Modification of Position and Attitude Determination of a Test Article through Photogrammetry to Account for Structural Deformation

Sean A. Krolikowski

Follow this and additional works at: https://scholar.afit.edu/etd

Part of the Aerospace Engineering Commons, and the Mechanics of Materials Commons

Recommended Citation

This Thesis is brought to you for free and open access by the Student Graduate Works at AFIT Scholar. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of AFIT Scholar. For more information, please contact richard.mansfield@afit.edu.
MODIFICATION OF POSITION AND ATTITUDE
DETERMINATION OF A TEST ARTICLE THROUGH
PHOTOGRAMMETRY TO ACCOUNT FOR STRUCTURAL
DEFORMATION

THESIS

Sean A. Krolikowski, First Lieutenant, USAF

AFIT/GA/ENY/01M-03

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.
MODIFICATION OF POSITION AND ATTITUDE DETERMINATION OF A TEST ARTICLE THROUGH PHOTOGRAMMETRY TO ACCOUNT FOR STRUCTURAL DEFORMATION

THESIS

Presented to the Faculty
Department of Aeronautics and Astronautics
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Astronautical Engineering

Sean Andrew Krolikowski, B.S.
First Lieutenant, USAF

March 2001

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED
MODIFICATION OF POSITION AND ATTITUDE
DETERMINATION OF A TEST ARTICLE THROUGH
PHOTOGRAMMETRY TO ACCOUNT FOR STRUCTURAL
DEFORMATION

Sean Andrew Krolkowski, B.S.
First Lieutenant, USAF

Approved:

Dr. Steven Tragesser (Chairman) 7 MAR 01
Dr. Curtis Spenny (Member) 7 March 01
Maj Montgomery Hughson (Member) 7 MAR 01
Acknowledgements

I would like to thank my advisor, Dr. Steven Tragesser, for all of his help on this thesis. I would also like to thank Dr. Wim Ruyten for his guidance and for providing the code for the rigid model which I modified to include deformation. If I had to start from scratch I think I would still be coding. Finally, I would like to thank my fiancee, for all of her love and support, even if it was from the other side of the country.

Sean Andrew Krolikowski
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>iv</td>
</tr>
<tr>
<td>List of Figures</td>
<td>vii</td>
</tr>
<tr>
<td>List of Symbols</td>
<td>viii</td>
</tr>
<tr>
<td>List of Abbreviations</td>
<td>ix</td>
</tr>
<tr>
<td>Abstract</td>
<td>x</td>
</tr>
<tr>
<td><strong>I. Introduction</strong></td>
<td></td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1-1</td>
</tr>
<tr>
<td>1.2 Problem Statement</td>
<td>1-2</td>
</tr>
<tr>
<td>1.3 Methodology</td>
<td>1-2</td>
</tr>
<tr>
<td>1.4 Assumptions/Limitations</td>
<td>1-3</td>
</tr>
<tr>
<td><strong>II. Position and Attitude Determination</strong></td>
<td></td>
</tr>
<tr>
<td>2.1 Non-Topographic Photogrammetry</td>
<td>2-1</td>
</tr>
<tr>
<td>2.2 Nonlinear Fitting Scheme</td>
<td>2-5</td>
</tr>
<tr>
<td>2.3 Levenberg-Marquardt</td>
<td>2-8</td>
</tr>
<tr>
<td><strong>III. Deformation Modeling</strong></td>
<td></td>
</tr>
<tr>
<td>3.1 Parabolic Bending</td>
<td>3-1</td>
</tr>
<tr>
<td>3.2 Linear Twist</td>
<td>3-4</td>
</tr>
<tr>
<td>3.3 Implementation of Deformation Models</td>
<td>3-6</td>
</tr>
<tr>
<td>3.4 Evaluating the Deformation Model</td>
<td>3-9</td>
</tr>
<tr>
<td>3.4.1 Target Distribution</td>
<td>3-10</td>
</tr>
<tr>
<td>3.4.2 Camera Views</td>
<td>3-11</td>
</tr>
</tbody>
</table>
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1.</td>
<td>NTP Set Up</td>
<td>2-2</td>
</tr>
<tr>
<td>2.2.</td>
<td>Wind Tunnel Set Up</td>
<td>2-5</td>
</tr>
<tr>
<td>3.1.</td>
<td>Parabolic Bending Set Up</td>
<td>3-1</td>
</tr>
<tr>
<td>3.2.</td>
<td>Wire-frame Bent Wing</td>
<td>3-2</td>
</tr>
<tr>
<td>3.3.</td>
<td>Linear Twisting Set Up</td>
<td>3-4</td>
</tr>
<tr>
<td>3.4.</td>
<td>Wire-Frame Twisted Wing</td>
<td>3-5</td>
</tr>
<tr>
<td>3.5.</td>
<td>Set-up of Truth Model Test Article</td>
<td>3-9</td>
</tr>
<tr>
<td>3.6.</td>
<td>Set-Up of Y Interval Determination</td>
<td>3-11</td>
</tr>
<tr>
<td>3.7.</td>
<td>Wind Tunnel Camera Set Up</td>
<td>3-12</td>
</tr>
<tr>
<td>3.8.</td>
<td>Sample View of Cameras 1-4</td>
<td>3-13</td>
</tr>
<tr>
<td>3.9.</td>
<td>Sample View of Cameras 5-8</td>
<td>3-13</td>
</tr>
<tr>
<td>4.1.</td>
<td>Relative Error Versus Number of Data Points, Severely Deformed</td>
<td>4-1</td>
</tr>
<tr>
<td>4.2.</td>
<td>Relative Error Versus Number of Data Points, Moderately Deformed</td>
<td>4-2</td>
</tr>
<tr>
<td>4.3.</td>
<td>Relative Error Versus YCF</td>
<td>4-3</td>
</tr>
<tr>
<td>4.4.</td>
<td>Comparison of YCF=1 to YCF=1.25</td>
<td>4-4</td>
</tr>
<tr>
<td>4.5.</td>
<td>Relative error versus number of cameras, moderate bending</td>
<td>4-4</td>
</tr>
<tr>
<td>4.6.</td>
<td>Relative Error Versus Number of Cameras, Severe bending</td>
<td>4-5</td>
</tr>
<tr>
<td>4.7.</td>
<td>Relative Error Versus Bending Coefficient for Bending and Rigid Models</td>
<td>4-6</td>
</tr>
<tr>
<td>4.8.</td>
<td>Relative Error Versus Twisting Coefficient for Bending and Rigid Models</td>
<td>4-7</td>
</tr>
<tr>
<td>4.9.</td>
<td>Relative Error Versus Noise Level for Bending and Rigid Models</td>
<td>4-8</td>
</tr>
</tbody>
</table>
## List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f$ Focal Length</td>
<td>2-1</td>
</tr>
<tr>
<td>$\vec{A}$ vector from $f$ to target on model</td>
<td>2-1</td>
</tr>
<tr>
<td>$\vec{a}$ vector from $f$ to target in image</td>
<td>2-2</td>
</tr>
<tr>
<td>$\phi$ roll angle of model</td>
<td>2-5</td>
</tr>
<tr>
<td>$\alpha$ pitch angle of model</td>
<td>2-5</td>
</tr>
<tr>
<td>$\beta$ yaw angle of model</td>
<td>2-5</td>
</tr>
<tr>
<td>$\Delta x$ displacement of model frame from tunnel frame in $x$ direction</td>
<td>2-6</td>
</tr>
<tr>
<td>$\Delta y$ displacement of model frame from tunnel frame in $x$ direction</td>
<td>2-6</td>
</tr>
<tr>
<td>$\Delta z$ displacement of model frame from tunnel frame in $x$ direction</td>
<td>2-6</td>
</tr>
<tr>
<td>$\vec{q}$ unknown parameter vector</td>
<td>2-7</td>
</tr>
<tr>
<td>$\chi^2$ merit function</td>
<td>2-7</td>
</tr>
<tr>
<td>$K_{bend}$ bending coefficient</td>
<td>3-2</td>
</tr>
<tr>
<td>$L$ length of wing</td>
<td>3-2</td>
</tr>
<tr>
<td>$K_{twist}$ twisting coefficient</td>
<td>3-5</td>
</tr>
<tr>
<td>XDF X Density Factor</td>
<td>3-10</td>
</tr>
<tr>
<td>YDF Y Density Factor</td>
<td>3-10</td>
</tr>
<tr>
<td>YCF Y Cluster Factor</td>
<td>3-10</td>
</tr>
</tbody>
</table>
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>L-M Levenberg-Marquardt</td>
<td>x</td>
</tr>
<tr>
<td>AEDC Arnold Engineering Development Center</td>
<td>1-1</td>
</tr>
<tr>
<td>PSP Pressure-Sensitive Paint</td>
<td>1-1</td>
</tr>
<tr>
<td>NTP Non-Topographic Photogrammetry</td>
<td>2-1</td>
</tr>
<tr>
<td>TRS Tunnel Reference System</td>
<td>2-5</td>
</tr>
<tr>
<td>rms Root Mean Squared</td>
<td>2-8</td>
</tr>
</tbody>
</table>
Abstract

The Arnold Engineering Development Center (AEDC) at Arnold AFB, TN currently has a computer program which, through a process known as photogrammetry, combines multiple 2D images of a wind tunnel test article, affixed with numerous registration markers, and the known 3D coordinates of those markers. It can then accurately determine the unknown position and attitude of the test article relative to the wind tunnel. The current algorithm has a problem in that it assumes the test article is a rigid body, when, in fact, the test article experiences deformation under aerodynamic loads. Due to this deformation, the 3D coordinates of the markers are not precisely known.

This research looks at modifying the current program to account for this deformation and to improve the accuracy of the position and attitude determination of the test article. The current program uses the Levenberg-Marquardt method of multi-parameter optimization to solve for the unknown parameters of position and attitude. In this work, deformation is modeled in two modes, simple parabolic bending and linear twisting, and uses the L-M method to solve for these additional parameters. This work also determines the minimum number of targets and cameras required to obtain the maximum accuracy. It varies the model targets from about 20 to 200, and looks at using 1, 2, 4, 6, and 8 cameras. The results are a great improvement in accuracy over the original program. The results also show that optimal accuracy is obtained with approximately 50 targets and 2 cameras. Any more than this produces an extremely small improvement in accuracy, with no real added benefit.

It is clear that by adding simple bending and twisting parameters to the list of unknowns in the L-M solver, a much greater accuracy can be achieved in the determination of the position and attitude.
MODIFICATION OF POSITION AND ATTITUDE
DETERMINATION OF A TEST ARTICLE THROUGH
PHOTOGRAMMETRY TO ACCOUNT FOR STRUCTURAL
DEFORMATION

I. Introduction

1.1 Background

One of the tasks of the Arnold Engineering Development Center (AEDC) at Arnold Air Force Base, Tennessee is to place the customer’s small-scale model of their vehicle, be it airplane, heavy lift rocket, etc., in a wind tunnel and measure the aerodynamic loading, giving an approximation of what the real loading environment will be. The old method was to use a “a pressure loads model, one instrumented with hundreds of pressure orifices” [8] to measure the pressures across the model’s surface. Today that method is changing to one that is more efficient and cost-effective. That method utilizes pressure-sensitive paint PSP. Dr. Wim Ruyten, AEDC, describes this process [10]:

In PSP measurements, we paint scale models of aircraft or other objects with a special paint that glows when ultraviolet light shines on it. The glow has a different color than the light that produces it, so we can use optical filters to separate the two. Even more important, the brightness of the glow depends on the air pressure on the model. So by taking pictures of the model, we can back out what the pressure is.

A crucial element of the pressure determination process is knowing the exact position and attitude of the test article relative to the wind tunnel. In the old method, this was done with “a complex procedure for combining and calibrating data from sting-mounted balance sensors and strain gages.” [5] Dr. Ruyten goes on the
say that "...there is a growing interest to measure angles of attack with an accuracy that surpasses .01 deg. This level of accuracy cannot be obtained using traditional measurements based on balance sensors and strain gages." [6] Once again, a more accurate and efficient method was found. It is an optical method using registration markers placed on the test article. Through a process known as photogrammetry, it is possible to take one or more 2D images of the article and its registration markers, or targets, and combine that with the known 3D coordinates of the targets to back out the position and attitude of the article.

There is a problem with the current method. This method assumes that the test article is a rigid body. This means that the three-dimensional coordinates of the targets in the model reference frame are known at all times. This is not the case. After repeated exposure to aerodynamic loading, the article, notably appendages such as wings and stabilizers, will experience small structural deformations. This means that the three-dimensional coordinates of the targets in the model frame are constantly changing. As time progresses and the model becomes more deformed, the current method will become more and more inaccurate at position and attitude determination.

1.2 Problem Statement

Improve the accuracy of position and attitude determination of a wind tunnel test article by accounting for structural deformation. Determine the optimal number of targets and cameras needed to obtain the acceptable accuracy.

1.3 Methodology

The current method utilizes the Levenberg-Marquardt method of multi-parameter optimization. The known parameters are the camera location(s) and orientation(s) and the 3D coordinates of the registration markers in the model coordinate frame. The unknown parameters are the three position and three attitude parameters. It
then minimizes the least squares merit function of the predicted target coordinates in the camera frame and the measured target coordinates in the camera frame. This minimization produces the six position and attitude parameters.

This thesis adds two unknown parameters to the equations, a parabolic bending coefficient and a linear twisting coefficient. By adding these simple models of deformation, the program will more accurately compute the position and attitude of a deformed test article.

This thesis also completes many sample runs of data, varying both the number of targets and the number of cameras. Through this analysis it shows that there is an optimal number of targets and cameras where the accuracy is still kept at a maximum.

1.4 Assumptions/Limitations

The methodology employed here tries to model the deformation of the test article in two ways, parabolic bending and linear twist about a central line. There are some limitations in this method which will prevent it from ever precisely determining the position and attitude of the article. First, this method was conceived with the assumption that the deformed piece of the article would be a wing or a rocket fin or some other protrusion from a main body. The idea of a second order bending and a first order twisting are suited to this type of application. The majority of AEDC's test articles are an aircraft configuration of some type, which is why this method is used. However, this method may not be appropriate for every conceivable test article that AEDC may use. Second, second order bending and first order twisting are very simple approximations of the true deformation that occurs. These are believed to be good approximations, but they are by no means perfect.
II. Position and Attitude Determination

2.1 Non-Topographic Photogrammetry

Accurate determination of position and attitude of the wind tunnel test article is not only important for pressure-sensitive paint testing, but is in fact "one of the persistent interests in wind tunnel testing" [6] Balance sensor and strain gages are not meeting today's accuracy requirements. More accurate optical methods are being used, and those methods are based on a method known as Non-Topographic Photogrammetry, introduced in 1979 by H.M. Karara et al. [2]

Non-Topographic Photogrammetry (NTP) considers the case where an object has been photographed by one or more exterior cameras. The goal is to determine the coordinates of the targets (small black dots placed on the surface of the test article) in the two dimensional frame of the photograph. The three dimensional coordinates of the targets in the model frame are assumed known, and the position and orientation of the camera(s) need to be determined through calibration. In this thesis, one of the assumptions is that the position and orientation of the camera(s) are already known, because the calibration process is performed before any aerodynamic loading is placed on the model, and thus the model remains undeformed. Therefore, the details of the calibration process will not be presented. It should also be noted that any lens or image distortion are neglected in this study, as the cameras are assumed to be perfect.

The configuration of object and camera can be seen in Figure 2.1. From the figure, \( XYZ \) is the frame associated with the test article or model, and \( uv \) is the frame associated with the photograph (hence the inverted image). The camera lens is at the intersection of all the lines, and is denoted by the focal length, \( f \). For the derivation, we have selected one target point on the top of the canopy, denoted by \((x_i, y_i, z_i)\). \( \vec{A} \) is the vector from the focal point to the selected target point on
Figure 2.1  NTP Set Up

the model. $\vec{a}$ is the vector from the focal point to the selected target point in the photographic image, denoted by $(u_i, v_i)$

The relationship between the model frame and the image frame needs to be established. This relationship can be described by a three axis coordinate transformation. First, the model frame is rotated about it's $X$ axis by an angle $\omega$, arriving at the first intermediary axis. This axis is then rotated about it's $Y'$ axis by an angle $\phi$, taking it to the second intermediary axis. Finally, this axis can be rotated about it's $Z''$ axis by an angle $\kappa$, transforming it into the final image coordinate frame.

\[
\begin{bmatrix}
X' \\
Y' \\
Z'
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \omega & \sin \omega \\
0 & -\sin \omega & \cos \omega
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix}
\]

\[
\begin{bmatrix}
X'' \\
Y'' \\
Z''
\end{bmatrix} = \begin{bmatrix}
\cos \phi & 0 & -\sin \phi \\
0 & 1 & 0 \\
\sin \phi & 0 & \cos \phi
\end{bmatrix}
\begin{bmatrix}
X' \\
Y' \\
Z'
\end{bmatrix}
\]
Equations (2.1), (2.2), (2.3) can be combined to form one large transformation matrix, Equation (2.4).

\[
M = \begin{bmatrix}
\cos \kappa \cos \phi & \cos \kappa \sin \phi \sin \omega + \sin \kappa \cos \omega & -\cos \kappa \sin \phi \cos \omega + \sin \kappa \sin \omega \\
-\sin \kappa \cos \phi & -\sin \kappa \sin \phi \sin \omega + \cos \kappa \cos \omega & \sin \kappa \sin \phi \cos \omega + \cos \kappa \sin \omega \\
\sin \phi & -\cos \phi \sin \omega & \cos \phi \cos \omega
\end{bmatrix}
\]  

(2.4)

The focal point is given coordinates in both reference frames. In the model frame it is denoted as \((x_c, y_c, z_c)\). In the image frame, the focal coordinates are \((u_c, v_c)\). Thus \(\bar{A}\) now becomes Equation (2.5), and \(\bar{a}\) now becomes Equation (2.6).

\[
\bar{A} = \begin{bmatrix}
x_i - x_c \\
y_i - y_c \\
z_i - z_c
\end{bmatrix}
\]  

(2.5)

\[
\bar{a} = \begin{bmatrix}
u_i - u_c \\
v_i - v_c \\
-f
\end{bmatrix}
\]  

(2.6)

To compare the two vectors \(\bar{a}\) and \(\bar{A}\), we need to get them both in terms of coordinates in the same reference frame. Thus, \(M\) will transform \(\bar{A}\) to the image coordinate frame. The result is shown below where \(U, V,\) and \(W\) are the coordinates of \(\bar{A}\) in the image frame.

\[
U = (\cos \kappa \cos \phi)(x_i - x_c) + (\cos \kappa \sin \phi \sin \omega + \sin \kappa \cos \omega)(y_i - y_c)
+ (\cos \kappa \sin \phi \cos \omega + \sin \kappa \sin \omega)(z_i - z_c)
\]
\[ V = (\sin \kappa \cos \phi)(x_i - x_c) + (- \sin \kappa \sin \phi \sin \omega + \cos \kappa \cos \omega)(y_i - y_c) \]
\[ + (\cos \kappa \sin \phi \cos \omega + \cos \kappa \sin \omega)(z_i - z_c) \]
\[ W = (\sin \phi)(x_i - x_c) + (- \cos \phi \sin \omega)(y_i - y_c) + (\cos \phi \cos \omega)(z_i - z_c) \]

The key to this whole process is realizing that, due to the nature of imaging, \( \vec{A} \) and \( \vec{a} \) are collinear. As H.M. Karara says, "The imaging process requires that the image and object rays be collinear, that is, that the components of the two vectors expressed in the same coordinate system be equal, except for a scale factor." [2] Thus we can say that \( \vec{a} = k \vec{M} \vec{A} \), where \( k \) is the scale factor. Expressing this equation in the image frame coordinates, we get

\[ u_i - u_c = kU \]
\[ v_i - v_c = kV \]
\[ -f = kW \]

The exact value of the scale factor \( k \) is unknown, but we can solve the third equation of (2.8) for \( k \), and substitute that result into the first and second equations of (2.8). This gives us equation (2.9).

\[ u_i = u_c - f \frac{U}{W} \]
\[ v_i = v_c - f \frac{V}{W} \]

This is the result of Non-Topographic Photogrammetry. Knowing the location of the camera lens (or focus), the coordinates of the desired target in the model frame, and the orientation of the model frame with respect to the image frame, we can calculate what the coordinates of that target will be in the image frame. Position and attitude determination will turn this around and, knowing what the image...
coordinates of the target are from the image taken, determine what the orientation of the model is with respect to the camera.

2.2 Nonlinear Fitting Scheme

Non-Topographic Photogrammetry can now be applied to the test article in the wind tunnel to determine the position and orientation of the article with respect to the tunnel. The initial set up can be seen in Figure 2.2, where $X^*Y^*Z^*$ is the coordinate frame associated with the tunnel, also known as the Tunnel Reference System or TRS.

![Figure 2.2 Wind Tunnel Set Up](image)

The TRS and the model frame may be offset by three Euler angles. A rotation about the $X^*$ axis will be denoted by $\phi$, and is also known as roll. A rotation about the $Y^*$ axis will be denoted by $\alpha$, and is also known as pitch. A rotation about the $Z^*$ axis will be denoted by $\beta$, and is also known as yaw. The first task is to transform the coordinates of the target from the model frame to the tunnel frame. This is accomplished in Equation (2.10), which shows that this is just a matter of rotating the model frame to the tunnel frame, similar to what was done in the
previous section, but \( R \) is the rotation matrix using the angles \( \alpha, \beta, \) and \( \phi \), whereas \( M \) was the rotation matrix using the angles \( \kappa, \phi, \) and \( \omega \). Also, the displacements of the model frame from the tunnel frame, \( \Delta x, \Delta y, \) and \( \Delta z \) have been added in.

\[
x_i^* = \Delta x + M x_i \\
y_i^* = \Delta y + M y_i \\
z_i^* = \Delta z + M z_i
\]  

(2.10)

Where \( x_i, y_i, \) and \( z_i \) are the coordinates of the target point in the model frame, and \( x_i^*, y_i^*, \) and \( z_i^* \) are the coordinates of the target point in the tunnel reference frame.

The coordinates of the target in the TRS are given by Equation (2.11).

\[
x_i^* = \Delta x + x_i(\cos \alpha \cos \beta) + y_i(\sin \beta \cos \phi + \sin \alpha \cos \beta \sin \phi) \\
+ z_i(-\sin \beta \sin \phi + \sin \alpha \cos \beta \cos \phi) \\
y_i^* = \Delta y + x_i(-\cos \alpha \sin \beta) + y_i(\cos \beta \cos \phi - \sin \alpha \sin \beta \sin \phi) \\
+ z_i(-\cos \beta \sin \phi - \sin \alpha \sin \beta \cos \phi) \\
z_i^* = \Delta z + x_i(-\sin \alpha) + y_i(\cos \alpha \sin \phi) + z_i(\cos \alpha \cos \phi)
\]  

(2.11)

It is assumed that both the camera location and orientation are known from calibration. The location and orientation parameters, instead of being with respect to the model frame as in the previous section, are here with respect to the TRS. The location parameters are given by \( x_i^*, y_i^*, \) and \( z_i^* \). The orientation angles of the camera are given by \( \phi_i^*, \kappa_i^*, \) and \( \omega_i^* \). The coordinates of the target are now transformed from the tunnel frame to the image frame, using the same camera rotation matrix as before. The coordinates in the image frame \( U, V, \) and \( W \) are now affixed with an asterisk to indicate that they came from tunnel coordinates. This is shown by
Equation 2.12.

\[
U_{ci}^* = (\cos \kappa^* \cos \phi^*)(x_i^* - x_c^*) + (\cos \kappa^* \sin \phi^* \sin \omega^* + \sin \kappa^* \cos \omega^*)(y_i^* - y_c^*) \\
+(-\cos \kappa^* \sin \phi^* \cos \omega^* + \sin \kappa^* \sin \omega^*)(z_i^* - z_c^*)
\]

\[
V_{ci}^* = (-\sin \kappa^* \cos \phi^*)(x_i^* - x_c^*) + (-\sin \kappa^* \sin \phi^* \sin \omega^* + \cos \kappa^* \cos \omega^*)(y_i^* - y_c^*) \\
+(\sin \kappa^* \sin \phi^* \cos \omega^* + \cos \kappa^* \sin \omega^*)(z_i^* - z_c^*)
\]

\[
W_{ci}^* = (\sin \phi^*)(x_i^* - x_c^*) + (-\cos \phi^* \sin \omega^*)(y_i^* - y_c^*) + (\cos \phi^* \cos \omega^*)(z_i^* - z_c^*)
\]

Applying the same collinearity principles from the previous section, we arrive at the same result as (2.9), except that now the image coordinates are a function of the tunnel coordinates, not the model coordinates. The result is

\[
u_{ci} = u_c - \frac{U_{ci}^*}{W_{ci}^*} = u_{ci}(\vec{q})
\]

\[
v_{ci} = v_c - \frac{V_{ci}^*}{W_{ci}^*} = v_{ci}(\vec{q})
\]

Equation (2.13)

The right side of Equation (2.13) shows that now the only unknown parameters remaining in \( u_{ci} \) and \( v_{ci} \) are the position and attitude parameters of the model. They have been grouped into the vector \( \vec{q} \), given by

\[
\vec{q}^p = [\Delta x, \Delta y, \Delta z, \alpha, \beta, \phi]
\]

According to Dr. Ruyten, to solve for the six unknown position and attitude parameters, the minimization of a least squares sum is used. [7] That sum is called the \( \chi^2 \) merit function, and it is the difference between the photographed image coordinates of the targets, denoted by \( \bar{u}_{ci} \) and \( \bar{v}_{ci} \), and the image coordinates as a function of the unknown parameters \( \vec{q} \), given by

\[
\chi^2(\vec{q}) = \sum_c \sum_i \{(u_{ci}(\vec{q}) - \bar{u}_{ci})^2 + (v_{ci}(\vec{q}) - \bar{v}_{ci})^2\}
\]

2-7
The summation index $c$ shows that this function is summed over all cameras, if there are more than one, and $i$ indicates that it is summed over all target coordinates. Dr. Ruyten also explains that a function closely related to the $\chi^2$ merit function is the rms fit error. “This error gives the rms deviation (in pixels) between measured and fitted image coordinates.” [7] The rms fit error function is given by

$$\sigma(q) = \left[ \frac{1}{N} \chi^2(q) \right]^{\frac{1}{2}}$$

(2.16)

where N is the total number of image coordinate pairs.

