directory for the output in the output directory. Next, using the files shown in Figure A.1b block-1, *main.m* determines the inputs necessary for trimmed flight based on the initial conditions stated in the input file. Figure A.1b block-2 represents the linearization of the system. Using the inputs found during the trim routine and initial conditions, the system is linearized.

Upon completion of the linearization *main.m* calls the *gain_calc.m* routine to determine the control gains based on the linear model. As discussed in the control chapter, these gains are determined through an LQR routine and the weights for this routine are specified in the input file. Using these control gains the non-linear system is simulated, which is represented by fig:blkdiag2b block-4. Once the simulation has completed, *main.m* uses the output to post-process other information such as power.

At this point the code has all the information necessary, and *main.m* stores this information in a file called *simout.mat*. This file is in a structure format and can be used by many codes. From here the code proceeds to format the information in a form that is useful to the user. This format is in the form of five output files as shown in Figure A.1b blocks 6 and 7.

Upon completion, *main.m* outputs one final file, *output.dat*. This file consists of all the files used for the simulation and the dates these files were last modified. This file also contains all of the input data, trim data, linearized data, gains, and filenames of the output. The final step occurs when *main.m* moves the input file to the *complete* directory followed by a return to *run_this.m*. If another input file is in the *input* directory the process repeats. This repetition allows for the code run unattended until all of the input files are processed.

A.2 Simulation Input

The input to the simulation is contained in a text file. This text file is formated very similar to a MATLAB m-file, if a line is preceeded with a % symbol then the line will be ignored. The file contains four sets of data: shape, dynamic, control, and flight path. The shape information, described in the Shape Generation Chapter, is in the form of a structure as shown below: For example, when specifying the chord as a function of span, $shape(8).data$, we specify a matrix with two rows. The first row specifies span location and the second row is the actual $\frac{chord}{halfspan}$. This can be seen in detail in Figures 2.5a and b.

```
%=====================================
% shape functions structure
% desc - description of parameter
```

```
% var - variable for data
% data - the actual data
%        first row is don-dimensionalized span
%        second row is value associated with span
%        in-between is linearly interpolated
%=======================================
shape(1).desc='xcfunc describes the distribution of the xc mapping parameter';
shape(1).var='xcfunc';
shape(1).data='[-1 1; 1 1]';

shape(2).desc='ycfunc describes the distribution of the yc mapping parameter';
shape(2).var='ycfunc';
shape(2).data='[-1 0 1; 1 0 1]';

shape(3).desc='xtfunc describes the distribution of the xt mapping parameter';
shape(3).var='xtfunc';
shape(3).data='[-1 1; 1 1]';

shape(4).desc='ytfunc describes the distribution of the yt mapping parameter';
shape(4).var='ytfunc';
shape(4).data='[-1 0 1; 1 0 1]';

shape(5).desc='deltafunc describes the distribution of the delta mapping parameter';
shape(5).var='deltafunc';
shape(5).data='[-1 1; 1 1]';

shape(6).desc='xstartfunc describes the leading edge sweep of the wing';
shape(6).var='xstartfunc';
shape(6).data='[-1 0 1; tan(35*pi/180) 0 tan(35*pi/180)]';

shape(7).desc='zstartfunc describes the dihedral of the wing';
shape(7).var='zstartfunc';
shape(7).data='[-1 0 1; 0 0 0]';

shape(8).desc='chordfunc describes the chord of the wing -- nondimensionlized by the half span --';
shape(8).var='chordfunc';
shape(8).data='[-1 -.442 0 .442 1; .372 .558 1.21 .558 .372]';

shape(9).desc='twistfunc describes where the twist is applied to the wing';
shape(9).var='twistfunc';
shape(9).data='[-1 0 1; 1 0 1]';

shape(10).desc='twistaxisfunc describes where the twist axis is on the wing -- percent chord --';
shape(10).var='twistaxisfunc';
shape(10).data='[-1 1; .25 .25]';

shape(11).desc='pretwistfunc describes the geometric pretwist in the wing -- in radians --';
shape(11).var='pretwistfunc';
shape(11).data='[-1  -.471 -.021 .021 .471 1; -1 4.42 3.70 3.70 4.42 -1]';

shape(12).desc='vortex_panels describes the number of panels in the chordwise direction';
shape(12).var='vortex_panels';
shape(12).data='40';

shape(13).desc='chord_panels describes the number of panels on each surface in the chordwise direction';
shape(13).var='chord_panels';
shape(13).data='20';

shape(14).desc='span describes the half span of the wing -- b/2 ft --';
shape(14).var='span';
shape(14).data='15';

shape(15).desc='thickness value 0 to -.1';
shape(15).var='xc';
shape(15).data='-.1';

shape(16).desc='camber value -1 to 1';
shape(16).var='yc';
shape(16).data='0';

shape(17).desc='thickness value towards trailing edge 1 to 1.1';
shape(17).var='xt';
shape(17).data='1';

shape(18).desc='reflex value -1 to 1';
shape(18).var='yt';
shape(18).data='0';

shape(19).desc='trailing edge thickness distribution 0 to .8';
shape(19).var='delta';
shape(19).data='0';

shape(20).desc='twist value -1 to 1 degrees';
shape(20).var='th_t';
shape(20).data='0';
%=======================================
```

