Chapter 1

Introduction

An ad in the classified section of a national newspaper reads, "Wanted, factory personnel, required to prepare sub-assemblies for small engine manufacturer. Contract position. $8/hr." Common manufacturers with low volume contracts of moderately difficult assembly operations greatly benefit from innate human dexterity. From an early age a human being has the ability to put simple objects together. For example, a child can be seen trying to put a star shaped peg into its appropriate hole. The child may try the round hole first, and then the square, but eventually the youngster will insert the peg into the star shaped hole. As the peg is inserted the child automatically corrects for misalignment by moving the peg around until the peg slides in successfully. It is simple to see that if a child can handle such a task, then an adult can handle a more complicated task that requires fine control of their motor skills. However, such assembly tasks in modern assembly lines are tedious slow, and expensive. Potential assembly failures arise due to human error that may be attributed to many consecutive hours of repetitive tasks. Though operator rotation throughout the shift is one way to reduce assembly failure, it still does not provide a high quality cost-effective solution to the assembly of large volume contracts.

However, the desire to automatically assemble products within a manufacturing line is not entirely instigated by a company's demand for more profits, but instead, is driven by the global marketplace. Productivity was seen as a main area for improvement in order to be competitive in the global economy (Nevins, 1980). An increase in productivity meant that a manufacturer must decrease the man-hours, materials, energy, and capital required in the production of industrial products. Simultaneously, good morale in the workforce is obtained via a better quality of life for the laborer.

Though the need to be more productive is receiving current headlines, the Russians first considered the actual problem of automatic assembly in the 1960's. Though the motivation for these studies was not completely identified, they did provide the groundwork for the Americans once the documents were “edited” and translated. In one study, the author proposed sending a peg on a pre-described search pattern (Savishchenko, 1965). Here the peg, being held in a fixture, is moved across the surface of the mating part in a sinusoidal or spiral search pattern. Through experimentation, the author found that the type of search pattern depended on the geometry of the parts to be mated. In a study as to the automatic assembly of large parts, the author found through static load analysis, that horizontal assembly was most effective. In this form of assembly the female part has one contact point on a centering mandrel and another on the peg (Andreev, 1966). The female part corrects its orientation while it slides onto the peg. During this period, work was underway to introduce compliance in the automatic assembly problem of the peg-in-hole. Though the analysis was still two-dimensional, a research effort was conducted that considered angular as well as axial misalignments of the peg. The primary focus here was to avoid assembly failure by minimizing the insertion force. Through a geometric analysis, the author found that the chamfer on the peg should be constructed of a curved radius that is a function of the tilt angle of the peg. This minimized the insertion force thereby reducing the chances of failure (Laktionov, 1966).

These previous studies considered the peg and hole as rigid bodies held in rigid fixtures. Compliance in the gripper, or fixture that held the peg, was introduced a year later in an experimental analysis. Here the author devised two alternative work-holding systems, one that modeled the elasticity of the once rigid gripper, and another that permitted six degrees of freedom in the deflection of the peg. The author found that to compensate for the alignment errors of the peg it was possible to displace the assembly fixtures, but the permissible amount of displacement in the assembly fixture must be greater than the sum of all the misalignment errors (Karelin, 1967). Researchers were still interested in reducing the assembly force, and another experimental study determined that the contact area which produced the smallest contact stress was obtained with a logarithmic chamfer at the end of the peg (Andreev, 1969). This type of chamfer, though difficult to machine at that time, outperformed the angular and radial chamfers that are common today. Though the assembly problems were primarily interested in peg-in-hole applications, they did cover press-fit and force-fit assemblies. Here, the researchers avoided the large assembly forces by heating up the hole until a clearance was available (Andreev, 1972). An equation was derived that modeled the permissible amount of misalignment as a function of the heat applied to the hole. All of the results of the Russian researchers were partially based on minimizing the assembly force in order to reduce but not eliminate the chance for a failed assembly. Actually, the idea of introducing some form of compliance at the peg was not forgotten and re-emerged in the American literature several years later.

Whitney (1982) published a collection of research results that defined the mating events that occur between a peg and hole and the forces that arise during a failure mode. Here, a failure mode is defined as either *jamming*, where the applied forces are out of proportion with the contact state no longer permitting assembly, or as *wedging*, where the contact forces deform the parts and assembly is not possible without reorienting. The Remote Center Compliance (RCC) device was utilized in the research to provide the support for the peg since it offered the elasticity necessary for successful assembly. The remote center of compliance in the device is a point projected into space from the support. Any load acting at this point will cause the support to deflect in pure translation, and any moment about this point will deflect the support in pure rotation. This type of support for the peg provided the ability to model its successful path into the hole. Qiao (et al., 1994) conducted a similar analysis as to the forces that permit successful assembly, but came up with a pre-described approach such that the peg assembled successfully without feed-back or with the use of a RCC. Similarly, a study was done that developed a hybrid force-position strategy using active compliance to successfully insert convex three-dimensional pegs (Strip, 1988). However, both studies did not consider the potential for wedging conditions.

The techniques of passive assembly have proved valuable since successful insertion can be completed without the need for feedback devices. Other assembly primitives, such as the peg-in-hole, multiple peg-in-hole and polygonal part insertions have been successfully analyzed with constraint networks and demonstrated in practice (Sathirakul, 1998). Here, a network of contact states is created that determines the path of successful assembly based on static load relations and failure modes. Contact states are the collection of contacts between two parts at each moment during an assembly operation.

Threaded fastener insertion, a ubiquitous assembly primitive and the primary focus of this thesis, is an assembly problem that has faced some scrutiny in the literature. It was first considered by Blaer (1962) who found that in order to assemble threaded parts automatically, the bolt must be centered in the nut and supported by a longitudinal force before rotation of the bolt can begin. A purely geometric and non-frictional study by Romanov (1964) determined that the chamfer angle on the bolt must be less than the chamfer angle of the nut to avoid jamming. Nevins and Whitney (1989) defined the potential errors associated with threaded assembly as angular or helical thread mismatch - a condition more commonly understood as cross threading. During angular thread mismatch, the rotational position of the fastener causes a portion of the thread to contact the nut below the first full thread. After a small rotation, the fastener becomes wedged in the nut. In the case of helical thread mismatch, the fastener is not in phase with the nut thread, resulting in a thread deformation also known as “stripping”.

A more recent attempt to analyze threaded fastener insertion was made by Nicolson and Fearing (1991,1993), but their goal was not a passive solution to the thread-mating problem. Instead, an apparatus consisting of a two degree of freedom robot, two servomotors to correct for lateral misalignment, and a rotational stepper motor was designed to correct the positional uncertainty. Although their results were successful, there is a relative cost to incorporate the control mechanism. A compliant mechanism attached to a robotic arm would allow the assembly of threaded fasteners without the need for feedback devices, hence, achieving greater assembly speeds limited only by the positional accuracy of the robotic manipulator and the inertia of the parts involved.

Dhayagude, Gao, and Mrad (1996) developed fuzzy logic control of an automated threaded assembly. Here, the control system is able to handle large variability in operational parameters. The clamping forces of the bolt coupled with the torque of the driver are two of the measurements used by the fuzzy logic controller. Torque levels are compared to specifications during assembly to determine if the bolt jammed or slipped. In addition to the input, the control unit captures the system dynamics qualitatively, and executes the control heuristic in real-time. The authors conclude that, though the method is complex and costly, it will prove valuable in delicate assembly situations where the precise clamp load must be known.

1.1 Research Objectives

Although threaded assembly of C-mount camera lenses has been demonstrated with a spatial remote center of compliance (SRCC), it is not fully understood, and therefore can not be extrapolated to other cases. Currently, the assembly is completed by the conventional heuristic of rotating the bolt counter clockwise until a "snap" is heard. The "snap" results from a sudden change in the height of the bolt and occurs when the bolt is in phase with the nut. From this point, it is known that any clockwise rotation results in a successful assembly provided that the orientation is maintained. It is of interest to determine the nature of contact between the bolt and nut prior to an in-phase condition, and how the multiple contact states can be constrained to permit successful completion of the threaded fastener task.

A contact analysis between the bolt and nut will provide the information necessary to verify that the SRCC responds correctly to the forces involved during assembly. The main objectives of the contact analysis are:

- Identify the surface regions in contact and the resultant normals;
- Enumerate the multiple contact point configurations; and
- Determine the contact state network that defines assembly paths.

Once the surface regions are known, the resultant normal for the bolt and nut can be calculated for each orientation. This data will be utilized on a contact state network where the optimal path (i.e., the path of least resistance) can be determined. The knowledge of multiple contact points and their depth will allow one to determine the potential for jamming or wedging. The analysis of this problem shall begin with a two-dimensional analysis of the problem.

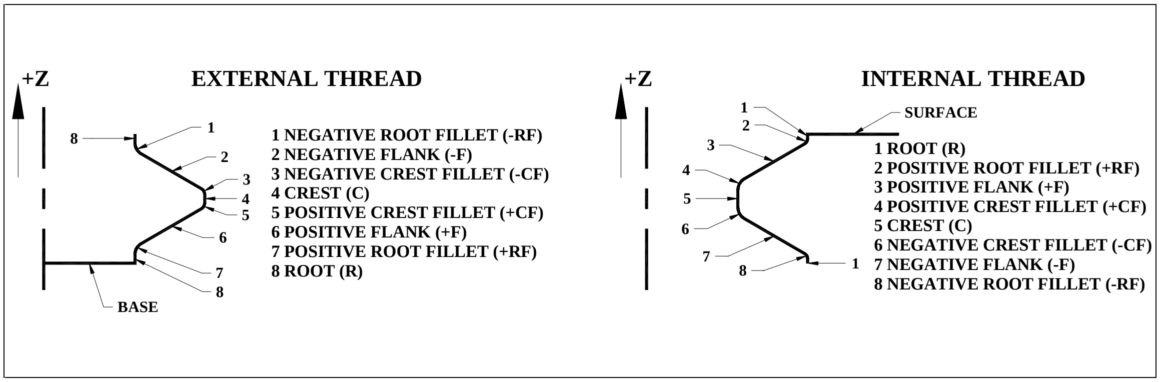
Chapter 2

Planar Contact Analysis

Instead of trying to visualize spatial contact configurations, it will be beneficial to begin the analysis of this problem in two dimensions. Any insight into the development of contact location algorithms can then be gleaned and introduced into the inevitable three-dimensional analysis. In this chapter we identify the standard thread design for internal and external threads, and establish the parametric equations that define the cross section of such threads so that the contact points can be identified.

2.1 Screw Thread Definitions

The internal and external threads are defined by the American National Standard for Unified Screw Threads (Oberg, 1992). Fillet radii bound the crest and root in order to approximate an actual manufactured thread.



+Z
EXTERNAL THREAD
8
1
2
3
4
5
6
7
8
BASE
1 NEGATIVE ROOT FILLET (-RF)
2 NEGATIVE FLANK (-F)
3 NEGATIVE CREST FILLET (-CF)
4 CREST (C)
5 POSITIVE CREST FILLET (+CF)
6 POSITIVE FLANK (+F)
7 POSITIVE ROOT FILLET (+RF)
8 ROOT (R)
+Z
INTERNAL THREAD
1
2
SURFACE
3
4
5
6
7
8
1
1 ROOT (R)
2 POSITIVE ROOT FILLET (+RF)
3 POSITIVE FLANK (+F)
4 POSITIVE CREST FILLET (+CF)
5 CREST (C)
6 NEGATIVE CREST FILLET (-CF)
7 NEGATIVE FLANK (-F)
8 NEGATIVE ROOT FILLET (-RF)

Figure 2. 1 Internal & External Thread Profiles

The bolt (or external thread) is made up of 9 regions, as shown in Figure 2.1 above. The flank is defined as positive if the radius of the bolt increases when measured in the positive azimuth (+Z) direction. Likewise, the flank is defined as negative if the radius of the bolt decreases in the positive azimuth direction. This logic is identical for the crest fillet and the root fillet. In addition to these regions the bolt has constant radius surfaces defined as the crest and the root. The ninth, and final, region is known as the base, since the bottom of the bolt completes a surface region and has the potential to become involved during contact.

The nut follows a similar definition, however, due to the nature of an internal thread, the value of the radius is a maximum at the root and a minimum at the crest as measured from the nut azimuth axis. The designation of positive or negative as applied to the flank, crest fillet, and root fillet of the nut follows the same definition as provided for the bolt. The ninth region is known as the surface, and like the bolt, it completes the surface region of the nut that has the potential to come into contact. The top of the bolt and the bottom of the nut are not included here because these regions will not come into contact once the position and orientation errors are introduced.

The pitch of a thread is defined as the vertical change of the thread per one revolution. It provides a threaded assembly the ability to mesh. In order for two threaded parts to mate, the pitch of each must be the same, or the parts will wedge together and become damaged. The amount of vertical change can be calculated with the following equation:

$$z = \frac{p\theta}{2\pi} \qquad 0 \le \theta \le 2\pi \tag{2. 1}$$

Here, p is the pitch in inches (or millimeters) and θ is the angular location in radians. Note that any profile, such as the external thread, which is helically swept, will repeat itself after one full revolution about the azimuth axis.

2.2 Parametric Equation Derivations

When the bottom of the bolt is viewed parallel to the negative azimuth axis, a distinct cross-section will be seen. It has the appearance of being elliptical, and can easily be defined by mapping the cross section of the bolt from the X-Z plane to the X-Y plane. If one cut the bolt perpendicular to its azimuth axis, then the radius as measured counterclockwise will pass through all of the regions (with the exception of the base), since one full rotation about the azimuth axis is equivalent to one pitch of the external thread. Figure 2.2 illustrates what the cross-section looks like from this view.

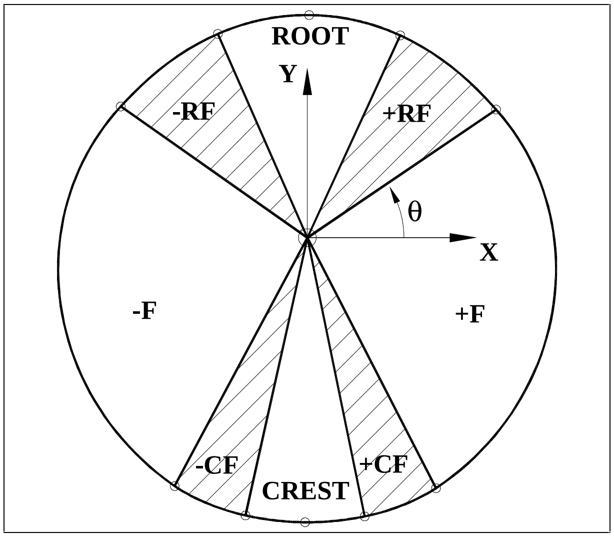


Figure 2. 2 General Thread Cross-Section

Now, as this cross section rotates counter-clockwise, the radius as measured along the X-axis will increase for the positive regions and decrease for the negative ones. As the radii are measured this way, the positive regions begin at their maximum and end at their minimum values while the negative regions begin at their minimum and end at their maximum values. The reason for this lies in the handedness of the thread, if the thread is right-handed, then you move in the positive azimuth direction when rotating counter-clockwise. If the thread is left-handed, then you move in the positive azimuth direction when rotating clockwise. In this analysis, we will only deal with right-handed threaded fasteners.

Let's begin with two examples where the equation that defines the radius of the bolt as a function of θ will be derived — where θ is the angle measured from the origin of the bolt to a point on the bolt cross section. As will be explained in chapter 3, solid modeling software provided the original orientation of the bolt and nut. Thus, for the remainder of this chapter, the initial values are predetermined by the solid modeling software.

2.2.1 Negative Root Fillet

Two known points of the root fillet are selected which are the starting and ending points of the negative root fillet region (Figure 2.3). The starting point of the negative root fillet (-RF) is simply the root radius, b_{min}. The ending point is the root radius plus one half the root fillet radius (denoted rr$_b$) which is calculated through a simple geometric analysis of the external thread profile.

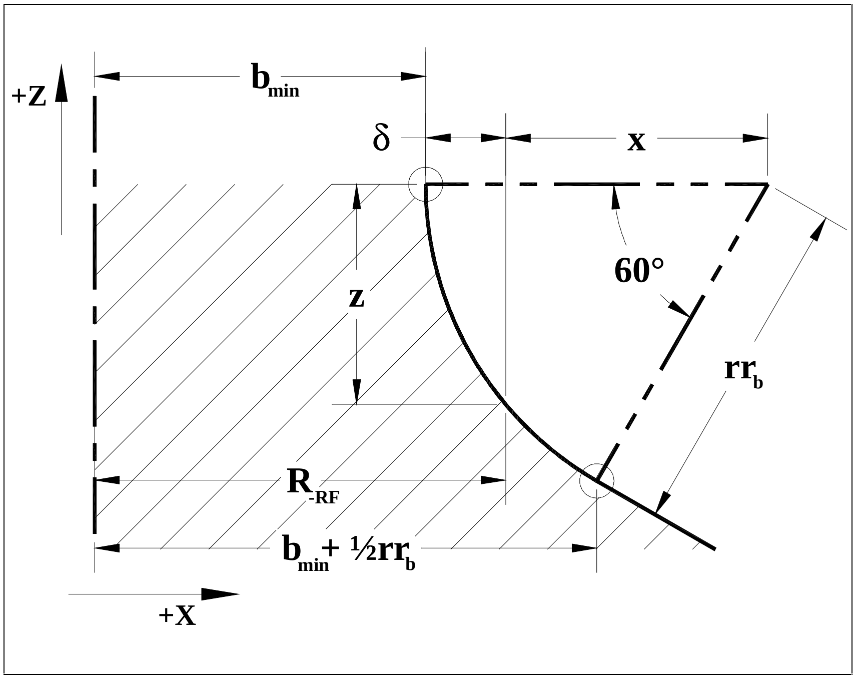


Figure 2. 3 Negative Root Fillet

Here, δ refers to the amount the radius changes as one moves in the negative azimuth (z) direction. The change in radius is added to the initial value and forms the following equation:

$$x = \sqrt{rr_b^2 - z^2} \tag{2. 2}$$

$$\delta = rr_b - x = rr_b - \sqrt{rr_b^2 - z^2} \tag{2. 3}$$

$$R_{-RF}(z) = (b_{\min} + rr_b) - \sqrt{rr_b^2 - z^2} \tag{2. 4}$$

Substituting equation 2.1 for the azimuth height, *z*, one obtains the radius of the negative root fillet as a function of θ:

$$R_{-RF}(\theta) = (b_{\min} + rr_b) - \sqrt{rr_b^2 - L^2(\theta - \theta_{ini})^2} \qquad L = \left(\frac{p}{2\pi}\right) \tag{2. 5}$$

The θ$_{ini}$ value is a correction factor that allows this part of the thread profile to begin at a point other than zero.