A successful minimization of the $\chi^2$ merit function will result in values for each of the unknown position and attitude parameters. However, because there are six unknown parameters, minimizing this function is difficult. It requires a multi-parameter optimization scheme. The method employed is called a Levenberg-Marquardt algorithm, and is explained in the next section.

2.3 Levenberg-Marquardt

Levenberg-Marquardt is one of many non-linear methods of data modeling, or multi-parameter optimization. However, Dr. Ruyten has chosen to use the LM method because, as he says [7]

Experience has shown that (even when employing as many as 94 fit parameters — six for model alignment and 11 parameters for 8 cameras each) satisfactory convergence of the LM algorithm is typically reached in 1-10 iterations. This constitutes a significant speed-up over the simplex method that was employed [before].

The book *Numerical Recipes in Fortran* [11] does an excellent job of explaining the LM algorithm. In general, LM follows these steps:

1. Pick initial values for the unknown parameters. Usually this will be 0, but in the case of the actual wind tunnel these could be the preliminary values read from the machine gages.
(2) Evaluate $\chi^2$ using initial values and image data.

(3) Increment the unknown parameters by a small amount, and re-evaluate $\chi^2$.

(4) If the new $\chi^2$ is greater than the previous one, increase the increment by a factor of 10, and evaluate again.

(5) If the new $\chi^2$ is less than the previous one, decrease the increment by a factor of 10, and evaluate again.

(6) Continue until the difference in the functions is less than some tolerance, typically $10^{-3}$.

The first two steps are relatively easy, as are evaluating whether $\chi^2$ has increased or decreased. The true heart of this nonlinear method is determining the magnitude and direction in which to increment the unknown parameters. Close to the minimum, the $\chi^2$ function is expected to be well approximated by a quadratic form, which can be written as

$$\chi^2(q) \approx \gamma - d \cdot q + \frac{1}{2} q \cdot D \cdot q$$  \hspace{1cm} (2.17)

where $d$ is an $M$-vector, and $D$ is an $M \times M$ matrix. If this approximation is a good one, we can jump from the current trial parameters, $q_{\text{cur}}$, to the minimizing ones, $q_{\text{min}}$, in a single leap, given by

$$q_{\text{min}} = q_{\text{cur}} + D^{-1} \cdot [ -\nabla \chi^2(q_{\text{cur}}) ]$$  \hspace{1cm} (2.18)

However, this may be a poor local approximation to the shape of the function that we are trying to minimize at $q_{\text{cur}}$. If this is true, the best we can do is to step down the gradient using the steepest decent, given by

$$q_{\text{next}} = q_{\text{cur}} - \text{constant} \times \nabla \chi^2(q_{\text{cur}})$$  \hspace{1cm} (2.19)
where the constant is small enough not to exhaust the downhill direction.

To use Equations 2.18 and 2.19, we need to be able to compute the gradient of the \( \chi^2 \) function at any set of parameters \( \vec{q} \). To use Equation 2.18 we also need the matrix \( D \), which is the second derivative matrix (Hessian matrix) of the \( \chi^2 \) merit function, at any \( \vec{q} \).

We have specified the \( \chi^2 \) merit function, therefore the Hessian matrix is known to us. Therefore, we can use Equation 2.18 whenever we choose to. The only reason to use Equation 2.19 will be if Equation 2.18 fails to improve the fit, signaling failure of Equation 2.17 as a good local approximation.

First, we need to determine partial derivatives of \( \chi^2 \) with respect to the set of \( M \) unknown parameters in \( \vec{q} \). Taking partial derivatives once arrives at the gradient (Equation 2.20), which will be zero at the \( \chi^2 \) minimum.

\[
\frac{\partial \chi^2}{\partial q_k} = -2 \sum_c \sum_i \left[ (u_{ci}(\vec{q}) - \bar{u}_{ci}) \frac{\partial u_{ci}(\vec{q})}{\partial q_k} + (v_{ci}(\vec{q}) - \bar{v}_{ci}) \frac{\partial v_{ci}(\vec{q})}{\partial q_k} \right] \\
k = 1, 2, ..., M \tag{2.20}
\]

Taking an additional partial derivative yields Equation 2.21

\[
\frac{\partial^2 \chi^2}{\partial q_k \partial q_l} = 2 \sum_c \sum_i \left[ \frac{\partial u_{ci}(\vec{q})}{\partial q_k} \frac{\partial u_{ci}(\vec{q})}{\partial q_l} - \frac{\partial u_{ci}(\vec{q})}{\partial q_l} \frac{\partial u_{ci}(\vec{q})}{\partial q_k} \right] + \\
\quad \frac{\partial v_{ci}(\vec{q})}{\partial q_k} \frac{\partial v_{ci}(\vec{q})}{\partial q_l} - \frac{\partial v_{ci}(\vec{q})}{\partial q_l} \frac{\partial v_{ci}(\vec{q})}{\partial q_k} \right] \tag{2.21}
\]

However, the \( \frac{\partial^2}{\partial q_l \partial q_k} \) terms are deemed sufficiently small, and the equation reduces to

\[
\frac{\partial^2 \chi^2}{\partial q_k \partial q_l} = 2 \sum_c \sum_i \left[ \frac{\partial u_{ci}(\vec{q})}{\partial q_k} \frac{\partial u_{ci}(\vec{q})}{\partial q_l} + \frac{\partial v_{ci}(\vec{q})}{\partial q_k} \frac{\partial v_{ci}(\vec{q})}{\partial q_l} \right] \tag{2.22}
\]
We now need to solve for the partial derivatives of $u_{ci}(q)$ and $v_{ci}(q)$ with respect to each of the six unknown parameters. The partial derivatives are given as

\[
\frac{\partial u_{ci}(q)}{\partial q_k} = -\frac{f}{W_{ci}} \left[ (\cos \kappa^* \cos \phi^*) \left( \frac{\partial x_i^*}{\partial q_k} \right) + (\cos \kappa^* \sin \phi^* \sin \omega^* + \sin \kappa^* \cos \omega^*) \left( \frac{\partial y_i^*}{\partial q_k} \right) \\
+ (-\cos \kappa^* \sin \phi^* \cos \omega^* + \sin \kappa^* \sin \omega^*) \left( \frac{\partial z_i^*}{\partial q_k} \right) \right] \\
+ \frac{fU_{ci}}{W_{ci}^2} \left[ (\sin \phi^*) \left( \frac{\partial x_i^*}{\partial q_k} \right) + (-\cos \phi^* \sin \omega^*) \left( \frac{\partial y_i^*}{\partial q_k} \right) + (\cos \phi^* \cos \omega^*) \left( \frac{\partial z_i^*}{\partial q_k} \right) \right]
\]

(2.23)

\[
\frac{\partial v_{ci}(q)}{\partial q_k} = -\frac{f}{W_{ci}} \left[ (-\sin \kappa^* \cos \phi^*) \left( \frac{\partial x_i^*}{\partial q_k} \right) + (-\sin \kappa^* \sin \phi^* \sin \omega^* + \cos \kappa^* \cos \omega^*) \left( \frac{\partial y_i^*}{\partial q_k} \right) \\
+ (\sin \kappa^* \sin \phi^* \cos \omega^* + \cos \kappa^* \sin \omega^*) \left( \frac{\partial z_i^*}{\partial q_k} \right) \right] \\
+ \frac{fV_{ci}}{W_{ci}^2} \left[ (\sin \phi^*) \left( \frac{\partial x_i^*}{\partial q_k} \right) + (-\cos \phi^* \sin \omega^*) \left( \frac{\partial y_i^*}{\partial q_k} \right) + (\cos \phi^* \cos \omega^*) \left( \frac{\partial z_i^*}{\partial q_k} \right) \right]
\]

Notice in Equation 2.23 that only $\frac{\partial x_i^*}{\partial q_k}$, $\frac{\partial y_i^*}{\partial q_k}$, and $\frac{\partial z_i^*}{\partial q_k}$ change now as the unknown parameter, $q$, with which the partial derivative is taken with respect to changes. These partial derivatives with respect to the six unknown position and attitude parameters are given as

\[
\begin{align*}
\frac{\partial x_i^*}{\partial \Delta x} &= 1 \\
\frac{\partial x_i^*}{\partial \Delta y} &= 0 \\
\frac{\partial x_i^*}{\partial \Delta z} &= 0 \\
\frac{\partial y_i^*}{\partial \alpha} &= \cos \beta (z_i^* - \Delta z) \\
\frac{\partial z_i^*}{\partial \beta} &= (y_i^* - \Delta y) \\
\frac{\partial x_i^*}{\partial \phi} &= \gamma_i (-\sin \beta \sin \phi + \sin \alpha \cos \beta \cos \phi) - \\
&(\sin \beta \cos \phi + \sin \alpha \cos \beta \sin \phi) z_i
\end{align*}
\]

(2.24)
\[
\frac{\partial y^*_i}{\partial \Delta x} = 0 \\
\frac{\partial y^*_i}{\partial \Delta y} = 1 \\
\frac{\partial y^*_i}{\partial \Delta z} = 0 \\
\frac{\partial y^*_i}{\partial \alpha} = -\sin \beta (z^*_i - \Delta z) \\
\frac{\partial y^*_i}{\partial \beta} = -(x^*_i - \Delta x) \\
\frac{\partial y^*_i}{\partial \phi} = y_i (-\cos \beta \sin \phi - \sin \alpha \sin \beta \cos \phi) - \\
(cos \beta \cos \phi - \sin \alpha \sin \beta \sin \phi) z_i \\
\]

According to *Numerical Recipes* [11], it is conventional to remove the factors of 2 by defining

\[
\beta_k = -\frac{1}{2} \frac{\partial \chi^2}{\partial q_k} \\
\alpha_{kl} = \frac{1}{2} \frac{\partial^2 \chi^2}{\partial q_k \partial q_l} \\
\]

(2.27)
making \( [\alpha] = \frac{1}{2} \mathbf{D} \) in Equation (2.18), in terms of which that equation can be rewritten as the set of linear equations

\[
\sum_{l=1}^{M} \alpha_{kl} \delta q_l = \beta_k
\]  

(2.28)

This set is then solved for \( \delta q_l \), which is the increment that is added to the unknown parameters. The key to the LM method is that it makes one big improvement over this standard method. Normally, \( \delta q_l \) equals some constant times \( \beta_l \). However, Marquardt realized that the scale of this constant is dictated by the reciprocal of the diagonal element of the alpha matrix. He also inserted another factor, \( \lambda \), which could be set to much less than one to reduce the step size. The result of these realizations is Equation (2.29).

\[
\delta q_l = \frac{1}{\lambda \alpha_{ll}} \beta_l
\]  

(2.29)

Marquardt also realized that Equation (2.29) could be combined with Equation (2.28) if a new matrix, \( \alpha' \), is defined by

\[
\alpha'_{jj} = \alpha_{jj}(1 + \lambda)
\]

\[
\alpha'_{jk} = \alpha_{jk}
\]  

(2.30)

Which then yields Equation (2.31)

\[
\sum_{l=1}^{M} \alpha'_{kl} \delta q_l = \beta_k
\]  

(2.31)

This now is the set of linear equations the LM method uses to determine the increment to apply to the unknown parameters in \( \bar{q} \).

The last step that remains in this process is to determine the precision of the fitted parameters. According to Dr. Ruyten, the precision of each fit parameter, \( q_k \), is given by [7]

\[
\sigma_{q_k} = \left[ \frac{N}{2N - M} \right]^\frac{1}{2} \sigma(q) \sigma_{kk}^\frac{1}{2}
\]  

(2.32)
where $N$ is the number of image coordinate pairs, $M$ is the number of fit parameters, $\sigma(\bar{q})$ is the rms fit error, given by Equation (2.16), and $C_{kk}$ are the diagonal elements of the covariance matrix $C$. The covariance matrix $C$ is found by inverting the curvature matrix, $\alpha_{kl}$, given by Equation (2.27).
III. Deformation Modeling

Modeling of structural deformation can be an extremely complicated field, typically requiring some type of finite element analysis. We tried to compromise somewhere between a rigid model, which is what the current program uses, and a finite element analysis, which may be too complicated to implement in a program such as this. Since the wings, horizontal, and vertical stabilizers of small scale aircraft test articles undergo significant deformation, we tailored our model for these structures. From his testing experience, Dr. Ruyten suggested that the deformation could be modeled by superposition of parabolic bending and linear twisting. [4]

3.1 Parabolic Bending

![Parabolic Bending Set Up](image)

Figure 3.1 Parabolic Bending Set Up

Realistically, the wing would not bend linearly, such that the entire wing is deflected at a constant angle. It would be much more rigid near the fuselage where all of the structural support is, and would be more flexible near the tip due to the moment arm from the base of the wing to the tip. Thus, under severe aerodynamic loading,
The wing should deflect in a curved manner. This behavior can be approximated as parabolic bending. Figure 3.1 shows the deflection for parabolic bending.

The equation used to model parabolic bending is

\[ z = K_{bend} \left( \frac{y}{L} \right)^2 \]  

(3.1)

where \( z \) is the deflection value, \( K_{bend} \) is the bending coefficient, \( y \) is the distance from the base of the wing to the target point, and \( L \) is the total length of the wing. Figure 3.2 shows a MATLAB-generated wire-frame model of a wing displaying parabolic bending. As will be discussed in the next chapter, the wing is the approximate dimensions of a Lockheed Martin F-22A Raptor wing, with a wing length of 6.78 meters. The bending coefficient is .1, meaning that the tip of the wing is .1m lower than an undeformed wing as shown in the figure.

There is a problem with simply applying the bend and twist equations to the undeformed coordinates to get deformed coordinates. The new \( Z \) value is calculated based on the bend and twist functions and the wing is essentially "stretched". For example a point that was on the wing tip, with a \( Y \) value of 6.78, would have a
new Z value of, say, -.5, but would still have a Y value of 6.78. The wing is being elongated, and this is not a very accurate representation.

One way to account for this is to first calculate the path length from the origin to the undeformed point. Then, follow the curve of the bending function until it reaches that same path length. Find the new Y value for that path length and replace the old Y with the new one. In this way, the function no longer stretches the wing and is more accurate. To apply this to our bending function, we use a method prescribed in Advanced Engineering Mathematics [3]. We first find a parametric representation of the bend function, which is given by

\[ r(t) = t \hat{i} + \frac{BCt^2}{Y_{\text{max}}} \hat{j} \]  

(3.2)

Now, find the derivative with respect to the parameter \( t \), which is given by

\[ r'(t) = t \hat{i} + \frac{2BCt}{Y_{\text{max}}} \hat{j} \]  

(3.3)

We now find \( r' \cdot r' \), which is given by

\[ r' \cdot r' = 1 + t \left[ \frac{2BC}{Y_{\text{max}}} \right]^2 \]  

(3.4)

We can now apply this to the general equation for the arc length of a curve, which is given by

\[ l = \int_a^b \sqrt{r' \cdot r'} \, dt \]  

(3.5)

where, in our case \( a = 0 \) and \( b = Y \). For each point, we simply set \( l \) equal to the undeformed path length, and solve for the new Y value, which is the upper limit of integration. The undeformed path length is simply

\[ l_{\text{und}} = \sqrt{X^2 + Y^2} \]  

(3.6)
This method of correction is not applied to the twisting function for two reasons. First, the twisting displacement is generally smaller than the bending displacement. Second, bending is only a function of one variable, and the path length will only vary in one direction. Thus it is correctable. Twisting is a function of $X$ and $Y$, and therefore the parameterization of the path length is significantly more complex.

3.2 Linear Twist

The other mode of deformation being modeled is linear twisting. For this model we assume the base at the wing is rigidly attached to the fuselage and that the deflection is linear at each chord line, meaning that the positive deflection on the leading edge is equal in magnitude to the negative deflection on the trailing edge. However, at each increasing chord interval, that angle is increased. Thus the twist gets more and more severe. Figure 3.3 shows the set-up for linear twist.

\[ z = K_{\text{twist}} \left( \frac{y}{L} \right) \left( \frac{x}{X_{\text{max}}} \right) \]
where $z$ is the deflection value, $K_{\text{twist}}$ is the twisting coefficient, $y$ is the distance from the base of the wing to the target point, $L$ is the total length of the wing, $x$ is distance along the chord, from the origin to the target point, and $X_{\text{max}}$ is the total chord length, at that particular target point. This function will provide no twist at the base, where $y$ is equal to zero. It will also provide maximum twisting upwards at the tip on the leading edge, and maximum twisting downward on the trailing edge.

Figure 3.4 shows a MATLAB-generated wire-frame model of a wing displaying linear twisting. This graph is a little deceptive, as it appears to have some curve to it. One difference is that the set up shows a rectangular wing, where as Figure 3.4 is again the F-22 modeled wing. The midline of the wing goes from the midpoint of the base to the midpoint of the tip, exactly dividing the wing in half at each chord interval. The twist is about this line, which is at an angle, compared to the rectangular wing which has it's midline perfectly straight. The other factor to account for is that this is a severely twisted wing (twist coefficient of 4), to show the effects of twisting. As previously mentioned, there is no way to correct for the stretching of the wing where twisting is concerned. Because of this, some stretching of the wing is evident in the graph. Thus, while the graph seems to show some
curvature, it is in fact linear twisting. Hence, the equation for the total deflection, accounting for both parabolic bending and linear twist is

\[ z = K_{bend} \left( \frac{y}{L} \right)^2 + K_{twist} \left( \frac{y}{L} \right) \left( \frac{x}{X_{\text{max}}} \right) \]  

(3.8)

3.3 Implementation of Deformation Models

With equations for both the bending and twisting of the wing, we now need to integrate these into the Levenberg-Marquardt optimizer. Initially there were six unknown parameters, three for position and three for attitude. Now we will introduce two more unknown parameters, the bending and twisting coefficients. When the program tries to match position and attitude parameters to the given images, it understands there exists the possibility the images were taken from a deformed article. Solving for these deformation parameters will yield a more accurate solution for the position and attitude.

As stated in the previous chapter, the Levenberg-Marquardt method uses partial derivatives of the equations with respect to the unknown parameters to determine the step size and direction. Since we have included two new unknown parameters to solve for, this means calculating two new sets of partial derivatives.

Recall from Equation (2.23) that only \( \frac{\partial x_i^*}{\partial q_k}, \frac{\partial y_i^*}{\partial q_k}, \) and \( \frac{\partial z_i^*}{\partial q_k} \) change as the unknown parameter with which the partial derivative is taken with respect to changes. Essentially, this means that to add in \( K_{\text{bend}} \) and \( K_{\text{twist}} \) as parameters, all that really needs to be solved are the partial derivatives of \( x_i^* \), \( y_i^* \), and \( z_i^* \) with respect to the unknown parameters, now including \( K_{\text{bend}}, \) and \( K_{\text{twist}}. \) Of course, \( x_i^* \), \( y_i^* \), and \( z_i^* \) now include the deformation functions.

First, we need to combine the equations governing deformation into the equations that transform target coordinates from the model frame to the image frame.
That is,

\[ x_i^* = \Delta x + x_i (\cos \alpha \cos \beta) + y_i (\sin \beta \cos \phi + \sin \alpha \cos \beta \sin \phi) \]

\[ + \left[ K_{\text{twist}} \left( \frac{y}{L} \right) \left( \frac{x}{x_{\text{max}}} \right) - K_{\text{bend}} \left( \frac{y}{L} \right)^2 \right] (- \sin \beta \sin \phi + \sin \alpha \cos \beta \cos \phi) \]

\[ y_i^* = \Delta y + x_i (- \cos \alpha \sin \beta) + y_i (\cos \beta \cos \phi - \sin \alpha \sin \beta \sin \phi) \]

\[ + \left[ K_{\text{twist}} \left( \frac{y}{L} \right) \left( \frac{x}{x_{\text{max}}} \right) - K_{\text{bend}} \left( \frac{y}{L} \right)^2 \right] (- \cos \beta \sin \phi - \sin \alpha \sin \beta \cos \phi) \]

\[ z_i^* = \Delta z + x_i (- \sin \alpha) + y_i (\cos \alpha \sin \phi) + \left[ K_{\text{twist}} \left( \frac{y}{L} \right) \left( \frac{x}{x_{\text{max}}} \right) - K_{\text{bend}} \left( \frac{y}{L} \right)^2 \right] (\cos \alpha \cos \phi) \]

Now, simply take the partial derivatives of each with respect to all the unknown parameters, including \( K_{\text{bend}} \) and \( K_{\text{twist}} \). This is given by

\[
\frac{\partial x_i^*}{\partial \Delta x} = 1 \\
\frac{\partial x_i^*}{\partial \Delta y} = 0 \\
\frac{\partial x_i^*}{\partial \Delta z} = 0 \\
\frac{\partial x_i^*}{\partial \alpha} = \cos \beta (z_i^* - \Delta z) \\
\frac{\partial x_i^*}{\partial \phi} = y_i (- \sin \beta \sin \phi + \sin \alpha \cos \beta \cos \phi) - \\
(\sin \beta \cos \phi + \sin \alpha \cos \beta \sin \phi) \left[ K_{\text{twist}} \left( \frac{y}{L} \right) \left( \frac{x}{x_{\text{max}}} \right) - K_{\text{bend}} \left( \frac{y}{L} \right)^2 \right] \\
\frac{\partial x_i^*}{\partial K_{\text{bend}}} = -(- \sin \beta \sin \phi + \sin \alpha \cos \beta \cos \phi) \left( \frac{y}{L} \right)^2 \\
\frac{\partial x_i^*}{\partial K_{\text{twist}}} = (- \sin \beta \sin \phi + \sin \alpha \cos \beta \cos \phi) \left( \frac{x}{x_{\text{max}}} \right) \left( \frac{y}{L} \right) \\
\]

3-7
The equations which convert the coordinates of the targets from the model frame into pixel coordinates in the image frame were modified to include the deformation functions prescribed. Partial derivatives of those functions were taken with respect to the old unknown position and attitude parameters, as well as the new coefficients and bending and twisting. These partial differential equations can now

\[
\begin{align*}
\frac{\partial y^*_t}{\partial \Delta x} &= 0 \\
\frac{\partial y^*_t}{\partial \Delta y} &= 1 \\
\frac{\partial y^*_t}{\partial \Delta z} &= 0 \\
\frac{\partial y^*_t}{\partial \alpha} &= -\sin \beta (z^*_i - \Delta z) \\
\frac{\partial y^*_t}{\partial \beta} &= -(x^*_i - \Delta x) \\
\frac{\partial y^*_t}{\partial \phi} &= y_i(-\cos \beta \sin \phi - \sin \alpha \sin \beta \cos \phi) - \\
&= (\cos \beta \cos \phi - \sin \alpha \sin \beta \sin \phi)\left[K_{twist}\left(\frac{y}{L}\right)\left(\frac{x}{X_{max}}\right) - K_{bend}\left(\frac{y}{L}\right)\right]^2 \\
\frac{\partial y^*_t}{\partial K_{bend}} &= -(-\cos \beta \sin \phi - \sin \alpha \sin \beta \cos \phi)\left(\frac{y}{L}\right)^2 \\
\frac{\partial y^*_t}{\partial K_{twist}} &= (-\cos \beta \sin \phi - \sin \alpha \sin \beta \cos \phi)\left(\frac{y}{L}\right)\left(\frac{x}{X_{max}}\right) \\
\frac{\partial z^*_t}{\partial \Delta x} &= 0 \\
\frac{\partial z^*_t}{\partial \Delta y} &= 0 \\
\frac{\partial z^*_t}{\partial \Delta z} &= 1 \\
\frac{\partial z^*_t}{\partial \alpha} &= -\cos \beta (x^*_i - \Delta x) + \sin \beta (y^*_i - \Delta y) \\
\frac{\partial z^*_t}{\partial \beta} &= 0 \\
\frac{\partial z^*_t}{\partial \phi} &= y_i(\cos \alpha \cos \phi) - (\cos \alpha \sin \phi)\left[K_{twist}\left(\frac{y}{L}\right)\left(\frac{x}{X_{max}}\right) - K_{bend}\left(\frac{y}{L}\right)\right]^2 \\
\frac{\partial z^*_t}{\partial K_{bend}} &= -(\cos \alpha \cos \phi)\left(\frac{y}{L}\right)^2 \\
\frac{\partial z^*_t}{\partial K_{twist}} &= (\cos \alpha \cos \phi)\left(\frac{y}{L}\right)\left(\frac{x}{X_{max}}\right) \\
\end{align*}
\]
be coded into SUBROUTINE mrqfun1 of the Fortran code (see Appendix B). The program is now modified and ready to account for deformation of the test article.