The dynamic section contains data the will have an effect on the equations of motion. These values such as mass, density, and time constants, are specified in a structure format like that of the shape, and are shown below: If one wanted to change the time constant of the camber input to .2, then they would change the value for $dynamic(7).data$ to $'.2'$.

```
%=======================================
% dynamic data for the simulation
% desc - description of parameter
% var - variable for data
% data - the actual data
%=======================================
dynamic(1).desc='mass of the wing in slugs';
dynamic(1).var='mass';
dynamic(1).data='310.5';

dynamic(2).desc='acceleration of gravity ft/s^2';
dynamic(2).var='g';
dynamic(2).data='32.2';

dynamic(3).desc='center of gravity [x,y,z] relative to zeta';
dynamic(3).var='cg';
dynamic(3).data='[7,0,0]';

dynamic(4).desc='mass moment of inertia Iyy slugs ft^2';
dynamic(4).var='Iyy';
dynamic(4).data='50000';

dynamic(5).desc='air density slugs/ft^3';
dynamic(5).var='rho';
dynamic(5).data='2.38e-3';

dynamic(6).desc='initial conditions [uo wo qo tho xo zo tau_camber tau_reflex tau_twist tau_thrust]';
dynamic(6).var='xinit';
dynamic(6).data='[400 0 0 0 0 0 0 0 0 0]';

dynamic(7).desc='time constant for camber';
dynamic(7).var='tau_camber';
dynamic(7).data='.3';

dynamic(8).desc='time constant for reflex';
dynamic(8).var='tau_reflex';
dynamic(8).data='.3';

dynamic(9).desc='time constant for twist';
dynamic(9).var='tau_twist';
dynamic(9).data='.3';

dynamic(10).desc='time constant for thrust';
dynamic(10).var='tau_thrust';
dynamic(10).data='.3';
%=======================================
```

The control inputs consist of MATLAB commands and are entered so that the end result are two weight matrices for the LQR algorithm. These matrices should be set equal to $Q1$, state weights, and $Q2$, control weights.

```
%=======================================
% LQR weights
%=======================================
%=======================================
% Q2/Q1 larger = more control effort
%=======================================
ratio=1;
%=======================================
%=======================================
% state weights for LQR
% [int_th int_x int_z u w q th x z camber_x reflex_x twist_x thrust_x]
%=======================================
H=[1000 0 0 0 0 0 0 0 0 0 0 0; 0 1 0 0 0 0 0 0 0 0 0 0 0; 0 0 1 0 0 0 0 0 0 0 0 0 0];
Q1=1/ratio*H'*eye(3)*H;
```

```
%=======================================
%=======================================
% control weights for LQR
% [camber_u reflex_u twist_u thrust_u]
%=======================================
Q2=ratio*[100 0 0 0; 0 100 0 0; 0 0 100 0; 0 0 0 100];
%=======================================
```