2.2.2 Negative Flank

The negative flank (-F) precedes the negative root fillet, and a similar procedure is used to derive the radius of the negative flank as a function of azimuth height (Figure 2.4). The radius is equal to b_{min} plus one half the root fillet radius at the starting point, and equals b_{max} minus one half the crest fillet radius at the termination point.

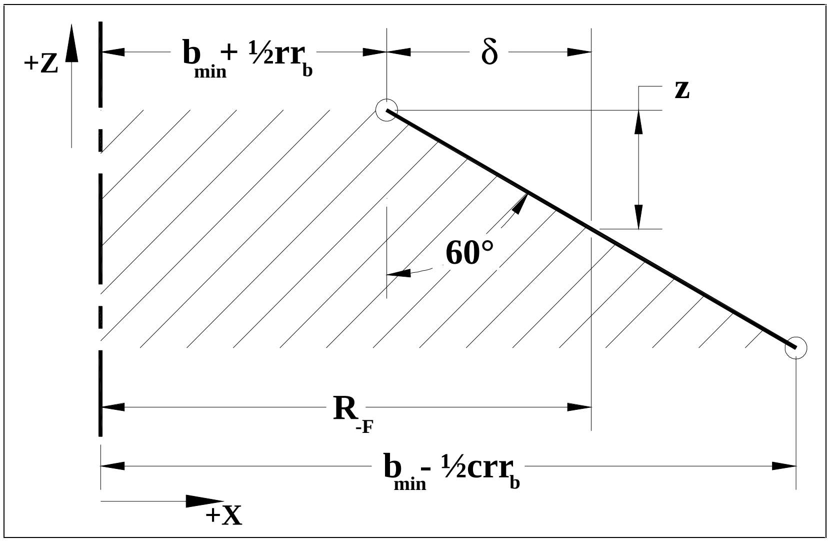


Figure 2. 4 Negative Flank

The radius changes at a rate dependent on the slope of the flank line, which again is denoted by δ. The change in radius is added to the initial value and the following equation of the negative flank radius is derived:

$$\tan(60°) = \frac{\delta}{z} \qquad \textbf{(2. 6)}$$

$$\delta = z\tan(60°) = \sqrt{3}z \qquad \textbf{(2. 7)}$$

$$R_{-F}(z) = \left(b_{\min} + \frac{1}{2}rr_b\right) + \sqrt{3}z \qquad \textbf{(2. 8)}$$

Substituting in equation 2.1 for the azimuth height, *z*, one obtains the radius for the negative flank as a function of θ:

$$R_{-F}(\theta) = \left(b_{\min} + \frac{1}{2}rr_b\right) + K(\theta - \theta_{ini}) \qquad K = \frac{\sqrt{3}p}{2\pi} \qquad \textbf{(2. 9)}$$

This process is repeated for the remaining six regions of the bolt and for all of the regions of the nut. The entire set of parametric equations can be found in the appendix.

2.3 Planar Contact Regions

A solid model of the threaded parts will be introduced later on in chapter three, which will provide the orientations of the bolt and nut cross sections *a priori*. Since the two-dimensional analysis was constructed based on the solid models, it is necessary to show how the two bodies contact one another prior to any rotation about the Z-axis. Figure 2.5 shows a planar view when looking down onto the XY plane and a cross-sectional view when looking perpendicular to the XZ plane. The relative alignment of the bolt and nut when the phase angle is zero is shown in the cross section for reference.

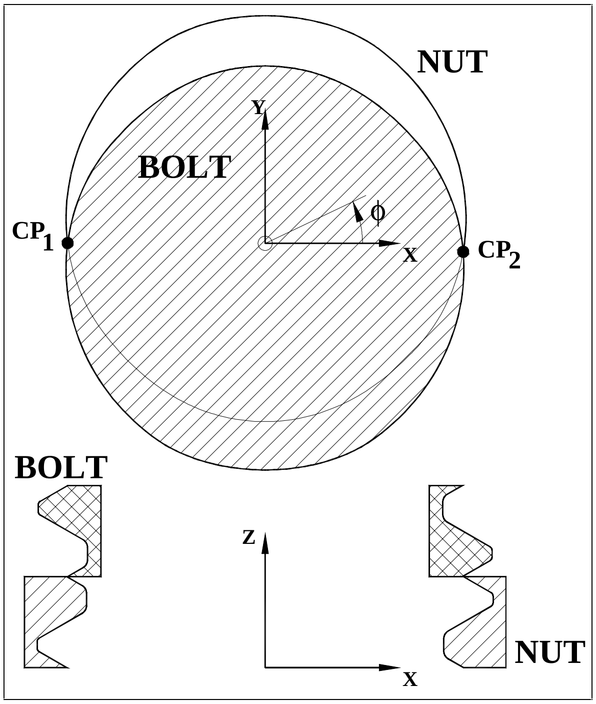


Figure 2. 5 Initial Orientation of Bolt & Nut

Since the bolt and nut are made up of nine regions each that have the potential to come into contact, there will be a set of combinations of these regions that represent the contacting surfaces. Given a value of phase, one simply would rotate the bolt cross section by that amount about the Z-axis, and equate the parametric equations to ascertain the contact point. In this planar example, there will always be two points of contact, since the only degree of freedom is a rotation about the Z-axis. The pair of combinations can be determined with the aid of an annular surface disk (Figure 2.6), which has been derived from the equations to follow in this section for each contact point. It describes the surface regions in contact given a value of phase, ϕ.

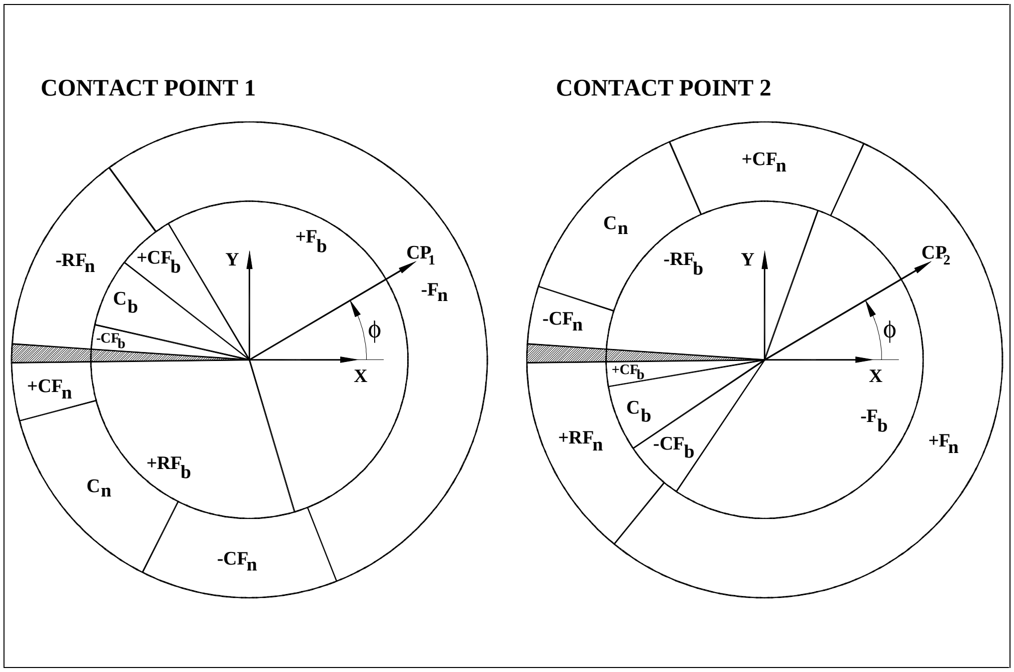


Figure 2. 6 Contact Region Combination Disk

The outer annulus identifies contact regions of the nut; the corresponding contact regions of the bolt are identified on the inner disk. The terminology used here is identical to that of Figure 2.1, above. For the example shown in Figure 2.6, at a relative phase angle of 30° (ϕ=30°), contact point 1 occurs between the negative flank of the nut and the positive flank of the bolt. The location of contact point 1 is not given in the figure, but can be found, along with the orientation of its contact normal, by referring to the data in appendix A. At this same phase angle, Figure 2.6 identifies contact point 2 comprising the positive flank of the nut touching the negative flank of the bolt. The dark region shown in the inner disk and outer annulus represents an in-phase condition at which time no contact occurs. Though the locations of the contact points vary with phase, it is of primary interest to identify the *regions* in contact so as to be able to calculate the common normal at the contact point, rather than establishing a relationship between phase (ϕ) and the contact locations.

2.4 Two Dimensional Contact Point Analysis

We are now able to set up an analytical method for locating where the cross section of the bolt intersects the cross section of the nut. Given the cross-section of the bolt placed on top of the nut cross-section, locate the two points where the radius of the bolt and nut are equal. The only degree of freedom for the bolt in this special case is the rotation about the azimuth axis, since there are no translation errors in the X and Y-axes, which are the only other remaining degrees of freedom. The azimuth rotation will be defined by the phase angle, ϕ. Once ϕ is given, then in order to solve for the intersecting points one must locate which radius equation of the bolt equals the radius equation of the nut. A simple example follows.

2.4.1 Example: $\phi = 30°$

Figure 2.7 highlights the regions that contain radii equality points at this value of ϕ. For the first contact point, the positive flank of the bolt intersects the negative flank of the nut. For the second contact point, the negative flank of the bolt intersects the positive flank of the nut.

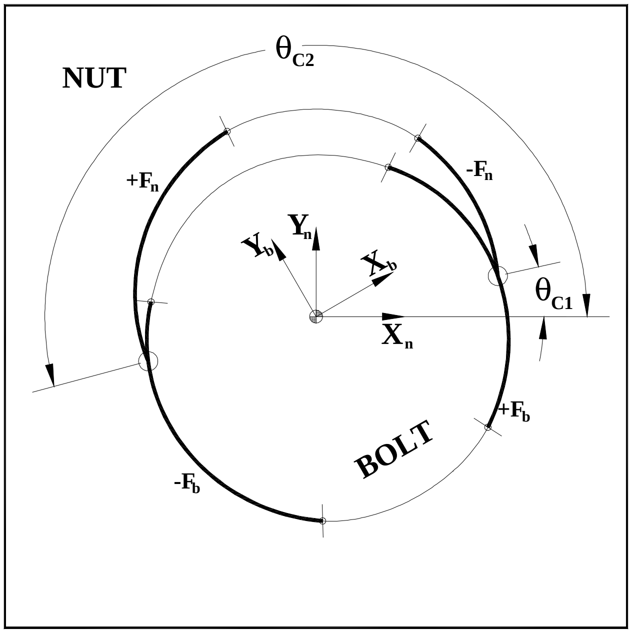


Figure 2. 7 Contact Points at $\phi = 30°$

The two contact points can be found by solving the following equations:

$$\text{Contact One} \quad R_b^{+F} = R_n^{-F} \tag{2. 10}$$

$$\text{Contact Two} \quad R_b^{-F} = R_n^{+F} \tag{2. 11}$$

Each of these equations can be expanded to the following:

$$\text{Contact One} \quad \left(b_{\max} - \frac{1}{2}crr_b\right) - K(\theta_b - \theta_{ini}^{b}) = \left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n + \theta_{ini}^{n}) \tag{2. 12}$$

$$\text{Contact Two} \quad \left(b_{\min} + \frac{1}{2}rr_b\right) + K(\theta_b - \theta_{ini}^{b}) = \left(n_{\max} - \frac{1}{2}rr_n\right) - K(\theta_n - \theta_{ini}^{n}) \tag{2. 13}$$

At first glance, it appears that we have one equation with two unknowns. However, a second equation that defines θ_b as a function of θ_n is available. From Figure 2.8, θ_b can be easily calculated by the following equation:

$$\#1 \quad \theta_b = 2\pi - (\phi - \theta_n) \tag{2. 14}$$

$$\#2 \quad \theta_b = \theta_n - \phi \tag{2. 15}$$

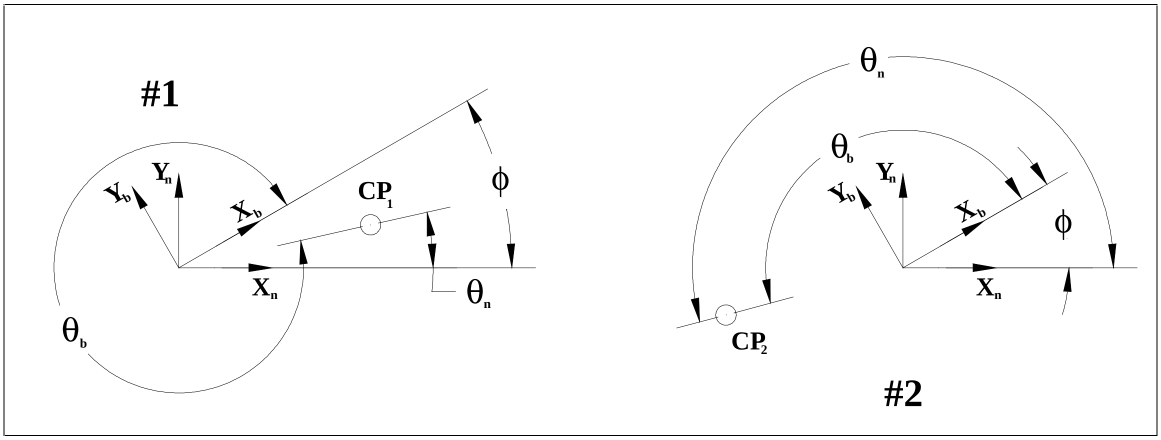


Figure 2. 8 θ_b Derivation

After a healthy round of algebraic manipulation the following two equations can be used to identify the angular location of the two contact points (Fig. 2.5). Their position in the X-Y plane can be determined by converting from Polar to Cartesian coordinates:

$$\theta_{c1}=\frac{1}{2}\left\{\frac{1}{K}\left[b_{\max}-n_{\min}-\frac{1}{2}(crr_b+crr_n)\right]+\theta_{ini}^{b}+\theta_{ini}^{n}+\phi\right\} \tag{2. 16}$$

$$\theta_{c2}=\frac{1}{2}\left\{\frac{1}{K}\left[n_{\max}-b_{\min}-\frac{1}{2}(rr_n+rr_b)\right]+\theta_{ini}^{b}+\theta_{ini}^{n}+\phi\right\} \tag{2. 17}$$

$$X_{c1}=R_n^{-F}(\theta_{c1})\cos(\theta_{c1}) \qquad Y_{c1}=R_n^{-F}(\theta_{c1})\sin(\theta_{c1}) \tag{2. 18}$$

$$X_{c2}=R_n^{+F}(\theta_{c2})\cos(\theta_{c2}) \qquad Y_{c2}=R_n^{+F}(\theta_{c2})\sin(\theta_{c2}) \tag{2. 19}$$

2.5 Critical Values of ϕ

By determining the critical angles of phase, one will know how to isolate the appropriate radii equations so as to enumerate the location of each contact point. Different radii formulae (versions of equations 2.5 and 2.9) are involved depending on the value of phase, since different regions of the bolt as well as the nut cross section will intersect. The following example illustrates the method to identify all of the critical ranges of phase. Here, the basic question is “What limiting value of ϕ is the negative flank of the bolt still equal to the positive flank of the nut as shown in Figure 2.5?” From visual inspection, it appears that the positive flank switches to the positive crest fillet of the nut, while the negative flank of the bolt contacts both regions of the nut. The value of the nut radius at the moment the switch occurs is:

$$R_n=n_{\min}+\left(\frac{1}{2}crr_n\right) \tag{2. 20}$$

This relationship allows the equation for the location of contact two (eq. 2.13) to be written as:

$$\left(b_{\min}+\frac{1}{2}rr_b\right)+K(\theta_b-\theta_{ini}^{b})=R_n \tag{2. 21}$$

$$\theta_b=\theta_n-\phi \tag{2. 22}$$

Subbing in for θ_b, and noting that θ_n is equal to the starting angle of the positive crest fillet (since that is where the switch from the positive flank to the positive crest fillet occurs) one obtains the following equation for the critical value of phase:

$$\phi_{cr} = \theta_n - \theta_{ini}^b + \frac{1}{K}\left[\left(b_{\min} + \frac{1}{2}rr_b\right) - R_n\right] \qquad \textbf{(2. 23)}$$

In order to automate the process of graphing multiple pivot axes, all of the critical phase angles must be calculated, so that the correct equalities can be analyzed. The critical values, along with the appropriate contact angle formulas are listed in appendix A. Figure 2.9 illustrates how the contact points vary with phase. The line which has a phase value attached to the endpoints connects each pair of contact points.

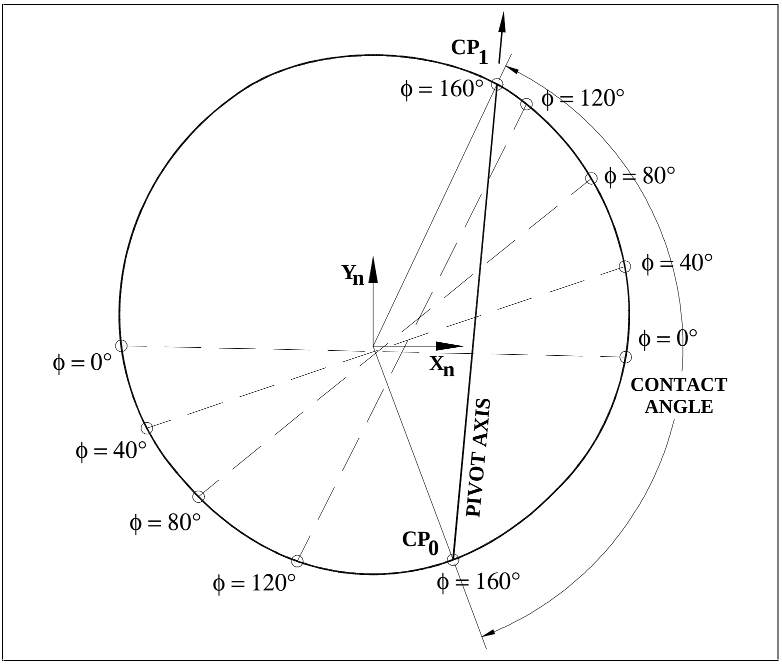


Figure 2. 9 Pivot Axis Position vs. Phase Angle

The angle that depicts the range where the bolt radius is larger than the nut radius is referred to as the contact angle. The line that connects the two contact points will be known as the pivot axis. Physically, the pivot axis is an axis that the bolt will rotate about in order to obtain the third contact point. Notice how the contact angle diminishes as the phase value (ϕ) increases until the bolt and nut are in-phase. It is of interest to determine if the distance between contact points tend towards zero, or if there is some limiting value. Fortunately, this question can be answered via proof by induction.

2.6 Curvature Verification

Hypothesis: When the bolt reaches the in-phase value of ϕ (3.0744 radians), line-line contact exists between the two bodies, thereby reaching a minimum straight-line distance between the two contact points. By line-line contact, we are stating that all of the bolt radii are equal to all of the nut radii, thus providing an infinite number of contact points between the two bodies.