3.4 Evaluating the Deformation Model

Now needed is some way to evaluate the model with deformation against the original rigid body program to determine how much of an improvement has been made. To aid in the evaluation, a program was written in MATLAB to construct a hypothetical test article to be used as the "truth model". By comparing the original and modified programs to this truth model, quantitative error improvement results can be obtained. The code for this MATLAB program is shown in Appendix A.

The test article in the truth model is based on the approximate dimensions of a Lockheed Martin F-22A Raptor. The test article includes a rigid fuselage which is 19 meters long and 4 meters wide, and a wing that is 6.78 meters from base to tip, 9.85 meters long along the base, and 1.66 meters long along the tip. The wing can bend according to the parabolic bending and linear twisting defined in the previous chapter. This set up is shown in Figure 3.5.

![Figure 3.5 Set-up of Truth Model Test Article](image)
3.4.1 Target Distribution. One of the features of the truth model program is the ability to easily change the number and density of target locations on the wing. This helps answer the question of how many targets is optimal, and what kind of spacing is desired.

The program uses three variables, X Density Factor (XDF), Y Density Factor (YDF), and Y Cluster Factor (YCF), to set the number and spacing of the targets. The density factor divides the wing into that number of sections, with a target on the wing edge and targets between sections. Thus, with an XDF of 3 and a YDF of 4, you will get a total of 20 targets on the wing. In the x-direction, we space the targets equally using the interval

\[ m = \frac{X_{\text{max}} - X_{\text{min}}}{XDF} \]  

(3.13)

So, an XDF of 3 divides the wing, in the X direction, into 3 sections of the same size, meaning that at each Y interval there will be 4 targets, 2 on the edges and 2 in between.

In the y-direction, a grid of equally spaced targets on the wing is not desired because the majority of the deformation will be occurring near the wing tip. The desired grid is one more densely populated near the wing tip. Thus, the YCF variable is introduced. YCF determines by what order the spacing between Y intervals decreases. For example, a YCF of 2 indicates that the spacing between each interval will decrease parabolically.

Figure 3.6 shows the setup to determine the Y interval. We first define a curve given by \( Y^{YCF} \), where the endpoint is the wingspan, \( L \), which gives a function value of \( Y^{YCF}_{\text{max}} \). This ensures that the Y intervals end on the wing tip. To determine the Y spacing, the interval size is first determined by

\[ n = \frac{(L)^{YCF}}{YDF} \]  

(3.14)
Figure 3.6  Set-Up of Y Interval Determination

Then, at each interval $N = n, 2n, 3n, ..., (YDF - 1)n, (YDF)n$, the $Y$ value is determined by

$$Y = N^{\frac{1}{vCF}}$$

(3.15)

3.4.2 Camera Views. After all the coordinates of the test article have been calculated, the program will then show the perspective of each of eight cameras, and how the test article will look to that camera. This gives the user a nice sense of how much the test article has been deformed.

The program can show anywhere from one to eight camera views. Cameras are placed at 45 degree intervals, all in a plane that is approximately in the center (midway from tail to nose) of the fuselage. The camera set up is seen in Figure 3.7, which is looking down the wind tunnel at the test article head on. Sample camera views in pixel coordinates are shown in Figures 3.8 and 3.9. This sample has the test article at $\alpha = 0$, $\beta = 0$, $\phi = 0$, a bending coefficient of .7, and a twisting coefficient of .1. This makes for a fairly deformed wing, as cameras 3 and 7 show. In
an undeformed case, cameras 3 and 7 would only show a straight line because they are stationed directly off the wingtips.
Figure 3.8  Sample View of Cameras 1-4

Figure 3.9  Sample View of Cameras 5-8
IV. Results

An analysis is performed to determine how the error in position and attitude varies as the number of targets, YCF, and number and location of cameras are changed. The aim is to optimize these parameters so that we may better evaluate the performance of the new bending model versus the old rigid model. Many runs of each program were accomplished to make these charts, and the raw data for each run can be found in Appendix C. In all test cases, the test article was set at the following position and attitude: $\Delta x = 5 \text{ m}$, $\Delta y = 0 \text{ m}$, $\Delta z = -20 \text{ m}$, $\alpha = 15 \text{ deg}$, $\beta = 10 \text{ deg}$, and $\phi = 5 \text{ deg}$.

![Figure 4.1](image.png)

**Figure 4.1** Relative Error Versus Number of Data Points, Severely Deformed

Figures 4.1 and 4.2 show the results of the target number study, computed using 4 cameras. As seen in the graphs, after a certain number of data points the relative error of position and attitude due to number of targets is fairly constant. This study was performed on both a severely deformed wing (BC=0.7 TC=0.1) and
a moderately deformed one (BC=.4 TC=.01) to ensure consistency. Fifty targets was deemed to be sufficiently into this regime. Bear in mind that 50 is the number of data points, not necessarily the number of targets on the test article. Thus, in a 4-camera configuration, the actual number of targets is about 13.

Figure 4.3 shows the results of the Y cluster factor study. The Y density factor was bumped up to 8 to give more divisions in the Y axis. This was done to capture the spectrum from evenly spaced to tightly packed towards the wing tip. The graph shows a pretty even trend that error gets worse as the points are packed tighter and tighter towards the wing tip. It also shows a drop in error around YCF=1.25, which is not quite evenly spaced, but still provides good coverage of the whole wing. Figure 4.4 shows the difference in the target layouts of YCF=1 and YCF=1.25.

Figures 4.5 and 4.6 show the results of the camera study, one with moderate bending and one with more severe bending. This graph uses the same camera set up as in Figure 3.7. In the graph, 4 denotes cameras 1-4, and 8 denotes all 8 cameras.
A surprising result is that more cameras does not necessarily seem to be better. In fact, 2 cameras offset by 90 degrees performs as well as, if not better than, 4 or 8 cameras.

We now have optimum camera and target conditions, and can evaluate the two position and attitude models; the old rigid model, and the new model which includes bending and twisting. Figure 4.7 shows the results of the bending study. As seen in the graph, and as is expected, as the bending coefficient gets more and more severe, the new bending model outperforms the old rigid model by greater margins. The same can be said for the performance in the presence of twist, shown in Figure 4.8.

One last area to evaluate is how each model performs in the presence of noise. Neither method is going to have perfect measurements, and thus noise will affect each. Figure 4.9 shows the effects of increasing noise on each model. At low levels of noise, the margin between the rigid model and the bend/twist model remains fairly constant. In extremely noisy conditions, behavior begins to diminish.
Figure 4.4  Comparison of YCF=1 to YCF=1.25

Figure 4.5  Relative error versus number of cameras, moderate bending
Figure 4.6 Relative Error Versus Number of Cameras, Severe bending
Figure 4.7 Relative Error Versus Bending Coefficient for Bending and Rigid Models
Figure 4.8 Relative Error Versus Twisting Coefficient for Bending and Rigid Models
Figure 4.9  Relative Error Versus Noise Level for Bending and Rigid Models
V. Conclusions

The main objective of this thesis was to improve AEDC's current method of position and attitude determination to account for deformation of the test article. The results showed that by adding in bending and twisting coefficients, dramatic increases in accuracy of position and attitude determination could be achieved for simulated data with a simple deformation model. The next step to continue the work of this thesis would be to incorporate more complex deformation models, possibly using a finite element analysis. Also, the improved deformation model should be compared against the original using actual test data from real wind tunnel models.