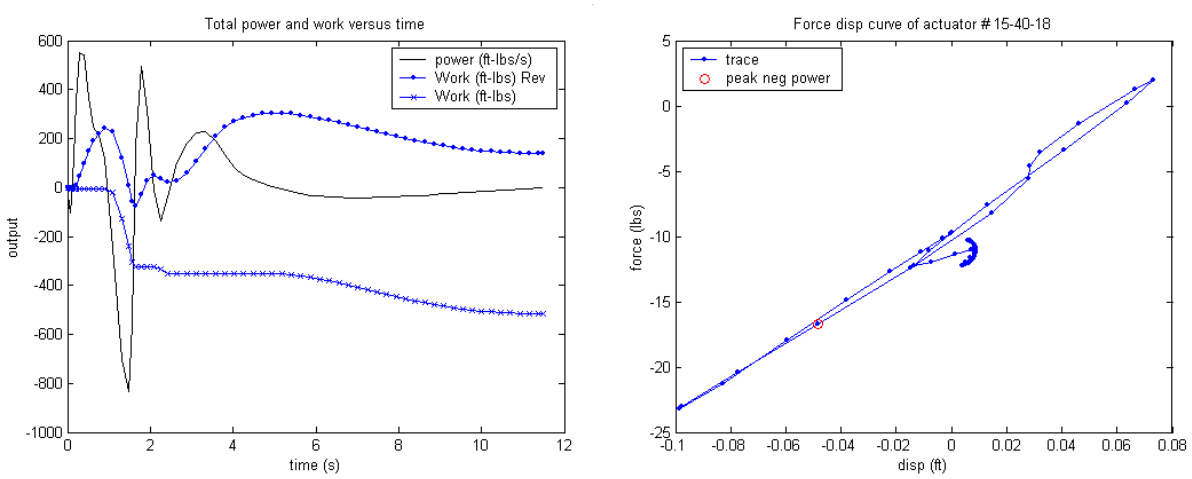
The final input is the desired trajectory as a function of time. Simulink will read in these inputs and linearly interpolate between them.

```
%=======================================
% inputs to simulation
% t th x z
%=======================================
in(1,:)=[0 0 0 0];
in(2,:)=[.5 .041 200 0];
in(3,:)=[1 .082 400 0];
in(4,:)=[1.5 .124 600 0];
in(5,:)=[11.5 .124 4600 -500];
%=======================================
```

A.3 Simulation Output

The simulation outputs five files that help in visualizing the results of the simulation. Figures A.2a and b shows the first of these files, which are the total power requirements and the force displacement requirements of the most demanding actuator. The overall power requirements refer to the total from all the actuators added together. Figure A.2a also shows the overall energy requirements assuming both the case if we could capture energy from the free stream velocity, reversible, and the case if we could not capture this energy, irreversible. While Figure A.2a refers to the overall power, Figure A.2b refers to the power of only one actuator, which is the most demanding actuator on the outer surface. This force displacement plot can take on some unique shapes, due to the decoupled twist and camber inputs.

The next output, Figure A.3, is a plot of the control inputs that are feed into the shape generation. These inputs are actually the states $camber_x$, $reflex_x$, and $twist_x$. We chose to monitor these states as opposed to the actual input to the system, so that it could be determined when these inputs became to large.



(a) (b)

Figure A.2: Plots showing the power requirements of the wing: (a) plot of the power and energy requirements of the wing versus time; and (b) plot of the force displacement requirements of the most demanding actuator.