The only way this could be true is if the curvature of the bolt cross-section is identical to the curvature of the nut cross-section. If so, then the points where the curvatures cease to equal are the limiting values of the pivot axis. Just before the bolt and nut are in-phase, the region of the nut edge that comes into contact with the bolt is defined by:

$$R_n^{-F}(\theta_n) = \left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^{n}) \qquad \textbf{(2. 24)}$$

The equation that locates a point on the curve defined by equation 2.24 is (in vector form):

$$\vec{\mathbf{P}} = R_n^{-F}(\theta_n)\cos(\theta_n)\hat{\mathbf{i}} + R_n^{-F}(\theta_n)\sin(\theta_n)\hat{\mathbf{j}} \qquad \textbf{(2. 25)}$$

The curvature of equation 2.25 is defined by:

$$k_n = \left[\frac{\mathbf{P}_{\mathbf{i}}'\mathbf{P}_{\mathbf{j}}'' - \mathbf{P}_{\mathbf{i}}''\mathbf{P}_{\mathbf{j}}'}{\left(\left|\mathbf{P}'\right|\right)^{3/2}}\right] \qquad \textbf{(2. 26)}$$

The first derivative of the position equation (2.25) is:

$$\begin{aligned}\left(\frac{d\vec{\mathbf{P}}}{d\theta_n}\right) = \vec{\mathbf{P}}' = &\left\{K\cos(\theta_n) - \left[\left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^{n})\right]\sin(\theta_n)\right\}\hat{\mathbf{i}} \\ &+ \left\{K\sin(\theta_n) + \left[\left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^{n})\right]\cos(\theta_n)\right\}\hat{\mathbf{j}}\end{aligned} \qquad \textbf{(2. 27)}$$

The second derivative of the position equation (2.25) is:

$$\begin{aligned}\left(\frac{d\vec{\mathbf{P}}'}{d'\theta_n}\right) = \vec{\mathbf{P}}'' = &\left\{-2K\sin(\theta_n) - \left[\left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^{n})\right]\cos(\theta_n)\right\}\hat{\mathbf{i}} \\ &+ \left\{2K\cos(\theta_n) - \left[\left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^{n})\right]\sin(\theta_n)\right\}\hat{\mathbf{j}}\end{aligned} \qquad \textbf{(2. 28)}$$

After substituting the necessary values into equation 2.26, the following relationship defines the curvature of the nut:

$$k_n = \frac{2K^2 + \left[\left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^n)\right]^2}{\left\{4K^2 + \left[\left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^n)\right]^2\right\}^{\frac{3}{2}}} \qquad \textbf{(2. 29)}$$

Just before the bolt and nut are in-phase, the radius equation for the bolt is:

$$R_b^{-F}(\theta_b) = b_{\min} + \left(\frac{1}{2}rr_b\right) + K(\theta_b - \theta_{ini}^b) \qquad \textbf{(2. 30)}$$

Following the same procedure as before, the curvature equation of the bolt is:

$$k_b = \frac{2K^2 + \left[\left(b_{\min} + \frac{1}{2}rr_b\right) + K(\theta_n - (\phi + \theta_{ini}^b))\right]^2}{\left\{4K^2 + \left[\left(b_{\min} + \frac{1}{2}rr_b\right) + K(\theta_n - (\phi + \theta_{ini}^b))\right]^2\right\}^{\frac{3}{2}}} \qquad \textbf{(2. 31)}$$

In order to prove that the curvature is the same it is sufficient to show that the following two portions of the equation are identical:

$$k_n = k_b \qquad \textbf{(2. 32)}$$

$$\frac{2K^2 + A_n^2}{\left[4K^2 + A_n^2\right]^{\frac{3}{2}}} = \frac{2K^2 + A_b^2}{\left[4K^2 + A_b^2\right]^{\frac{3}{2}}} \qquad \textbf{(2. 33)}$$

$$A_n = A_b \qquad \textbf{(2. 34)}$$

$$\left(b_{\min} + \frac{1}{2}rr_b\right) + K(\theta_n - (\phi + \theta_{ini}^b)) = \left(n_{\min} + \frac{1}{2}crr_n\right) + K(\theta_n - \theta_{ini}^n) \qquad \textbf{(2. 35)}$$

Rearranging equation 2.35, and solving for ϕ, one obtains:

$$\phi = \theta_{ini}^n - \left(\theta_{ini}^b + \frac{a}{K}\right) \qquad a = n_{\min} - b_{\min} \qquad \textbf{(2. 36)}$$

When the appropriate values are substituted, one obtains the following value for ϕ:

$$\phi = 176.15° = 3.0744\ rad$$

This value is identical to the critical value of ϕ, therefore, the curvature of the bolt and nut are identical. Thus, the contact angle reaches a minimum value instead of tending towards zero. This is beneficial to know, because if the line connecting the two contact points reduces to a point, the bolt will not have a predefined direction of rotation that was available with the pivot axis. Without a fully defined pivot axis, the next direction of rotation would not be known.

Though the assembly of a threaded fastener is truly a three-dimensional operation, a less complicated two-dimensional analysis of the problem provided some insight as to how the bolt and nut interact during assembly. The next chapter will extend this work into three dimensions by considering a spatial orientation of the bolt.

Chapter 3

Spatial Contact Analysis

Although the two dimensional analysis provided some insight into how the contacts between a bolt and nut behave during counter-clockwise rotation of the bolt, it did not provide the opportunity to analyze the potential for out-of-plane contacts. By modeling the assembly problem in three dimensions, one will be able to determine if any out-of-plane contacts exist. We begin by assuming that the bolt will obtain an out-of-plane contact once it achieves stability, which is defined as three distinct contact points. The following steps predict the path that the bolt will take to obtain stability:

1. Initial contact at predefined spatial orientation.
2. Rotation about a vector tangent to the surface of the nut at the initial contact point until the second contact is made.
3. Rotation about a pivot axis defined by the two contact points until a third contact is obtained.

The direction of rotation at the initial contact point will be negative to ensure that the bolt will rotate in the direction that corrects its spatial orientation. The direction of rotation about the pivot axis will vary depending on the location of the two contact points. The prediction above is based on the action following a minimization of the virtual work done on an unbiased elastic structure, such as an RCC.

3.1 Numerical Contact Analysis

In order to prove that the bolt follows this contact path, a supervisory code was written to isolate the contacts as they occurred in space. To facilitate the detection of these contacts, a software library, known as *RAPID* (Gottschalk, 1996, version 2.01), was used to determine contacts between two solid bodies given some initial orientation in space[1]. However, RAPID required the input of a solid model in a tessellated form. This meant that the bolt and nut must be modeled through solid modeling software, tessellated, and exported into a binary data file. RAPID uses the vertices of the triangles contained in the binary data file to store the model in memory. Once the models are stored, the user then

[1] The author gratefully acknowledges the creator of RAPID, Steven Gottschalk, for allowing the incorporation of his collision detection library into this analysis.

provides the spatial orientation of each model, and RAPID determines if the two parts are in contact. If they are in contact, RAPID returns a list of contact pairs – one triangle for the bolt and one triangle for the nut. However, these pairs are only the identification numbers of the triangles, not the coordinate values. It is up to the user to augment the library to calculate the geometric location of the contact coordinates.

3.2 Solid Model Definition

A solid modeler, *Pro/Engineer release 19*, was used to create the images of one pitch of the bolt and nut thread (#1/4-20 UNC). Pro/Engineer provides the user with the ability to export Stereolithography (.STL) files. .STL files represent the surfaces of a solid model as groups of small polygons, or in this case, groups of triangles. The quality of an .STL file can be controlled through the Pro/Engineer interface, and it depends on the deviation of the actual surface from the tessellated surface. One parameter that can be adjusted is the chord height. This specifies the maximum distance between a chord and a surface (Fig. 3.1).

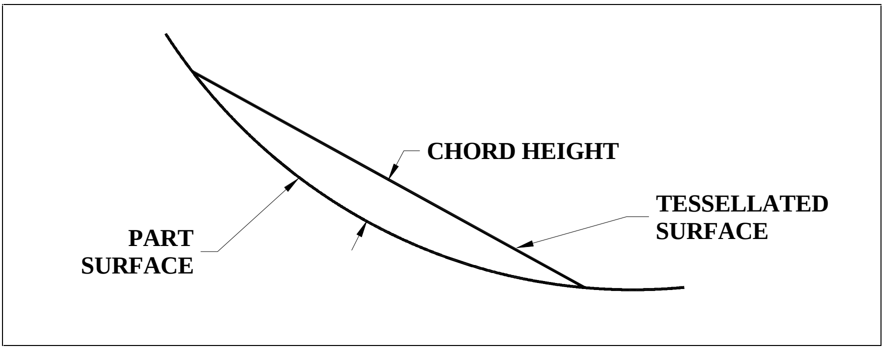


Figure 3. 1 Chord Height

The minimum value of chord height is determined by the software, and is a function of part accuracy. Another parameter to control the quality of the tessellated output is angle control. If a part has small radial features, like the dimple on a golf ball, then these surfaces will have very little definition in the tessellated output. Angle control regulates the amount of additional improvement provided along curves with small radii. Like the chord height parameter, angle control is bounded with the minimum being determined by the software. For the contact analysis between the bolt and nut, the best tessellation was desired, so the minimum values of chord height (1.0 x 10^{-4}) and angle control (0.5) were chosen for both the nut and bolt. The bolt was made hollow, and the nut was cut out of a 16-sided polygon to reduce the size of triangles that appeared on the top and bottom surfaces of the bolt and nut. This produced a model that deviated from the true surface by a few millionths of an inch, which is within the surface noise of a real threaded part (Fig. 3.2).

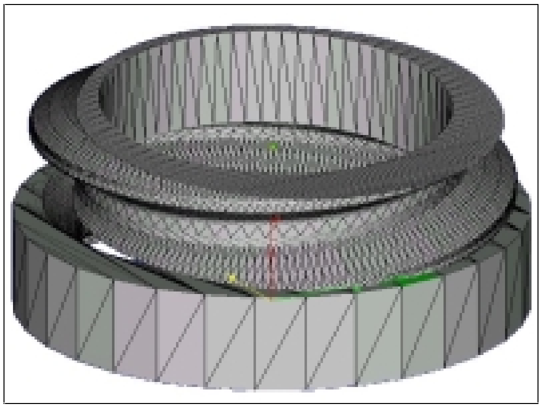


Figure 3. 2 Tessellated Solid Model – Bolt & Nut

The data in the .STL files is in binary format to maintain accuracy, and to reduce the size so that the models could be transported via diskette. Each triangle in the solid model is allotted 50 bytes, while the first 84 bytes of the file contain header information and the number of facets the model contains. The following depicts the breakdown of the binary file format (Fig. 3.3):

Address	Length	Type	Description
0	80	char	Header Information
80	4	long	#of facets in solid
First Facet (50 bytes):			
84	4	float	Normal (θ_x)
88	4	float	Normal (θ_y)
92	4	float	Normal (θ_z)
96	4	float	Vertex 1 (X-component)
100	4	float	Vertex 1 (Y-component)
104	4	float	Vertex 1 (Z-component)
108	4	float	Vertex 2 (X-component)
112	4	float	Vertex 2 (Y-component)
116	4	float	Vertex 2 (Z-component)
120	4	float	Vertex 3 (X-component)
124	4	float	Vertex 3 (Y-component)
128	4	float	Vertex 3 (Z-component)
132	2	short	Attribute Info. (Not used)
Second Facet (50 bytes):			
134 ….			

Figure 3.3 .STL File Output Description

When the model is sent into RAPID, only the vertex information is provided. A copy of what is sent to RAPID is maintained in memory, so when RAPID returns the list of contact pairs one can locate the triangle vertices involved. The type identifier indicates the variable declaration, which is single precision (float) for all of the vertices. The long and short identifiers are integers that require 4 and 2 bytes, respectively. Pro/Engineer did not provide the ability to export the vertices of the tessellation in double precision, because the .STL file format only uses single precision.

Before describing the contact location algorithm, we will begin with a brief introduction to vector algebra, which is required to understand the derivation of a line in symmetric form, and move immediately into the general equations that locate the approximate contact point between two intersecting triangles in space.

3.3 General Vector Algebra

A vector in space is defined as a mathematical expression possessing magnitude and direction. If a point 'A' is located in space by (x_0, y_0, z_0) and point 'B' is located at (x_1, y_1, z_1) then the vector from 'A' to 'B' is:

$$\mathbf{AB} = (x_1 - x_0)\hat{\mathbf{i}} + (y_1 - y_0)\hat{\mathbf{j}} + (z_1 - z_0)\hat{\mathbf{k}} \qquad \textbf{(3. 1)}$$

Given two vectors in space, the dot product is defined as the product of the magnitudes of the two given vectors and of the cosine of the angle between them.

$$\mathbf{P} = a\hat{\mathbf{i}} + b\hat{\mathbf{j}} + c\hat{\mathbf{k}} \qquad \mathbf{Q} = x\hat{\mathbf{i}} + y\hat{\mathbf{j}} + z\hat{\mathbf{k}}$$

$$\mathbf{P} \bullet \mathbf{Q} = ax + by + cz = |\mathbf{P}||\mathbf{Q}|\cos(\theta) \qquad \textbf{(3. 2)}$$

$$\hat{\mathbf{i}} \bullet \hat{\mathbf{i}} = \hat{\mathbf{j}} \bullet \hat{\mathbf{j}} = \hat{\mathbf{k}} \bullet \hat{\mathbf{k}} = \mathbf{1}$$

$$\hat{\mathbf{i}} \bullet \hat{\mathbf{j}} = \hat{\mathbf{i}} \bullet \hat{\mathbf{k}} = \hat{\mathbf{j}} \bullet \hat{\mathbf{k}} = \mathbf{0}$$

Given any vector in space, the direction cosines can be computed by fixing a reference frame at the base of the vector, and then dividing the dot product of the vector with each respective axis by its magnitude.

$$\cos(\alpha) = \frac{\mathbf{A} \bullet \hat{\mathbf{i}}}{|\mathbf{A}|} \qquad \cos(\beta) = \frac{\mathbf{A} \bullet \hat{\mathbf{j}}}{|\mathbf{A}|} \qquad \cos(\gamma) = \frac{\mathbf{A} \bullet \hat{\mathbf{k}}}{|\mathbf{A}|} \qquad \textbf{(3. 3)}$$

The sum of the squares of the direction cosines is always equal to 1. In the contact study that follows in chapter 4, the direction cosines will serve as an invaluable tool in the determination of assembly motions. Given any two nonparallel vectors in space, the cross product is a vector whose length is numerically equal to the area of the parallelogram spanned by the two given vectors. The cross product obeys the right hand rule such that the direction of computation occurs counterclockwise when looking down onto the plane created by two given vectors. Unlike the dot product, the cross product is not a commutative operation; therefore, the order in which a cross product is performed must be monitored. The standard basis vectors involved in the cross product of the right-handed system are shown below:

$$\hat{\mathbf{i}}\times\hat{\mathbf{j}}=\hat{\mathbf{k}} \quad \hat{\mathbf{j}}\times\hat{\mathbf{k}}=\hat{\mathbf{i}} \quad \hat{\mathbf{k}}\times\hat{\mathbf{i}}=\hat{\mathbf{j}}$$

$$\hat{\mathbf{j}}\times\hat{\mathbf{i}}=-\hat{\mathbf{k}} \quad \hat{\mathbf{k}}\times\hat{\mathbf{j}}=-\hat{\mathbf{i}} \quad \hat{\mathbf{i}}\times\hat{\mathbf{k}}=-\hat{\mathbf{j}}$$

$$\hat{\mathbf{i}}\times\hat{\mathbf{i}}=\mathbf{0} \quad \hat{\mathbf{j}}\times\hat{\mathbf{j}}=\mathbf{0} \quad \hat{\mathbf{k}}\times\hat{\mathbf{k}}=\mathbf{0}$$

$$\mathbf{P}=a\hat{\mathbf{i}}+b\hat{\mathbf{j}}+c\hat{\mathbf{k}} \quad \mathbf{Q}=d\hat{\mathbf{i}}+e\hat{\mathbf{j}}+f\hat{\mathbf{k}}$$

$$\mathbf{P}\times\mathbf{Q}=(bf-ce)\hat{\mathbf{i}}+(cd-af)\hat{\mathbf{j}}+(ae-bd)\hat{\mathbf{k}} \tag{3. 4}$$

3.4 Equation of a Line in Symmetric Form

A normal vector, or a vector perpendicular to a plane in space, can be computed from three points that lie on the plane. If $\mathbf{V}_1$ is the vector from point 1 to 3 and $\mathbf{V}_2$ is the vector from point 1 to 2, then the normal vector is defined by the cross product of $\mathbf{V}_2$ and $\mathbf{V}_1$ (Fig. 3.4).

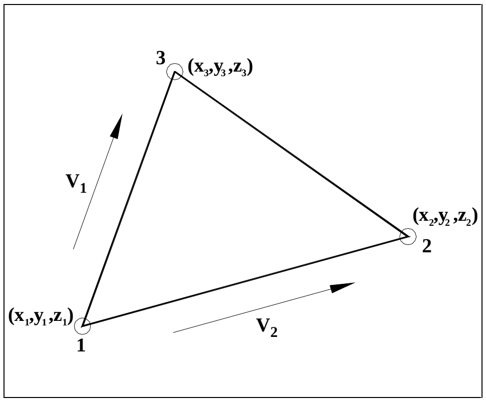


Figure 3. 4 Normal Vector to a Plane

$$\mathbf{N} = \mathbf{V_2} \times \mathbf{V_1} \tag{3. 5}$$

Where:

$$\mathbf{V_1} = (x_3 - x_1)\hat{\mathbf{i}} + (y_3 - y_1)\hat{\mathbf{j}} + (z_3 - z_1)\hat{\mathbf{k}} \tag{3. 6}$$

$$\mathbf{V_2} = (x_2 - x_1)\hat{\mathbf{i}} + (y_2 - y_1)\hat{\mathbf{j}} + (z_2 - z_1)\hat{\mathbf{k}} \tag{3. 7}$$

Therefore the normal vector becomes:

$$\begin{aligned}\mathbf{N} = {} & [(y_2 - y_1)(z_3 - z_1) - (y_3 - y_1)(z_2 - z_1)]\hat{\mathbf{i}} + \\ & [(x_3 - x_1)(z_2 - z_1) - (x_2 - x_1)(z_3 - z_1)]\hat{\mathbf{j}} + \\ & [(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)]\hat{\mathbf{k}}\end{aligned} \tag{3. 8}$$

A line in space can be considered as the result of the intersection of two non-parallel infinite planes. Therefore, given two sets of points, each of which defines an infinite plane in space, one can derive the equation of the line of intersection. The general form for the vector equation of the plane with normal vector N is:

$$\mathbf{N} \bullet (\mathbf{R} - \mathbf{R}_1) = \mathbf{0} \tag{3. 9}$$

The vector **R**$_1$ is drawn from the origin to any given point on the plane (point 1), and the vector **R** is drawn from point 1 to any arbitrary point on the plane (Fig. 3.5).