This thesis was also to determine the optimal number of targets and cameras to achieve the greatest accuracy, while staying in reasonable numbers. It was found that at least 50 targets are required to achieve optimal accuracy, while any more than that did not add a whole lot of benefit. A YCF of 1.25 was found to provide the best accuracy. This was more clustered than a straight linear distribution, but not quite as dense at the wing tip as a parabolic distribution. It was expected from previous data runs that 4 cameras would provide the optimal solution, but when actually graphed out, 2 cameras spaced at 90 degrees provided slightly better results.
Appendix A. MATLAB Code

```matlab
% Thesis: F-22 Wing Target Assignment  
% Author: ILt Sean A. Krolikowski  
% Date: 31 August 2000

clear all

% Establish wing and fuselage boundaries from specs
Xr=[-4.925 4.925 -1.185 -2.852 -4.925]; Yr=[0 0 6.78 6.78 0];
Yf=[2 -2 2 2 0]; Xf=[0 0 15 19 15 0]; Yf=Yf-2; Xf=Xf-4.925;

figure(1),clf plot(Yr,Xr,'*-'),axis square,hold on
plot(Yf,Xf,'*-')

% Set the Density Factors: XDF and YDF
% This will specify how dense the grid points are in the X and Y directions
XDF=5; YDF=7;

% Set the Y Cluster Factor, YCF
% This will specify how clustered the grid points are towards the wing tip
% NOTE: If for some reason the wing is reconfigured to allow a negative y value, you should not enter an odd number for the YCF
YCF=1.25;

% Compute Grid Points
Ymin=0; Ymax=6.78; w=Ymin; t=1; i=0; n=((Ymax-Ymin)^YCF)/YDF; N=n;
while w <= Ymax
    Xmin=.30575252*w-4.925;
    Xmax=-.9*w+4.925;
    Xmid=(Xmax+Xmin)/2;
    q=Xmin;
    index=1;
    m=(Xmax-Xmin)/XDF;
    for index=1:(XDF+1)
        if X(t,i)q; Xbar(t,i)=Xmid;
        XM(t,i)=Xmax;
        Y(t,i)=w;
        t=t+1;
    
A-1
\[
q = q + m; \\
\text{index} = \text{index} + 1; \\
\text{end}
\]

\[
w = N^{\frac{1}{2/YF}}; \\
N = N + n; \\
i = i + 1; \\
\text{end}
\]

\% Establish Fuselage Targets
\[
YF = [-2 0 2 -1 1 -2 0 2 -1 1 -2 0 2 -1 1 -2 0 2 -1 1 -2 0 2 -1 1 -2 0 2 -1 1 -2 0 2 -1 1 -2 0 2 -1 1 -2]; \\
XF = [0 0 0 1.5 1.5 3 3 3 4.5 4.5 6 6 6 7.5 7.5 9 9 9 10.5 10.5 12 12 12 13.5 13.5 15 15 15 17 17 19]; \\
YF = YF - 2; \\
XF = XF - 4.925; \\
\text{Flength} = \text{length}(XF);
\]

\% Draw Grid
\[
\text{plot}(Y, X, '*r'), \text{plot}(YF, XF, '*r'), \text{plot}(Y, Xbar, 'g'), \text{hold off}
\]

\% Set Bending and Twisting Coefficients
\[
BC = 1; \text{TC} = .5;
\]

\% Determine Undeformed Path Lengths
\[
\text{length} = \text{length}(X); \\
1 = 1; \text{for } l = 1: \text{length}
\]
\[
L(l, 1) = \sqrt{(X(l, 1) - 2 + Y(l, 1) - 2)^2}; \\
1 = 1 + 1;
\]
\% Calculate Twist Angle
\[
T\alpha = \text{atan}(Xbar(\text{length}, 1) / Y(\text{length}, 1));
\]

\% Solve for corrected Y coords, given path length
\[
a = (2 * \text{BC} / Y_{\text{max}}) \cdot 2;
\]

\[
\text{for } j = 1: \text{length}
\]
\[
\text{tlen} = Y(j, 1); \text{told} = 0; \text{tnew} = 7; \text{if } \text{BC} = 0
\]
\[
Y_{\text{new}}(j, 1) = Y(j, 1);
Z(j, 1) = 0;
\]
\% else
\[
\text{while } \text{abs}(\text{tnew} - \text{told}) > 1e-12
\]
\[
\text{told} = \text{tnew};
\text{tnew} = \text{told} - (1/2 * \text{told} * (a * \text{told}^3) * (1/2)) / (a * \text{told}^2 - 2 + a * \text{told} + a (3/2) * \text{told}^2 / a * a (2 * \text{told} - 2 + a (1/2) * \text{told}^2) * (1/2) * a (1/2) / a (1/2) / a (1/2) / a (1/2) / a (1/2) / a (1/2));
\]
\% end

A-2
Ynew(j,l)=tnew;
XT(j,l)=X(j,l)*cos(TA)-Y(j,l)*sin(TA);
XTM(j,l)=XM(j,l)*cos(TA)-Y(j,l)*sin(TA);
YT(j,l)=X(j,l)*sin(TA)+Ynew(j,l)*cos(TA);
Z(j,l)=-BC*(tnew/Ymax)^2+TC*(YT(j,l)/Ymax)*(XT(j,l)/XTM(j,l));
end end

% Set Up for 3D Grid
figure(2), clf for j=0:i-1
    plot3(Ynew(j*XDF+j+1:(j+1)*XDF+j+1,l),X(j*XDF+j+1:(j+1)*XDF+j+1,l),Z(j*XDF+j+1:(j+1)*XDF+j+1,l),'--',hold on
end

s=1; for k=1:XDF+1
    for l=0:i-1
        YY(s,l)=Ynew(k+l*(XDF+1));
        XX(s,l)=X(k+l*(XDF+1));
        ZZ(s,l)=Z(k+l*(XDF+1));
        s=s+1;
    end
end

for j=0:XDF
    plot3(YY(j*i+1:(j+1)*i+1,l),XX(j*i+1:(j+1)*i+1,l),ZZ(j*i+1:(j+1)*i+1,l),'--')
end
hold off

% Convert target coords from wing frame to model frame
DX=4.925; DY=2; for j=1:length
    Xi(j,1)=X(j,1)+DX;
    Yi(j,1)=Ynew(j,1)+DY;
    Yunbent(j,1)=Y(j,1)+DY;
    Zi(j,1)=Z(j,1);
    Zunbent(j,1)=0;
end
XF=XF+DX; YF=YF+DY;

% Add fuselage points to data set
% for j=1:length
%    Xi(length+j,1)=XF(1,j);
%    Yi(length+j,1)=YF(1,j);
%    Yunbent(length+j,1)=YF(1,j);
%    Zi(length+j,1)=0;
%    Zunbent(length+j,1)=0;
% end
% length=max(size(Xi));
%Print Model Coords to file for FORTRAN
data = [transpose(Xi); transpose(Yi); transpose(Zi)]; data2 = [transpose(Xi); transpose(Yunbent); transpose(Zunbent)]; fid = fopen('data.in', 'w');
fprintf(fid, '7.5f
', length);
fprintf(fid, '7.5f
', BC);
fprintf(fid, '7.5f
', TC);
fprintf(fid, '7.5f
', DX);
fprintf(fid, '7.5f
', Ymax);
fprintf(fid, '7.4f
    7.4f
', data);
fprintf(fid, '7.4f
    7.4f
', data2);
fclose(fid);

%Set Model Orientation, alpha is pitch, beta is yaw, and phi is roll
alpha = 0; beta = 0; phi = 0;

%Convert angles to radians and evaluate sin and cos
alpha = alpha*(pi/180); beta = beta*(pi/180); phi = phi*(pi/180);
cal = cos(alpha); sal = sin(alpha); cab = cos(beta); sab = sin(beta);
cap = cos(phi); sap = sin(phi);

%Set displacement of model frame origin from TRS
delxk = 5; delyk = 0; delzk = -20;

%Convert target coords from model frame to TRS
for j = 1:length
    Xistar(j, 1) = delxk + Xi(j, 1)*cab + Yi(j, 1)*(-sab*cp + sal*cb*sp) + Zi(j, 1)*(-sal*cp + cab*sp);
    Yistar(j, 1) = delyk + Xi(j, 1)*-cab*sb + Yi(j, 1)*(cb*cp - sal*sb*sp) + Zi(j, 1)*(-cb*sp + sal*sb*cp);
    Zistar(j, 1) = delzk + Xi(j, 1)*-sa + Yi(j, 1)*cab*sp + Zi(j, 1)*cab;
end

%Set camera parameters:
%uc and vc are the location of the camera focus in the camera frame, should be the same for each camera
%f is the focal length, also should be the same
%Assume the camera uses a resolution of 1024x1024, with the origin at the bottom right corner
uc = 512; vc = 512; f = 1000;

%Define position and attitude of Camera 1
xcl = 14; ycl = 0; zcl = -40; phicl = 0; kappacl = 0; omegac1 = 0;
phicl = phicl*(pi/180); kappacl = kappacl*(pi/180);
omegac1 = omegac1*(pi/180); cli = cos(phicl); spi = sin(phicl);
\( c_{kl} = \cos(kappa_{cl}); \quad s_{kl} = \sin(kappa_{cl}); \quad c_{ol} = \cos(omega_{cl}); \quad s_{ol} = \sin(omega_{cl}); \)

% Find target coords in camera frame
for \( j = 1:length \)
\[
\begin{align*}
U_{cil}(j,l) &= (X_{istar}(j,l) - x_{cl}) \cdot c_{pl} \cdot c_{kl} + (Y_{istar}(j,l) - y_{cl}) \cdot (s_{kl} \cdot c_{ol} + s_{pl} \cdot c_{kl} \cdot s_{ol}) + \\
&\quad (Z_{istar}(j,l) - z_{cl}) \cdot (s_{kl} \cdot s_{ol} - s_{pl} \cdot c_{kl} \cdot c_{ol}); \\
V_{cil}(j,l) &= (X_{istar}(j,l) - x_{cl}) \cdot -c_{pl} \cdot s_{kl} + (Y_{istar}(j,l) - y_{cl}) \cdot (c_{kl} \cdot c_{ol} - s_{pl} \cdot s_{kl} \cdot s_{ol}) + \\
&\quad (Z_{istar}(j,l) - z_{cl}) \cdot (c_{kl} \cdot s_{ol} + s_{pl} \cdot s_{kl} \cdot c_{ol}); \\
W_{cil}(j,l) &= (X_{istar}(j,l) - x_{cl}) \cdot s_{pl} + (Y_{istar}(j,l) - y_{cl}) \cdot -c_{pl} \cdot s_{ol} + (Z_{istar}(j,l) - z_{cl}) \cdot c_{pl} \cdot c_{ol};
\end{align*}
\]
end
for \( j = 1:length \)
\[
\begin{align*}
ucil(j,l) &= uc - f \cdot (U_{cil}(j,l)/W_{cil}(j,l)); \\
vcil(j,l) &= vc - f \cdot (V_{cil}(j,l)/W_{cil}(j,l));
\end{align*}
\]
end

% Define position and attitude of Camera 2
\[
x_{c2} = 14; \quad y_{c2} = 20; \quad z_{c2} = -40; \quad \phi_{c2} = 0; \quad kappa_{c2} = 0; \quad omega_{c2} = 45; \\
\phi_{c2} = \phi_{c2} \cdot (\pi/180); \quad kappa_{c2} = kappa_{c2} \cdot (\pi/180); \\
omega_{c2} = omega_{c2} \cdot (\pi/180); \quad cp2 = \cos(\phi_{c2}); \quad sp2 = \sin(\phi_{c2}); \\
ck2 = \cos(kappa_{c2}); \quad sk2 = \sin(kappa_{c2}); \quad co2 = \cos(omega_{c2}); \quad so2 = \sin(omega_{c2});
\]

% Find target coords in camera frame
for \( j = 1:length \)
\[
\begin{align*}
U_{cil}(j,l) &= (X_{istar}(j,l) - x_{c2}) \cdot cp2 \cdot ck2 + (Y_{istar}(j,l) - y_{c2}) \cdot (sk2 \cdot co2 + sp2 \cdot ck2 \cdot so2) + \\
&\quad (Z_{istar}(j,l) - z_{c2}) \cdot (sk2 \cdot so2 - sp2 \cdot ck2 \cdot co2); \\
V_{cil}(j,l) &= (X_{istar}(j,l) - x_{c2}) \cdot -cp2 \cdot sk2 + (Y_{istar}(j,l) - y_{c2}) \cdot (ck2 \cdot co2 - sp2 \cdot sk2 \cdot so2) + \\
&\quad (Z_{istar}(j,l) - z_{c2}) \cdot (ck2 \cdot so2 + sp2 \cdot sk2 \cdot co2); \\
W_{cil}(j,l) &= (X_{istar}(j,l) - x_{c2}) \cdot sp2 + (Y_{istar}(j,l) - y_{c2}) \cdot -cp2 \cdot so2 + (Z_{istar}(j,l) - z_{c2}) \cdot cp2 \cdot co2;
\end{align*}
\]
end
for \( j = 1:length \)
\[
\begin{align*}
ucil(j,l) &= uc - f \cdot (U_{cil}(j,l)/W_{cil}(j,l)); \\
vcil(j,l) &= vc - f \cdot (V_{cil}(j,l)/W_{cil}(j,l));
\end{align*}
\]
end

% Define position and attitude of Camera 3
\[
x_{c3} = 14; \quad y_{c3} = 20; \quad z_{c3} = -20; \quad \phi_{c3} = 0; \quad kappa_{c3} = 0; \quad omega_{c3} = 90; \\
\phi_{c3} = \phi_{c3} \cdot (\pi/180); \quad kappa_{c3} = kappa_{c3} \cdot (\pi/180); \\
omega_{c3} = omega_{c3} \cdot (\pi/180); \quad cp3 = \cos(\phi_{c3}); \quad sp3 = \sin(\phi_{c3}); \\
ck3 = \cos(kappa_{c3}); \quad sk3 = \sin(kappa_{c3}); \quad co3 = \cos(omega_{c3});
\]
so3=sin(omegac3);

%Find target coords in camera frame
for j=1:length
    Uci3(j,1)=(Xistar(j,1)-xc3)*cp3*ck3+(Yistar(j,1)-yc3)*(sk3*co3+sp3*ck3*so3)+...  
                                 (Zistar(j,1)-zc3)*(sk3*so3-sp3*ck3*co3);
    Vci3(j,1)=(Xistar(j,1)-xc3)*-cp3*sk3+(Yistar(j,1)-yc3)*(ck3*co3-sp3*sk3*so3)+...  
                                 (Zistar(j,1)-zc3)*(ck3*so3+sp3*sk3*co3);
    Wci3(j,1)=(Xistar(j,1)-xc3)*sp3+(Yistar(j,1)-yc3)*-cp3*so3+(Zistar(j,1)-zc3)*cp3*co3;
end

for j=1:length
    uci3(j,1)=uc-f*(Uci3(j,1)/Wci3(j,1));
    vci3(j,1)=vc-f*(Vci3(j,1)/Wci3(j,1));
end

%Define postion and attitude of Camera 4
xc4=14; yc4=20; zc4=0; phic4=0; kappac4=0; omegac4=135;
phic4=phic4*(pi/180); kappac4=kappac4*(pi/180);
omegac4=omegac4*(pi/180); cp4=cos(phic4); sp4=sin(phic4);
ck4=cos(kappac4); sk4=sin(kappac4); co4=cos(omegac4);
so4=sin(omegac4);

%Find target coords in camera frame
for j=1:length
    Uci4(j,1)=(Xistar(j,1)-xc4)*cp4*ck4+(Yistar(j,1)-yc4)*(sk4*co4+sp4*ck4*so4)+...
                                 (Zistar(j,1)-zc4)*(sk4*so4-sp4*ck4*co4);
    Vci4(j,1)=(Xistar(j,1)-xc4)*-cp4*sk4+(Yistar(j,1)-yc4)*(ck4*co4-sp4*sk4*so4)+...
                                 (Zistar(j,1)-zc4)*(ck4*so4+sp4*sk4*co4);
    Wci4(j,1)=(Xistar(j,1)-xc4)*sp4+(Yistar(j,1)-yc4)*-cp4*so4+(Zistar(j,1)-zc4)*cp4*co4;
end

for j=1:length
    uci4(j,1)=uc-f*(Uci4(j,1)/Wci4(j,1));
    vci4(j,1)=vc-f*(Vci4(j,1)/Wci4(j,1));
end

%Define postion and attitude of Camera 5
xc5=14; yc5=0; zc5=0; phic5=0; kappac5=0; omegac5=180;
phic5=phic5*(pi/180); kappac5=kappac5*(pi/180);
omegac5=omegac5*(pi/180); cp5=cos(phic5); sp5=sin(phic5);
ck5=cos(kappac5); sk5=sin(kappac5); co5=cos(omegac5);
so5=sin(omegac5);
%Find target coords in camera frame
for j=1:length
    Uci5(j,1)=(Xistar(j,1)-xc5)*cp5*ck5+(Yistar(j,1)-yc5)*(sk5*co5+sp5*ck5*so5)+...
               (Zistar(j,1)-zc5)*(sk5*so5-sp5*ck5*co5);
    Vci5(j,1)=(Xistar(j,1)-xc5)*-cp5*sk5+(Yistar(j,1)-yc5)*(ck5*co5-sp5*sk5*so5)+...
               (Zistar(j,1)-zc5)*(ck5*so5+sp5*sk5*co5);
    Wci5(j,1)=(Xistar(j,1)-xc5)*sp5+(Yistar(j,1)-yc5)*-cp5*so5+(Zistar(j,1)-zc5)*cp5*co5;
end

for j=1:length
    uci5(j,1)=uc-f*(Uci5(j,1)/Wci5(j,1));
    vci5(j,1)=vc-f*(Vci5(j,1)/Wci5(j,1));
end

%Define position and attitude of Camera 6
xc6=14; yc6=-20; zc6=0; phic6=0; kappac6=0; omegac6=225;
phic6=phic6*(pi/180); kappac6=kappac6*(pi/180);
omegac6=omegac6*(pi/180); cp6=cos(phic6); sp6=sin(phic6);
ck6=cos(kappac6); sk6=sin(kappac6); co6=cos(omegac6);
so6=sin(omegac6);

%Find target coords in camera frame
for j=1:length
    Uci6(j,1)=(Xistar(j,1)-xc6)*cp6*ck6+(Yistar(j,1)-yc6)*(sk6*co6+sp6*ck6*so6)+...
               (Zistar(j,1)-zc6)*(sk6*so6-sp6*ck6*co6);
    Vci6(j,1)=(Xistar(j,1)-xc6)*-cp6*sk6+(Yistar(j,1)-yc6)*(ck6*co6-sp6*sk6*so6)+...
               (Zistar(j,1)-zc6)*(ck6*so6+sp6*sk6*co6);
    Wci6(j,1)=(Xistar(j,1)-xc6)*sp6+(Yistar(j,1)-yc6)*-cp6*so6+(Zistar(j,1)-zc6)*cp6*co6;
end

for j=1:length
    uci6(j,1)=uc-f*(Uci6(j,1)/Wci6(j,1));
    vci6(j,1)=vc-f*(Vci6(j,1)/Wci6(j,1));
end

%Define position and attitude of Camera 7
xc7=14; yc7=-20; zc7=-20; phic7=0; kappac7=0; omegac7=270;
phic7=phic7*(pi/180); kappac7=kappac7*(pi/180);
omegac7=omegac7*(pi/180); cp7=cos(phic7); sp7=sin(phic7);
ck7=cos(kappac7); sk7=sin(kappac7); co7=cos(omegac7);
so7=sin(omegac7);
%Find target coords in camera frame
for j=1:length
    Uci7(j,1)=((Xistar(j,1)-xc7)*cp7*ck7+(Yistar(j,1)-yc7)*(sk7*co7+sp7*ck7*so7)+... 
               (Zistar(j,1)-zc7)*(sk7*so7-sp7*ck7*co7);
    Vci7(j,1)=((Xistar(j,1)-xc7)*-cp7*sk7+(Yistar(j,1)-yc7)*(ck7*co7-sp7*sk7*so7)+... 
               (Zistar(j,1)-zc7)*(ck7*so7+sp7*sk7*co7);
    Wci7(j,1)=((Xistar(j,1)-xc7)*sp7+(Yistar(j,1)-yc7)*-cp7*so7+(Zistar(j,1)-zc7)*cp7*co7;
end

for j=1:length
    uci7(j,1)=uc-f*(Uci7(j,1)/Wci7(j,1));
    vci7(j,1)=vc-f*(Vci7(j,1)/Wci7(j,1));
end

%Define position and attitude of Camera 8
xc8=14; yc8=-20; zc8=-80; phic8=0; kappac8=0; omegac8=315;
phic8=phic8*(pi/180); kappac8=kappac8*(pi/180);
omegac8=omegac8*(pi/180); cp8=cos(phic8); sp8=sin(phic8);
ck8=cos(kappac8); sk8=sin(kappac8); co8=cos(omegac8);
so8=sin(omegac8);

%Find target coords in camera frame
for j=1:length
    Uci8(j,1)=((Xistar(j,1)-xc8)*cp8*ck8+(Yistar(j,1)-yc8)*(sk8*co8+sp8*ck8*so8)+... 
               (Zistar(j,1)-zc8)*(sk8*so8-sp8*ck8*co8);
    Vci8(j,1)=((Xistar(j,1)-xc8)*-cp8*sk8+(Yistar(j,1)-yc8)*(ck8*co8-sp8*sk8*so8)+... 
               (Zistar(j,1)-zc8)*(ck8*so8+sp8*sk8*co8);
    Wci8(j,1)=((Xistar(j,1)-xc8)*sp8+(Yistar(j,1)-yc8)*-cp8*so8+(Zistar(j,1)-zc8)*cp8*co8;
end

for j=1:length
    uci8(j,1)=uc-f*(Uci8(j,1)/Wci8(j,1));
    vci8(j,1)=vc-f*(Vci8(j,1)/Wci8(j,1));
end

%Plot Camera 1-4 perspective
figure(3),clf subplot(2,2,1), plot(vci1,uci1,'*') grid on
subplot(2,2,2), plot(vci2,uci2,'*') grid on
subplot(2,2,3), plot(vci3,uci3,'*') grid on
subplot(2,2,4), plot(vci4,uci4,'*') grid on

%Plot Camera 5-8 perspective
figure(4), clf subplot(2,2,1), plot(vci5,uci5,'*') grid on
title('Camera 5') subplot(2,2,2), plot(vci6,uci6,'*') grid on
title('Camera 6') subplot(2,2,3), plot(vci7,uci7,'*') grid on
title('Camera 7') subplot(2,2,4), plot(vci8,uci8,'*') grid on
title('Camera 8')
Appendix B. Fortran Code

program fit
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

***** COPYRIGHT NOTICE ***** c c
Subroutines in this file are based, in part, on the following c
subroutines from Numerical Recipes in Fortran, Second Edition, c
Cambridge University Press: c c   - CHOLDC: Cholesky
decomposition of pos. def. sym. matrix c   - CHOLSL: Solution of
associated linear system c   - MRQMIN: Levenberg-Marquardt
nonlinear parameter optimization c   - MRQCOF: Calculate
matrices and chi-square for MRQMIN c   - MRQSRT: Rearrangement
of covariance matrix for MRQCOF c   - GASDEV: Random number
generator for Gaussian noise c   - RAN1: Random number
generator for uniform noise c c The following licence
information and warranty disclaimer apply c c
to the use of these routines: c c
 C C   Numerical Recipes Fortran Diskette Documentation v2.01 C C
License Information and WARRANTY DISCLAIMER C C

What does your license cover? C C As the owner of this free
Numerical Recipes diskette in IBM/PC C format, you are licensed
to install the programs on this diskette C onto a single
IBM/PC-compatible computer. You are not licensed C to move the
files to any other type of computer, nor to use them C on more
than a single IBM/PC-compatible computer for each diskette C
purchased. By installing or using the programs, you acknowledge C
acceptance of the following DISCLAIMER OF WARRANTY: C C
DISCLAIMER OF WARRANTY C C THE PROGRAMS ACCESSED BY THIS
ROUTINE (AND ON THE ORIGINAL C DISKETTE) ARE PROVIDED "AS
IS" WITHOUT WARRANTY OF ANY KIND. C WE MAKE NO WARRANTIES,
EXPRESS OR IMPLIED, THAT THEY ARE FREE OF C ERROR, OR ARE
CONSISTENT WITH ANY PARTICULAR STANDARD OF C
MERCHANTABILITY, OR THAT THEY WILL MEET YOUR REQUIREMENTS FOR ANY
C PARTICULAR APPLICATION. THEY SHOULD NOT BE RELIED ON FOR
SOLVING C A PROBLEM WHOSE INCORRECT SOLUTION COULD RESULT IN
INJURY TO A C PERSON OR LOSS OF PROPERTY. IF YOU DO USE THEM
IN SUCH A MANNER, C IT IS AT YOUR OWN RISK. THE AUTHORS AND
PUBLISHER DISCLAIM ALL C LIABILITY FOR DIRECT, INDIRECT,
OR CONSEQUENTIAL DAMAGES C RESULTING FROM YOUR USE OF THE
PROGRAMS. C C Can you redistribute Numerical Recipes in your
programs? C C If you want to include Numerical Recipes routines
in programs that C are further distributed (either commercially
or non-commercially) you can obtain permission to do so from Numerical Recipes Software. If the routines are bound into your program executable and are not separately visible to or usable by your users, there is generally no charge, provided that (i) advance permission is obtained, and (ii) a copyright notice like that on this diskette is embedded in your program executable. Contact Numerical Recipes Software at P.O. Box 243, Cambridge, MA 02238 (USA) [fax 617-863-1739] for details. In distributing a program containing Numerical Recipes routines, you acknowledge acceptance of the above DISCLAIMER OF WARRANTY, and of the fact that no business relationship is created between your program's users and Numerical Recipes Software, the authors of the Numerical Recipes books, or Cambridge University Press. If you want to distribute software that has Numerical Recipes in the form of source code or individually callable object modules, then you must contact Numerical Recipes Software for further information. A fee (per Recipe) is charged, and we normally limit the total number of Recipes distributed to 20. Licenses for other types of computers are available from Numerical Recipes Software (press F9 for address). For educational and noncommercial users, we offer two 'streamlined' procedures:

Educational License for Single-Screen Workstation

If you are affiliated with an educational or not-for-profit research institution that is connected to the Internet, you may license the programs for use on a single workstation (one screen) as follows: Mail your name, address, and email address; your workstation's hostname, internet address, brand and model number; and a $50 one-time license fee (must accompany order) to Numerical Recipes Software, at the address below (press F9). Be sure to specify the language you want (FORTRAN or C). You will receive, by return mail or email, instructions for downloading the programs electronically. Upon payment of the fee you may also, if you desire, upload the contents of this diskette (after unpacking the files with the NRCOPY program) to your workstation.

Right-to-Copy License for Courses

Instructors at accredited educational institutions who have adopted Numerical Recipes for a course, and who already own a diskette, may license the programs for use in that course as follows: Mail your name, title, and address; the course name, number, dates, and
estimated enrollment; and advance payment of $5 per
(estimated) student to Numerical Recipes Software, at the C
address below (F9). You will receive by return mail a license C
authorizing you to make copies of your diskette for use by your C
students, and/or to transfer the programs to a machine C
accessible to your students (but only for the duration of the C
course). C C

How to contact Numerical Recipes Software
C C
Our address is: Numerical Recipes Software C
P.O. Box 243 C Cambridge, MA 02238 (USA) C
Our fax number is: 617-863-1739. C C Sorry, we do not take
telephone calls other than fax. C

! Variables associated with target points (corners of cube):
! parameter (nmax=8)
real x(5000),y(5000),z(5000) ! Model coordinates
real xU(5000),yU(5000),zU(5000) ! Unbent Model coordinates
real xt(5000),yt(5000),zt(5000) ! Tunnel coordinates
real u(5000),v(5000) ! Image coordinates
real u2(5000),v2(5000) ! Image coordinates
real u3(5000),v3(5000) ! Image coordinates
real u4(5000),v4(5000) ! Image coordinates
real Xmax(5000) ! Distance from midline to leading edge

! Wing frame displacement from model frame
real DDX,DDY

! Chord length of wing
real Ymax, xmid, div, TA, w, tip, xf, xtip, ytip

! Variables associated with camera (star superscript not shown):
real phi c, phi c2, kappa c, kappa c2, omega c, omega c2, bc, tc,
* phi c3, kappa c3, omega c3, phi c4, kappa c4, omega c4
integer nmax
common /camera/ uc, vc, fc, xc, yc, xc2, yc2, xC2, yC2, xc3, yC3, zc3,
* xC4, yC4, zC4, uxc, uyc, uzc, vx c, vy c, wxc, wyc, wzc,
* uxc2, uyc2, uzc2, vx c2, vy c2, wxc2, wyc2, wzc2,
* uxc3, uyc3, uzc3, vx c3, vy c3, wxc3, wyc3, wzc3,
* uxc4, uyc4, uzc4, vx c4, vy c4, wxc4, wyc4, wzc4

! Variables associated with position and attitude:
parameter (npar=8)
real posatt(npar), fitrms(npar)
c.... Degrees/radians conversion:
    raddeg = atan(1.)/45.

c.... Assume target points on corners of cube: !  data x
/12.,12.,0.,0.,12.,12.,0.,0./ !  data y
/0.,12.,12.,0.,0.,12.,12.,0./ !  data z
/0.,0.,0.,0.,12.,12.,12.,12./

100 FORMAT(I5)
200 FORMAT(3(F16.12))
300 FORMAT(F16.12,3X,F16.12,3X,F16.12)
400 FORMAT(F5.5)

open(2,FILE='data.in',STATUS='OLD')
read(2,100) nmax
read(2,400) be
cread(2,400) tc
cread(2,400) DDX
cread(2,400) DDY
cread(2,400) Ymax
DO I=1,nmax
    read(2,200) x(I),y(I),z(I)
ENDDO

DO I=1,nmax
    read(2,200) xU(I),yU(I),zU(I)
ENDDO

write(3,100) nmax
write(3,* ) bc
cwrite(3,* ) tc
cwrite(3,* ) DDX
cwrite(3,* ) DDY
cwrite(3,* ) Ymax
DO I=1,nmax
write(3,300) x(I),y(I),z(I)
enddo
cDO I=1,nmax
cwrite(3,300) xU(I),yU(I),zU(I)
cenddo

accurate = 0.

B-4
div = 0.
do i=1,nmax
   w = yU(i)-DDY
tip = yU(nmax)-DDY
   if (w.EQ.tip) then
      xmid = xmid + xU(i)-DDX
div = div + 1
   endif
endo
xmid = xmid/div
TA = atan(xmid/Ymax)
c.... Input coords of leading edge endpoints, in wing frame
 xf=4.925
 yf=0.
 xtip=-1.177
 ytip=6.78
c.... Calculate Xmax for each Y along the wing
do i=1,nmax
   Xmax(i)=((xtip-xf)/(ytip-yf))*(yU(i)-DDY)+xf
endo
do i=1,nmax
   Xmax(i)=Xmax(i)*cos(TA)-(yU(i)-DDY)*sin(TA)
endo
c.... Specify the camera parameters:
   uc = 512.  ! pixels
   vc = 512.  ! pixels
   fc = 1000. ! pixels
   xc = 14.
   yc = 0.
   zc = -40.
   phic = 0.  ! degrees
   kappac = 0.  ! degrees
   omegac = 0.  ! degrees
   xc2 = 14.
   yc2 = 20.
   zc2 = -40.
   phic2 = 0.  ! degrees
   kappac2 = 0.  ! degrees
   omegac2 = -45.  ! degrees
xc3 = 14.
yc3 = 20.
zc3 = -20.
phic3 = 0. ! degrees
kappac3 = 0. ! degrees
omegac3 = -90. ! degrees

xc4 = 14.
yc4 = 20.
zc4 = 0.
phic4 = 0. ! degrees
kappac4 = 0. ! degrees
omegac4 = -135. ! degrees

c.... Convert angles to radians:
    phic = phic*raddeg
    kappac = kappac*raddeg
    omegac = omegac*raddeg
    phic2 = phic2*raddeg
    kappac2 = kappac2*raddeg
    omegac2 = omegac2*raddeg
    phic3 = phic3*raddeg
    kappac3 = kappac3*raddeg
    omegac3 = omegac3*raddeg
    phic4 = phic4*raddeg
    kappac4 = kappac4*raddeg
    omegac4 = omegac4*raddeg

c.... Calculate the camera orientation matrices:
    call setmatrix (phic,kappac,omegac,
    * uxc,uyc,uzc,vxc,vyc,vzc,wxc,wyc,wzc)
    call setmatrix (phic2,kappac2,omegac2,
    * uxc2,uyc2,uzc2,vxc2,vyc2,vzc2,wxc2,wyc2,wzc2)
    call setmatrix (phic3,kappac3,omegac3,
    * uxc3,uyc3,uzc3,vxc3,vyc3,vzc3,wxc3,wyc3,wzc3)
    call setmatrix (phic4,kappac4,omegac4,
    * uxc4,uyc4,uzc4,vxc4,vyc4,vzc4,wxc4,wyc4,wzc4)

c.... Specify position and attitude of test article:
    dxk = 5
dyk = 0
dzk = -20
alphak = 15.*raddeg
betak = 10.*raddeg
phik = 5.*raddeg

c.... Calculate tunnel coordinates of targets:
call setmatrix (alphak,betak,phik,
* r11,r12,r13,r21,r22,r23,r31,r32,r33)
do i = 1, nmax
x(i) = dxk + r11*x(i) + r12*y(i) + r13*z(i)
y(i) = dyk + r21*x(i) + r22*y(i) + r23*z(i)
zt(i) = dzk + r31*x(i) + r32*y(i) + r33*z(i)
endo

c.... Specify noise level on image coordinates:
spread = .00001 ! pixel
idum = -911 ! initial seed for random number generator

c.... Calculate corresponding image coordinates:
open(l,FILE='fit.out',STATUS='UNKNOWN')
write(l,*)
write(l,*) 'Camera 1:
write(l,'(a)') '  i    u(i)    v(i)'
do i = 1, nmax
uki = uxc*(xt(i)-xc) + uyc*(yt(i)-yc) + uzc*(zt(i)-zc)
vki = vxc*(xt(i)-xc) + vyc*(yt(i)-yc) + vzc*(zt(i)-zc)
wki = wxc*(xt(i)-xc) + wyc*(yt(i)-yc) + wzc*(zt(i)-zc)
write(*,*) uki,vki,wki
u(i) = uc - fc*uki/wki + spread*gasdev(idum)
v(i) = vc - fc*vki/wki + spread*gasdev(idum)
write(l,'((i4,2f9.3))') i, u(i), v(i)
endo

write(l,*) 'Camera 2:'
write(l,'(a)') '  i    u(i)    v(i)'
do i = 1, nmax
uki2 = uxc2*(xt(i)-xc2) + uyc2*(yt(i)-yc2) + uzc2*(zt(i)-zc2)
vki2 = vxc2*(xt(i)-xc2) + vyc2*(yt(i)-yc2) + vzc2*(zt(i)-zc2)
wki2 = wxc2*(xt(i)-xc2) + wyc2*(yt(i)-yc2) + wzc2*(zt(i)-zc2)
u2(i) = uc - fc*uki2/wki2 + spread*gasdev(idum)
v2(i) = vc - fc*vki2/wki2 + spread*gasdev(idum)
write(l,'((i4,2f9.3))') i, u2(i), v2(i)
enddo
write(1,*)
write(1,"('Camera 3:')")
write(1,"((a))' i u(i) v(i)'
doi = 1, nmax
uki3 = uxc3*(xt(i)-xc3) + uyc3*(yt(i)-yc3) + uzc3*(zt(i)-zc3)
vki3 = vxc3*(xt(i)-xc3) + vyc3*(yt(i)-yc3) + vzc3*(zt(i)-zc3)
wiki3 = wxc3*(xt(i)-xc3) + wyc3*(yt(i)-yc3) + wzc3*(zt(i)-zc3)
u3(i) = uc - fc*uki3/wki3 + spread*gasdev(idum)
v3(i) = vc - fc*vki3/wiki3 + spread*gasdev(idum)
write(1,"((i4,2f9.3))') i, u3(i), v3(i)
enddo
write(1,*)
write(1,"('Camera 4:')")
write(1,"((a))' i u(i) v(i)'
doi = 1, nmax
uki4 = uxc4*(xt(i)-xc4) + uyc4*(yt(i)-yc4) + uzc4*(zt(i)-zc4)
vki4 = vxc4*(xt(i)-xc4) + vyc4*(yt(i)-yc4) + vzc4*(zt(i)-zc4)
wiki4 = wxc4*(xt(i)-xc4) + wyc4*(yt(i)-yc4) + wzc4*(zt(i)-zc4)
u4(i) = uc - fc*uki4/wiki4 + spread*gasdev(idum)
v4(i) = vc - fc*vki4/wiki4 + spread*gasdev(idum)
write(1,"((i4,2f9.3))') i, u4(i), v4(i)
enddo
c.... Initialize the least-squares fit:
do ipar = 1, npar
   posatt(ipar) = 0. ! initial guess for posatt values
endo
c.... Estimate the noise level (in this case known exactly):
sigma = spread
c.... Perform the fit:
call pafit (nmax,x,y,z,xU,yU,u,v,u2,v2,u3,v3,u4,v4,
   *sigma,posatt,fitrms,rmepix,TA,Xmax,Ymax,DDX,DDY)
c.... Report results:
write(1,*)
write(1,"('... Final results of LM fit: '")
write(1,"((ix,a))")
   * 'FIT   EXACT   ERROR   PRECISION'
write(1,"((ix,a,4f10.5))' 'DeltaX_k: ')
   * posatt(i), dXk, posatt(i)-dXk, fitrms(i)
write(l,"(ix,a,4fl0.5)") 'DeltaY_k:',
+ posatt(2), dyk, posatt(2)-dyk, fitrms(2)
write(l,"(ix,a,4fl0.5)") 'DeltaZ_k:',
+ posatt(3), dzk, posatt(3)-dkz, fitrms(3)
write(l,"(ix,a,4fl0.5)") 'Alpha_k:',
+ posatt(4)/raddeg, alphak/raddeg, (posatt(4)-alphak)/raddeg,
+ fitrms(4)/raddeg
write(l,"(ix,a,4fl0.5)") 'Beta_k:',
+ posatt(5)/raddeg, betak/raddeg, (posatt(5)-betak)/raddeg,
+ fitrms(5)/raddeg
write(l,"(ix,a,4fl0.5)") 'Phi_k:',
+ posatt(6)/raddeg, phik/raddeg, (posatt(6)-phik)/raddeg,
+ fitrms(6)/raddeg
write(l,"(ix,a,4fl0.5)") 'BC:',
+ posatt(7), bc, posatt(7)-bc, fitrms(7)
write(l,"(ix,a,4fl0.5)") 'TC:',
+ posatt(8), tc, posatt(8)-tc, fitrms(8)

c.... Compare calculated and actual noise:
write(1,*) '... Compare calculated and actual noise amplitude: '
write(1, "(ix,a)") 'CALCULATED ACTUAL'
write(1, "(ix,2f10.5)") rmspix, spread

open(7,FILE='ERROR.out',STATUS='UNKNOWN')
write(7,*) posatt(1)-dxk
write(7,*) posatt(2)-dyk
write(7,*) posatt(3)-dzk
write(7,*) (posatt(4)-alphak)/raddeg
write(7,*) (posatt(5)-betak)/raddeg
write(7,*) (posatt(6)-phik)/raddeg
write(7,*) posatt(7)-bc
write(7,*) posatt(8)-tc
end

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

! From Ruyten, Appendix B, Eq.(B-1)

subroutine setmatrix (alpha,beta,phi,
+ r11,r12,r13,r21,r22,r23,r31,r32,r33)

c = cos(alpha)
sa = sin(alpha)
cb = cos(beta)
sb = sin(beta)
cp = cos(phi)
sp = sin(phi)

sasp = sa*sp
sacp = sa*cp

r11 = ca*cb
r12 = sb*cp + sasp*cb
r13 = -sb*sp + sacp*cb
r21 = -ca*sb
r22 = cb*cp - sasp*sb
r23 = -cb*sp - sacp*sb
r31 = -sa
r32 = ca*sp
r33 = ca*cp

cccc... Convergence criteria for LM optimization:
dcmin = 0.01
nconv = 4

cccc Initialize parameters:
do ipar = 1, npar
   coef(ipar) = posatt(ipar)
   ifit(ipar) = 0
endo

subroutine pafit (nmax,x,y,z,xU,yU,zU,u,v,u2,v2,u3,v3,u4,v4,
*sigma,posatt,fitrms,rmspix,TAXmax,Ymax,DDX,DDY)

real x(*),y(*),z(*),u(*),v(*),u2(*),v2(*),u3(*),v3(*),u4(*),v4(*),Xmax(*)

parameter (npar=8)
real coef(npar),covar(npar,npar),alfa(npar,npar)
integer ifit(npar)
... Initialize Levenberg-Marquardt:
    alambda = -1.
call mrqmini (nmax,x,y,z,xU,yU,zU,u,v,u2,v2,u3,v3,u4,v4,
        *sigma,coef,ifit,covar,alfa,npar,chi2q,alambda,nfree,TA,Xmax,
        *Ya,DDX,DDY)

... Iterate Levenberg-Marquardt to convergence:
    ktot = 1
    knew = 0
write(1,*)
write(1,’... Progress of LM fit:’)
write(1,’ITER   CHISQ    RMSPIX    LAMBDA’)
do while (knew.lt.nconv)
    rmspix = sigma * sqrt(chisq/float(nmax))
write(1,’(i5,1p,9e12.3)’) ktot, chisq, rmspix, alambda
    ktot = ktot + 1
    ochisq = chisq
call mrqmini (nmax,x,y,z,xU,yU,zU,u,v,u2,v2,u3,v3,u4,v4,
        *sigma,coef,ifit,covar,alfa,npar,chi2q,alambda,nfree,TA,Xmax,
        *Ya,DDX,DDY)
    if (chisq.gt.ochisq) then
        knew = 0
    elseif (abs(ochisq-chisq).lt.dcmin) then
        knew = knew + 1
    endif
enddo

... Transfer parameters back to posatt:
    do ipar = 1, npar
        posatt(ipar) = coef(ipar)
    enddo

... Calculate precision:
    alambda = 0.
call mrqmini (nmax,x,y,z,xU,yU,zU,u,v,u2,v2,u3,v3,u4,v4,
        *sigma,coef,ifit,covar,alfa,npar,chi2q,alambda,nfree,TA,Xmax,
        *Ya,DDX,DDY)
    sigsca = sqrt(chisq/float(nfree))
    do ipar = 1, npar
        fitrms(ipar) = sigsca*sqrt(covar(ipar,ipar))
    enddo
Substantial modifications have been made to the routines MRQMIN, MRQCOF, and MRQSRT:
1. Replaced x(*), y(*), sig(*), ndata in calling sequence with alternate pass-through argument lists
2. Using scalar sigma instead of vector.
3. Replaced ma and nca in argument lists with npar.
4. Returning number of degrees of freedom: nfree.
5. Removed external reference to funcs.
6. Replaced gaussj with choldc, chols.
7. Replaced covsrt with makecovar: See makecovar.
8. Changed from y=y(xi,a) to u=u(i,a) + v=v(i,a).
9. Calling functions mrqfun: initial call to set parameters.
10. Changed ia=0 to signify fitting parameter.

Adaptation of Numerical Recipes "covsrt". Based on Cholesky decomposition of covar: adapted calculation of $L^{-1}$ from Num. Rec. p91. Results checked against gaussj OK.