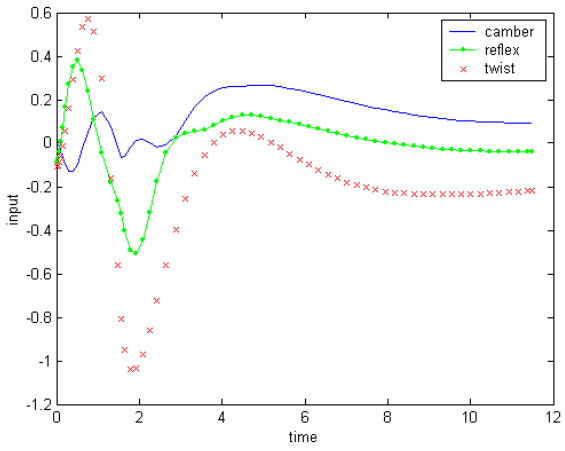
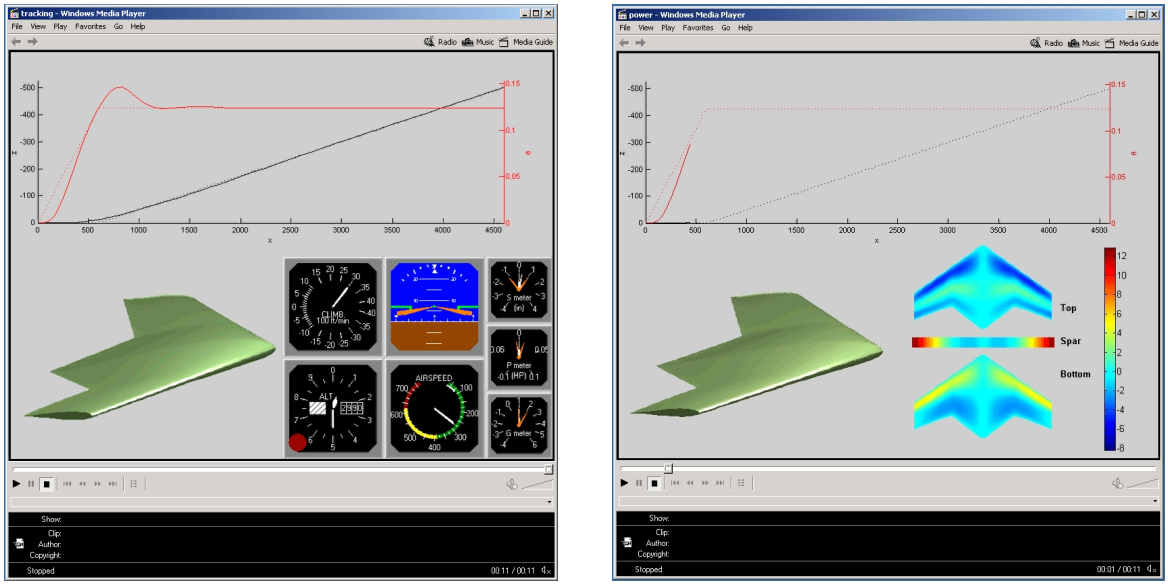


Figure A.3: Plot of the control input to the wing as a function of time.

The next two output files are movies showing the real time results as the wing maneuvers the desired course by morphing from one shape to another. The first movie Figure A.4a shows the wing in the lower left hand corner while the tracking results are displayed in the top. The lower right hand corner contains a series of gauges that help to give a sense of what is happening. Most of the gauges are the kind you would find in a real aircraft panel, vertical speed indicator, artificial horizon, altimeter, air speed indicator, and a load meter. The other two gauges display: the current maximum and minimum displacement of the outer surface of the wing; and the minimum and maximum peak power of the most demanding actuator. By displaying these two gauges the user can see where the plane required the largest input and possible adjust the shape, control law, or input accordingly. The second movie is the same as the first, except the gauges have been replaced by the localized real time power in and out of the wing.



(a) (b)

Figure A.4: Screen shots from the output movies: (a) screen shot of the movie that demonstrates the real time tracking and performance capabilities of the wing; and (b) screen shot of the movie that demonstrates the real time power requirements of the wing.