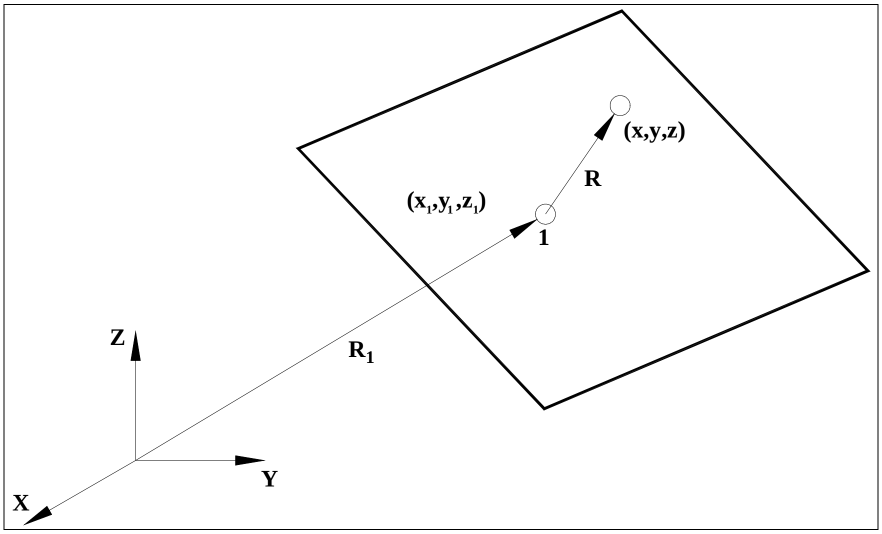


Figure 3. 3 Arbitrary Plane in Space

Choosing point 1 as the given point, the quantity (**R**-**R$_1$**) becomes:

$$(\mathbf{R} - \mathbf{R_1}) = (x - x_1)\hat{\mathbf{i}} + (y - y_1)\hat{\mathbf{j}} + (z - z_1)\hat{\mathbf{k}} \tag{3. 10}$$

Utilizing the normal vector calculated from the given coordinates, the equation of the plane becomes:

Given: $\mathbf{N} = a\hat{\mathbf{i}} + b\hat{\mathbf{j}} + c\hat{\mathbf{k}}$

$$\mathbf{N} \bullet (\mathbf{R} - \mathbf{R_1}) = a(x - x_1) + b(y - y_1) + c(z - z_1) = 0 \qquad \textbf{(3. 11)}$$

Equation of a plane:

$$ax + by + cz = (ax_1 + by_1 + cz_1) \qquad \textbf{(3. 12)}$$

Since the two planes are assumed non-parallel, there will be two distinct normal vectors **N**$_1$ and **N**$_2$. The normal vectors are perpendicular at any point on their respective planes, so they can be placed such that the origins of the vectors meet at the line of intersection.

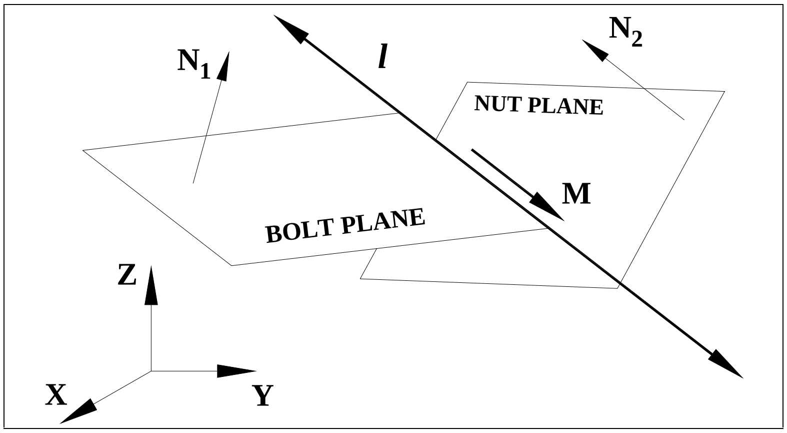


Figure 3. 4 Direction & Normal Vectors to a Plane

The direction vector (**M**) is defined as a vector parallel to the line created from two intersecting planes, and it is computed from the cross product of the normal vectors:

$$\mathbf{M} = \mathbf{N_1} \times \mathbf{N_2} \qquad \textbf{(3. 13)}$$

Where:

$$\mathbf{N_1} = a_b\hat{\mathbf{i}} + b_b\hat{\mathbf{j}} + c_b\hat{\mathbf{k}}$$
$$\mathbf{N_2} = a_n\hat{\mathbf{i}} + b_n\hat{\mathbf{j}} + c_n\hat{\mathbf{k}}$$

Therefore:

$$\mathbf{M} = (b_b c_n - c_b b_n)\hat{\mathbf{i}} + (c_b a_n - a_b c_n)\hat{\mathbf{j}} + (a_b b_n - b_b a_n)\hat{\mathbf{k}} \qquad \textbf{(3. 14)}$$

The line of intersection, ***l***, can be derived from the two equations for the plane and the direction vector in the following manner. Since the line is infinite, it will intersect at least one of the XZ, YZ, or XY planes. For the initial derivation it is assumed that the line intersects the YZ plane, therefore, let Y$_O$=0 as a point on the line. Then the two plane equations become:

$$a_b X_O + c_b Z_O = (a_b X_1 + b_b Y_1 + c_b Z_1) \tag{3. 15}$$

$$a_n X_O + c_n Z_O = (a_n X_1 + b_n Y_1 + c_n Z_1) \tag{3. 16}$$

Thus, given the value of Y$_O$, the remaining two simultaneous linear equations can be solved for X$_O$ and Z$_O$.

$$X_O = X_1 + \frac{(c_n b_b - b_n c_b)}{(c_n a_b - a_n c_b)} Y_1 \tag{3. 17}$$

$$Z_O = Z_1 + \frac{(b_n a_b - a_n b_b)}{(c_n a_b - a_n c_b)} Y_1 \tag{3. 18}$$

The symmetric equation of the line becomes:

$$\frac{X - X_O}{(\mathbf{M})\hat{\mathbf{i}}} = \frac{Y - Y_O}{(\mathbf{M})\hat{\mathbf{j}}} = \frac{Z - Z_O}{(\mathbf{M})\hat{\mathbf{k}}} \tag{3. 19}$$

Subbing in for the direction vector **M** and Y$_0$:

$$\frac{X - X_O}{b_b c_n - c_b b_n} = \frac{Y}{c_b a_n - a_b c_n} = \frac{Z - Z_O}{a_b b_n - b_b a_n} \tag{3. 20}$$

If any one of the components of the direction vector is zero then the equation of the line has only one dependent variable. For example, if the direction vector has only Y-axis and Z-axis components, then the line must be parallel to the YZ plane at a position of X$_O$ on the X-axis. The derivation of the symmetric equations of the line must be repeated to account for all possible cases, since up to two values of the direction vector can be zero. The derivation for these cases is similar to the one above, thus, only the resulting equations are listed below:

When Z$_0$ = 0:

$$X_O = X_1 + \frac{(c_b b_n - b_b c_n)}{(a_b b_n - b_b a_n)} Z_1 \qquad Y_O = Y_1 + \frac{(a_b c_n - c_b a_n)}{(a_b b_n - b_b a_n)} Z_1 \tag{3. 21}$$

When X$_0$ = 0

$$Y_O = Y_1 + \frac{(a_b c_n - c_b a_n)}{(b_b c_n - c_b b_n)} X_1 \qquad Z_O = Z_1 + \frac{(b_b a_n - a_b b_n)}{(b_b c_n - c_b b_n)} X_1 \tag{3. 22}$$

If two of the directional components are zero, the equations reduce to the trivial case of a line parallel to the X, Y, or Z-axis.

3.5 Contact Location Methodology

Given two intersecting triangles in space, one can locate the approximate coordinates of the contact point at the moment the intersection occurs. Knowing that each triangle defines a plane in space, and as long as the planes are not parallel, then there will be a line that results from the intersection of the two planes. Now, the vertices of the triangle define three intersecting lines that bound one another and form a triangle in space. Hence, given two intersecting triangles, one can determine the equation of the plane that each triangle defines via its vertices. From this, the equation of the line resulting from the intersection of the two planes can be derived. Now, the line can be reduced to a line segment by determining where it intersects the sides of the bolt triangle. The bolt triangle is chosen here, because in the tessellation of the solid model it is generally smaller than the nut triangle, and therefore would be more accurate in locating the contact point.

At this point the boundary of the nut triangle is analyzed to see what part of the line segment is included in the plane bounded by the sides of the nut triangle. Basically, part of the line segment that lies in the plane of the bolt will intersect at least one of the sides of the nut triangle. Vectors are drawn from the intersecting points in the bolt plane towards the intersecting points in the nut plane. The magnitudes of these vectors are analyzed to determine the appropriate end points of the line segment. Finally, the midpoint of the line segment is taken as the estimated contact point for that contact pair. This process is repeated for each set of contacts and all of the midpoints are summed and averaged in order to produce the approximate contact point. Ideally, the smaller number of contact pairs the better the approximation. The number of triangles in the tessellated model determines the accuracy of this result. For our purposes, by moving the bolt down in steps of 1 instead of 100 micro inches, we are able to reduce the maximum variation of the calculated contact point by an average of 0.0025 inches. The following section details the mathematics involved in the contact point calculation.

3.6 Contact Location Derivation

Currently, we know the symmetric equation of the line defined by a single contact pair. However, there are multiple contact pairs involved in a theoretical contact location, since the surface of the bolt will intersect the surface of the nut to some degree thereby increasing the number of contacting triangle pairs. If each contact pair is considered, then how does one determine where the actual contact point occurred? The following algorithm will be presented to approximate the contact location.

Ideally, we are interested in the segment of the line bounded by the plane of the bolt and nut triangle. In fact, both planes must share the line segment because the plane of the bolt and nut defined it. However, the nut triangle may not fully intersect the bolt triangle, as it has the potential to intersect the bolt triangle in four ways (Fig. 3.7).

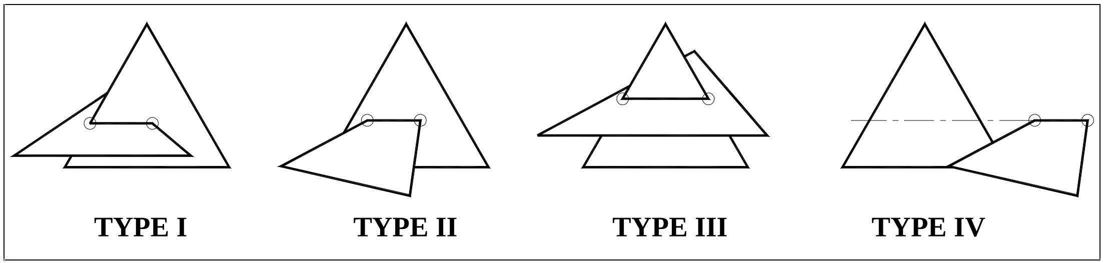


Figure 3. 5 Types of Bolt & Nut Triangle Intersection

Note that in the fourth case, the nut triangle does not actually intersect the bolt triangle, but it has the potential to be returned as a contact pair by *RAPID*. This is due to the internal accuracy of the collision detection library and cannot be avoided. Hence, the contact location algorithm will check for this special case and only use the line segment defined by the bolt triangle. For the remaining types, the portion of the infinite line that is actually involved with the intersection of the bolt and nut triangle will be used to locate the contact point. The approximate location of the contact point will be the average of the sum of the midpoints for each line segment. Hence, if there are 10 contact pairs involved (that fall within the 4 valid intersection types above), then there will be 10 midpoints that will be averaged to define one contact point. The process of enumerating the line segment helps eliminate potential error associated with triangle size. For instance, if the triangles are large, then the line segment bounded by just the bolt triangle will be large, subsequently creating error when calculating the midpoint. However, the length of the line segment can be reduced by determining where it is bounded by the bolt and nut triangle. This, in turn, will reduce the potential for error in the contact point location. Moreover, note that it is of primary interest to minimize the number of contact pairs involved in a contact so as to eliminate the potential for error.

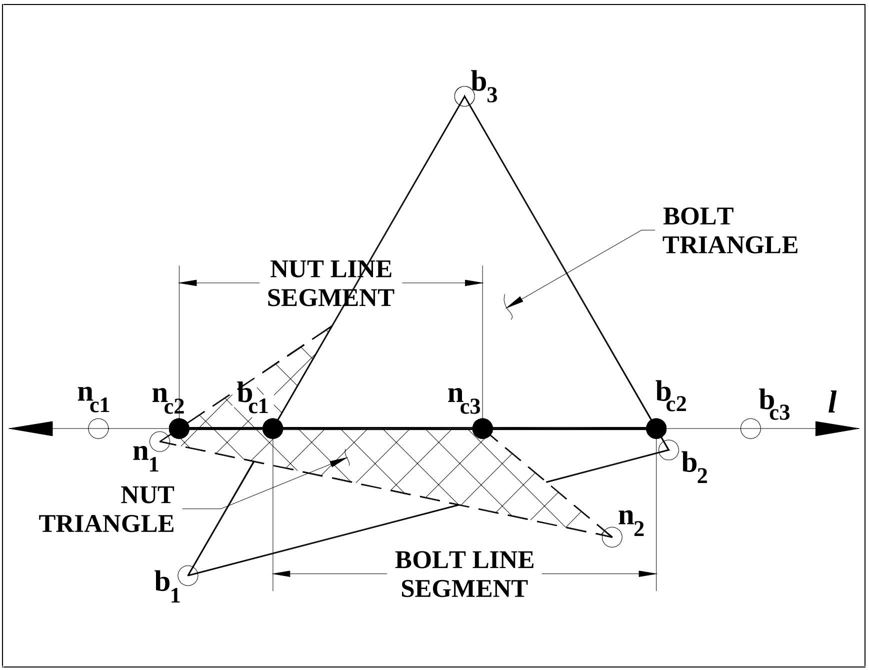


Figure 3. 6 Endpoints of Intersecting Line Segment

The following steps summarize how we approximate the contact point (Fig. 3.8):

1. Derive an equation of a line that defines each boundary of the bolt triangle
2. Enumerate where line, ***l***, intersects the bolt triangle (b_{c1}, b_{c2}, b_{c3})
3. Check and identify which two of the three points are bounded by the bolt triangle
4. Derive an equation of a line that defines each boundary of the nut triangle
5. Enumerate where line, ***l***, intersects the nut triangle (n_{c1}, n_{c2}, n_{c3})
6. Check and identify which two of the three points are bounded by the nut triangle
7. Determine if any of the nut points fall within the boundary of the bolt triangle
8. Extract the endpoints of the line segment
9. Calculate the midpoint, and repeat steps 1-8 for next contact pair
10. Average midpoints from each contact pair

The first step will be to derive the equation of a line for each side of the bolt triangle. A simplified method of determining the equation of a line in symmetric form will be used.

Here, one can simply determine the direction vector **M** from the vertex points since the line is already parallel to the plane of the bolt triangle. The line equations become:

From Vertex 1 to 3:

$$\frac{x-x_{b1}}{x_{b3}-x_{b1}}=\frac{y-y_{b1}}{y_{b3}-y_{b1}}=\frac{z-z_{b1}}{z_{b3}-z_{b1}} \tag{3. 23}$$

From Vertex 2 to 3:

$$\frac{x-x_{b2}}{x_{b3}-x_{b2}}=\frac{y-y_{b2}}{y_{b3}-y_{b2}}=\frac{z-z_{b2}}{z_{b3}-z_{b2}} \tag{3. 24}$$

From Vertex 1 to 2:

$$\frac{x-x_{b1}}{x_{b2}-x_{b1}}=\frac{y-y_{b1}}{y_{b2}-y_{b1}}=\frac{z-z_{b1}}{z_{b2}-z_{b1}} \tag{3. 25}$$

The contact points (b_{c1}, b_{c2}, b_{c3}) are determined by solving a system of equations. Two of the equations are supplied by the triangle line equations while the third is provided by the symmetric equation of the line:

$$\begin{array}{lll}
\mathrm{b}_{c1}: & \mathrm{b}_{c2}: & \mathrm{b}_{c3}: \\
\dfrac{X-X_{b1}}{X_{b3}-X_{b1}}=\dfrac{Z-Z_{b1}}{Z_{b3}-Z_{b1}} & \dfrac{X-X_{b2}}{X_{b3}-X_{b2}}=\dfrac{Z-Z_{b2}}{Z_{b3}-Z_{b2}} & \dfrac{X-X_{b1}}{X_{b2}-X_{b1}}=\dfrac{Z-Z_{b1}}{Z_{b2}-Z_{b1}} \\
\dfrac{Y-Y_{b1}}{Y_{b3}-Y_{b1}}=\dfrac{Z-Z_{b1}}{Z_{b3}-Z_{b1}} & \dfrac{Y-Y_{b2}}{Y_{b3}-Y_{b2}}=\dfrac{Z-Z_{b2}}{Z_{b3}-Z_{b2}} & \dfrac{Y-Y_{b1}}{Y_{b2}-Y_{b1}}=\dfrac{Z-Z_{b1}}{Z_{b2}-Z_{b1}} \\
\dfrac{X-X_{O}}{(\mathbf{M})\hat{\mathbf{i}}}=\dfrac{Z-Z_{O}}{(\mathbf{M})\hat{\mathbf{k}}} & \dfrac{X-X_{O}}{(\mathbf{M})\hat{\mathbf{i}}}=\dfrac{Z-Z_{O}}{(\mathbf{M})\hat{\mathbf{k}}} & \dfrac{X-X_{O}}{(\mathbf{M})\hat{\mathbf{i}}}=\dfrac{Z-Z_{O}}{(\mathbf{M})\hat{\mathbf{k}}}
\end{array} \tag{3. 26 a,b,c}$$

The line resulting from the intersection of the bolt and nut triangle will intersect the lines that make up the sides of the bolt triangle three times. It is of interest to find out which two of the three points are bounded by the bolt triangle. In order to verify that b_{c1} and b_{c2} meet the criteria, the magnitude of the vector drawn from b_1 to b_{c1} must be less than the magnitude of the vector drawn from b_1 to b_3, and the magnitude of the vector drawn from b_2 to b_3 must be less than the magnitude of the vector drawn from b_2 to b_{c2}. This will succeed unless one of the lines that make up a boundary of the bolt triangle is parallel to ***l***, in which case a third point of intersection can not be calculated. The entire process is defined in the steps below:

1. Calculate the vector magnitudes for each side of the bolt triangle:

$$\left|\vec{\mathbf{V}}_1^3\right| = \sqrt{\left(b_{x3} - b_{x1}\right)^2 + \left(b_{y3} - b_{y1}\right)^2 + \left(b_{z3} - b_{z1}\right)^2} \qquad \textbf{(3. 27)}$$

$$\left|\vec{\mathbf{V}}_2^3\right| = \sqrt{\left(b_{x3} - b_{x2}\right)^2 + \left(b_{y3} - b_{y2}\right)^2 + \left(b_{z3} - b_{z2}\right)^2} \qquad \textbf{(3. 28)}$$

$$\left|\vec{\mathbf{V}}_1^2\right| = \sqrt{\left(b_{x2} - b_{x1}\right)^2 + \left(b_{y2} - b_{y1}\right)^2 + \left(b_{z2} - b_{z1}\right)^2} \qquad \textbf{(3. 29)}$$

2. Calculate the vector magnitudes for each vector drawn from a vertex of the bolt triangle to the intersection point:

$$\left|\vec{\mathbf{V}}_1^{c1}\right| = \sqrt{\left(b_x^{c1} - b_{x1}\right)^2 + \left(b_y^{c1} - b_{y1}\right)^2 + \left(b_z^{c1} - b_{z1}\right)^2} \qquad \textbf{(3. 30)}$$

$$\left|\vec{\mathbf{V}}_2^{c2}\right| = \sqrt{\left(b_x^{c2} - b_{x2}\right)^2 + \left(b_y^{c2} - b_{y2}\right)^2 + \left(b_z^{c2} - b_{z2}\right)^2} \qquad \textbf{(3. 31)}$$

$$\left|\vec{\mathbf{V}}_1^{c3}\right| = \sqrt{\left(b_x^{c3} - b_{x1}\right)^2 + \left(b_y^{c3} - b_{y1}\right)^2 + \left(b_z^{c3} - b_{z1}\right)^2} \qquad \textbf{(3. 32)}$$

3. Calculate the angle between the vector that defines the side of the triangle and the vector drawn from the vertex to the intersection point:

$$\theta_{c1} = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_1^3 \bullet \vec{\mathbf{V}}_1^{c1}}{\left|\vec{\mathbf{V}}_1^3\right|\left|\vec{\mathbf{V}}_1^{c1}\right|}\right) \quad \theta_{c2} = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_2^3 \bullet \vec{\mathbf{V}}_2^{c2}}{\left|\vec{\mathbf{V}}_2^3\right|\left|\vec{\mathbf{V}}_2^{c2}\right|}\right) \quad \theta_{c3} = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_1^2 \bullet \vec{\mathbf{V}}_1^{c3}}{\left|\vec{\mathbf{V}}_1^2\right|\left|\vec{\mathbf{V}}_1^{c3}\right|}\right) \qquad \textbf{(3. 33 a,b,c)}$$

4. If the vectors are coincident (the angle is zero) and the magnitude of the vector drawn from the vertex to the contact point is less than the magnitude of the vector that defines a side of the bolt triangle, then the intersection point falls on the triangle boundary.

This process is repeated for the nut triangle, resulting in two sets of two points. Thus we are left with the endpoints of the nut and bolt line segment. Now we must check and see if the location of the nut endpoints match any of the five intersection types listed above. We want to determine where the nut endpoints are in order to reduce the length of the contact line segment. Once again, vector magnitudes as well as vector directions are used to correctly isolate the appropriate endpoints. This process is emulated with the following steps (assuming that b_{c1}, b_{c2}, n_{c2}, n_{c3} are the two intersection points that lie within the boundaries of the bolt and nut triangle, respectively):

1. Determine the magnitude of the vector drawn from b$_{c1}$ to b$_{c2}$:

$$\left|\vec{\mathbf{V}}_{b_{c1}}^{b_{c2}}\right| = \sqrt{\left(b_x^{c2} - b_x^{c1}\right)^2 + \left(b_y^{c2} - b_y^{c1}\right)^2 + \left(b_z^{c2} - b_z^{c1}\right)^2} \tag{3. 34}$$

2. Determine the magnitude of the vector drawn from b$_{c1}$ to n$_{c2}$:

$$\left|\vec{\mathbf{V}}_{b_{c1}}^{n_{c1}}\right| = \sqrt{\left(n_x^{c2} - b_x^{c1}\right)^2 + \left(n_y^{c2} - b_y^{c1}\right)^2 + \left(n_z^{c2} - b_z^{c1}\right)^2} \tag{3. 35}$$

3. Determine the magnitude of the vector drawn from b$_{c1}$ to n$_{c3}$:

$$\left|\vec{\mathbf{V}}_{b_{c1}}^{n_{c2}}\right| = \sqrt{\left(n_x^{c3} - b_x^{c1}\right)^2 + \left(n_y^{c3} - b_y^{c1}\right)^2 + \left(n_z^{c3} - b_z^{c1}\right)^2} \tag{3. 36}$$

4. Calculate the angle between the vector from b$_{c1}$ to b$_{c2}$ and the vector from b$_{c1}$ to n$_{c2}$:

$$\theta_0^1 = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_{b_{c1}}^{b_{c2}} \bullet \vec{\mathbf{V}}_{b_{c1}}^{n_{c2}}}{\left|\vec{\mathbf{V}}_{b_{c1}}^{b_{c2}}\right|\left|\vec{\mathbf{V}}_{b_{c1}}^{n_{c2}}\right|}\right) \tag{3. 37}$$

5. Calculate the angle between the vector from b$_{c1}$ to b$_{c2}$ and the vector from b$_{c1}$ to n$_{c3}$:

$$\theta_0^2 = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_{b_{c1}}^{b_{c2}} \bullet \vec{\mathbf{V}}_{b_{c1}}^{n_{c3}}}{\left|\vec{\mathbf{V}}_{b_{c1}}^{b_{c2}}\right|\left|\vec{\mathbf{V}}_{b_{c1}}^{n_{c3}}\right|}\right) \tag{3. 38}$$

6. Determine the magnitude of the vector drawn from b$_{c2}$ to n$_{c2}$:

$$\left|\vec{\mathbf{V}}_{b_{c2}}^{n_{c1}}\right| = \sqrt{\left(n_x^{c2} - b_x^{c2}\right)^2 + \left(n_y^{c2} - b_y^{c2}\right)^2 + \left(n_z^{c2} - b_z^{c2}\right)^2} \tag{3. 39}$$

7. Determine the magnitude of the vector drawn from b$_{c2}$ to n$_{c3}$:

$$\left|\vec{\mathbf{V}}_{b_{c2}}^{n_{c2}}\right| = \sqrt{\left(n_x^{c3} - b_x^{c2}\right)^2 + \left(n_y^{c3} - b_y^{c2}\right)^2 + \left(n_z^{c3} - b_z^{c2}\right)^2} \tag{3. 40}$$

8. Calculate the angle between the vector from b$_{c2}$ to b$_{c1}$ and the vector from b$_{c2}$ to n$_{c2}$:

$$\theta_1^1 = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_{b_{c2}}^{b_{c1}} \bullet \vec{\mathbf{V}}_{b_{c2}}^{n_{c2}}}{\left|\vec{\mathbf{V}}_{b_{c2}}^{b_{c1}}\right|\left|\vec{\mathbf{V}}_{b_{c2}}^{n_{c2}}\right|}\right) \tag{3. 41}$$

9. Calculate the angle between the vector from b$_{c2}$ to b$_{c1}$ and the vector from b$_{c2}$ to n$_{c3}$:

$$\theta_1^2 = \cos^{-1}\left(\frac{\vec{\mathbf{V}}_{b_{c2}}^{b_{c1}} \bullet \vec{\mathbf{V}}_{b_{c2}}^{n_{c3}}}{\left|\vec{\mathbf{V}}_{b_{c2}}^{b_{c1}}\right|\left|\vec{\mathbf{V}}_{b_{c2}}^{n_{c3}}\right|}\right) \tag{3. 42}$$

All of these calculations are necessary, since we do not have prior knowledge of the order of the intersection points (b_{c1}, b_{c2}, b_{c3}). In the first contact situation, the four ordered pairs of the bolt and nut intersection points produce four possible arrangements of the angle between them as shown in Table 3.1:

Table 3. 1

	θ_0^1	θ_0^2	θ_1^1	θ_1^2
(n_{c2}, b_{c1}, n_{c3}, b_{c2})	π	0	0	0
(n_{c3}, b_{c1}, n_{c2}, b_{c2})	0	π	0	0
(n_{c2}, b_{c2}, n_{c3}, b_{c1})	0	0	π	0
(n_{c3}, b_{c2}, n_{c2}, b_{c1})	0	0	0	π

Thus, the intersection between the bolt and nut will fall into the first type provided that the angle between the respective vectors matches one line in the table above. If not, then the angular results are compared with the next intersection type. In addition to the angular check, the magnitudes of the vectors (shown in steps 1-3 and 5-7) are compared in order to verify that the intersection point falls within the plane of the bolt triangle. If the nut intersection point does not fall within the plane of the triangle, then either the fourth or fifth intersection type will define the resulting line segment. The remaining cases can be found in appendix B and highlighted within the C++ code in appendix D.

Before defining the multiple contact state models, we will begin with a brief review of coordinate system transformations since this technique is required to understand the progression of the bolt from the initial to the final contact state.

3.7 Coordinate System Transformations

One can efficiently conduct an assembly analysis between two bodies by utilizing Euler rotations. Here, the orientation of one body that is fixed is considered the fixed reference frame. This is also known as the fixed inertial reference frame, as discussed in any dynamics text, but this analysis is purely geometric, and therefore, the fixed inertial reference frame will be known as the fixed reference frame. The position of the active body, or the body being manipulated, is firstly defined in its local reference frame. Instead of dealing with two reference frames, it is far simpler to first define the active body's orientation in its local reference frame, and then transform the orientation to the fixed reference frame. Once this is done, then a contact analysis between two bodies in space can be accomplished. Since the local reference frame is a moving reference, the orientation of the moving body will be transformed to the fixed reference frame in order to keep track of its orientation in space. Consider two coordinate systems that share a common origin. The unprimed and primed axes will denote the fixed and local reference frame, respectively (Fig. 3.9).

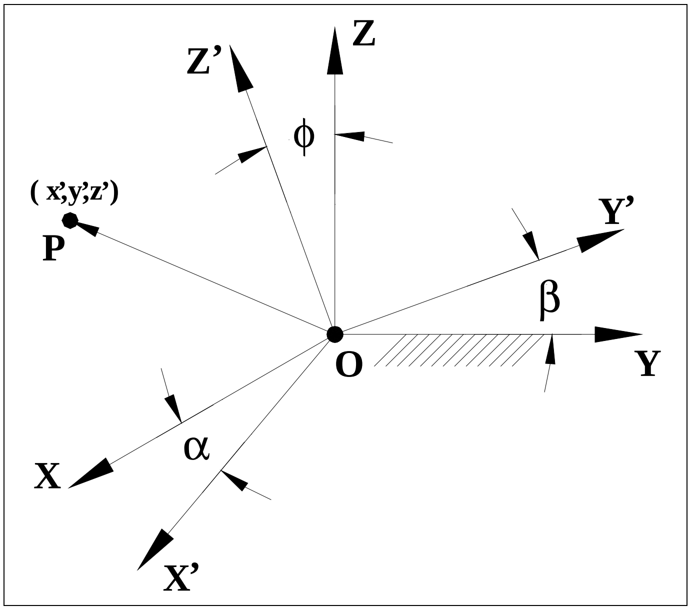


Figure 3. 7 Rotating Reference Frames

Consider a fixed body that contains a set of mutually orthogonal unprimed coordinate axes located at point O. Also, consider a moving body containing another set of mutually orthogonal primed coordinate axes, initially coincident at point O and containing an arbitrary point P at (x’, y’, z’). If the primed axes are rotated about the OX axis an amount α, then the coordinates that locate the point P in the fixed frame are (x, y, z). These two sets of coordinates are related by the following transformation (Meirovitch, 1970):

$$\begin{Bmatrix} x' \\ y' \\ z' \end{Bmatrix} = [\mathbf{R}_X] \begin{Bmatrix} x \\ y \\ z \end{Bmatrix} \qquad \textbf{(3. 43)}$$

$$[\mathbf{R}_{\mathrm{X}}]=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\alpha) & \sin(\alpha)\\ 0 & -\sin(\alpha) & \cos(\alpha)\end{bmatrix} \tag{3. 44}$$

[$\mathbf{R}_{\mathrm{X}}$] will be referred to as the X-axis rotation matrix. Similarly, if the primed axes are rotated about the OY axis by an amount β, the Y-axis rotation matrix becomes:

$$[\mathbf{R}_{\mathrm{Y}}]=\begin{bmatrix}\cos(\beta) & 0 & -\sin(\beta)\\ 0 & 1 & 0\\ \sin(\beta) & 0 & \cos(\beta)\end{bmatrix} \tag{3. 45}$$

Finally, if the primed axes are rotated about the OZ axis by an amount ϕ, then the Z-axis rotation matrix becomes:

$$[\mathbf{R}_{\mathrm{Z}}]=\begin{bmatrix}\cos(\phi) & \sin(\phi) & 0\\ -\sin(\phi) & \cos(\phi) & 0\\ 0 & 0 & 1\end{bmatrix} \tag{3. 46}$$

When one considers the orientation of a body in space, the order of rotation becomes paramount. In two dimensions, only one rotation, a rotation about an axis perpendicular to the plane, is possible, therefore the orientation of a body is defined as two translations and one rotation. Translations are commutative in two and three dimensions, but rotations are not commutative in three dimensions. A body will possess a different orientation depending on the order of rotation. There are 12 unique parameterizations of a rotation matrix using successive rotations about a set of mutually orthogonal axes (Craig, 1989), however; only a Z-Y-X rotation will be used in this analysis. Here, the moving body rotates about the OZ axis by an amount ϕ first, followed by a rotation about the OY axis by an amount β, and finished with a rotation about the OX axis by an amount α. The coordinates of point P in the local frame are transformed to coordinates in the fixed reference frame by sequentially multiplying the rotation matrices. Using the order of rotation described above, the matrix multiplication begins by multiplying the rotation matrices in the prescribed order right to left:

$$\begin{Bmatrix}x'\\ y'\\ z'\end{Bmatrix}=[\mathbf{R}_{\mathrm{X}}][\mathbf{R}_{\mathrm{Y}}][\mathbf{R}_{\mathrm{Z}}]\begin{Bmatrix}x\\ y\\ z\end{Bmatrix} \tag{3. 47}$$

$$[\mathbf{R}] = [\mathbf{R}_{\mathbf{X}}][\mathbf{R}_{\mathbf{Y}}][\mathbf{R}_{\mathbf{Z}}] = \begin{bmatrix} c\beta c\phi & c\beta s\phi & -s\beta \\ s\alpha s\beta c\phi - c\alpha s\phi & s\alpha s\beta s\phi + c\alpha c\phi & s\alpha c\beta \\ c\alpha s\beta c\phi + s\alpha s\phi & c\alpha s\beta s\phi - s\alpha c\phi & c\alpha c\beta \end{bmatrix} \qquad \textbf{(3. 48)}$$

Here cβ is defined as the cos(β), and sβ is defined as the sin(β). Since the columns of a rotation matrix all have unit magnitude and are orthogonal to one another, the inverse of a rotation matrix is equivalent to its transpose (Crane, 1998). Therefore, by providing known coordinates of any point in the local reference frame the equivalent values of those points can be obtained through the following matrix calculation:

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = [\mathbf{R}]^{\mathrm{T}} \begin{Bmatrix} x' \\ y' \\ z' \end{Bmatrix} \qquad \textbf{(3. 49)}$$

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{bmatrix} c\beta c\phi & s\alpha s\beta c\phi - c\alpha s\phi & c\alpha s\beta c\phi + s\alpha s\phi \\ c\beta s\phi & s\alpha s\beta s\phi + c\alpha c\phi & c\alpha s\beta s\phi - s\alpha c\phi \\ -s\beta & s\alpha c\beta & c\alpha c\beta \end{bmatrix} \begin{Bmatrix} x' \\ y' \\ z' \end{Bmatrix} \qquad \textbf{(3. 50)}$$

When dealing with coordinate transformations that require translations as well as rotations it is more efficient to deal with 4×4 transformation matrices. Here, the local coordinate system is translated from and rotated about a fixed reference frame (Fig. 3.10). The coordinates that locate the origin of the moving frame are homogenous; meaning that they are all scaled by a fourth variable. This fourth variable permits the use of a 4×4 matrix when representing coordinate transformations. In homogenous coordinates, a three-dimensional point given by *X*, *Y*, and *Z* is represented by four scalar values, *x*, *y*, *z*, and *w*. The homogenous and three-dimensional coordinates are related by:

$$X = \frac{x'}{w} \qquad Y = \frac{y'}{w} \qquad Z = \frac{z'}{w} \qquad \textbf{(3. 51 a,b,c)}$$

Therefore, the first three components of the homogenous coordinates of a point are the same as the three-dimensional coordinates of the point when *w* = 1 (McCrea, 1947).