```fortran
subroutine makecovar (covar, alpha, pivot, ifix, maxpar, npar, mfit)

real covar(maxpar,maxpar),alpha(maxpar,maxpar),pivot(*)
integer ifix(*)

! Determine $L^{-1}$ according to Num. Rec. p91:
do i = 1, mfit
  covar(i,i) = 1./pivot(i)
do j = i+1, mfit
  sum = 0.
do k = i, j-1
    sum = sum - covar(j,k)*covar(k,i)
  enddo
  covar(j,i) = sum/pivot(j)
enddo

do i = 1, mfit
  do j = i, mfit
    sum = 0.
do k = max(i,j), mfit
      sum = sum + covar(k,i)*covar(k,j)
    enddo
    alpha(i,j) = sum
  alpha(j,i) = sum
enddo
```

B-12
enddo
enddo

! Set remainder of matrix to zero:
do i = mfit+1, npar
  do j = 1, i
    alpha(i,j) = 0.
    alpha(j,i) = 0.
  enddo
enddo

! Copy from alpha to covar:
do i = 1, npar
  do j = 1, npar
    covar(i,j) = alpha(i,j)
  enddo
enddo

! Redistribute:
k = mfit
  do j = npar, 1, -1
    if (ifix(j).eq.0) then
      do i = 1, npar
        swap = covar(i,k)
        covar(i,k) = covar(i,j)
        covar(i,j) = swap
      enddo
      do i = 1, npar
        swap = covar(k,i)
        covar(k,i) = covar(j,i)
        covar(j,i) = swap
      enddo
      k = k - 1
    endif
  enddo
end

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
SUBROUTINE mrqminI (nmax,x,y,xU,yU,u,v,u2,v2,u3,v3,u4,v4,
  *  sigma,a,ifix,covar,alpha,npar,chisq,alambda,nfree,TX,Xmax,
  *  Ymax,DX,DY)

B-13
real x(*), y(*), z(*), xU(*), yU(*), u(*), v(*), u2(*), v2(*),
  u3(*), v3(*), u4(*), v4(*), Xmax(*)

INTEGER ifix(npar)
REAL a(npar), alpha(npar, npar), covar(npar, npar)

PARAMETER (MMAX=8)
REAL atry(MMAX), beta(MMAX), da(MMAX), pivot(MMAX)
SAVE ochisq, atry, beta, da, mfit

if (npar.gt.MMAX) stop '*** mrqmin1: npar.gt.MMAX ***'

if (alambda.lt.0.) then
   mfit = 0
   do j = 1, npar
      if (ifix(j).eq.0) mfit = mfit + 1
   enddo
   alambda = 0.001
endif

j = 0
do 1 = 1, npar
   if (ifix(1).eq.0) then
      j = j + 1
      k = 0
      do m = 1, npar
         if (ifix(m).eq.0) then
            k = k + 1
            covar(j,k) = alpha(j,k)
         endif
      enddo
      covar(j,j) = alpha(j,j)*(1.+alambda)
      da(j) = beta(j)
   endif
1   j = j + 1
   k = 0
! Prepare for linear system solution or matrix inverse:
open(4,FILE='Amatrix.out',STATUS='UNKNOWN')
write(4,*) covar
write(4,*)
call choldc (covar,mfit,npar,pivot,ierr)

! Compute covariance matrix (was: "covert"):
if (alambda.eq.0.) then
  call makecovar (covar,alpha,pivot,ifax,npar,npar,mfit)
  return
endif

! Proceed with solution of linear system:
call cholsl (covar,mfit,npar,pivot,da,da)
j = 0
do 1 = 1, npar
  if (ifax(l).eq.0) then
    j = j + 1
    atry(l) = a(l) + da(j)
  endif
enddo

call mrqcofl (nmax,x,y,z,xU,yU,u,v,u2,v2,u3,v3,u4,v4, sigma, atrv, ifix, covar, da, npar, chisq, nf, TA, Xmax, Ymax, DDX, DDY)

if (chisq.lt.ochisq) then
  alambda = 0.1*alambda
  ochisq = chisq
  j = 0
  do 1 = 1, npar
    if (ifix(l).eq.0) then
      j = j + 1
      k = 0
      do m = 1, npar
        if (ifix(m).eq.0) then
          k = k + 1
          alpha(j,k) = covar(j,k)
        endif
      enddo
    beta(j) = da(j)
    a(l) = atry(l)
endif
enddo
else
  alambda = 10.0 * alambda
  chisq = ochisq
endif

END

C (C) Copr. 1986-92 Numerical Recipes Software --259u. c
Modified by W. M. Ruyten

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

SUBROUTINE mrqcof1 (nmax,x,y,xU,yU,u,v,u2,v2,u3,v3,u4,v4,
  *         sigma,a,ifix,alpha,beta,npar,chisq,nfree,TA,Xmax,DX,DDY)
real x(*),y(*),z(*),xU(*),yU(*),zU(*),u(*),v(*),u2(*),v2(*),
  *         u3(*),v3(*),u4(*),v4(*),Xmax(*)

INTEGER ifix(npar)
REAL a(npar),alpha(npar,npar),beta(npar)

PARAMETER (MMAX=8)
REAL duda(MMAX),dvda(MMAX),duda2(MMAX),dvda2(MMAX)
REAL duda3(MMAX),dvda3(MMAX),duda4(MMAX),dvda4(MMAX)

if (npar.gt.MMAX) stop '*** mrqcof1: npar.gt.MMAX ***'

c.... Initialize arrays:
mfit = 0
  do j = 1, npar
    if (ifix(j).eq.0) mfit = mfit + 1
  enddo
  do j = 1, mfit
    do k = 1, j
      alpha(j,k) = 0.
    enddo
    beta(j) = 0.
  enddo

c.... Initialize rotation matrix and derivative:
i = 0

B-16
call mrqfunl (i,x,y,z,xU,yU,zU,a,upred,upred2,vpred,vpred2, 
*upred3,upred4,vpred3,vpred4,duda,duda2,duda3,duda4,dvda, 
*dvda2,dvda3,dvda4,TA,Xmax,Ymax,DDX,DDY)

c.... Build alpha and beta by summing over all points:
chisq = 0.
do i = 1, nmax

call mrqfunl (i,x,y,z,xU,yU,zU,a,upred,upred2,vpred,vpred2, 
*upred3,upred4,vpred3,vpred4,duda,duda2,duda3,duda4,dvda, 
*dvda2,dvda3,dvda4,TA,Xmax,Ymax,DDX,DDY)

du = u(i) - upred
dv = v(i) - vpred
du2 = u2(i) - upred2
dv2 = v2(i) - vpred2
du3 = u3(i) - upred3
dv3 = v3(i) - vpred3
du4 = u4(i) - upred4
dv4 = v4(i) - vpred4

ej = 0
do l = 1, npar
  if (ifix(l).eq.0) then
    j = j + 1
    wtu = duda(l)
    wtv = dvda(l)
    wtu2 = duda2(l)
    wtv2 = dvda2(l)
    wtu3 = duda3(l)
    wtv3 = dvda3(l)
    wtu4 = duda4(l)
    wtv4 = dvda4(l)
  end
  k = 0
  do m = 1, 1
    if (ifix(m).eq.0) then
      k = k + 1
      alpha(j,k) = alpha(j,k) 
      + wtu*duda(m) + wtv*dvda(m)
      + wtu2*duda2(m) + wtv2*dvda2(m)
      + wtu3*duda3(m) + wtv3*dvda3(m)
      + wtu4*duda4(m) + wtv4*dvda4(m)
  enddo
endo
endif
endo
dotoj = beta(j) + du*wtu + dv*wtv
* + du2*wtu2 + dv2*wtv2
* + du3*wtu3 + dv3*wtv3
* + du4*wtu4 + dv4*wtv4
endo
doto
chisq = chisq + du*du + dv*dv + du2*du2 + dv2*dv2
* + du3*du3 + dv3*dv3 + du4*du4 + dv4*dv4
endo
c.... Perform scaling by sigma:
sig2i = 1./(sigma*sigma)
do j = 1, mfit
do k = 1, j
alpha(j,k) = alpha(j,k)*sig2i
doto
beta(j) = beta(j)*sig2i
doto
chisq = chisq*sig2i
c.... Fill out matrix:
do j = 2, mfit
do k = 1, j-1
alpha(k,j) = alpha(j,k)
doto
endo
doto
c.... Determine number of degrees of freedom:
nfree = 2*nmax - mfit
END

C (C) Copr. 1986-92 Numerical Recipes Software "-259u.. c
Modified by W. M. Ruyten

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

SUBROUTINE mrqfunl (i,x,y,z,xU,yU,zU,coef,upred,upred2,vpred,
*vpred2,upred3,upred4,vpred4,duda,duda2,duda3,duda4,dvda,
*dvda2,dvda3,dvda4,TA,Xmax,Ymax,DDX,DDY)
real x(*), y(*), z(*), xU(*), yU(*), zU(*), Xmax(*), maxX

REAL coef(*), duda(*), dvda(*), duda2(*), dvda2(*)
REAL duda3(*), dvda3(*), duda4(*), dvda4(*)

save dx, dy, dz, r11, r12, r13, r21, r22, r23, r31, r32, r33, sb, cb

! Camera common block copied from top of program
common /camera/ uc, vc, xc, yc, xc2, yc2, xc3, yc3, xc4,
* xc4, yc4, uxc, uyc, uxc2, uyc2, uxc3, uyc3, uxc4, uyc4,
* uxc2, uyc2, vxc2, vyc2, wxc, wyc, wxc2, wyc2,
* uxc3, uyc3, vxc3, vyc3, wxc3, wyc3,
* uxc4, uyc4, vxc4, vyc4, wxc4, wyc4

c.... Calculate trig factors only on initial call:
if (i.gt.0) goto 10

dx = coef(1)
dy = coef(2)
dz = coef(3)

alpha = coef(4)
beta = coef(5)
phi = coef(6)

BC = coef(7)
TC = coef(8)

ta = cos(alpha)
sa = sin(alpha)
ct = cos(beta)
sb = sin(beta)
cp = cos(phi)
sp = sin(phi)

sasp = sa*sp
sacp = sa*cp

r11 = ca*cb
r12 = sb*cp + sasp*cb
r13 = -sb*sp + sacp*cb
r21 = -ca*sb
r22 = cb*cp - sasp*sb
Perform actual calculation: 10

\[ \begin{align*}
\mathbf{r}_{23} &= -cb\mathbf{sp} - sacp\mathbf{sb} \\
\mathbf{r}_{31} &= -sa \\
\mathbf{r}_{32} &= ca\mathbf{sp} \\
\mathbf{r}_{33} &= ca\mathbf{sp} \\
\text{return} \\
\end{align*} \]

c.

\[ \begin{align*}
\mathbf{xi} &= \mathbf{xU}(i) \\
\mathbf{yi} &= \mathbf{yU}(i) \\
\mathbf{zi} &= \mathbf{zU}(i) \\
\end{align*} \]

\[ \begin{align*}
\mathbf{xtw} &= (\mathbf{xU}(i) - \mathbf{DDX}) \cos(\mathbf{TA}) - (\mathbf{yU}(i) - \mathbf{DDY}) \sin(\mathbf{TA}) \\
\mathbf{ytw} &= (\mathbf{xU}(i) - \mathbf{DDX}) \sin(\mathbf{TA}) + (\mathbf{yU}(i) - \mathbf{DDY}) \cos(\mathbf{TA}) \\
\mathbf{maxX} &= \mathbf{Xmax}(i) \\
\end{align*} \]

! Tunnel coordinates of targets:

\[ \begin{align*}
\mathbf{xt} &= \mathbf{dx} + r_{11} \mathbf{xi} + r_{12} \mathbf{yi} + r_{13} \mathbf{(yI-DDY)/(Ymax)} \\
&+ \mathbf{(yI-DDY)/(Ymax)} \mathbf{maxX) \\
\mathbf{yt} &= \mathbf{dy} + r_{21} \mathbf{xi} + r_{22} \mathbf{yi} + r_{23} \mathbf{(yI-DDY)/(Ymax)} \\
&+ \mathbf{(yI-DDY)/(Ymax)} \mathbf{maxX) \\
\mathbf{zt} &= \mathbf{dz} + r_{31} \mathbf{xi} + r_{32} \mathbf{yi} + r_{33} \mathbf{(yI-DDY)/(Ymax)} \\
&+ \mathbf{(yI-DDY)/(Ymax)} \mathbf{maxX) \\
\end{align*} \]

! Implied image coordinates:

\[ \begin{align*}
\mathbf{uki} &= \mathbf{uxc}(\mathbf{xt} - \mathbf{xc}) + \mathbf{uyc}(\mathbf{yt} - \mathbf{yc}) + \mathbf{uzc}(\mathbf{zt} - \mathbf{zc}) \\
\text{open}(5,\text{FILE}='\text{duda.out}',\text{STATUS}='\text{UNKNOWN}') \\
\text{write}(5,*)i \\
\text{write}(5,*)\mathbf{maxX} \\
\mathbf{vki} &= \mathbf{vxc}(\mathbf{xt} - \mathbf{xc}) + \mathbf{vyc}(\mathbf{yt} - \mathbf{yc}) + \mathbf{vzc}(\mathbf{zt} - \mathbf{zc}) \\
\mathbf{wki} &= \mathbf{wxc}(\mathbf{xt} - \mathbf{xc}) + \mathbf{wyc}(\mathbf{yt} - \mathbf{yc}) + \mathbf{wzc}(\mathbf{zt} - \mathbf{zc}) \\
\mathbf{upred} &= \mathbf{uc} - \mathbf{fc}\mathbf{uki}/\mathbf{wki} \\
\mathbf{vpred} &= \mathbf{vc} - \mathbf{fc}\mathbf{vki}/\mathbf{wki} \\
\end{align*} \]

\[ \begin{align*}
\mathbf{uki2} &= \mathbf{uxc2}(\mathbf{xt} - \mathbf{xc2}) + \mathbf{uyc2}(\mathbf{yt} - \mathbf{yc2}) + \mathbf{uzc2}(\mathbf{zt} - \mathbf{zc2}) \\
\mathbf{vki2} &= \mathbf{vxc2}(\mathbf{xt} - \mathbf{xc2}) + \mathbf{vyc2}(\mathbf{yt} - \mathbf{yc2}) + \mathbf{vzc2}(\mathbf{zt} - \mathbf{zc2}) \\
\mathbf{wki2} &= \mathbf{wxc2}(\mathbf{xt} - \mathbf{xc2}) + \mathbf{wyc2}(\mathbf{yt} - \mathbf{yc2}) + \mathbf{wzc2}(\mathbf{zt} - \mathbf{zc2}) \\
\mathbf{upred2} &= \mathbf{uc} - \mathbf{fc}\mathbf{uki2}/\mathbf{wki2} \\
\mathbf{vpred2} &= \mathbf{vc} - \mathbf{fc}\mathbf{vki2}/\mathbf{wki2} \\
\end{align*} \]

\[ \begin{align*}
\mathbf{uki3} &= \mathbf{uxc3}(\mathbf{xt} - \mathbf{xc3}) + \mathbf{uyc3}(\mathbf{yt} - \mathbf{yc3}) + \mathbf{uzc3}(\mathbf{zt} - \mathbf{zc3}) \\
\mathbf{vki3} &= \mathbf{vxc3}(\mathbf{xt} - \mathbf{xc3}) + \mathbf{vyc3}(\mathbf{yt} - \mathbf{yc3}) + \mathbf{vzc3}(\mathbf{zt} - \mathbf{zc3}) \\
\mathbf{wki3} &= \mathbf{wxc3}(\mathbf{xt} - \mathbf{xc3}) + \mathbf{wyc3}(\mathbf{yt} - \mathbf{yc3}) + \mathbf{wzc3}(\mathbf{zt} - \mathbf{zc3}) \\
\mathbf{upred3} &= \mathbf{uc} - \mathbf{fc}\mathbf{uki3}/\mathbf{wki3} \\
\end{align*} \]
\[ v_{\text{pred}3} = v_c - f_c v_{k3}/w_{k3} \]

\[ u_{k4} = u_{x4}(x_t-x_{c4}) + u_{y4}(y_t-y_{c4}) + u_{z4}(z_t-z_{c4}) \]

\[ v_{k4} = v_{x4}(x_t-x_{c4}) + v_{y4}(y_t-y_{c4}) + v_{z4}(z_t-z_{c4}) \]

\[ w_{k4} = w_{x4}(x_t-x_{c4}) + w_{y4}(y_t-y_{c4}) + w_{z4}(z_t-z_{c4}) \]

\[ u_{\text{pred}4} = u_c - f_c u_{k4}/w_{k4} \]

\[ v_{\text{pred}4} = v_c - f_c v_{k4}/w_{k4} \]

c.... Calculate partial derivatives w.r.t. fit parameters:

! Use trick for derivatives w.r.t. alpha_k:
! \((dR/dalpha_k)^T(R^{-1}) = (0,0,cb, 0,0,-sb, -cb, sb, 0)\)

! Start with tunnel coordinates:
\[ d_{xt1} = 1. \]
\[ d_{xt2} = 0. \]
\[ d_{xt3} = 0. \]
\[ d_{xt4} = cb(z_t-dz) \]
\[ d_{xt5} = (yt-dy) \]
\[ d_{xt6} = r_{13}yi - r_{12}((-BC)*(yi-DDY/Ymax)*((yi-DDY/Ymax)^+ \]
\[ * \]
\[ TC*(ytw/Ymax)*(xtw/maxX) \]
\[ d_{xt7} = -r_{13}((-yi-DDY/Ymax)*((yi-DDY/Ymax) \]
\[ d_{xt8} = r_{13}((ytw/Ymax)*(xtw/maxX) \]

\[ d_{yt1} = 0. \]
\[ d_{yt2} = 1. \]
\[ d_{yt3} = 0. \]
\[ d_{yt4} = -sb(z_t-dz) \]
\[ d_{yt5} = -(xt-dx) \]
\[ d_{yt6} = r_{23}yi - r_{22}((-BC)*(yi-DDY/Ymax)*((yi-DDY/Ymax) \]
\[ * \]
\[ TC*(ytw/Ymax)*(xtw/maxX) \]
\[ d_{yt7} = -r_{23}((-yi-DDY/Ymax)*((yi-DDY/Ymax) \]
\[ d_{yt8} = r_{23}((ytw/Ymax)*(xtw/maxX) \]

\[ d_{zt1} = 0. \]
\[ d_{zt2} = 0. \]
\[ d_{zt3} = 1. \]
\[ d_{zt4} = -cb(x_t-dx) + sb(yt-dy) \]
\[ d_{zt5} = 0. \]
\[ d_{zt6} = r_{33}yi - r_{32}((-BC)*(yi-DDY/Ymax)*((yi-DDY/Ymax) \]
\[ * \]
\[ TC*(ytw/Ymax)*(xtw/maxX) \]
\[ d_{zt7} = -r_{33}((-yi-DDY/Ymax)*((yi-DDY/Ymax) \]
\[ d_{zt8} = r_{33}((ytw/Ymax)*(xtw/maxX) \]

B-21
! Continue by chain rule with U,V,W product terms:

\[
duki_{1a} = u_x c_x * d_x t_1 + u_y c_y * d_y t_1 + u_z c_z * d_z t_1 \\
duki_{2a} = u_x c_x * d_x t_2 + u_y c_y * d_y t_2 + u_z c_z * d_z t_2 \\
duki_{3a} = u_x c_x * d_x t_3 + u_y c_y * d_y t_3 + u_z c_z * d_z t_3 \\
duki_{4a} = u_x c_x * d_x t_4 + u_y c_y * d_y t_4 + u_z c_z * d_z t_4 \\
duki_{5a} = u_x c_x * d_x t_5 + u_y c_y * d_y t_5 + u_z c_z * d_z t_5 \\
duki_{6a} = u_x c_x * d_x t_6 + u_y c_y * d_y t_6 + u_z c_z * d_z t_6 \\
duki_{7a} = u_x c_x * d_x t_7 + u_y c_y * d_y t_7 + u_z c_z * d_z t_7 \\
duki_{8a} = u_x c_x * d_x t_8 + u_y c_y * d_y t_8 + u_z c_z * d_z t_8 \\
\]

\[
dvk_{1a} = v_x c_x * d_x t_1 + v_y c_y * d_y t_1 + v_z c_z * d_z t_1 \\
dvk_{2a} = v_x c_x * d_x t_2 + v_y c_y * d_y t_2 + v_z c_z * d_z t_2 \\
dvk_{3a} = v_x c_x * d_x t_3 + v_y c_y * d_y t_3 + v_z c_z * d_z t_3 \\
dvk_{4a} = v_x c_x * d_x t_4 + v_y c_y * d_y t_4 + v_z c_z * d_z t_4 \\
dvk_{5a} = v_x c_x * d_x t_5 + v_y c_y * d_y t_5 + v_z c_z * d_z t_5 \\
dvk_{6a} = v_x c_x * d_x t_6 + v_y c_y * d_y t_6 + v_z c_z * d_z t_6 \\
dvk_{7a} = v_x c_x * d_x t_7 + v_y c_y * d_y t_7 + v_z c_z * d_z t_7 \\
dvk_{8a} = v_x c_x * d_x t_8 + v_y c_y * d_y t_8 + v_z c_z * d_z t_8 \\
\]

\[
duki_{1b} = u_x c_x * d_x t_1 + u_y c_y * d_y t_1 + u_z c_z * d_z t_1 \\
duki_{2b} = u_x c_x * d_x t_2 + u_y c_y * d_y t_2 + u_z c_z * d_z t_2 \\
duki_{3b} = u_x c_x * d_x t_3 + u_y c_y * d_y t_3 + u_z c_z * d_z t_3 \\
duki_{4b} = u_x c_x * d_x t_4 + u_y c_y * d_y t_4 + u_z c_z * d_z t_4 \\
duki_{5b} = u_x c_x * d_x t_5 + u_y c_y * d_y t_5 + u_z c_z * d_z t_5 \\
duki_{6b} = u_x c_x * d_x t_6 + u_y c_y * d_y t_6 + u_z c_z * d_z t_6 \\
duki_{7b} = u_x c_x * d_x t_7 + u_y c_y * d_y t_7 + u_z c_z * d_z t_7 \\
duki_{8b} = u_x c_x * d_x t_8 + u_y c_y * d_y t_8 + u_z c_z * d_z t_8 \\
\]

\[
dvk_{1b} = v_x c_x * d_x t_1 + v_y c_y * d_y t_1 + v_z c_z * d_z t_1 \\
dvk_{2b} = v_x c_x * d_x t_2 + v_y c_y * d_y t_2 + v_z c_z * d_z t_2 \\
dvk_{3b} = v_x c_x * d_x t_3 + v_y c_y * d_y t_3 + v_z c_z * d_z t_3 \\
dvk_{4b} = v_x c_x * d_x t_4 + v_y c_y * d_y t_4 + v_z c_z * d_z t_4 \\
dvk_{5b} = v_x c_x * d_x t_5 + v_y c_y * d_y t_5 + v_z c_z * d_z t_5 \\
\]

B-22
dvki6b = vxc2*dxt6 + vyc2*dyt6 + vzc2*dzt6
dvki7b = vxc2*dxt7 + vyc2*dyt7 + vzc2*dzt7
dvki8b = vxc2*dxt8 + vyc2*dyt8 + vzc2*dzt8

dvki1b = vxc2*dxt1 + vyc2*dyt1 + vzc2*dzt1
dvki2b = vxc2*dxt2 + vyc2*dyt2 + vzc2*dzt2
dvki3b = vxc2*dxt3 + vyc2*dyt3 + vzc2*dzt3
dvki4b = vxc2*dxt4 + vyc2*dyt4 + vzc2*dzt4
dvki5b = vxc2*dxt5 + vyc2*dyt5 + vzc2*dzt5
dvki6b = vxc2*dxt6 + vyc2*dyt6 + vzc2*dzt6
dvki7b = vxc2*dxt7 + vyc2*dyt7 + vzc2*dzt7
dvki8b = vxc2*dxt8 + vyc2*dyt8 + vzc2*dzt8

duki1c = uxc3*dxt1 + uyc3*dyt1 + uzc3*dzt1
duki2c = uxc3*dxt2 + uyc3*dyt2 + uzc3*dzt2
duki3c = uxc3*dxt3 + uyc3*dyt3 + uzc3*dzt3
duki4c = uxc3*dxt4 + uyc3*dyt4 + uzc3*dzt4
duki5c = uxc3*dxt5 + uyc3*dyt5 + uzc3*dzt5
duki6c = uxc3*dxt6 + uyc3*dyt6 + uzc3*dzt6
duki7c = uxc3*dxt7 + uyc3*dyt7 + uzc3*dzt7
duki8c = uxc3*dxt8 + uyc3*dyt8 + uzc3*dzt8

dvkic1 = vxc3*dxt1 + vyc3*dyt1 + vzc3*dzt1
dvkic2 = vxc3*dxt2 + vyc3*dyt2 + vzc3*dzt2
dvkic3 = vxc3*dxt3 + vyc3*dyt3 + vzc3*dzt3
dvkic4 = vxc3*dxt4 + vyc3*dyt4 + vzc3*dzt4
dvkic5 = vxc3*dxt5 + vyc3*dyt5 + vzc3*dzt5
dvkic6 = vxc3*dxt6 + vyc3*dyt6 + vzc3*dzt6
dvkic7 = vxc3*dxt7 + vyc3*dyt7 + vzc3*dzt7
dvkic8 = vxc3*dxt8 + vyc3*dyt8 + vzc3*dzt8

duki1d = uxc4*dxt1 + uyc4*dyt1 + uzc4*dzt1
duki2d = uxc4*dxt2 + uyc4*dyt2 + uzc4*dzt2
duki3d = uxc4*dxt3 + uyc4*dyt3 + uzc4*dzt3
duki4d = uxc4\cdot dxt4 + uyc4\cdot dyt4 + uzc4\cdot dzt4

duki5d = uxc4\cdot dxt5 + uyc4\cdot dyt5 + uzc4\cdot dzt5

duki6d = uxc4\cdot dxt6 + uyc4\cdot dyt6 + uzc4\cdot dzt6

duki7d = uxc4\cdot dxt7 + uyc4\cdot dyt7 + uzc4\cdot dzt7

duki8d = uxc4\cdot dxt8 + uyc4\cdot dyt8 + uzc4\cdot dzt8

dvki1d = vxc4\cdot dxt1 + vyc4\cdot dyt1 + vzc4\cdot dzt1

dvki2d = vxc4\cdot dxt2 + vyc4\cdot dyt2 + vzc4\cdot dzt2

dvki3d = vxc4\cdot dxt3 + vyc4\cdot dyt3 + vzc4\cdot dzt3

dvki4d = vxc4\cdot dxt4 + vyc4\cdot dyt4 + vzc4\cdot dzt4

dvki5d = vxc4\cdot dxt5 + vyc4\cdot dyt5 + vzc4\cdot dzt5

dvki6d = vxc4\cdot dxt6 + vyc4\cdot dyt6 + vzc4\cdot dzt6

dvki7d = vxc4\cdot dxt7 + vyc4\cdot dyt7 + vzc4\cdot dzt7

dvki8d = vxc4\cdot dxt8 + vyc4\cdot dyt8 + vzc4\cdot dzt8

dwki1d = wxc4\cdot dxt1 + wyc4\cdot dyt1 + wzc4\cdot dzt1

dwki2d = wxc4\cdot dxt2 + wyc4\cdot dyt2 + wzc4\cdot dzt2

dwki3d = wxc4\cdot dxt3 + wyc4\cdot dyt3 + wzc4\cdot dzt3

dwki4d = wxc4\cdot dxt4 + wyc4\cdot dyt4 + wzc4\cdot dzt4

dwki5d = wxc4\cdot dxt5 + wyc4\cdot dyt5 + wzc4\cdot dzt5

dwki6d = wxc4\cdot dxt6 + wyc4\cdot dyt6 + wzc4\cdot dzt6

dwki7d = wxc4\cdot dxt7 + wyc4\cdot dyt7 + wzc4\cdot dzt7

dwki8d = wxc4\cdot dxt8 + wyc4\cdot dyt8 + wzc4\cdot dzt8

\begin{align*}
&\text{! Finish with image coordinates themselves:} \\
&\text{fac1} = -fc/\text{wiki} \\
&\text{fac2} = fc/\text{wiki}^2 \\
&\text{duda}(1) = \text{fac1}\cdot \text{duki1a} + \text{fac2}\cdot \text{dwki1a} \\
&\text{duda}(2) = \text{fac1}\cdot \text{duki2a} + \text{fac2}\cdot \text{dwki2a} \\
&\text{duda}(3) = \text{fac1}\cdot \text{duki3a} + \text{fac2}\cdot \text{dwki3a} \\
&\text{duda}(4) = \text{fac1}\cdot \text{duki4a} + \text{fac2}\cdot \text{dwki4a} \\
&\text{duda}(5) = \text{fac1}\cdot \text{duki5a} + \text{fac2}\cdot \text{dwki5a} \\
&\text{duda}(6) = \text{fac1}\cdot \text{duki6a} + \text{fac2}\cdot \text{dwki6a} \\
&\text{duda}(7) = \text{fac1}\cdot \text{duki7a} + \text{fac2}\cdot \text{dwki7a} \\
&\text{duda}(8) = \text{fac1}\cdot \text{duki8a} + \text{fac2}\cdot \text{dwki8a} \\
&\text{fac2} = fc/\text{wiki}^2 \\
&\text{dvda}(1) = \text{fac1}\cdot \text{dvki1a} + \text{fac2}\cdot \text{dwki1a} \\
&\text{dvda}(2) = \text{fac1}\cdot \text{dvki2a} + \text{fac2}\cdot \text{dwki2a} \\
&\text{dvda}(3) = \text{fac1}\cdot \text{dvki3a} + \text{fac2}\cdot \text{dwki3a} \\
&\text{dvda}(4) = \text{fac1}\cdot \text{dvki4a} + \text{fac2}\cdot \text{dwki4a} \\
&\text{dvda}(5) = \text{fac1}\cdot \text{dvki5a} + \text{fac2}\cdot \text{dwki5a} \\
&\text{dvda}(6) = \text{fac1}\cdot \text{dvki6a} + \text{fac2}\cdot \text{dwki6a}
\end{align*}
\[ dvda(7) = \text{fac1}\cdot dvki7a + \text{fac2}\cdot dwki7a \]