The last file in the output is the *output.dat* file. This file contains all pertinent information used in running the simulation and is shown below: With this file and the *simout.mat* file all of the information shown above could be re-created in a form suitable to the user.

```
%============================================================
% Files used in the simulation
%============================================================
Simulation was run at 14-Nov-2001 14:03:41
control_plot.m 02-Oct-2001 07:32:30
force_disp_plot.m 14-Nov-2001 12:27:48
gain_calc.m 14-Nov-2001 13:22:41
lin_sim.mdl 11-Nov-2001 23:24:46
main.m 14-Nov-2001 12:41:13
plant_10x_sfun.m 12-Nov-2001 00:17:34
plant_4x_sfun.m 11-Nov-2001 22:51:03
power_calc.m 28-Aug-2001 06:00:00
power_mov.m 14-Nov-2001 13:26:25
power_plot.m 14-Nov-2001 12:27:23
quasi_wing_fun.m 13-Nov-2001 18:38:12
tracking_mov.m 14-Nov-2001 13:25:44
trim_sim.mdl 11-Nov-2001 21:03:01
ucav_sim.mdl 11-Nov-2001 23:31:05



%============================================================
% Input to simulation
%============================================================
shape(1).desc='xcfunc describes the distribution of the xc mapping parameter';
shape(1).var='xcfunc';
shape(1).data='[-1 1; 1 1]';
shape(2).desc='ycfunc describes the distribution of the yc mapping parameter';
shape(2).var='ycfunc';
shape(2).data='[-1 0 1; 1 0 1]';
shape(3).desc='xtfunc describes the distribution of the xt mapping parameter';
shape(3).var='xtfunc';
shape(3).data='[-1 1; 1 1]';
shape(4).desc='ytfunc describes the distribution of the yt mapping parameter';
shape(4).var='ytfunc';
shape(4).data='[-1 0 1; 1 0 1]';
shape(5).desc='deltafunc describes the distribution of the delta mapping parameter';
shape(5).var='deltafunc';
shape(5).data='[-1 1; 1 1]';
shape(6).desc='xstartfunc describes the leading edge sweep of the wing';
shape(6).var='xstartfunc';
shape(6).data='[-1 0 1; tan(35*pi/180) 0 tan(35*pi/180)]';
shape(7).desc='zstartfunc describes the dihedral of the wing';
shape(7).var='zstartfunc';
shape(7).data='[-1 0 1; 0 0 0]';
shape(8).desc='chordfunc describes the chord of the wing -- nondimensionlized by the half span --';
shape(8).var='chordfunc';
shape(8).data='[-1 -.442 0 .442 1; .372 .558 1.21 .558 .372]';
shape(9).desc='twistfunc describes where the twist is applied to the wing';
shape(9).var='twistfunc';
shape(9).data='[-1 0 1; 1 0 1]';
shape(10).desc='twistaxisfunc describes where the twist axis is on the wing -- percent chord --';
shape(10).var='twistaxisfunc';
shape(10).data='[-1 1; .25 .25]';
shape(11).desc='pretwistfunc describes the geometric pretwist in the wing -- in radians --';
shape(11).var='pretwistfunc';
shape(11).data='[-1  -.471 -.021 .021 .471 1; -1 4.42 3.70 3.70 4.42 -1]';
shape(12).desc='vortex_panels describes the number of panels in the chordwise direction';
shape(12).var='vortex_panels';
shape(12).data='40';
shape(13).desc='chord_panels describes the number of panels on each surface in the chordwise direction';
shape(13).var='chord_panels';
shape(13).data='20';
shape(14).desc='span describes the half span of the wing -- b/2 ft --';
shape(14).var='span';
shape(14).data='15';
shape(15).desc='thickness value 0 to -.1';
shape(15).var='xc';
shape(15).data='-.1';
shape(16).desc='camber value -1 to 1';
shape(16).var='yc';
shape(16).data='0';
shape(17).desc='thickness value towards trailing edge 1 to 1.1';
shape(17).var='xt';
```

```
shape(17).data='1';
shape(18).desc='reflex value -1 to 1';
shape(18).var='yt';
shape(18).data='0';
shape(19).desc='trailing edge thickness distribution 0 to .8';
shape(19).var='delta';
shape(19).data='0';
shape(20).desc='twist value -1 to 1 degrees';
shape(20).var='th_t';
shape(20).data='0';
dynamic(1).desc='mass of the wing in slugs';
dynamic(1).var='mass';
dynamic(1).data='310.5';
dynamic(2).desc='acceleration of gravity ft/s^2';
dynamic(2).var='g';
dynamic(2).data='32.2';
dynamic(3).desc='center of gravity [x,y,z] relative to zeta';
dynamic(3).var='cg';
dynamic(3).data='[7,0,0]';
dynamic(4).desc='mass moment of inertia Iyy slugs ft^2';
dynamic(4).var='Iyy';
dynamic(4).data='50000';
dynamic(5).desc='air density slugs/ft^3';
dynamic(5).var='rho';
dynamic(5).data='2.38e-3';
dynamic(6).desc='initial conditions [uo wo qo tho xo zo tau_camber tau_reflex tau_twist tau_thrust]';
dynamic(6).var='xinit';
dynamic(6).data='[400 0 0 0 0 0 0 0 0 0]';
dynamic(7).desc='time constant for camber';
dynamic(7).var='tau_camber';
dynamic(7).data='.3';
dynamic(8).desc='time constant for reflex';
dynamic(8).var='tau_reflex';
dynamic(8).data='.3';
dynamic(9).desc='time constant for twist';
dynamic(9).var='tau_twist';
dynamic(9).data='.3';
dynamic(10).desc='time constant for thrust';
dynamic(10).var='tau_thrust';
dynamic(10).data='.3';
ratio=1;
H=[1000 0 0 0 0 0 0 0 0 0 0 0; 0 1 0 0 0 0 0 0 0 0 0 0; 0 0 1 0 0 0 0 0 0 0 0 0];
Q1=1/ratio*H'*eye(3)*H;
Q2=ratio*[100 0 0 0; 0 100 0 0; 0 0 100 0; 0 0 0 100];
in(1,:)=[0 0 0 0];
in(2,:)=[.5 .041 200 0];
in(3,:)=[1 .082 400 0];
in(4,:)=[1.5 .124 600 0];
in(5,:)=[11.5 .124 4600 -500];



%============================================================
% Trim information
%============================================================
Inputs were found to be:
Camber = -0.000
Reflex = -0.085
Twist = -0.109
Thrust = -0.276

States at the trim point:
u = 400.000
w = -0.000
th = -0.000
q = 0.000

The change in states when the trim was found:
du = -0.00000
dw = 0.00000
th = 0.00000
dq = -0.00000



%============================================================
% Linearization information
%============================================================
The linearized matrices

A matrix
0.0014  0.0724  -0.2221  -32.2000  0.0000  0.0000 0.0308 -0.0115 -0.0806 1.0000
-0.1610  -1.4219  403.8124  0.0000  0.0000  0.0000 -21.4709 21.4709 -20.6110 0.0000
0.0000  -0.0014  -0.0193  0.0000  0.0000  0.0000 -0.4040 0.5742 -0.0966 0.0000
0.0000  0.0000  1.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
```

```
1.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  1.0000  0.0000  -400.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 -3.3333 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 -3.3333 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 -3.3333 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 -3.3333

 B matrix
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
3.3333  0.0000  0.0000 0.0000
0.0000  3.3333  0.0000 0.0000
0.0000  0.0000  3.3333 0.0000
0.0000  0.0000  0.0000 3.3333

 C matrix
1.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  1.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  0.0000  1.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  1.0000  0.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  1.0000  0.0000 0.0000 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  1.0000 0.0000 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 1.0000 0.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 1.0000 0.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 1.0000 0.0000
0.0000  0.0000  0.0000  0.0000  0.0000  0.0000 0.0000 0.0000 0.0000 1.0000

 D matrix
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000
0.0000  0.0000  0.0000 0.0000



%============================================================
% Control gains
%============================================================
Open loop eigenvalues
0.000  0.000 i
0.000  0.000 i
-0.734  0.293 i
-0.734  -0.293 i
0.014  0.106 i
0.014  -0.106 i
-3.333  0.000 i
-3.333  0.000 i
-3.333  0.000 i
-3.333  0.000 i

Closed loop eigenvalues
-1.562  3.362 i
-1.562  -3.362 i
-3.945  1.279 i
-3.945  -1.279 i
-1.548  0.000 i
-0.654  0.825 i
-0.654  -0.825 i
-0.232  0.400 i
-0.232  -0.400 i
-0.468  0.000 i
-3.329  0.000 i
-3.333  0.000 i
-3.333  0.000 i

Control gains K
-59.528  0.004  0.000  0.041  -0.001  -13.067  -37.691  0.018  -0.007 0.744 -1.049 0.192 0.011
68.527  -0.009  -0.051  -0.023  -0.051  21.777  80.420  -0.027  -0.097 -1.049 1.595 -0.071 -0.005
-41.937  -0.018  -0.084  -0.079  -0.090  2.381  37.092  -0.062  -0.176 0.192 -0.071 0.394 -0.020
1.331  0.098  -0.020  1.042  0.001  -0.145  -1.071  0.450  -0.069 0.011 -0.005 -0.020 0.275



%============================================================
% Simulation output data
```

```
%============================================================
The data is stored in "simout.mat" in the form of a structure array.
The description is under simout.desc as shown below:
The values can be found under simout.data
1 time
2 desired theta
3 desired x
4 desired z
5 camber
6 reflex
7 twist
8 thrust
9 u, velocity in x-direction relative to the airfoil
10 w, velocity in z-direction relative to the airfoil
11 q, angular velocity about the y-axis relative to the airfoil
12 th, pitch angle relative to inertial-axis
13 x, position relative to the inertial-axis
14 z, position relative to the inertial-axis
15 Fx, force in x-direction at cg
16 Fz, force in z-direction at cg
17 M, moment about cg
18 x_zeta, x-locations that make up the wing
19 y_zeta, y-locations that make up the wing
20 z_zeta, z-locations that make up the wing
21 ptx, x-locations for forces [fx, fy, fz]
22 pty, y-locations for forces [fx, fy, fz]
23 ptz, z-locations for forces [fx, fy, fz]
24 fx, forces in the x-direction located at [ptx, pty, ptz]
25 fy, forces in the y-direction located at [ptx, pty, ptz]
26 fz, forces in the z-direction located at [ptx, pty, ptz]
27 theta_t, twist about the rotation axis as a function of span
28 m, moment about the rotation axis
29 px, power in the x-direction located at [ptx, pty, ptz]
30 pz, power in the z-direction located at [ptx, pty, ptz]
31 pth, power to twist the spanwise section located at pty(1,:)

 For example to plot the state u as a function of time type --> "plot(simout(1).data,simout(9).data)"



%============================================================
% Simulation output data
%============================================================
A figure showing the total power and work of the simulation has been exported as power.bmp
A figure showing the force vs disp of the actuator with the most costly power consumption has been exported as force_disp.bmp
A figure showing the control inputs of the simulation has been exported as control.bmp




%============================================================
% Movie output data
%============================================================
A (800x600) movie demonstrating the tracking capabilities of the wing has been exported as tracking.avi
A (800x600) movie demonstrating the power requirements of the wing has been exported as power.avi



%============================================================
% Ending simulation time
%============================================================
Simulation ended at 14-Nov-2001 14:14:47
```