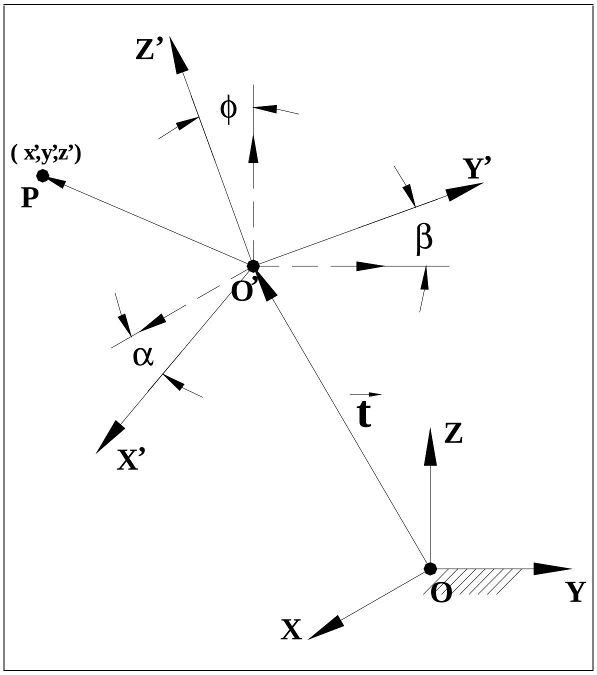


Figure 3. 8 Translated & Rotated Reference Frames

The homogenous coordinates are then utilized in a matrix multiplication that enumerates a homogenous transform. A homogenous transformation is constructed as:

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{z} \\ \hline \mathbf{1} \end{bmatrix} = \left[\begin{array}{ccc:c} & \mathbf{R}^{\mathrm{T}} & & \vec{\mathbf{t}} \\ \hdashline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{array}\right] \begin{bmatrix} \mathbf{x'} \\ \mathbf{y'} \\ \mathbf{z'} \\ \hline \mathbf{1} \end{bmatrix} \qquad \textbf{(3. 52)}$$

In the matrix equation above, the vector **t** is drawn with respect to the unprimed frame, thus, given a point in the primed reference frame; one can find the location of that point as measured from the unprimed frame. Given the primed frame as body **A** and the unprimed frame as body **B**, the matrix equation becomes:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{B}} = {}^{\mathbf{B}}_{\mathbf{A}}\mathbf{T} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{A}} \qquad \textbf{(3. 53)}$$

In the case of multiple rotations, it may occur that the user is interested in the location of a point on a rigid body after the solid body has undergone several transformations. This is where the user can take advantage of the simplicity of the 4×4 homogenous transformation method by using it to map multiple coordinate systems. For example, consider a moving body that underwent two translations and rotations in the figure below:

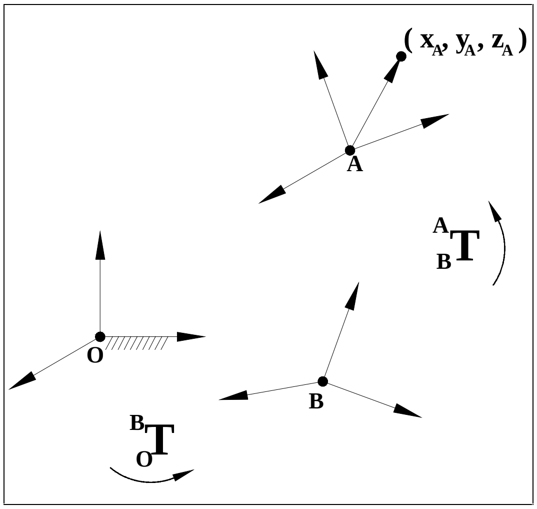


Figure 3. 9 Multiple Homogenous Transformations

A point located on the moving body can be mapped by the following set of equations:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{B}} = {}^{\mathbf{A}}_{\mathbf{B}}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{A}} \qquad \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{O}} = {}^{\mathbf{B}}_{\mathbf{O}}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{B}} \qquad \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{O}} = {}^{\mathbf{B}}_{\mathbf{O}}\mathbf{T}^{-1}\, {}^{\mathbf{A}}_{\mathbf{B}}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{A}} \qquad \textbf{(3. 54 a,b,c)}$$

The benefit of keeping track of the position of a moving frame will prove to be an essential tool during the contact analysis of a threaded assembly. The next topic will discuss how the spatial orientation of the bolt was modeled as the bolt moved through multiple contact phases.

3.8 1st Contact State Model

The data files of the bolt and nut are exported from Pro/Engineer sharing the same reference frame. This reference frame will be referred to as the "fixed" frame. All the vertices of the bolt undergo a transformation that orients and translates each coordinate in space. The bolt is then translated along the Z-axis of the fixed frame in a series of regressing steps of finer adjustment until contact with the nut is made, since it is of primary interest to cause a contact with the least amount of penetration (smallest number of contact pairs). Once the final move is made, at a value of 10 micro inches, the contact pairs are sent to the contact location subroutine. Figure 3.12 shows the orientation of the reference frames and the transformation between them once the first contact point (**CP$_0$**) is obtained.

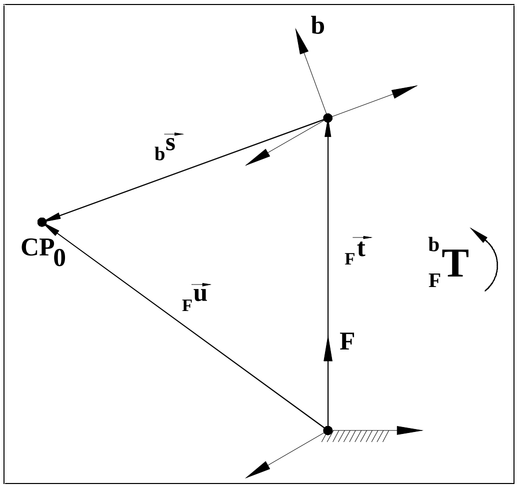


Figure 3. 10 1st Contact State Model

The vector **u** locates the contact point **CP$_0$**, and the vector **t** locates the origin of the bolt reference frame; both of which are measured with respect to the fixed reference frame. The vector **s** locates the contact point as measured in the bolt reference frame. The vertices of the bolt are transformed to the fixed frame via equation 3.55:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{F}} = {}^{\mathbf{b}}_{\mathbf{F}}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{b}} \quad \textbf{(3. 55)}$$

The transformation is defined as:

$${}^{\mathbf{b}}_{\mathbf{F}}\mathbf{T} = \begin{bmatrix} c\beta c\phi & s\alpha s\beta c\phi - c\alpha s\phi & c\alpha s\beta c\phi + s\alpha s\phi & 0 \\ c\beta s\phi & s\alpha s\beta s\phi + c\alpha c\phi & c\alpha s\beta s\phi - s\alpha s\phi & 0 \\ -s\beta & s\alpha c\beta & c\alpha c\beta & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(3. 56)}$$

Note that α, β, and ϕ are given because the orientation of the bolt is known *a priori*, and that the value of t_z is determined numerically through the series of regressing steps mentioned earlier. Now, the contact point as viewed in the bolt frame can easily be calculated through the following homogenous transformation:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{b}} = {}^{\mathbf{b}}_{\mathbf{F}}\mathbf{T} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{F}} \tag{3. 57}$$

This is useful because the parametric equation that defines the cross-section of the bolt at the contact can only be utilized once the contact point in the bolt reference frame is known.

3.9 2nd Contact State Model

Once the first contact point is obtained, a new reference frame, $\mathbf{C_0}$, is created at the location of the contact point, $\mathbf{CP_0}$. This is done because the collision detection library, RAPID, needs to start with both tessellated models sharing the same reference frame. This is the sole reason why new models of the bolt and nut need to be created in the code. Thus, the bolt triangles maintain the first contact point configuration prior to a subsequent rotation about the X-axis of the $\mathbf{C_0}$ frame. The actual contact locations are stored in memory and converted back to the fixed reference frame so the user has knowledge of where the contacts occurred with reference to the actual position of the bolt and nut. The X-axis of the $\mathbf{C_0}$ frame is aligned with a vector drawn tangent to the nut surface at the contact location. The bolt and nut vertices will be rotated about this X-axis until the second contact point is obtained (Fig. 3.13).

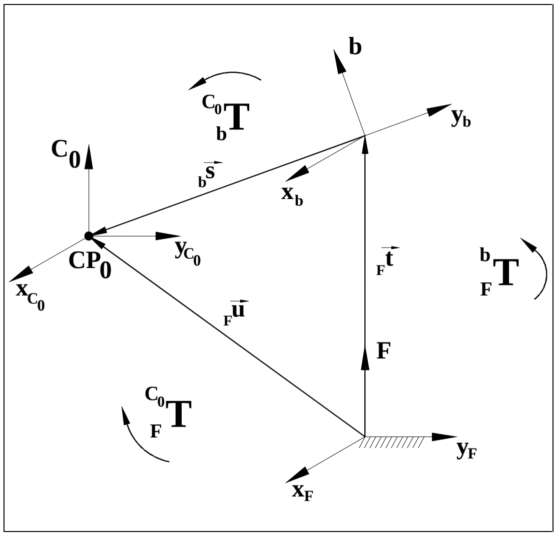


Figure 3. 11 2nd Contact State Model

Before commencing with a rotation about the X-axis of the **$\mathbf{C_0}$** frame, the bolt and nut vertices must be transformed by the following matrix equations:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{C_0}} = {}^{\mathbf{C_0}}_{\mathbf{F}}\mathbf{T}\;{}^{\mathbf{b}}_{\mathbf{F}}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{b}} \qquad \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{C_0}} = {}^{\mathbf{C_0}}_{\mathbf{F}}\mathbf{T} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{n}} \qquad \textbf{(3. 58 a,b)}$$

$${}^{\mathbf{C_0}}_{\mathbf{F}}\mathbf{T} = \begin{bmatrix} c\delta_2 c\delta_1 & c\delta_2 s\delta_1 & -s\delta_2 & (-c\delta_2 c\delta_1)u_x - (c\delta_2 s\delta_1)u_y + (s\delta_2)u_z \\ -s\delta_1 & c\delta_1 & 0 & (s\delta_1)u_x - (c\delta_1)u_y \\ s\delta_2 c\delta_1 & s\delta_2 s\delta_1 & c\delta_2 & (-s\delta_2 c\delta_1)u_x - (s\delta_2 s\delta_1)u_y - (c\delta_2)u_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(3. 59)}$$

δ_1 and δ_2 are calculated from the direction cosines of the vector tangent to the surface of the nut at the contact point and are represented in the fixed reference frame. Figure 3.14 depicts the two-step process of transforming the X_F-axis to a general orientation of a vector tangent to the nut surface.

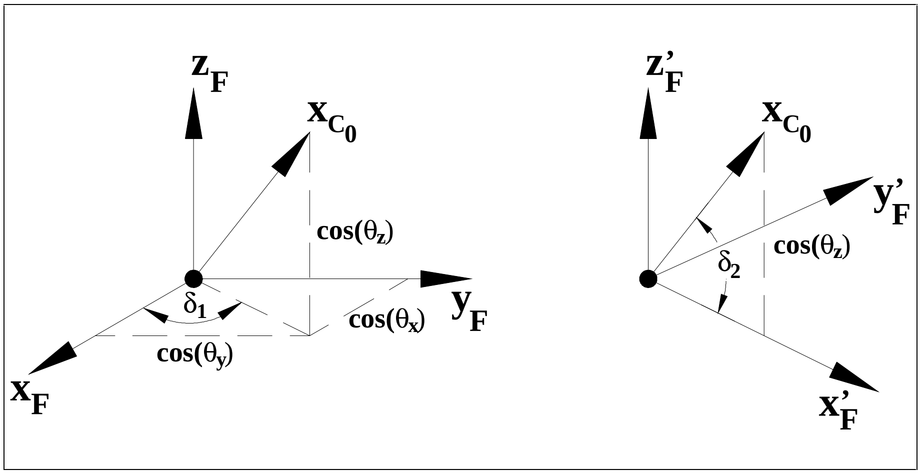


Figure 3. 12 Vector Rotation Angles

The fixed frame is rotated about the Z_F-axis first by an amount δ_1. Then the new fixed frame (denoted by a prime) is rotated about the Y'_F-axis by an amount δ_2. The formulae for calculating δ_1 and δ_2 are given below:

$$\delta_1 = a\cos\left[\frac{\theta_x}{\sin(a\cos\theta_z)}\right] \qquad \delta_2 = a\cos(\theta_z) - \left(\frac{\pi}{2}\right) \qquad \textbf{(3. 60 a,b)}$$

Note that θ_x and θ_z are expressed in radians, since the vector tangent to the surface of the nut at the contact point is of unit magnitude. The transformation that orients the bolt frame with the $\mathbf{C_0}$ frame follows a similar derivation. Here, the X_B-axis must be aligned with the X_{C0}-axis via two successive rotations, ψ_1 and ψ_2. However, the direction cosines of the vector tangent to the nut surface must be expressed in the bolt frame prior to the computation of these angles.

$$ {}^{\mathbf{C_0}}_{\mathbf{b}}\mathbf{T} = \begin{bmatrix} c\psi_2 c\psi_1 & c\psi_2 s\psi_1 & -s\psi_2 & (-c\psi_2 c\psi_1)s_x - (c\psi_2 s\psi_1)s_y + (s\psi_2)s_z \\ -s\psi_1 & c\psi_1 & 0 & (s\psi_1)s_x - (c\psi_1)s_y \\ s\psi_2 c\psi_1 & s\psi_2 s\psi_1 & c\psi_2 & (-s\psi_2 c\psi_1)s_x - (s\psi_2 s\psi_1)s_y - (c\psi_2)s_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(3. 61)} $$

$$ \psi_1 = a\cos\left[\frac{\theta_x^b}{\sin\left(a\cos\theta_z^b\right)}\right] \qquad \psi_2 = a\cos\left(\theta_z^b\right) - \left(\frac{\pi}{2}\right) \qquad \textbf{(3. 62 a,b)} $$

At this point the bolt is rotated about the X-axis of the $\mathbf{C_0}$ frame until the second contact point is obtained. The direction of rotation is negative, because the vector tangent to the surface of the nut is directed counter-clockwise, and the assembly force is acting along the negative azimuth axis of the bolt. Subsequently, a negative rotation about this axis will rotate the bolt towards the nut surface. Figure 3.15 depicts the position of the bolt frame once it obtains the second contact point, $\mathbf{CP_1}$.

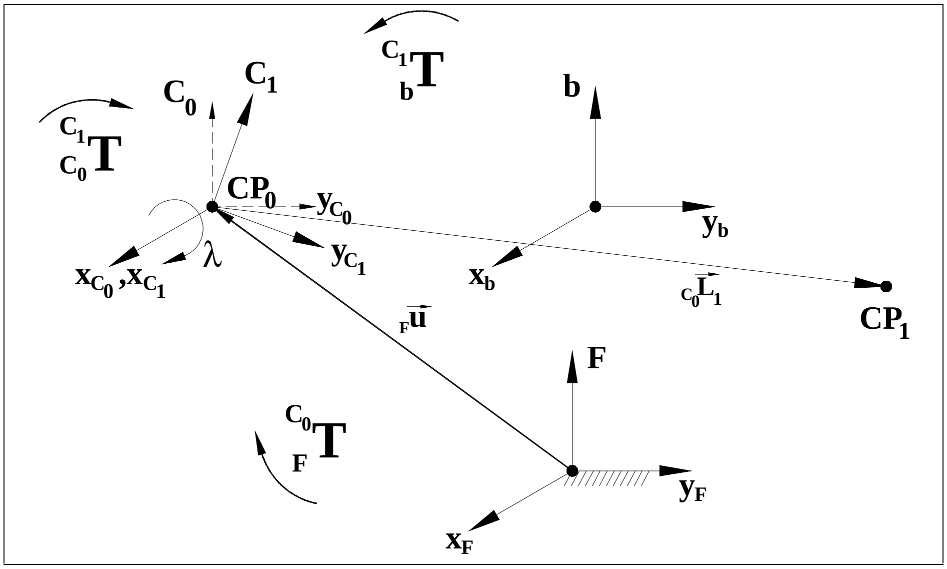


Figure 3. 13 Coordinate Frames at Two-Point Contact

The transformation that carries the bolt to the second contact state is simply comprised of a rotation about the X_{C0}-axis:

$$ {}^{C_1}_{C_0}\mathbf{T}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\lambda & \sin\lambda & 0 \\ 0 & -\sin\lambda & \cos\lambda & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(3. 63)} $$

The advent of the second contact point provides a more constrained situation, as now the bolt has a maximum of two degrees of freedom provided it still maintains both contact points. Once the bolt has achieved this situation, a new model of the nut and bolt must be created as observed from a new reference frame defined by the contact points. At all times the contact states are validated as being quasi-stable with respect to an elastic support, such as an RCC.

3.10 3rd Contact State Model

Once again, an additional reference frame will be established at the initial contact point **CP_0**, but the X-axis of this coordinate system will be defined by a vector drawn from the first to the second contact point (Fig. 3.16).

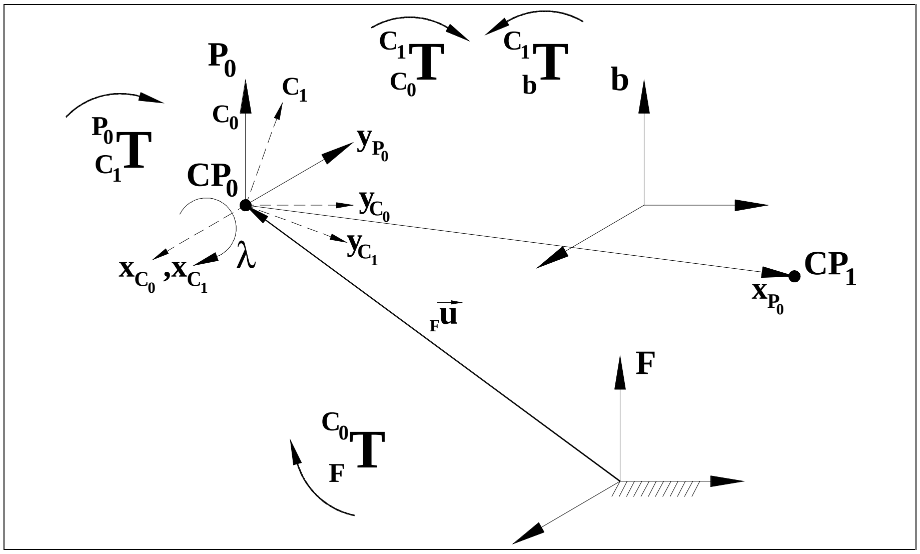


Figure 3. 14 3rd Contact State Model

The transformation from the fixed frame to the **P_0** frame will follow a similar derivation for the transformation from the fixed frame to the **C_0** frame. However, in this case, the transformation will occur in several steps. The first step, already known, is the transformation of the fixed frame to the $\mathbf{C_0}$ frame. The second step is the transformation from the $\mathbf{C_0}$ frame to the $\mathbf{P_0}$ frame. Again, there are two rotations that line up the $\mathbf{C_0}$ coordinate axes with a general vector in space. The two rotations are:

$$\theta_{P1} = a\cos\left[\frac{\theta_x^P}{\sin\left(a\cos\left(\theta_z^P\right)\right)}\right] \quad \theta_{P2} = a\cos\left(\theta_z^P\right) - \left(\frac{\pi}{2}\right) \tag{3. 64}$$

Recall that in a computer code the direction cosines are given in radians not degrees. Now, the transformation follows directly from the previous derivation:

$$ {}^{\mathbf{P_0}}_{\mathbf{C_0}}\mathbf{T} = \begin{bmatrix} c\theta_{P2}c\theta_{P1} & c\theta_{P2}s\theta_{P1} & -s\theta_{P2} & 0 \\ -s\theta_{P1} & c\theta_{P1} & 0 & 0 \\ s\theta_{P2}c\theta_{P1} & s\theta_{P2}s\theta_{P1} & c\theta_{P2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3. 65}$$

Given a set of coordinates in the fixed frame, one can transform them to the $\mathbf{P_0}$ frame via the following calculation:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{P_0}} = {}^{\mathbf{P_0}}_{\mathbf{C_0}}\mathbf{T} \; {}^{\mathbf{C_0}}_{\mathbf{F}}\mathbf{T} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{F}} \tag{3. 66}$$

The transformation equation for coordinates initially given in the bolt frame is somewhat more complicated. Recall that the bolt vertices begin by being represented in the fixed frame. The vertices must be transformed to the current location of the bolt coordinate system, so they are transformed to the initial contact configuration first (such that the bolt coordinate system at the moment of initial contact is viewed from the fixed reference frame). Then they are transformed from the fixed to the $\mathbf{C_1}$ frame, followed by an inverse transformation to record the second contact configuration of the bolt (the moment when the bolt obtains two contact points) as represented in the $\mathbf{C_0}$ frame. Finally, the bolt vertices are transformed from the $\mathbf{C_0}$ to the $\mathbf{P_0}$ frame. The cyclic representation of the transformations involved is shown in Figure 3.17:

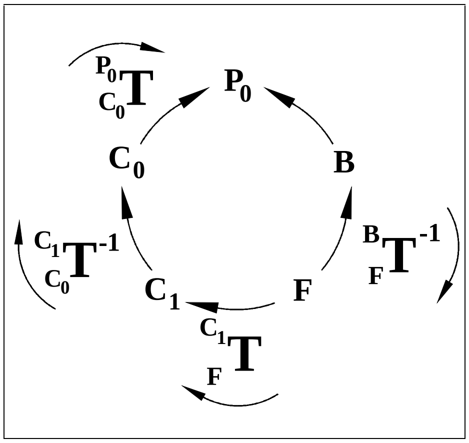


Figure 3. 15 Cyclic Diagram for 3rd Contact State Model

The entire homogenous transformation is shown in the equation below:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{P_0} = {}^{P_0}_{C_0}\mathbf{T}\ {}^{C_1}_{C_0}\mathbf{T}^{-1}\ {}^{C_1}_{F}\mathbf{T}\ {}^{b}_{F}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{b}} \qquad \textbf{(3. 67)}$$

Unlike the second contact point analysis, when the direction of rotation was known, the direction of rotation about the X_P-axis must be determined. The X-Y plane of the $\mathbf{P_0}$ frame is assumed to be parallel to the X-Y plane of the fixed frame, since a majority of the contact points will occur at or near the surface. Hence, the fixed frame is simulated at the surface of the nut while the direction of rotation about the X_P-axis is determined. The enumeration of the quadrant location of $\mathbf{CP_0}$ and $\mathbf{CP_1}$, the determination of whether the X_P-axis intersects the positive or negative X_F-axis and whether the slope of the X_P-axis is positive or negative will determine the direction of rotation. All of these parameters are measured with respect to the fixed reference frame. A portion of the rotation direction algorithm is shown below:

Table 3. 2

CP_0	X_P int.	Slope	CP_1	DIR. ROT
Q1	$+X_F$	Negative	Q2	Negative
			Q4	Positive
		Positive	Q3	Positive
			Q4	Positive
	$-X_F$	Negative	IMP	
		Positive	Q2	Negative
			Q3	Negative

For each of the quadrants that $\mathbf{CP_0}$ has the potential to occur in, there is one configuration that is geometrically impossible. In the previous example, it is geometrically impossible for $\mathbf{CP_0}$ to occur in quadrant one, the X_P-axis to intersect the negative X_F-axis, and for the slope of the X_P-axis to be negative. The remainder of the pivot direction algorithm is located in appendix C.