\[ dvda(8) = \text{fac1}\cdot dvki8a + \text{fac2}\cdot dwki8a \]

\[ \text{fac3} = -\frac{fc}{wki2} \]
\[ \text{fac4} = \frac{fc\cdot uki2}{wki2^2} \]
\[ duda2(1) = \text{fac3}\cdot duki1b + \text{fac4}\cdot dwki1b \]
\[ duda2(2) = \text{fac3}\cdot duki2b + \text{fac4}\cdot dwki2b \]
\[ duda2(3) = \text{fac3}\cdot duki3b + \text{fac4}\cdot dwki3b \]
\[ duda2(4) = \text{fac3}\cdot duki4b + \text{fac4}\cdot dwki4b \]
\[ duda2(5) = \text{fac3}\cdot duki5b + \text{fac4}\cdot dwki5b \]
\[ duda2(6) = \text{fac3}\cdot duki6b + \text{fac4}\cdot dwki6b \]
\[ duda2(7) = \text{fac3}\cdot duki7b + \text{fac4}\cdot dwki7b \]
\[ duda2(8) = \text{fac3}\cdot duki8b + \text{fac4}\cdot dwki8b \]

\[ \text{fac4} = \frac{fc\cdot vki2}{wki2^2} \]
\[ dvda2(1) = \text{fac3}\cdot dvki1b + \text{fac4}\cdot dwki1b \]
\[ dvda2(2) = \text{fac3}\cdot dvki2b + \text{fac4}\cdot dwki2b \]
\[ dvda2(3) = \text{fac3}\cdot dvki3b + \text{fac4}\cdot dwki3b \]
\[ dvda2(4) = \text{fac3}\cdot dvki4b + \text{fac4}\cdot dwki4b \]
\[ dvda2(5) = \text{fac3}\cdot dvki5b + \text{fac4}\cdot dwki5b \]
\[ dvda2(6) = \text{fac3}\cdot dvki6b + \text{fac4}\cdot dwki6b \]
\[ dvda2(7) = \text{fac3}\cdot dvki7b + \text{fac4}\cdot dwki7b \]
\[ dvda2(8) = \text{fac3}\cdot dvki8b + \text{fac4}\cdot dwki8b \]

\[ \text{fac5} = -\frac{fc}{wki3} \]
\[ \text{fac6} = \frac{fc\cdot uki3}{wki3^2} \]
\[ duda3(1) = \text{fac5}\cdot duki1c + \text{fac6}\cdot dwki1c \]
\[ duda3(2) = \text{fac5}\cdot duki2c + \text{fac6}\cdot dwki2c \]
\[ duda3(3) = \text{fac5}\cdot duki3c + \text{fac6}\cdot dwki3c \]
\[ duda3(4) = \text{fac5}\cdot duki4c + \text{fac6}\cdot dwki4c \]
\[ duda3(5) = \text{fac5}\cdot duki5c + \text{fac6}\cdot dwki5c \]
\[ duda3(6) = \text{fac5}\cdot duki6c + \text{fac6}\cdot dwki6c \]
\[ duda3(7) = \text{fac5}\cdot duki7c + \text{fac6}\cdot dwki7c \]
\[ duda3(8) = \text{fac5}\cdot duki8c + \text{fac6}\cdot dwki8c \]

\[ \text{fac6} = \frac{fc\cdot vki3}{wki3^2} \]
\[ dvda3(1) = \text{fac5}\cdot dvki1c + \text{fac6}\cdot dwki1c \]
\[ dvda3(2) = \text{fac5}\cdot dvki2c + \text{fac6}\cdot dwki2c \]
\[ dvda3(3) = \text{fac5}\cdot dvki3c + \text{fac6}\cdot dwki3c \]
\[ dvda3(4) = \text{fac5}\cdot dvki4c + \text{fac6}\cdot dwki4c \]
\[ dvda3(5) = \text{fac5}\cdot dvki5c + \text{fac6}\cdot dwki5c \]
\[ dvda3(6) = \text{fac5}\cdot dvki6c + \text{fac6}\cdot dwki6c \]
\[ dvda3(7) = \text{fac5}\cdot dvki7c + \text{fac6}\cdot dwki7c \]
\[ dvda3(8) = \text{fac5}\times dvki8c + \text{fac6}\times dwki8c \]

\[ \text{fac7} = \frac{-\text{fc}}{wki4} \]
\[ \text{fac8} = \frac{\text{fc}\times uk14}{wki2^{**2}} \]

\[ duda4(1) = \text{fac7}\times duk11d + \text{fac8}\times dwk11d \]
\[ duda4(2) = \text{fac7}\times duk12d + \text{fac8}\times dwk12d \]
\[ duda4(3) = \text{fac7}\times duk13d + \text{fac8}\times dwk13d \]
\[ duda4(4) = \text{fac7}\times duk14d + \text{fac8}\times dwk14d \]
\[ duda4(5) = \text{fac7}\times duk15d + \text{fac8}\times dwk15d \]
\[ duda4(6) = \text{fac7}\times duk16d + \text{fac8}\times dwk16d \]
\[ duda4(7) = \text{fac7}\times duk17d + \text{fac8}\times dwk17d \]
\[ duda4(8) = \text{fac7}\times duk18d + \text{fac8}\times dwk18d \]

\[ \text{fac8} = \frac{\text{fc}\times vki4}{wki4^{**2}} \]

\[ dvda4(l) = \text{fac7}\times dvk1ld + \text{fac8}\times dwk1ld \]
\[ dvda4(2) = \text{fac7}\times dvk12d + \text{fac8}\times dwk12d \]
\[ dvda4(3) = \text{fac7}\times dvk13d + \text{fac8}\times dwk13d \]
\[ dvda4(4) = \text{fac7}\times dvk14d + \text{fac8}\times dwk14d \]
\[ dvda4(5) = \text{fac7}\times dvk15d + \text{fac8}\times dwk15d \]
\[ dvda4(6) = \text{fac7}\times dvk16d + \text{fac8}\times dwk16d \]
\[ dvda4(7) = \text{fac7}\times dvk17d + \text{fac8}\times dwk17d \]
\[ dvda4(8) = \text{fac7}\times dvk18d + \text{fac8}\times dwk18d \]

\[ \text{end} \]

FUNCTION gasdev(idum)
INTEGER idum
REAL gasdev

USES rani
INTEGER iset
REAL fac,gset,rsq,v1,v2,ranl
SAVE iset,gset

DATA iset/0/

if (iset.eq.0) then
  vi=2.\times ranl(idum)-1.
  v2=2.\times ranl(idum)-1.
  rsq=vi**2+v2**2
  if(rsq.ge.1.\ or.rsq.eq.0.)goto 1
  fac=sqrt(-2.*log(rsq)/rsq)
  gset=vi*fac
  gasdev=v2*fac
end
iset=1
else
gasdev=gset
iset=0
endif
return
END

FUNCTION ranl(idum)
INTEGER idum,IA,IM,IQ,IR,NTAB,NDIV
REAL ranl,AM,EPS,RNMX
PARAMETER (IA=16807,IM=2147483647,AM=1./IM,IQ=127773,IR=2836,
*NTAB=32,NDIV=1+(IM-1)/NTAB,EPS=1.2e-7,RNMX=1.-EPS)
INTEGER j,k,iv(NTAB),iy
SAVE iv,iy
DATA iv /NTAB*0/, iy /0/
if (idum.le.0.or.iy.eq.0) then
    idum=max(-idum,1)
do 11 j=NTAB+8,l,-l
        k=idum/IQ
        idum=IA*(idum-k*IQ)-IR*k
        if (idum.lt.0) idum=idum+IM
        if (j.le.NTAB) iv(j)=idum
11     continue
    iy=iv(l)
endif
k=idum/IQ
idum=IA*(idum-k*IQ)-IR*k
if (idum.lt.0) idum=idum+IM
j=1+i/NDIV
iy=iv(j)
iv(j)=idum
ranl=min(AM*iy,RNMX)
return
END

SUBROUTINE choldc(a,n,np,p,ierr)
INTEGER n,np
real a(np,np),p(n)
INTEGER i,j,k
real sum
ierr=0
do 13 i=1,n
   do 12 j=i,n
      sum=a(i,j)
      do 11 k=i-1,1,-1
         sum=sum-a(i,k)*a(j,k)
      11 continue
      if(i.eq.j)then
         if(sum.le.0.)then
            ierr=i
            return
         endif
         p(i)=sqrt(sum)
      else
         a(j,i)=sum/p(i)
      endif
   12 continue 13 continue
return
END

C (C) Copr. 1986-92 Numerical Recipes Software ~-259u..

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

SUBROUTINE cholsl(a,n,np,p,b,x)
INTEGER n,np
real a(np,np),b(n),p(n),x(n)
INTEGER i,k
real sum
do 12 i=1,n
   sum=b(i)
   do 11 k=i-1,1,-1
      sum=sum-a(i,k)*x(k)
   11 continue
   x(i)=sum/p(i)
  12 continue
do 14 i=n,1,-1
   sum=x(i)
   do 13 k=i+1,n
      sum=sum-a(k,i)*x(k)
  13 continue
  14 continue
continue
x(i)=sum/p(i)
continue
return
END
C (C) Copr. 1986-92 Numerical Recipes Software 
~259u..
## Appendix C. Data Runs

### C.1 Runs Varying Number of Data Points

**BC=.7, TC=.1, XDF=2, YDF=1**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.04862</td>
<td>5.00000</td>
<td>0.04862</td>
<td>0.03647</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.01996</td>
<td>0.00000</td>
<td>-0.01996</td>
<td>0.00680</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.80795</td>
<td>-20.00000</td>
<td>0.19205</td>
<td>0.13681</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99999</td>
<td>15.00000</td>
<td>-0.00016</td>
<td>0.04803</td>
</tr>
<tr>
<td>Beta_k</td>
<td>-0.70264</td>
<td>5.00000</td>
<td>-5.70264</td>
<td>4.03329</td>
</tr>
<tr>
<td>Phi_k</td>
<td>0.01686</td>
<td>0.70000</td>
<td>-0.68314</td>
<td>0.47691</td>
</tr>
<tr>
<td>BC</td>
<td>0.10088</td>
<td>0.10000</td>
<td>0.00088</td>
<td>0.00578</td>
</tr>
</tbody>
</table>

**BC=.7, TC=.1, XDF=3, YDF=2**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00349</td>
<td>5.00000</td>
<td>0.00349</td>
<td>0.00876</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02274</td>
<td>0.00000</td>
<td>-0.02274</td>
<td>0.00825</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.98313</td>
<td>-20.00000</td>
<td>0.01687</td>
<td>0.02371</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.98799</td>
<td>15.00000</td>
<td>-0.01121</td>
<td>0.13638</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.97992</td>
<td>10.00000</td>
<td>-0.02008</td>
<td>0.07218</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.52164</td>
<td>5.00000</td>
<td>-0.47836</td>
<td>0.37489</td>
</tr>
<tr>
<td>BC</td>
<td>0.63206</td>
<td>0.70000</td>
<td>-0.06794</td>
<td>0.04131</td>
</tr>
<tr>
<td>TC</td>
<td>0.09930</td>
<td>0.10000</td>
<td>-0.00070</td>
<td>0.00805</td>
</tr>
</tbody>
</table>

**BC=.7, TC=.1, XDF=4, YDF=3**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00374</td>
<td>5.00000</td>
<td>0.00374</td>
<td>0.00579</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02149</td>
<td>0.00000</td>
<td>-0.02149</td>
<td>0.00521</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.98270</td>
<td>-20.00000</td>
<td>0.01730</td>
<td>0.01614</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.98858</td>
<td>15.00000</td>
<td>-0.01042</td>
<td>0.09566</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.97997</td>
<td>10.00000</td>
<td>-0.02003</td>
<td>0.04839</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.55500</td>
<td>5.00000</td>
<td>-0.44500</td>
<td>0.23367</td>
</tr>
<tr>
<td>BC</td>
<td>0.63709</td>
<td>0.70000</td>
<td>-0.06201</td>
<td>0.02502</td>
</tr>
<tr>
<td>TC</td>
<td>0.09932</td>
<td>0.10000</td>
<td>-0.00068</td>
<td>0.00565</td>
</tr>
</tbody>
</table>

**BC=.7, TC=.1, XDF=5, YDF=4**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00419</td>
<td>5.00000</td>
<td>0.00419</td>
<td>0.00446</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02088</td>
<td>0.00000</td>
<td>-0.02088</td>
<td>0.00387</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.98127</td>
<td>-20.00000</td>
<td>0.01873</td>
<td>0.01275</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99018</td>
<td>15.00000</td>
<td>-0.00982</td>
<td>0.07670</td>
</tr>
</tbody>
</table>

C-1
| Beta_k | 9.97992 | 10.00000 | -0.02008 | 0.03721 |
| Phi_k  | 4.54965 | 5.00000  | -0.45035 | 0.17845 |
| BC     | 0.63745 | 0.70000  | -0.06255 | 0.01877 |
| TC     | 0.09935 | 0.10000  | -0.00065 | 0.00446 |

BC=.7, TC=.1, XDF=5, YDF=5

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00463</td>
<td>5.00000</td>
<td>0.00463</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02054</td>
<td>0.00000</td>
<td>-0.02054</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.97978</td>
<td>-20.00000</td>
<td>0.02022</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99060</td>
<td>15.00000</td>
<td>-0.00940</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.97989</td>
<td>10.00000</td>
<td>-0.02011</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.53620</td>
<td>5.00000</td>
<td>-0.46380</td>
</tr>
<tr>
<td>BC</td>
<td>0.63671</td>
<td>0.70000</td>
<td>-0.06329</td>
</tr>
<tr>
<td>TC</td>
<td>0.09938</td>
<td>0.10000</td>
<td>-0.00062</td>
</tr>
</tbody>
</table>

BC=.7, TC=.1, XDF=6, YDF=6

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00502</td>
<td>5.00000</td>
<td>0.00502</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02033</td>
<td>0.00000</td>
<td>-0.02033</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.97836</td>
<td>-20.00000</td>
<td>0.02164</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99095</td>
<td>15.00000</td>
<td>-0.00905</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.97990</td>
<td>10.00000</td>
<td>-0.02010</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.52056</td>
<td>5.00000</td>
<td>-0.47945</td>
</tr>
<tr>
<td>BC</td>
<td>0.63559</td>
<td>0.70000</td>
<td>-0.06441</td>
</tr>
<tr>
<td>TC</td>
<td>0.09940</td>
<td>0.10000</td>
<td>-0.00060</td>
</tr>
</tbody>
</table>

BC=.4, TC=.01, XDF=2, YDF=2

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.02026</td>
<td>5.00000</td>
<td>0.02026</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.01066</td>
<td>0.00000</td>
<td>-0.01066</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.91855</td>
<td>-20.00000</td>
<td>0.08145</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>15.00007</td>
<td>15.00000</td>
<td>0.00007</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.00007</td>
<td>10.00000</td>
<td>0.00007</td>
</tr>
<tr>
<td>Phi_k</td>
<td>2.57657</td>
<td>5.00000</td>
<td>-2.42343</td>
</tr>
<tr>
<td>BC</td>
<td>0.11146</td>
<td>0.40000</td>
<td>-0.28864</td>
</tr>
<tr>
<td>TC</td>
<td>0.01003</td>
<td>0.01000</td>
<td>-0.00003</td>
</tr>
</tbody>
</table>

BC=.4, TC=.01, XDF=2, YDF=2

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00064</td>
<td>5.00000</td>
<td>0.00064</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.00792</td>
<td>0.00000</td>
<td>-0.00792</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.99675</td>
<td>-20.00000</td>
<td>0.00325</td>
</tr>
</tbody>
</table>

C-2
### C.2 Runs Varying Number of Cameras

<table>
<thead>
<tr>
<th>BC= .4, TC= .01, Cameras= 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIT</td>
</tr>
<tr>
<td>14.99670</td>
</tr>
<tr>
<td>9.99206</td>
</tr>
<tr>
<td>4.89159</td>
</tr>
<tr>
<td>0.38430</td>
</tr>
<tr>
<td>0.00993</td>
</tr>
</tbody>
</table>

### C.2.1 Runs Varying Number of Cameras

<table>
<thead>
<tr>
<th>BC= .4, TC= .01, XDF=3, YDF=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIT</td>
</tr>
<tr>
<td>DeltaX_k: 5.00072</td>
</tr>
<tr>
<td>DeltaY_k: -0.00743</td>
</tr>
<tr>
<td>DeltaZ_k: -19.99608</td>
</tr>
<tr>
<td>Beta_k: 9.99198</td>
</tr>
<tr>
<td>Phi_k: 4.90041</td>
</tr>
<tr>
<td>BC: 0.38556</td>
</tr>
<tr>
<td>TC: 0.00994</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BC= .4, TC= .01, XDF=4, YDF=4</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIT</td>
</tr>
<tr>
<td>DeltaX_k: 5.00084</td>
</tr>
<tr>
<td>DeltaY_k: -0.00722</td>
</tr>
<tr>
<td>DeltaZ_k: -19.99636</td>
</tr>
<tr>
<td>Beta_k: 9.99185</td>
</tr>
<tr>
<td>Phi_k: 4.89945</td>
</tr>
<tr>
<td>BC: 0.38565</td>
</tr>
<tr>
<td>TC: 0.00994</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BC= .4, TC= .01, XDF=5, YDF=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIT</td>
</tr>
<tr>
<td>DeltaX_k: 5.00095</td>
</tr>
<tr>
<td>DeltaY_k: -0.00710</td>
</tr>
<tr>
<td>DeltaZ_k: -19.99602</td>
</tr>
<tr>
<td>Alpha_k: 14.99608</td>
</tr>
<tr>
<td>Beta_k: 9.99174</td>
</tr>
<tr>
<td>Phi_k: 4.89640</td>
</tr>
<tr>
<td>BC: 0.38547</td>
</tr>
<tr>
<td>TC: 0.00994</td>
</tr>
</tbody>
</table>
DeltaX_k: 4.99583 5.00000 -0.00417 0.00079
DeltaY_k: 0.00003 0.00000 0.00003 0.00064
DeltaZ_k: -19.98162 -20.00000 0.01838 0.00263
Alpha_k: 14.85465 15.00000 -0.14535 0.01806
Beta_k: 9.97667 10.00000 -0.02433 0.00436
Phi_k: 4.86896 5.00000 -0.13104 0.03536
BC: 0.37456 0.40000 -0.02544 0.00392
TC: 0.00753 0.01000 -0.00247 0.00067

BC= .4, TC= .01, Cameras= 1 and 2

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00255</td>
<td>5.00000</td>
<td>0.00255</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.00559</td>
<td>0.00000</td>
<td>-0.00559</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.98990</td>
<td>-20.00000</td>
<td>0.01010</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.99461</td>
<td>15.00000</td>
<td>-0.00539</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.98918</td>
<td>10.00000</td>
<td>-0.01082</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.79492</td>
<td>5.00000</td>
<td>-0.20508</td>
</tr>
<tr>
<td>BC:</td>
<td>0.37142</td>
<td>0.40000</td>
<td>-0.00115</td>
</tr>
<tr>
<td>TC:</td>
<td>0.00885</td>
<td>0.01000</td>
<td>-0.00115</td>
</tr>
</tbody>
</table>

BC= .4, TC= .01, Cameras= 1 and 3

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00062</td>
<td>5.00000</td>
<td>0.00062</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.00421</td>
<td>0.00000</td>
<td>-0.00421</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.99756</td>
<td>-20.00000</td>
<td>0.00245</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.99645</td>
<td>15.00000</td>
<td>-0.00455</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.99457</td>
<td>10.00000</td>
<td>-0.00543</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.92415</td>
<td>5.00000</td>
<td>-0.07585</td>
</tr>
<tr>
<td>BC:</td>
<td>0.38603</td>
<td>0.40000</td>
<td>-0.01397</td>
</tr>
<tr>
<td>TC:</td>
<td>0.00882</td>
<td>0.01000</td>
<td>-0.00118</td>
</tr>
</tbody>
</table>

BC= .4, TC= .01, Cameras= 1 and 4

<table>
<thead>
<tr>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00182</td>
<td>5.00000</td>
<td>0.00182</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.00508</td>
<td>0.00000</td>
<td>-0.00508</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.99754</td>
<td>-20.00000</td>
<td>0.00245</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.98318</td>
<td>15.00000</td>
<td>-0.01682</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.97089</td>
<td>10.00000</td>
<td>-0.02911</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.86932</td>
<td>5.00000</td>
<td>-0.13068</td>
</tr>
<tr>
<td>BC:</td>
<td>0.36655</td>
<td>0.40000</td>
<td>-0.03345</td>
</tr>
<tr>
<td>TC:</td>
<td>0.00986</td>
<td>0.01000</td>
<td>-0.00014</td>
</tr>
</tbody>
</table>

BC= .4, TC= .01, Cameras= 4

C-4
<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00081</td>
<td>5.00000</td>
<td>0.00081</td>
<td>0.00078</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.00472</td>
<td>0.00000</td>
<td>-0.00472</td>
<td>0.00069</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.99675</td>
<td>-20.00000</td>
<td>0.00325</td>
<td>0.00245</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99423</td>
<td>15.00000</td>
<td>-0.00577</td>
<td>0.01243</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.99360</td>
<td>10.00000</td>
<td>-0.00641</td>
<td>0.00671</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.90767</td>
<td>5.00000</td>
<td>-0.09233</td>
<td>0.03790</td>
</tr>
<tr>
<td>BC</td>
<td>0.38420</td>
<td>0.40000</td>
<td>-0.01580</td>
<td>0.00444</td>
</tr>
<tr>
<td>TC</td>
<td>0.00876</td>
<td>0.01000</td>
<td>-0.00124</td>
<td>0.00087</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00081</td>
<td>5.00000</td>
<td>0.00081</td>
<td>0.00078</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.00472</td>
<td>0.00000</td>
<td>-0.00472</td>
<td>0.00069</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.99675</td>
<td>-20.00000</td>
<td>0.00325</td>
<td>0.00245</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99423</td>
<td>15.00000</td>
<td>-0.00577</td>
<td>0.01243</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.99360</td>
<td>10.00000</td>
<td>-0.00641</td>
<td>0.00671</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.90767</td>
<td>5.00000</td>
<td>-0.09233</td>
<td>0.03790</td>
</tr>
<tr>
<td>BC</td>
<td>0.38420</td>
<td>0.40000</td>