Appendix B

Code Used in the Simulation

The code used in to simulate the model is a combination of MATLAB m-files and Simulink model files. Due to the size and integration of the code, it would not work well to include it in text form. Therefore, in order to obtain a copy of the code please contact Dr. Harry H. Robertshaw at the Center for Intelligent Material Systems and Structures, CIMSS.

Dr. Harry H. Robertshaw
Center for Intelligent Material Systems and Sturctures
Durham Hall
Blacksburg VA, 24060

References

[N. S. Khot, et. al., 1998] N. S. Khot, K. Appa, J. Ausman, and F. E. Eastep, "Deformation of a Flexible Wing Using an Actuating System for a Rolling Manuever without Ailerons", AIAA-98-1802, 39th AIAA Structures, Structural Dyns, and Materials Conference, Long Beach, California, April 1998

[F. Campanile, et. al., 2000] F. Campanile, O. Seack, and D. Sachau, "The Belt-Rib Concept for Variable-Camber Airfoils: Recent Developments," Paper No. 3985-11, presented at the SPIE Symposium on Smart Structures and Materials, SPIE Vol. 3985, Newport Beach, CA, March 6-9, 2000

[L. B. Scherer, et. al., 1999] L. B. Scherer, C. A. Martin, M. West, J. P. Florance, C. D. Wieseman, A. W. Burner, and G. A. Fleming, "DARPA/AFRL/NASA Smart Wing Second Wind Tunnel Test Results," Paper No. 3674-28, presented at the SPIE Symposium on Smart Structures and Materials, SPIE Vol. 3674, Newport Beach, CA, March 1-4, 1999

[Michael A. Scott, et. al., 1998] Michael A. Scott, Raymond C. Montgomery, and Robert P. Weston, "Subsonic Maneuvering Effectiveness of High Performance Aircraft Which Employ Quasi-Static Shape Change Devices," SPIE Vol. 33226, 1998