At the moment the bolt obtains a third contact point with the nut, it is assumed to have a stable configuration. At this point the motion of the bolt is complete, and the critical contact data can be exported into a data file. Figure 3.18 depicts the coordinate systems at the moment the third contact point ($\mathbf{CP_2}$) is obtained.

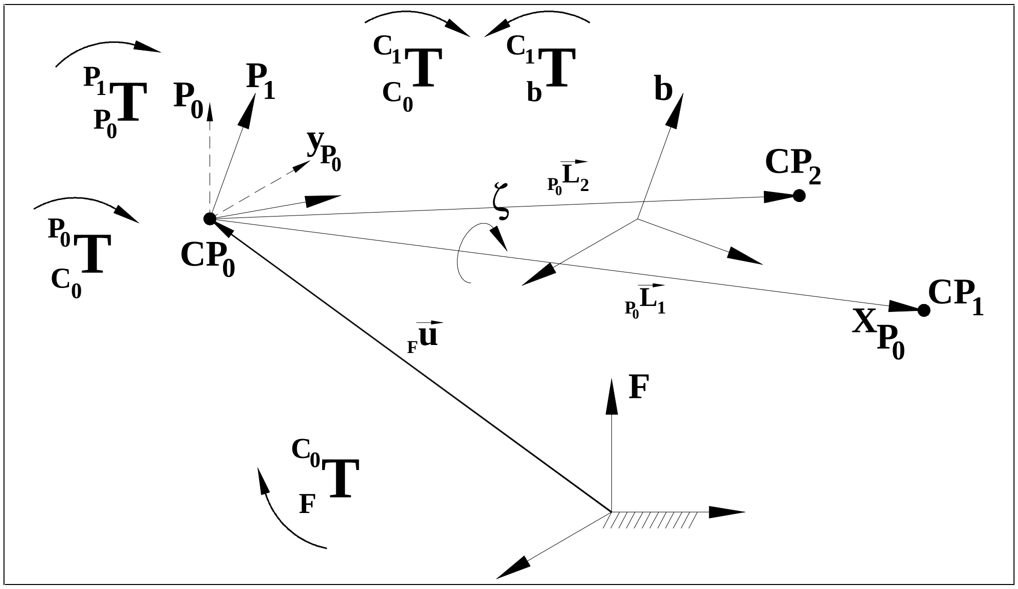


Figure 3. 16 Coordinate Frames at Three-Point Contact

The final transformation that pivots the bolt into the third contact point is defined as:

$$ {}^{P_1}_{P_0}\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\zeta & \sin\zeta & 0 \\ 0 & -\sin\zeta & \cos\zeta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{(3.\ 68)} $$

The series of transformations that convert a set of contact coordinates as measured in the $\mathbf{P_0}$ to the fixed frame is shown below:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{F}} = {}^{\mathbf{C_0}}_{\mathbf{F}}\mathbf{T}^{-1}\ {}^{\mathbf{P_0}}_{\mathbf{C_0}}\mathbf{T}^{-1} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{P_0}} \qquad \textbf{(3. 69)}$$

However, knowledge of where the contacts occurred as measured from the bolt frame is required to ascertain the appropriate parametric equation. Since the models of the bolt and nut were recreated in the $\mathbf{P_0}$ frame, any contact point will be represented by this coordinate system. Hence, the coordinates must be converted from the $\mathbf{P_0}$ frame to the bolt frame. This more complicated procedure begins with a transformation from the $\mathbf{P_0}$ frame to the $\mathbf{P_1}$ frame, followed by an inverse transformation that carries the coordinates from the $\mathbf{P_1}$ to the $\mathbf{C_0}$ frame. At this point the vertices are transformed from the $\mathbf{C_0}$ frame to the $\mathbf{C_1}$ frame, followed by another inverse transformation that converts the vertices from the $\mathbf{C_1}$ frame back to the bolt frame. A cyclic representation of the transformations involved in the conversion from the $\mathbf{P_0}$ to the bolt frame is shown in Figure 3.19:

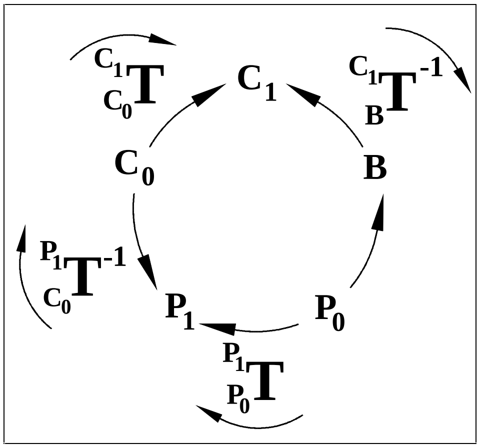


Figure 3. 17 Cyclic Diagram to Convert to Bolt Reference Frame

The entire transformation is shown in equation 3.70:

$$\begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{b}} = {}^{\mathbf{C_1}}_{\mathbf{b}}\mathbf{T}^{-1}\ {}^{\mathbf{C_1}}_{\mathbf{C_0}}\mathbf{T}\ {}^{\mathbf{P_1}}_{\mathbf{C_0}}\mathbf{T}^{-1}\ {}^{\mathbf{P_1}}_{\mathbf{P_0}}\mathbf{T} \begin{Bmatrix} x \\ y \\ z \\ 1 \end{Bmatrix}_{\mathbf{P_0}} \qquad \textbf{(3. 70)}$$

Note that the transformations from the bolt to the $\mathbf{C_1}$ frame and from the $\mathbf{C_0}$ to the $\mathbf{P_1}$ frame are equivalent to the transformations from the bolt to the $\mathbf{C_0}$ frame and from the $\mathbf{C_0}$ to the $\mathbf{P_0}$ frame, respectively.

The direction of the contact normal is critical in the determination of the potential for jamming and/or wedging. During a threaded assembly, it is possible to determine the common normal between two helically swept surfaces through the use of vectors tangent to the bolt and nut. The following section outlines the technique involved in the determination of a common normal at a contact point between the bolt and nut.

3.11 Common Normal between Two Curves in Space

If two known curves in space meet at a point, then it is possible to draw a tangent vector to each individual curve at that point. The cross product of the tangent vectors (since they are co-incident) will provide the common normal vector between them (Fig. 3.20). It is assumed here that each curve is planar to its respective coordinate system. By setting it up this way, one can find the unique tangent vector for each curve by drawing a position vector in Polar coordinates from the origin of each curve's reference frame.

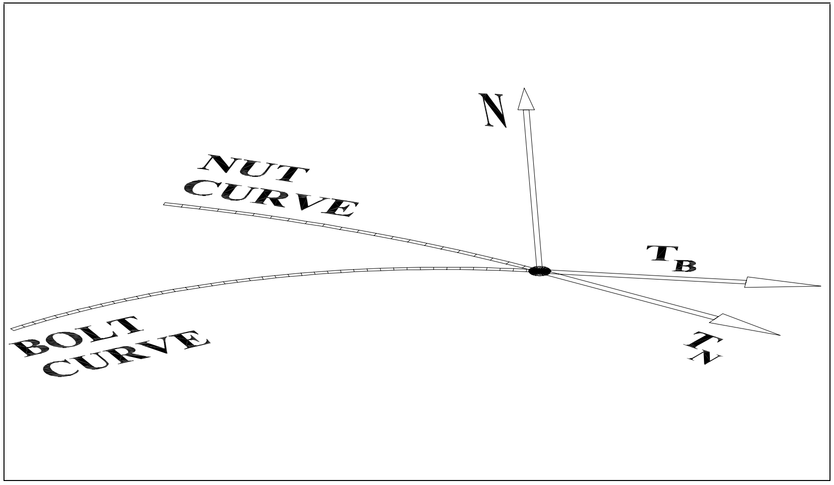


Figure 3. 18 Common Normal to Two Spatial Curves

$\mathbf{P_B}$ is the position vector, defined in Polar coordinates, drawn from the origin of the bolt frame to the contact point where R_B is a known scalar value. The tangent vector is defined as the derivative of the position vector with respect to its angular orientation (Munem, 1984). The unit tangent vector is found by normalizing the derivative, thereby elimination knowledge of the magnitude of the tangent vector. Since this study is consists of a purely geometric analysis, only the direction of the tangent is of importance. The procedure to obtain the tangent vector is defined in the following equations:

$$\mathbf{P_B} = R_B \cos(\theta)\hat{\mathbf{i}} + R_B \sin(\theta)\hat{\mathbf{j}} \tag{3. 71}$$

$$\mathbf{T_B} = \frac{d\mathbf{P_B}}{d\theta} = -R_B\left[\sin(\theta)\hat{\mathbf{i}} - \cos(\theta)\hat{\mathbf{j}}\right] \tag{3. 72}$$

$$\hat{\mathbf{T}}_\mathbf{B} = \frac{\frac{d\mathbf{P_B}}{d\theta}}{\left|\frac{d\mathbf{P_B}}{d\theta}\right|} = \frac{-R_B\left[\sin(\theta)\hat{\mathbf{i}} - \cos(\theta)\hat{\mathbf{j}}\right]}{R_B} \tag{3. 73}$$

$$\hat{\mathbf{T}}_\mathbf{B} = -\sin(\theta)\hat{\mathbf{i}} + \cos(\theta)\hat{\mathbf{j}} \tag{3. 74}$$

Now that the direction of the tangent vector is determined, it must be transformed back to the fixed reference frame, since it was initially defined in the bolt frame. Since the transformation is homogenous, the magnitude of the tangent vector will remain unity. The unit tangent vector of the planar curve located in the nut reference frame can be determined similarly. No transformation is needed, however, because this vector is defined in the fixed reference frame. Once the two tangent vectors are obtained, the cross product will yield the direction of the common normal between the two curves:

$$\mathbf{N} = \hat{\mathbf{T}}_\mathbf{B} \times \hat{\mathbf{T}}_\mathbf{N} \tag{3. 75}$$

Though the tangent vectors are unity, their cross product may not equal unity. Nevertheless, the direction cosines of the common normal can be obtained by normalizing its components. Recall that during the derivation of the parametric equations for the bolt and nut, R_B is not a scalar value but dependent on the angular location of the contact point. Therefore, it will not factor out of the equation unless the parametric equation has a constant radius as with the root and crest, respectively.

The next chapter will present the results of the contact analysis for multiple initial configurations. All of the techniques involved in this chapter were converted to computer code in order to automate the contact analysis via a computer simulation. For the interested reader, the computer code is available in appendix D.

Chapter 4

Spatial Contacts from Discretized Orientation Errors

Recall that the threaded assembly is completed by the conventional heuristic of rotating the bolt counter-clockwise until a “snap” is heard. The “snap” results from a sudden change in the height of the bolt and occurs when the bolt is in phase with the nut. From this point it is known that any clockwise rotation results in a successful assembly provided that the orientation is maintained. By utilizing the contact state models derived in the previous chapter, we conducted an exhaustive test sequence simulating this heuristic to ascertain the contact locations throughout a counter-clockwise rotation of the bolt.

The initial approach of the bolt has 6 degrees of freedom, three rotations (roll, pitch, and phase) and three translations (horizontal, vertical, and azimuth), resulting in a multitude of infinities. One way to reduce the number of free choices during the initial approach is to develop a pre-alignment strategy. An example of this is the Azimuth Rotation Strategy as discussed by Sturges (1996) that preconditions the constraint network for prismatic peg insertion. In our case, the bolt shall be considered as having two rotational freedoms (roll, α, and pitch, β) over the range –4, 0, and +4 degrees, as this is the extent of expected elastic compensation for an RCC. Thus, the bolt can be pre-aligned eight different ways based on the rotational degrees of freedom: $+\alpha$ only, $+\beta$ only, $-\alpha$ only, $-\beta$ only, $+\alpha$ and $+\beta$, $+\alpha$ and $-\beta$, $-\alpha$ and $+\beta$, $-\alpha$ and $-\beta$. For this analysis, it will be assumed that the translation errors in the vertical and horizontal directions (X and Y-axes) are corrected by the standard RCC. The azimuth translation (Z-axis) is constrained by the orientation of the bolt, as it is calculated once the bolt achieves its first contact. Hence the bolt will be positioned above the nut, and moved down the azimuth axis until RAPID detects a contact. Since the bolt will be rotated counter-clockwise, there are no restrictions on the phase angle ϕ, so it was discretized over one period (zero to 360°) in steps of 5 degrees, in order to determine the entire gamut of potential contact states. The complete set of motions involved in the contact analysis was numerically simulated by a supervisory code written in Microsoft Visual C++ version 5.0, which was compiled on a Pentium II 450 MHz computer with 128 Megabytes of RAM.

4.1 Computer Code Output File

During a simulation the code outputs the following information:

- Spatial Orientation Parameters (α,β,ϕ -- roll, pitch, and phase);
- The number of contact locations at the third contact state;
- The amount of rotation about the X_{C0} and X_{P0} axes (XROLL, PSI, respectively);
- The contact point coordinates once the bolt acquires its second contact point;
- The origin of the BOLT frame at the second contact state;
- The contact point coordinates once the bolt acquires its third contact point;
- The number of contact pairs involved in the determination of each of the contact points at the third contact state; and
- The origin of the BOLT frame at the third contact state, along with the coordinates of a point displaced along the Z-axis of the bolt frame by one pitch (0.050 inches).

A sample of the text file is shown in Figure 4.1 (coordinates measured with respect to the fixed reference frame):

```
ROLL =  -4           PITCH = 0           PHASE =6
Number of contact locations:  3          XROLL = -0.2199  PSI = -3.913
Contact Point Locations:
The original 1st two points are:
X0        -0.107471            Y0         -0.00764779         Z0         0.0499998
X1        0.109120             Y1         0.00327747          Z1         0.0500000
BOLT Origin at the Second Contact State
1.8258e-007          3.67833e-008        0.0498432
The three computed contact points are:
X                    Y                   Z                   # Contact Pairs
0.109120             0.00327661          0.0500000
0.109572             0.01437670          0.0499961
0.109408             0.00930305          0.0499955                    26
-0.107468            -0.00759809         0.0499928
-0.107471            -0.00764896         0.0500000
-0.107471            -0.00764779         0.0499998                    4
0.103651             0.04944990          0.0500000
0.103642             0.04947910          0.0500000
0.103647             0.04946450          0.0500000                    4
BOLT Origin at the Third Contact State
9.8259e-007          -1.58201e-005       0.0496918
BOLT Top Surface at the Third Contact State
-0.00034701          0.00688221          0.0992125
```

Figure 4.1 – Angles are in degrees, Coordinates are in inches

It is of interest to determine if the bolt purely rotates about the initial two contact points. If so, then we are assured that the model is sufficient to locate the three contact points that stabilize the bolt. However, if the number of contact pairs increases at either of the original contact points, then the model may not be providing accurate information as to the whereabouts of the third contact point. One way to determine if the model is actually locating three contact points is to analyze the depth of intersection between the bolt and nut, since excessive intersection at the endpoints of the pivot axis represents an impossible state.

4.2 Contact Embedment Analysis

In order to determine if there was excessive interference, the origin of the bolt reference frame at the two-point and three-point contact states needs to be recorded. If the origin of the bolt frame moved less then the amount that the bolt embedded in the nut, then we have an unstable case because we located a third contact point by doing something impossible (embedding a rigid body into another rigid body). If the origin of the bolt frame drops more than the amount that the bolt embeds in the nut, then this a possible movement because the bolt is releasing its potential energy. Furthermore, the location of the pivot axis will determine the stability of the bolt at the second contact state. If the pivot axis crosses at or near the origin of the bolt, then we could have a special "quasi-stable" two-point contact case. Here, the axis passes close to the compliance center (origin of the bolt frame) at which point the moment due to the applied load is nearly negligible. If the axis is far away from the origin of the bolt, then the code was unable to determine where the third contact point is because the pivot axis is *rolling* on the surface thereby producing an unstable contact state. A *rolling* axis is defined as acquiring more triangles at one or both of the original points once rotation about the pivot axis commences. Therefore, the output file was analyzed to determine whether the computed contact information was two-point unstable, two-point quasi-stable or three-point stable throughout each of the eight approaches.