<td>-0.01580</td>
<td>0.00444</td>
</tr>
<tr>
<td>TC</td>
<td>0.00876</td>
<td>0.01000</td>
<td>-0.00124</td>
<td>0.00087</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>1.39916</td>
<td>5.00000</td>
<td>-3.60084</td>
<td>1.72027</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>2.34957</td>
<td>0.00000</td>
<td>2.34957</td>
<td>1.25111</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-3.32290</td>
<td>-20.00000</td>
<td>16.67710</td>
<td>7.50294</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>-8.38708</td>
<td>15.00000</td>
<td>-23.38708</td>
<td>14.45716</td>
</tr>
<tr>
<td>Beta_k</td>
<td>2.17753</td>
<td>10.00000</td>
<td>-7.82247</td>
<td>7.62531</td>
</tr>
<tr>
<td>Phi_k</td>
<td>-4.85253</td>
<td>5.00000</td>
<td>-9.85253</td>
<td>24.85895</td>
</tr>
<tr>
<td>BC</td>
<td>-9.75151</td>
<td>1.00000</td>
<td>-10.75151</td>
<td>31.04010</td>
</tr>
<tr>
<td>TC</td>
<td>4.70599</td>
<td>0.50000</td>
<td>4.20599</td>
<td>17.06678</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.01123</td>
<td>5.00000</td>
<td>0.01123</td>
<td>0.00536</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.01977</td>
<td>0.00000</td>
<td>-0.01977</td>
<td>0.00437</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.92888</td>
<td>-20.00000</td>
<td>0.07112</td>
<td>0.01960</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.77107</td>
<td>15.00000</td>
<td>-0.22893</td>
<td>0.11429</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.03690</td>
<td>10.00000</td>
<td>0.03690</td>
<td>0.05043</td>
</tr>
<tr>
<td>Phi_k</td>
<td>3.12636</td>
<td>5.00000</td>
<td>-1.87364</td>
<td>0.29504</td>
</tr>
<tr>
<td>BC</td>
<td>0.67148</td>
<td>1.00000</td>
<td>-0.32852</td>
<td>0.04327</td>
</tr>
<tr>
<td>TC</td>
<td>0.43699</td>
<td>0.50000</td>
<td>-0.06301</td>
<td>0.00805</td>
</tr>
</tbody>
</table>

BC= .4, TC= .01, Cameras= 8

BC= 1, TC=.5, Cameras= 1

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>1.39916</td>
<td>5.00000</td>
<td>-3.60084</td>
<td>1.72027</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>2.34957</td>
<td>0.00000</td>
<td>2.34957</td>
<td>1.25111</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-3.32290</td>
<td>-20.00000</td>
<td>16.67710</td>
<td>7.50294</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>-8.38708</td>
<td>15.00000</td>
<td>-23.38708</td>
<td>14.45716</td>
</tr>
<tr>
<td>Beta_k</td>
<td>2.17753</td>
<td>10.00000</td>
<td>-7.82247</td>
<td>7.62531</td>
</tr>
<tr>
<td>Phi_k</td>
<td>-4.85253</td>
<td>5.00000</td>
<td>-9.85253</td>
<td>24.85895</td>
</tr>
<tr>
<td>BC</td>
<td>-9.75151</td>
<td>1.00000</td>
<td>-10.75151</td>
<td>31.04010</td>
</tr>
<tr>
<td>TC</td>
<td>4.70599</td>
<td>0.50000</td>
<td>4.20599</td>
<td>17.06678</td>
</tr>
</tbody>
</table>

BC= 1, TC= .5, Cameras= 1 and 2

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.01123</td>
<td>5.00000</td>
<td>0.01123</td>
<td>0.00536</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.01977</td>
<td>0.00000</td>
<td>-0.01977</td>
<td>0.00437</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.92888</td>
<td>-20.00000</td>
<td>0.07112</td>
<td>0.01960</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.77107</td>
<td>15.00000</td>
<td>-0.22893</td>
<td>0.11429</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.03690</td>
<td>10.00000</td>
<td>0.03690</td>
<td>0.05043</td>
</tr>
<tr>
<td>Phi_k</td>
<td>3.12636</td>
<td>5.00000</td>
<td>-1.87364</td>
<td>0.29504</td>
</tr>
<tr>
<td>BC</td>
<td>0.67148</td>
<td>1.00000</td>
<td>-0.32852</td>
<td>0.04327</td>
</tr>
<tr>
<td>TC</td>
<td>0.43699</td>
<td>0.50000</td>
<td>-0.06301</td>
<td>0.00805</td>
</tr>
</tbody>
</table>

C-5
<table>
<thead>
<tr>
<th>BC= 1, TC= .5, Cameras= 1 and 3</th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00420</td>
<td>5.00000</td>
<td>0.00420</td>
<td>0.00449</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.02007</td>
<td>0.00000</td>
<td>-0.02007</td>
<td>0.00436</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.97640</td>
<td>-20.00000</td>
<td>0.02360</td>
<td>0.01420</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.75491</td>
<td>15.00000</td>
<td>-0.24509</td>
<td>0.07321</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.99202</td>
<td>10.00000</td>
<td>-0.00798</td>
<td>0.04222</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>3.82213</td>
<td>5.00000</td>
<td>-1.17787</td>
<td>0.21012</td>
</tr>
<tr>
<td>BC:</td>
<td>0.73340</td>
<td>1.00000</td>
<td>-0.26660</td>
<td>0.02385</td>
</tr>
<tr>
<td>TC:</td>
<td>0.43463</td>
<td>0.50000</td>
<td>-0.06537</td>
<td>0.00464</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BC= 1, TC= .5, Cameras= 1 and 4</th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00966</td>
<td>5.00000</td>
<td>0.00966</td>
<td>0.00412</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.02296</td>
<td>0.00000</td>
<td>-0.02296</td>
<td>0.00360</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.97364</td>
<td>-20.00000</td>
<td>0.02636</td>
<td>0.01743</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.71916</td>
<td>15.00000</td>
<td>-0.28084</td>
<td>0.09659</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.88608</td>
<td>10.00000</td>
<td>-0.11392</td>
<td>0.03871</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>3.60765</td>
<td>5.00000</td>
<td>-1.39235</td>
<td>0.24747</td>
</tr>
<tr>
<td>BC:</td>
<td>0.64836</td>
<td>1.00000</td>
<td>-0.35164</td>
<td>0.02631</td>
</tr>
<tr>
<td>TC:</td>
<td>0.45221</td>
<td>0.50000</td>
<td>-0.04779</td>
<td>0.00632</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BC= 1, TC= .5, Cameras= 4</th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00494</td>
<td>5.00000</td>
<td>0.00494</td>
<td>0.00476</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.02169</td>
<td>0.00000</td>
<td>-0.02169</td>
<td>0.00432</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.97067</td>
<td>-20.00000</td>
<td>0.02933</td>
<td>0.01524</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.77376</td>
<td>15.00000</td>
<td>-0.22624</td>
<td>0.07916</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.99640</td>
<td>10.00000</td>
<td>-0.00360</td>
<td>0.04367</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>3.74709</td>
<td>5.00000</td>
<td>-1.25231</td>
<td>0.22587</td>
</tr>
<tr>
<td>BC:</td>
<td>0.72692</td>
<td>1.00000</td>
<td>-0.27308</td>
<td>0.02606</td>
</tr>
<tr>
<td>TC:</td>
<td>0.43588</td>
<td>0.50000</td>
<td>-0.06412</td>
<td>0.00549</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BC= 1, TC= .5, Cameras= 8</th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.00494</td>
<td>5.00000</td>
<td>0.00494</td>
<td>0.00476</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>-0.02169</td>
<td>0.00000</td>
<td>-0.02169</td>
<td>0.00432</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-19.97067</td>
<td>-20.00000</td>
<td>0.02933</td>
<td>0.01524</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.77376</td>
<td>15.00000</td>
<td>-0.22624</td>
<td>0.07916</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>9.99640</td>
<td>10.00000</td>
<td>-0.00360</td>
<td>0.04367</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>3.74709</td>
<td>5.00000</td>
<td>-1.25231</td>
<td>0.22587</td>
</tr>
<tr>
<td>BC:</td>
<td>0.72692</td>
<td>1.00000</td>
<td>-0.27308</td>
<td>0.02606</td>
</tr>
<tr>
<td>TC:</td>
<td>0.43588</td>
<td>0.50000</td>
<td>-0.06412</td>
<td>0.00549</td>
</tr>
</tbody>
</table>
### C.3 Runs Varying Bending Coefficient

**BC = .01, Deformation Model**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00000</td>
<td>5.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.0000</td>
<td>-20.0000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99999</td>
<td>15.00000</td>
<td>-0.00001</td>
<td>0.00001</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.00000</td>
<td>10.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.99998</td>
<td>5.00000</td>
<td>-0.00002</td>
<td>0.00003</td>
</tr>
<tr>
<td>BC</td>
<td>0.01000</td>
<td>0.01000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>TC</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

**BC = .01, Rigid Model**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00106</td>
<td>5.00000</td>
<td>0.00106</td>
<td>0.00024</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.00059</td>
<td>0.00000</td>
<td>-0.00059</td>
<td>0.00068</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.99875</td>
<td>-20.0000</td>
<td>0.00425</td>
<td>0.00033</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.9992</td>
<td>15.00000</td>
<td>-0.0008</td>
<td>0.00227</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.99888</td>
<td>10.00000</td>
<td>-0.0012</td>
<td>0.00232</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.91773</td>
<td>5.00000</td>
<td>-0.08227</td>
<td>0.00283</td>
</tr>
</tbody>
</table>

**BC = .1, Deformation Model**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00001</td>
<td>5.00000</td>
<td>0.00001</td>
<td>0.00005</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.00029</td>
<td>0.00000</td>
<td>-0.00029</td>
<td>0.00005</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.9997</td>
<td>-20.0000</td>
<td>0.0003</td>
<td>0.00016</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99990</td>
<td>15.00000</td>
<td>-0.0010</td>
<td>0.00080</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.9964</td>
<td>10.00000</td>
<td>-0.00046</td>
<td>0.00043</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.99824</td>
<td>5.00000</td>
<td>-0.00176</td>
<td>0.00248</td>
</tr>
<tr>
<td>BC</td>
<td>0.09966</td>
<td>0.10000</td>
<td>-0.00034</td>
<td>0.00029</td>
</tr>
<tr>
<td>TC</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00000</td>
<td>0.00006</td>
</tr>
</tbody>
</table>

**BC = .1, Rigid Model**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.01069</td>
<td>5.00000</td>
<td>0.01069</td>
<td>0.00245</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.00552</td>
<td>0.00000</td>
<td>-0.00552</td>
<td>0.00684</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.95743</td>
<td>-20.0000</td>
<td>0.04267</td>
<td>0.00328</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.99931</td>
<td>15.00000</td>
<td>-0.00069</td>
<td>0.02273</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.9917</td>
<td>10.00000</td>
<td>-0.00083</td>
<td>0.02322</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.17712</td>
<td>5.00000</td>
<td>-0.82288</td>
<td>0.02531</td>
</tr>
</tbody>
</table>

**BC = .5, Deformation Model**

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
</table>

C-7
<table>
<thead>
<tr>
<th>DeltaX_k</th>
<th>DeltaY_k</th>
<th>DeltaZ_k</th>
<th>Alpha_k</th>
<th>Beta_k</th>
<th>Phi_k</th>
<th>BC</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00164</td>
<td>0.00164</td>
<td>0.00121</td>
<td>14.99815</td>
<td>9.89995</td>
<td>4.84567</td>
<td>0.47671</td>
<td>0.00004</td>
</tr>
<tr>
<td>-0.00741</td>
<td>-0.00741</td>
<td>0.00006</td>
<td>-0.00185</td>
<td>-0.01005</td>
<td>-0.15443</td>
<td>-0.02329</td>
<td>0.00004</td>
</tr>
<tr>
<td>-19.09336</td>
<td>-20.00000</td>
<td>0.06664</td>
<td>0.01917</td>
<td>0.01033</td>
<td>0.06792</td>
<td>0.00678</td>
<td>0.00135</td>
</tr>
</tbody>
</table>

**BC= .5, Rigid Model**

<table>
<thead>
<tr>
<th>DeltaX_k</th>
<th>DeltaY_k</th>
<th>DeltaZ_k</th>
<th>Alpha_k</th>
<th>Beta_k</th>
<th>Phi_k</th>
<th>BC</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.05461</td>
<td>0.05461</td>
<td>0.01214</td>
<td>-0.01835</td>
<td>-0.01835</td>
<td>0.03389</td>
<td>0.00004</td>
<td>0.00478</td>
</tr>
<tr>
<td>-0.01835</td>
<td>0.01214</td>
<td>0.01835</td>
<td>-0.00366</td>
<td>0.06861</td>
<td>0.03708</td>
<td>0.03708</td>
<td>0.12628</td>
</tr>
</tbody>
</table>

**BC= 1, Deformation Model**

<table>
<thead>
<tr>
<th>DeltaX_k</th>
<th>DeltaY_k</th>
<th>DeltaZ_k</th>
<th>Alpha_k</th>
<th>Beta_k</th>
<th>Phi_k</th>
<th>BC</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.01074</td>
<td>5.01074</td>
<td>0.00429</td>
<td>14.99634</td>
<td>9.96896</td>
<td>4.06393</td>
<td>0.86144</td>
<td>1.00000</td>
</tr>
<tr>
<td>-0.02702</td>
<td>0.02702</td>
<td>0.00370</td>
<td>-0.03104</td>
<td>0.03708</td>
<td>0.19693</td>
<td>0.02288</td>
<td>0.02439</td>
</tr>
<tr>
<td>-19.58135</td>
<td>-20.00000</td>
<td>0.21994</td>
<td>0.22432</td>
<td>0.24509</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BC= 1, Rigid Model**

<table>
<thead>
<tr>
<th>DeltaX_k</th>
<th>DeltaY_k</th>
<th>DeltaZ_k</th>
<th>Alpha_k</th>
<th>Beta_k</th>
<th>Phi_k</th>
<th>BC</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.11041</td>
<td>5.11041</td>
<td>0.10721</td>
<td>14.99623</td>
<td>10.02420</td>
<td>3.08916</td>
<td>0.0033</td>
<td>0.00000</td>
</tr>
<tr>
<td>-0.01434</td>
<td>0.01434</td>
<td>0.06557</td>
<td>0.21994</td>
<td>0.24509</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-19.58135</td>
<td>20.00000</td>
<td>0.41865</td>
<td>0.42103</td>
<td>0.42103</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BC= 3, Deformation Model**

<table>
<thead>
<tr>
<th>DeltaX_k</th>
<th>DeltaY_k</th>
<th>DeltaZ_k</th>
<th>Alpha_k</th>
<th>Beta_k</th>
<th>Phi_k</th>
<th>BC</th>
<th>TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.10721</td>
<td>5.10721</td>
<td>0.10721</td>
<td>14.97874</td>
<td>9.99696</td>
<td>4.84567</td>
<td>0.47671</td>
<td>0.00004</td>
</tr>
<tr>
<td>-0.09868</td>
<td>-0.09868</td>
<td>0.06638</td>
<td>-0.02329</td>
<td>0.00678</td>
<td>0.00135</td>
<td>0.00135</td>
<td>0.00135</td>
</tr>
<tr>
<td>-19.57897</td>
<td>-20.00000</td>
<td>0.06638</td>
<td>0.00000</td>
<td>0.00000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

C-8
\[
\begin{array}{cccccc}
\text{Alpha}_k: & 15.00817 & 15.00000 & 0.00817 & 0.32444 \\
\text{Beta}_k: & 9.91008 & 10.00000 & -0.08992 & 0.19264 \\
\Phi_k: & -3.63004 & 5.00000 & -8.63004 & 0.81679 \\
\text{BC}: & 1.56135 & 3.00000 & -1.43865 & 0.09315 \\
\text{TC}: & 0.00268 & 0.00000 & 0.00268 & 0.02301 \\
\end{array}
\]

BC = 3, Rigid Model

<table>
<thead>
<tr>
<th>\text{FIT}</th>
<th>\text{EXACT}</th>
<th>\text{ERROR}</th>
<th>\text{PRECISION}</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>5.30461</td>
<td>5.00000</td>
<td>0.30461</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.14225</td>
<td>0.00000</td>
<td>0.14225</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-18.93527</td>
<td>-20.00000</td>
<td>1.06473</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>15.00500</td>
<td>15.00000</td>
<td>0.00500</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.21304</td>
<td>10.00000</td>
<td>0.21304</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>-16.40640</td>
<td>5.00000</td>
<td>-21.40640</td>
</tr>
</tbody>
</table>

C.4 Runs Varying Twisting Coefficient

TC = .01, Deformation Model

<table>
<thead>
<tr>
<th>\text{FIT}</th>
<th>\text{EXACT}</th>
<th>\text{ERROR}</th>
<th>\text{PRECISION}</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.99998</td>
<td>5.00000</td>
<td>-0.00002</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.00001</td>
<td>0.00000</td>
<td>0.00001</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.00007</td>
<td>-20.00000</td>
<td>-0.00007</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.99562</td>
<td>15.00000</td>
<td>-0.00438</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.00040</td>
<td>10.00000</td>
<td>0.00040</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.98921</td>
<td>5.00000</td>
<td>-0.01079</td>
</tr>
<tr>
<td>BC:</td>
<td>0.00676</td>
<td>0.01000</td>
<td>-0.00324</td>
</tr>
<tr>
<td>TC:</td>
<td>0.00876</td>
<td>0.01000</td>
<td>-0.00124</td>
</tr>
</tbody>
</table>

TC = .01, Rigid Model

<table>
<thead>
<tr>
<th>\text{FIT}</th>
<th>\text{EXACT}</th>
<th>\text{ERROR}</th>
<th>\text{PRECISION}</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.99906</td>
<td>5.00000</td>
<td>-0.00004</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.00081</td>
<td>0.00000</td>
<td>0.00081</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.00455</td>
<td>-20.00000</td>
<td>-0.00455</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.91409</td>
<td>15.00000</td>
<td>-0.06591</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.00640</td>
<td>10.00000</td>
<td>0.00641</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.94461</td>
<td>5.00000</td>
<td>-0.05539</td>
</tr>
</tbody>
</table>

TC = .05, Deformation Model

<table>
<thead>
<tr>
<th>\text{FIT}</th>
<th>\text{EXACT}</th>
<th>\text{ERROR}</th>
<th>\text{PRECISION}</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.99991</td>
<td>5.00000</td>
<td>-0.00009</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.00006</td>
<td>0.00000</td>
<td>0.00006</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.00034</td>
<td>-20.00000</td>
<td>-0.00034</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.97813</td>
<td>15.00000</td>
<td>-0.02187</td>
</tr>
</tbody>
</table>

C-9
<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.99128</td>
<td>5.00000</td>
<td>-0.00872</td>
<td>0.00569</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.00694</td>
<td>0.00000</td>
<td>0.00694</td>
<td>0.01592</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.03975</td>
<td>-20.00000</td>
<td>-0.03975</td>
<td>0.00765</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.57080</td>
<td>15.00000</td>
<td>-0.42920</td>
<td>0.05290</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.03350</td>
<td>10.00000</td>
<td>0.03350</td>
<td>0.05403</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>5.05175</td>
<td>5.00000</td>
<td>0.05175</td>