[D. J. Inman, et. al., 2000] Daniel J. Inman, Frank H. Gern, Harry H. Robertshaw, Rakesh K. Kapania, Greg Pettit, Anand Natarajan, and Erwin Sulaeman, "Comments on prospects of fully adaptive aircraft wings," SPIE Paper SS 4332-01, 8th International Symposium on Smart Structures and Materials, 2000

[J. J. Spillman, 1992] J. J. Spillman, "The use of variable camber to reduce drag, weight, and costs of transport aircraft," Aeronautical Journal, January 1992

[J. O. Simpson, et. al., 1998] J. O. Simpson, S. A. Wise, R. G. Bryant, R. J. Cano, T. S. Gates, J. A. Hinkley, R.S. Rogowski, and K. S. Whitley, "Innovative Materials for Aircraft Morphing," SPIE's 5th Annual International Symposium on Smart Structures and Materials, San Diego, CA, pp. 10, 1998

[Jones, 1990] Jones, Robert T. *Wing Theory*, Princeton University Press, Princeton, New Jersey, 1990

[Munson et. al., 1998] Muson B. R., Young D. F., and Okiishi T. H. *Fundamentals of Fluid Mechanics*, John Wiley and Sons, Inc, 1998

[Bertin and Smith, 1998] Bertin, John J. and Smith, Michael L. *Aerodynamics for Engineers*, Printice-Hall, Inc., New Jersey, 1998

[Bryson, 1994] Bryson, Authur E. Jr. *Control of Spacecraft and Aircraft*, Princeton University Press, Princeton, New Jersey, 1994

[Ashley, 1992] Ashley, Holt *Engineering Analysis of Flight Vehicles*, Dover Publications, Inc., New York, 1992

[Ogata, 1990] Ogata, Katsuhiko, *Modern Control Engineering*, Printice-Hall, Inc. Englewood Clifss, New Jersey 07632, 1990

[Gern, et. al., 2001] Gern, F.H., Inman, D.J., and Kapania, R.K, *Structural and Aeroelastic Modeling of General Planform UCAV Wings With Morphing Airfoils* AIAA Paper 2001-1369, SDM Conference, Seattle, WA, April 16-19, 2001.

[Abbott and Doenhoff, 1959] Abbott, H. Ira and Doenhoff, von E. Albert *Theory of Wing Sections*, Dover Publications, Inc., New York, 1959

Vita

Gregory W. Pettit

EDUCATION:

2000-2001 Virginia Polytechnic Institute and State University, Blacksburg, VA, Master of Science in Mechanical Engineering December 2001

1995-2000 Virginia Polytechnic Institute and State University, Blacksburg, VA, Bachelor of Science in Mechanical Engineering, Minor in Mathematics

1992-1994 New River Community College Dublin, VA, Associates Degree in Machine Technology

AWARDS:

5 Year BS/MS Degree Program

Virginia Space Grant Consortium Fellowship, 2001-2002

Pratt Fellowship, 2000-2001

CIMSS Undergraduate Research Internship, 1999-2000, Undergraduate Fellowship

Society of American Military Engineers Scholarship, 1999

PUBLICATIONS AND ACCOMPLISHMENTS:

Gregory W. Pettit, Harry H. Robertshaw, and Daniel J. Inman, "Morphing Wings for Unmanned Aircraft", Smart Materials Bulletin, ISSN 1471-3918 November 2001

Gregory W. Pettit, Harry H. Robertshaw, and Daniel J. Inman, "Model to Determine the Performance Requirements of Adaptive Materials used to Morph a Wing with an Aerodynamic Load," 12th International Conference on Adaptive Structures and Technologies

Gregory W. Pettit, Harry H. Robertshaw, Frank H. Gern, and Daniel J. Inman, "A Model to Evaluate the Aerodynamic Energy Requirements of Active Materials in Morphing

Wings", 2001 ASME Design Engineering Technical Conference

Daniel J. Inman, Frank H. Gern, Harry H. Robertshaw, Rakesh K. Kapania,Greg Pettit, Anand Natarajan, and Erwin Sulaeman, "Comments on prospects of fully adaptive aircraft wings," SPIE Paper SS 4332-01, 8th InternationalSymposium on Smart Structures and Materialsn Pilot, September 1998, Single Engine Land, 250 hours Hybrid Electric Vehicle Team, 1995-1996 and 1999-2000