4.2.1 Unstable Two-Point Contact State

Figure 4.1 depicts four contact points labeled 1, 2, 3A, and 3B. Points 1 and 2 define the endpoints of the pivot axis, or the X-axis of the $\mathbf{P_0}$ frame. The code simulates a rotation of the bolt about the pivot axis until RAPID returns (and the code categorizes) three distinct contact states. In a majority of the test runs, it was found that the third contact point was only obtained by embedding itself into the nut, thereby producing a discontinuity in the contact pair output. An example of an unstable state for the given orientation is shown below:

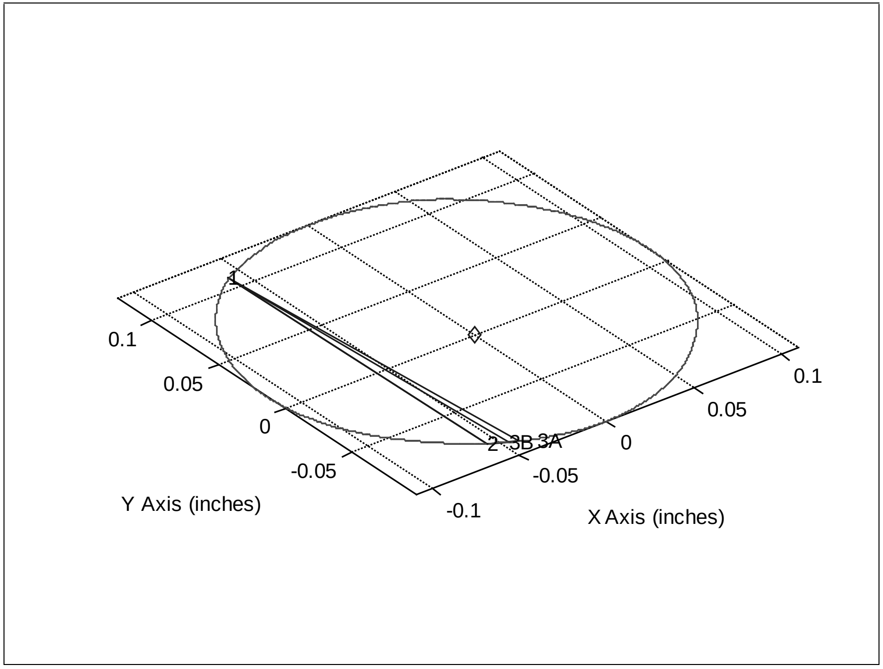


Figure 4. 2 Unstable Contact Point Locations @ (α=4°, β=0°, ϕ=185°)

The diamond in the center of the chart is the location of the bolt origin when the third contact state is reached. The circular object is the interior edge of the planar cross-section of the nut. The swath of embedment is pictorially shown by the close proximity of points 3A and 3B. Here, the code detected a discontinuity between points 2 and 3A, so it stopped rotating the bolt about the pivot axis and returned a series of contact pairs. In this case, it is determined that the bolt is unstable, because the origin of the bolt frame moved less than the amount embedded in the nut. The origin of the bolt frame moved from 0.047 to 0.046 inches (1.19 to 1.17 mm), or a displacement of 0.001 inches (0.03

mm). However, the depth of embedment was found to be 0.004 inches (0.10 mm), which is greater than the movement of the bolt reference frame, therefore we have an unstable case.

4.2.2 Quasi-Stable Two-Point Contact State

A “quasi-stable” two-point case is also present when approaching the nut with a positive rotation about the roll and pitch axes (Figure 4.2). Here the bolt reference frame dropped 0.0045 inches (0.11 mm) but the largest amount of interference was 0.0013 inches (0.03 mm), therefore the bolt released some potential energy as it moved down the azimuth axis of the fixed reference frame. Though only two contact points are present, the pivot axis passes near the origin of the bolt reference frame, thereby providing the potential for stability with an appropriately applied force.

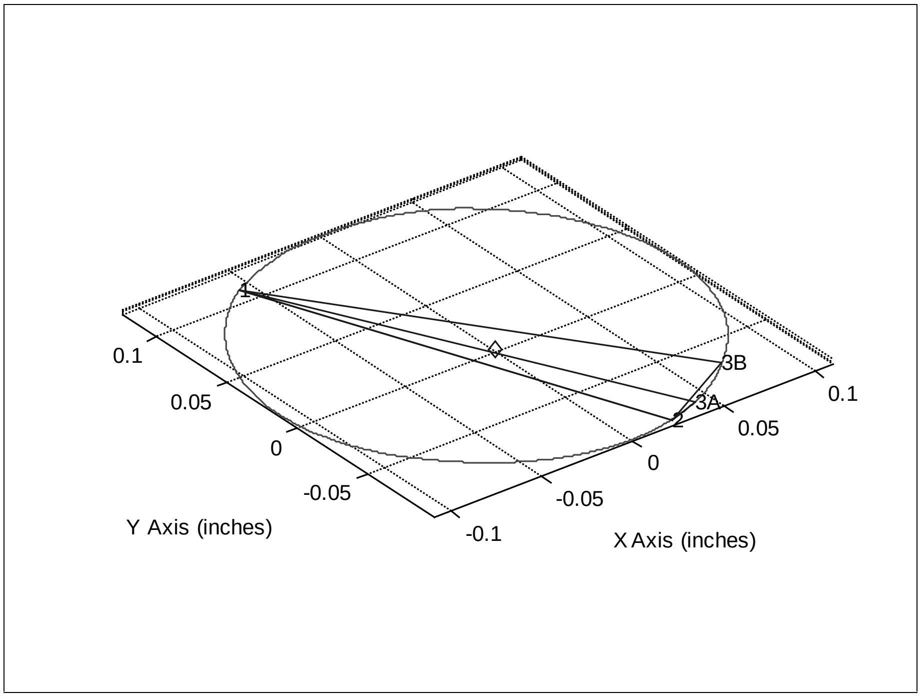


Figure 4. 3 Quasi-Stable Contact Point Locations @ (α=4°, β=4°, ϕ=235°)

Once again, the diamond in the center of the chart signifies the origin of the bolt reference frame, and points 1 and 2 represent the endpoints of the pivot axis. However, the code continued to rotate until it obtained a swath of triangles that begin with point 3A and end with 3B, since it is unable to decipher a quasi-stable contact configuration.

4.2.3 Stable Three-Point Contact State

An example of a stable three-point contact case is shown in Figure 4.3. Here, the number of contact pairs at each of the distinct contact points is minimized (4 pairs each). Moreover, the bolt did not embed itself into the nut as it pivoted about the axis defined by points 1 and 2. The origin of the bolt frame dropped slightly as it rotated about the X_{P0}-axis, however, all three contact points occurred at the surface of the nut which means that the two solid bodies did not interfere with one another.

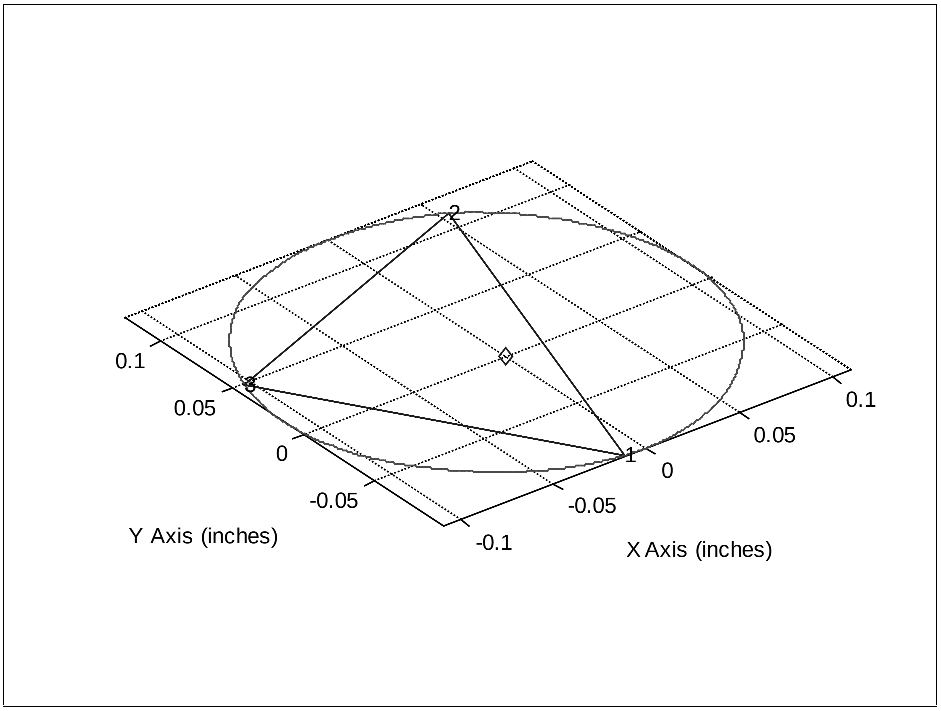


Figure 4. 4 Stable Contact Point Locations @ (α=-4°, β=0°, ϕ=131°)

This type of result was interspersed throughout each of the approaches, however, it occurred the least number of times.

4.3 Code Validity

The code functioned as intended, in that it was capable of isolating three distinct contact points. Yet, the model is limited in its ability to isolate actual non-interfering contact points since it constrained the bolt to rotate about an axis fixed in space. Moreover, there were instances when one of the endpoints of the pivot axis disappeared after the third contact point was recorded. This meant that the bolt lost a contact as it rotated about the pivot axis, and the code continued to rotate the bolt in order to obtain three distinct sets of contact pairs. In this case, the physical response of the bolt would be to release some potential energy by moving down the azimuth axis of the fixed reference frame, however, this was not accounted for in the model. An adjustment to the model, such as modeling a rolling pivot axis, or performing a compound rotation about the pivot axis, has the potential to improve the results.

Nevertheless, the parametric equations derived in chapter two can be used without modification to create a full spectrum of maps at any point in the history of a threaded assembly operation. Recall that, in the two-dimensional case, the contact points were calculated once the contacting surface regions were identified.

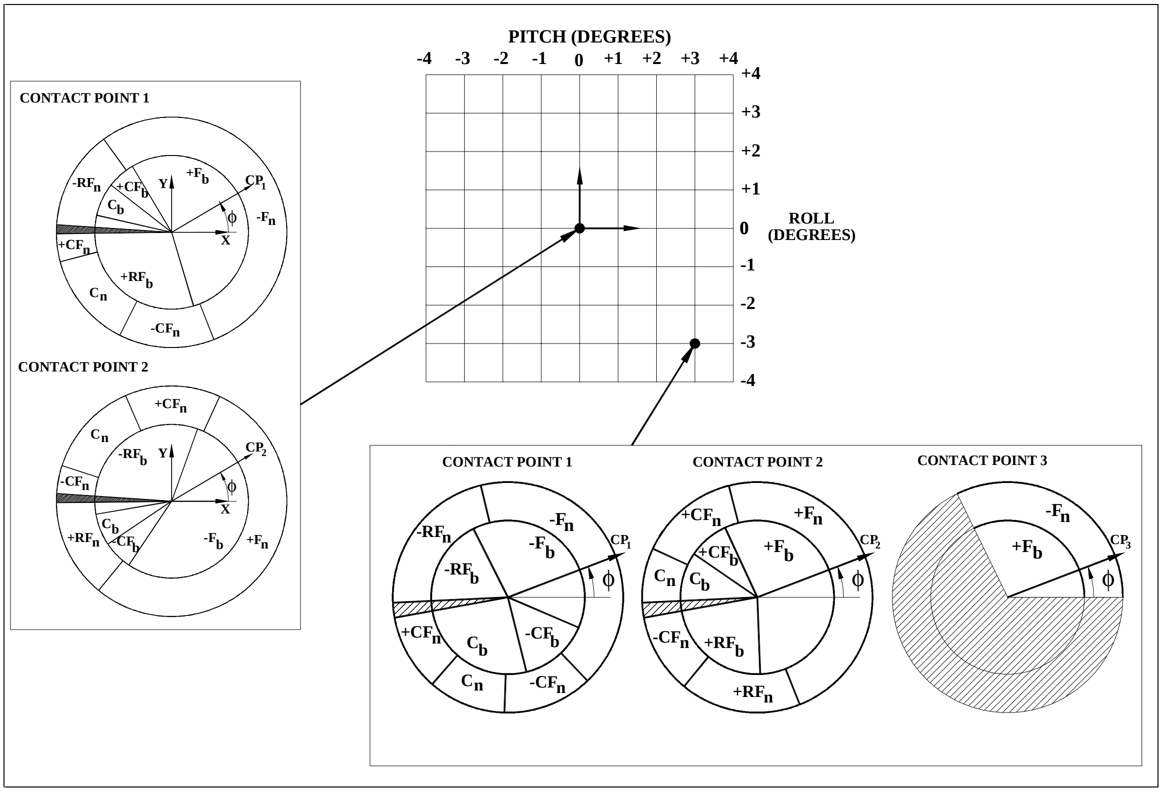


Figure 4. 5 Expansion of Two Dimensional Analysis

In the three-dimensional case, errors in rotation about the X and Y-axes of the bolt reference frame are permitted. This creates the opportunity for out-of-plane contacts, where each contact will have its own annulus describing the surface regions in contact. However, the annulus of chapter two must be modified to represent the parametric equations of the bolt cross-section at the moment a third contact point is obtained, or when the bolt is no longer parallel to the nut surface. Figure 4.4 shows the expansion of the two-dimensional analysis into the three-dimensional contact analysis. For our purposes, values of roll and pitch were calculated at every one-degree from –4 to +4 degrees. Thus, the three-dimensional analysis covers 81 cases of thread mating contacts for every possible phase angle between the nut and bolt. For example, figure 4.4 shows the contact states for a bolt with rotational errors of –3 degrees roll and +3 degrees pitch. The contact regions, once the third contact is obtained, are shown below the grid that represents the gamut of orientation errors.

In summary, we have investigated 81 potential orientations, each of which has its own set of contact points. This amount of variation constitutes a library that is comprised of all the contact states. From this library, the user will be able to correlate a contact state history, and develop a constraint network that permits successful assembly. The choice and number of grid points and phase angles used in this analysis convince us, by exhaustive examination, that there are no other possible contact states.

Chapter 5
Conclusion & Future Work

The demands for an increase in productivity and reduced assembly costs require engineers to automate solutions that replace manual labor. Currently, there does not exist a passive solution to assemble threaded fasteners, a common assembly primitive. Therefore, this work concentrated on threaded fastener insertion, in an effort to determine the nature of contact between the bolt and nut prior to an in-phase condition. This is the first step in the construction of a passive solution to this assembly problem, while the next two steps are the development of a constraint network guaranteeing successful assembly and experimentation to prove the theoretical design. In this chapter we summarize the results of this work and highlight the major contributions. We conclude with recommendations for future work concerning the analysis of threaded fastener assembly.

5.1 Planar Analysis

We started our research by simplifying the problem as a two-dimensional assembly analysis to gain an understanding about how contacts between the bolt and nut change during counter-clockwise motion. Here we were able to extract the planar parametric equations and the theoretical contact point locations when the bolt is flush with the surface. Though the location of the contact points varied with phase, it was of primary interest to identify the regions in contact so as to be able to calculate the common normal at the contact point, once the planar equations are extrapolated to a spatial reference system.

Yet, the tangent vectors can not be determined unless the contact point is known before hand. The technique used to locate the contact point began with the tessellation of one pitch of a #1/4-20 UNC thread, both internal and external. Tessellation was necessary since we utilized a collision detection library, RAPID, which required the solid bodies to be stored in memory as a series of triangles. During a collision, spatial geometry was used to locate an approximate contact point. Here, given a contact pair, the coordinates of the endpoints of a line segment created by the planar intersection of a bolt and nut triangle were enumerated through vector geometry. The midpoint of the line segment for each contact pair was averaged to produce an approximate location of the contact point. The minimum number of triangles was required to reduce the variation in position due to tessellation, so the bolt was dropped onto the nut in a series of iterations ending with 1 micro inch (25.4×10^{-9} m) adjustments.

5.2 Contact State Model & Results

Since the insertion force is assumed to be acting through the origin of the bolt reference frame and parallel to the azimuth axis of the bolt, the first contact state was constructed that rotated the bolt about a tangent vector to the surface or edge of the nut until a second contact point was obtained. The advent of a second contact point presented a more constrained contact state since we are interested in maintaining both contacts; thus it was hypothesized that the bolt would rotate about an axis defined by the original two contact points. The appropriate transformations were derived and the models entered into RAPID such that it was possible to rotate about the pivot axis until a third contact point is enumerated.

However, in order to show that the hypothesis was true, we analyzed the depth of intersection of the bolt into the nut as well as the vertical movement of the origin of the bolt reference frame. This determined whether the bolt was releasing potential energy or embedding itself into the nut. The calculated results show three types of contact states returned by the program:

- Unstable Two-Point Contact State
- Quasi-Stable Two-Point Contact State
- Stable Three Point Contact State

Though the unstable case still needs to be deciphered, the parametric equations derived in this work can be used without modification to create a full spectrum of maps at any point in the history of a threaded assembly problem. We investigated 81 potential orientations, each of which has its own set of contact points, which was described by a contact region disk. From this exhaustive examination, we are capable of detailing a contact state history and, from this, able to develop a constraint network.

5.3 Future Work

We have eliminated some of the unknown contact behavior between a bolt and nut during threaded assembly, but in order to complete the library of contacts one must continue to decipher the unstable two-point contact states. Here, the results can be improved by altering the model after the first contact point is obtained, such that the third contact point can be identified. One possibility would be to rotate the bolt about a dynamically rolling pivot axis. For instance, conduct a rotation about a vector tangent to the surface of the nut, and then determine if the contact pairs at the endpoints of the pivot axis increased. If the bolt lost some potential energy in the move, then alter the position of the pivot axis to represent the adjusted pair of contact points. If one of the endpoints disappears during an incremental rotation, then the bolt must be rotated about a vector tangent to the remaining contact point until a new endpoint can be found. If the bolt does not lose potential energy during a rotation about the pivot axis, then there is a possibility for a quasi-stable case provided that the pivot axis passes close to the origin of the bolt reference frame. Another possibility would be to rotate the bolt about a compound angle in such a way that it maintains the original two contacts and picks up a third without interfering with the nut. Such a rotation has the potential to be calculated through numerical simulation, which will condense several individual movements into one rotation.

Once the remaining regions are identified, and a constraint network is developed, it is our hope that designers of threaded fastener strategies will use the proposed generalized constraint network as an evaluation tool. For example, such a tool will provide the designer with the correct pitch angle setting (for an SRCC) based on thread geometry. Furthermore, it is recommended that experimentation be conducted with an RCC to determine what ranges of phase, if any, permit clockwise rotation of a threaded fastener without cross threading. This, in turn, could lead to an understanding of which contact states to avoid such that the automatic assembly system can reliably fasten a bolt and nut in a clockwise, and subsequently, more efficient manner.