<td>0.05894</td>
</tr>
</tbody>
</table>

TC= .1, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.99a79</td>
<td>5.00000</td>
<td>-0.00021</td>
<td>0.00053</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.00015</td>
<td>0.00000</td>
<td>0.00015</td>
<td>0.00049</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.00066</td>
<td>-20.00000</td>
<td>-0.00066</td>
<td>0.00169</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.95634</td>
<td>15.00000</td>
<td>-0.04366</td>
<td>0.00848</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.00447</td>
<td>10.00000</td>
<td>0.00447</td>
<td>0.00461</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.89178</td>
<td>5.00000</td>
<td>-0.10822</td>
<td>0.02640</td>
</tr>
<tr>
<td>BC:</td>
<td>-0.02242</td>
<td>0.01000</td>
<td>-0.03242</td>
<td>0.00311</td>
</tr>
<tr>
<td>TC:</td>
<td>0.08765</td>
<td>0.10000</td>
<td>-0.01235</td>
<td>0.00060</td>
</tr>
</tbody>
</table>

TC= .5, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.98203</td>
<td>5.00000</td>
<td>-0.01797</td>
<td>0.01136</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.01575</td>
<td>0.00000</td>
<td>0.01575</td>
<td>0.0180</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.08369</td>
<td>-20.00000</td>
<td>-0.08369</td>
<td>0.01531</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.78735</td>
<td>15.00000</td>
<td>-0.21265</td>
<td>0.04266</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.02972</td>
<td>10.00000</td>
<td>0.02972</td>
<td>0.02363</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.59648</td>
<td>5.00000</td>
<td>-0.50312</td>
<td>0.12920</td>
</tr>
<tr>
<td>BC:</td>
<td>-0.14748</td>
<td>0.01000</td>
<td>-0.15748</td>
<td>0.01519</td>
</tr>
</tbody>
</table>

TC= .5, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k:</td>
<td>4.99781</td>
<td>5.00000</td>
<td>-0.00219</td>
<td>0.00258</td>
</tr>
<tr>
<td>DeltaY_k:</td>
<td>0.00182</td>
<td>0.00000</td>
<td>0.00182</td>
<td>0.00245</td>
</tr>
<tr>
<td>DeltaZ_k:</td>
<td>-20.00473</td>
<td>-20.00000</td>
<td>-0.00473</td>
<td>0.00838</td>
</tr>
<tr>
<td>Alpha_k:</td>
<td>14.78735</td>
<td>15.00000</td>
<td>-0.21265</td>
<td>0.04266</td>
</tr>
<tr>
<td>Beta_k:</td>
<td>10.02972</td>
<td>10.00000</td>
<td>0.02972</td>
<td>0.02363</td>
</tr>
<tr>
<td>Phi_k:</td>
<td>4.59648</td>
<td>5.00000</td>
<td>-0.50312</td>
<td>0.12920</td>
</tr>
<tr>
<td>BC:</td>
<td>-0.14748</td>
<td>0.01000</td>
<td>-0.15748</td>
<td>0.01519</td>
</tr>
</tbody>
</table>
TC: 0.43906  0.50000  -0.06094  0.00301

TC=.5, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.92767</td>
<td>5.00000</td>
<td>-0.07233</td>
<td>0.06573</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.13080</td>
<td>0.00000</td>
<td>0.13080</td>
<td>0.15758</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.43194</td>
<td>-20.0000</td>
<td>-0.43194</td>
<td>0.07659</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>10.72178</td>
<td>15.00000</td>
<td>-4.27822</td>
<td>0.52660</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.44790</td>
<td>10.00000</td>
<td>0.44790</td>
<td>0.52917</td>
</tr>
<tr>
<td>Phi_k</td>
<td>6.21184</td>
<td>5.00000</td>
<td>1.21184</td>
<td>0.58048</td>
</tr>
</tbody>
</table>

TC= 1, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.99227</td>
<td>5.00000</td>
<td>-0.00773</td>
<td>0.00497</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.00623</td>
<td>0.00000</td>
<td>0.00623</td>
<td>0.00496</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.01830</td>
<td>-20.0000</td>
<td>-0.01830</td>
<td>0.01592</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.60260</td>
<td>15.00000</td>
<td>-0.39740</td>
<td>0.08536</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.06853</td>
<td>10.00000</td>
<td>0.06853</td>
<td>0.04955</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.20049</td>
<td>5.00000</td>
<td>-0.79951</td>
<td>0.23804</td>
</tr>
<tr>
<td>BC</td>
<td>-0.28052</td>
<td>0.01000</td>
<td>-0.29052</td>
<td>0.02802</td>
</tr>
<tr>
<td>TC</td>
<td>0.88034</td>
<td>1.00000</td>
<td>-0.11966</td>
<td>0.00609</td>
</tr>
</tbody>
</table>

TC= 1, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.90840</td>
<td>5.00000</td>
<td>-0.09160</td>
<td>0.10618</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.38357</td>
<td>0.00000</td>
<td>0.38357</td>
<td>0.30618</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.85516</td>
<td>-20.0000</td>
<td>-0.85516</td>
<td>0.15187</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>6.49211</td>
<td>15.00000</td>
<td>-8.50789</td>
<td>1.04132</td>
</tr>
<tr>
<td>Beta_k</td>
<td>11.12772</td>
<td>10.00000</td>
<td>1.12772</td>
<td>1.03227</td>
</tr>
<tr>
<td>Phi_k</td>
<td>7.40292</td>
<td>6.00000</td>
<td>2.40292</td>
<td>1.13983</td>
</tr>
</tbody>
</table>

TC= 2, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.97704</td>
<td>5.00000</td>
<td>-0.02296</td>
<td>0.00966</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.01739</td>
<td>0.00000</td>
<td>0.01739</td>
<td>0.01081</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.06331</td>
<td>-20.0000</td>
<td>-0.06331</td>
<td>0.02724</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.37409</td>
<td>15.00000</td>
<td>-0.62592</td>
<td>0.16181</td>
</tr>
<tr>
<td>Beta_k</td>
<td>10.13054</td>
<td>10.00000</td>
<td>0.13054</td>
<td>0.10814</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.12335</td>
<td>5.00000</td>
<td>-0.87665</td>
<td>0.37434</td>
</tr>
<tr>
<td>BC</td>
<td>-0.48618</td>
<td>0.01000</td>
<td>-0.49618</td>
<td>0.04486</td>
</tr>
<tr>
<td>TC</td>
<td>1.76974</td>
<td>2.00000</td>
<td>-0.23026</td>
<td>0.01232</td>
</tr>
</tbody>
</table>

TC= 2, Rigid Model

C-11
C.5 Runs Varying Noise

Noise=.01, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.02440</td>
<td>5.00000</td>
<td>0.02440</td>
<td>0.20041</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>1.21371</td>
<td>0.00000</td>
<td>1.21371</td>
<td>0.56798</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-21.63817</td>
<td>-20.00000</td>
<td>-1.63817</td>
<td>0.28936</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>-1.71328</td>
<td>15.00000</td>
<td>-16.71328</td>
<td>1.99238</td>
</tr>
<tr>
<td>Beta_k</td>
<td>13.04812</td>
<td>10.00000</td>
<td>3.04812</td>
<td>1.94719</td>
</tr>
<tr>
<td>Phi_k</td>
<td>9.47545</td>
<td>5.00000</td>
<td>4.47545</td>
<td>2.17946</td>
</tr>
</tbody>
</table>

Noise=.01, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.01658</td>
<td>5.00000</td>
<td>0.01658</td>
<td>0.06005</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02170</td>
<td>0.00000</td>
<td>-0.02170</td>
<td>0.00432</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.97081</td>
<td>-20.00000</td>
<td>-0.02919</td>
<td>0.15526</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.77310</td>
<td>15.00000</td>
<td>-0.22690</td>
<td>0.07927</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.99619</td>
<td>10.00000</td>
<td>-0.00381</td>
<td>0.04373</td>
</tr>
<tr>
<td>Phi_k</td>
<td>3.74983</td>
<td>5.00000</td>
<td>-1.25017</td>
<td>0.22617</td>
</tr>
<tr>
<td>BC</td>
<td>0.72717</td>
<td>1.00000</td>
<td>-0.27283</td>
<td>0.02609</td>
</tr>
<tr>
<td>TC</td>
<td>0.43586</td>
<td>0.50000</td>
<td>-0.06414</td>
<td>0.00549</td>
</tr>
</tbody>
</table>

Noise=.1, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00449</td>
<td>5.00000</td>
<td>0.00449</td>
<td>0.00489</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02180</td>
<td>0.00000</td>
<td>-0.02180</td>
<td>0.00443</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.97207</td>
<td>-20.00000</td>
<td>0.02793</td>
<td>0.15665</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.76726</td>
<td>15.00000</td>
<td>-0.23273</td>
<td>0.08131</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.99434</td>
<td>10.00000</td>
<td>-0.00566</td>
<td>0.04885</td>
</tr>
<tr>
<td>Phi_k</td>
<td>3.76914</td>
<td>5.00000</td>
<td>-1.23086</td>
<td>0.23192</td>
</tr>
<tr>
<td>BC</td>
<td>0.72947</td>
<td>1.00000</td>
<td>-0.27053</td>
<td>0.02675</td>
</tr>
<tr>
<td>TC</td>
<td>0.43569</td>
<td>0.50000</td>
<td>-0.06431</td>
<td>0.00564</td>
</tr>
</tbody>
</table>

Noise=.1, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.00449</td>
<td>5.00000</td>
<td>0.00449</td>
<td>0.00489</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02180</td>
<td>0.00000</td>
<td>-0.02180</td>
<td>0.00443</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.97207</td>
<td>-20.00000</td>
<td>0.02793</td>
<td>0.15665</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.76726</td>
<td>15.00000</td>
<td>-0.23273</td>
<td>0.08131</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.99434</td>
<td>10.00000</td>
<td>-0.00566</td>
<td>0.04885</td>
</tr>
<tr>
<td>Phi_k</td>
<td>3.76914</td>
<td>5.00000</td>
<td>-1.23086</td>
<td>0.23192</td>
</tr>
<tr>
<td>BC</td>
<td>0.72947</td>
<td>1.00000</td>
<td>-0.27053</td>
<td>0.02675</td>
</tr>
<tr>
<td>TC</td>
<td>0.43569</td>
<td>0.50000</td>
<td>-0.06431</td>
<td>0.00564</td>
</tr>
</tbody>
</table>

C-12
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Deformation Model (Noise=.25)</th>
<th>Rigid Model (Noise=.25)</th>
<th>Deformation Model (Noise=.5)</th>
<th>Rigid Model (Noise=.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX.k</td>
<td>5.01778 5.00000 0.01778 0.05998</td>
<td>5.01977 5.00000 0.01977 0.05977</td>
<td>5.00270 5.00000 0.00270 0.00652</td>
<td>5.02308 5.00000 0.02308 0.05975</td>
</tr>
<tr>
<td>DeltaY.k</td>
<td>0.09885 0.00000 0.09885 0.16829</td>
<td>0.10484 0.00000 0.10484 0.18796</td>
<td>-0.02224 0.00000 -0.02224 0.00592</td>
<td>0.11481 0.00000 0.11481 0.16755</td>
</tr>
<tr>
<td>DeltaZ.k</td>
<td>-20.01948 -20.00000 -0.01948 0.08195</td>
<td>-20.01896 -20.00000 -0.01896 0.08182</td>
<td>-19.97656 -20.00000 -19.97765 -20.00000</td>
<td>-20.01609 -20.00000 -0.01609 0.08167</td>
</tr>
<tr>
<td>Alpha.k</td>
<td>10.69551 15.00000 -4.30149 0.56911</td>
<td>10.70846 15.00000 -4.29154 0.56822</td>
<td>14.75756 15.00000 -4.25857 0.10857</td>
<td>14.74143 15.00000 -4.14569 0.30928</td>
</tr>
<tr>
<td>Beta.k</td>
<td>10.02623 10.00000 0.02623 0.56768</td>
<td>10.02369 10.00000 0.02370 0.56679</td>
<td>9.99125 10.00000 0.00875 0.4883</td>
<td>9.98610 10.00000 0.01390 0.05986</td>
</tr>
<tr>
<td>Phi.k</td>
<td>-1.62005 5.00000 -6.62005 0.62255</td>
<td>-1.61650 5.00000 -6.61650 0.62157</td>
<td>3.80120 5.00000 -1.19880 0.25243</td>
<td>3.85431 5.00000 -1.14569 0.30928</td>
</tr>
<tr>
<td>BC</td>
<td>0.73329 1.00000 -0.26671 0.02912</td>
<td>0.73329 1.00000 -0.26671 0.02912</td>
<td>0.73962 1.00000 -0.26038 0.03568</td>
<td>0.73962 1.00000 -0.26038 0.03568</td>
</tr>
<tr>
<td>TC</td>
<td>0.43539 0.50000 -0.06461 0.00614</td>
<td>0.43539 0.50000 -0.06461 0.00614</td>
<td>0.43490 0.50000 -0.06510 0.00752</td>
<td>0.43490 0.50000 -0.06510 0.00752</td>
</tr>
</tbody>
</table>

Noise= .25, Deformation Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Fit Exact Error Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX.k</td>
<td>5.00381 5.00000 0.00381 0.00532</td>
</tr>
<tr>
<td>DeltaY.k</td>
<td>-0.02197 0.00000 -0.02197 0.00483</td>
</tr>
<tr>
<td>DeltaZ.k</td>
<td>14.75756 15.00000 -0.24244 0.08854</td>
</tr>
<tr>
<td>Alpha.k</td>
<td>9.99125 10.00000 -0.00875 0.04883</td>
</tr>
<tr>
<td>Beta.k</td>
<td>3.60120 5.00000 -1.19880 0.25243</td>
</tr>
<tr>
<td>Phi.k</td>
<td>0.73329 1.00000 -0.26671 0.02912</td>
</tr>
<tr>
<td>BC</td>
<td>0.43539 0.50000 -0.06461 0.00614</td>
</tr>
</tbody>
</table>

Noise= .25, Rigid Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Fit Exact Error Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX.k</td>
<td>5.01977 5.00000 0.01977 0.05977</td>
</tr>
<tr>
<td>DeltaY.k</td>
<td>0.10484 0.00000 0.10484 0.16796</td>
</tr>
<tr>
<td>DeltaZ.k</td>
<td>-20.01896 -20.00000 -0.01896 0.08182</td>
</tr>
<tr>
<td>Alpha.k</td>
<td>10.70846 15.00000 -4.29154 0.56822</td>
</tr>
<tr>
<td>Beta.k</td>
<td>3.60120 5.00000 -1.19880 0.25243</td>
</tr>
<tr>
<td>Phi.k</td>
<td>0.73329 1.00000 -0.26671 0.02912</td>
</tr>
<tr>
<td>BC</td>
<td>0.43539 0.50000 -0.06461 0.00614</td>
</tr>
</tbody>
</table>

Noise= .5, Deformation Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Fit Exact Error Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX.k</td>
<td>5.00270 5.00000 0.00270 0.00652</td>
</tr>
<tr>
<td>DeltaY.k</td>
<td>-0.02224 0.00000 -0.02224 0.00592</td>
</tr>
<tr>
<td>DeltaZ.k</td>
<td>14.74143 15.00000 -0.25857 0.10857</td>
</tr>
<tr>
<td>Alpha.k</td>
<td>9.98610 10.00000 -0.01390 0.05986</td>
</tr>
<tr>
<td>Beta.k</td>
<td>3.85431 5.00000 -1.14569 0.30928</td>
</tr>
<tr>
<td>Phi.k</td>
<td>0.73962 1.00000 -0.26038 0.03568</td>
</tr>
<tr>
<td>BC</td>
<td>0.43490 0.50000 -0.06510 0.00752</td>
</tr>
</tbody>
</table>

Noise= .5, Rigid Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Fit Exact Error Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX.k</td>
<td>5.00270 5.00000 0.00270 0.00652</td>
</tr>
<tr>
<td>DeltaY.k</td>
<td>-0.02224 0.00000 -0.02224 0.00592</td>
</tr>
<tr>
<td>DeltaZ.k</td>
<td>14.74143 15.00000 -0.25857 0.10857</td>
</tr>
<tr>
<td>Alpha.k</td>
<td>9.98610 10.00000 -0.01390 0.05986</td>
</tr>
<tr>
<td>Beta.k</td>
<td>3.85431 5.00000 -1.14569 0.30928</td>
</tr>
<tr>
<td>Phi.k</td>
<td>0.73962 1.00000 -0.26038 0.03568</td>
</tr>
</tbody>
</table>

C-13
<table>
<thead>
<tr>
<th>Noise</th>
<th>Deformation Model</th>
<th>Fit</th>
<th>Exact</th>
<th>Error</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DeltaX_k</td>
<td>5.00157</td>
<td>5.00000</td>
<td>0.00157</td>
<td>0.00807</td>
</tr>
<tr>
<td></td>
<td>DeltaY_k</td>
<td>-0.02252</td>
<td>0.00000</td>
<td>-0.02252</td>
<td>0.00733</td>
</tr>
<tr>
<td></td>
<td>DeltaZ_k</td>
<td>-19.98118</td>
<td>-20.00000</td>
<td>0.01882</td>
<td>0.02588</td>
</tr>
<tr>
<td></td>
<td>Alpha_k</td>
<td>14.72526</td>
<td>15.00000</td>
<td>-0.27474</td>
<td>0.13457</td>
</tr>
<tr>
<td></td>
<td>Beta_k</td>
<td>9.98094</td>
<td>10.00000</td>
<td>-0.01906</td>
<td>0.07419</td>
</tr>
<tr>
<td></td>
<td>Phi_k</td>
<td>3.90825</td>
<td>5.00000</td>
<td>-1.09175</td>
<td>0.38306</td>
</tr>
<tr>
<td></td>
<td>BC</td>
<td>0.74606</td>
<td>1.00000</td>
<td>-0.25394</td>
<td>0.04419</td>
</tr>
<tr>
<td></td>
<td>TC</td>
<td>0.43441</td>
<td>0.50000</td>
<td>-0.06559</td>
<td>0.00932</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Noise</th>
<th>Rigid Model</th>
<th>Fit</th>
<th>Exact</th>
<th>Error</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DeltaX_k</td>
<td>5.00049</td>
<td>5.00000</td>
<td>-0.00049</td>
<td>0.00981</td>
</tr>
<tr>
<td></td>
<td>DeltaY_k</td>
<td>0.12452</td>
<td>0.12452</td>
<td>0.12452</td>
<td>0.16729</td>
</tr>
<tr>
<td></td>
<td>DeltaZ_k</td>
<td>-20.01722</td>
<td>-20.00000</td>
<td>-0.01722</td>
<td>0.08159</td>
</tr>
<tr>
<td></td>
<td>Alpha_k</td>
<td>10.01442</td>
<td>10.00000</td>
<td>-0.01442</td>
<td>0.56519</td>
</tr>
<tr>
<td></td>
<td>Beta_k</td>
<td>10.75814</td>
<td>15.00000</td>
<td>-4.24186</td>
<td>0.46571</td>
</tr>
<tr>
<td></td>
<td>Phi_k</td>
<td>0.75214</td>
<td>1.00000</td>
<td>-0.04069</td>
<td>0.48571</td>
</tr>
<tr>
<td></td>
<td>BC</td>
<td>0.43393</td>
<td>0.50000</td>
<td>-0.06607</td>
<td>0.1134</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Noise</th>
<th>Deformation Model</th>
<th>Fit</th>
<th>Exact</th>
<th>Error</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DeltaX_k</td>
<td>5.02971</td>
<td>5.00000</td>
<td>0.02971</td>
<td>0.05967</td>
</tr>
<tr>
<td></td>
<td>DeltaY_k</td>
<td>0.13472</td>
<td>0.13472</td>
<td>0.13472</td>
<td>0.16718</td>
</tr>
<tr>
<td></td>
<td>DeltaZ_k</td>
<td>-20.01636</td>
<td>-20.00000</td>
<td>-0.01636</td>
<td>0.08158</td>
</tr>
<tr>
<td></td>
<td>Alpha_k</td>
<td>10.01099</td>
<td>10.00000</td>
<td>-0.01099</td>
<td>0.56516</td>
</tr>
<tr>
<td></td>
<td>Beta_k</td>
<td>10.75814</td>
<td>15.00000</td>
<td>-4.24186</td>
<td>0.56558</td>
</tr>
<tr>
<td></td>
<td>Phi_k</td>
<td>-1.59878</td>
<td>5.00000</td>
<td>-6.59878</td>
<td>0.61979</td>
</tr>
</tbody>
</table>
### Noise= 5, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.98367</td>
<td>5.00000</td>
<td>-0.01633</td>
<td>0.04130</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02711</td>
<td>0.00000</td>
<td>-0.02711</td>
<td>0.03803</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.03678</td>
<td>-20.00000</td>
<td>-0.03678</td>
<td>0.13375</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.45712</td>
<td>15.00000</td>
<td>-0.54289</td>
<td>0.69989</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.89562</td>
<td>10.00000</td>
<td>-0.10438</td>
<td>0.38438</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.75190</td>
<td>5.00000</td>
<td>-0.24810</td>
<td>1.96557</td>
</tr>
<tr>
<td>BC</td>
<td>0.84682</td>
<td>1.00000</td>
<td>-0.15318</td>
<td>0.22666</td>
</tr>
<tr>
<td>TC</td>
<td>0.42655</td>
<td>0.50000</td>
<td>-0.07345</td>
<td>0.04838</td>
</tr>
</tbody>
</table>

### Noise= 5, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.08221</td>
<td>5.00000</td>
<td>0.08221</td>
<td>0.06634</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.29192</td>
<td>0.00000</td>
<td>0.29192</td>
<td>0.18466</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.00261</td>
<td>-20.00000</td>
<td>-0.00261</td>
<td>0.09098</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>11.02010</td>
<td>15.00000</td>
<td>-3.97990</td>
<td>0.63190</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.94238</td>
<td>10.00000</td>
<td>-0.06762</td>
<td>0.63032</td>
</tr>
<tr>
<td>Phi_k</td>
<td>-1.50586</td>
<td>5.00000</td>
<td>-6.50586</td>
<td>0.69128</td>
</tr>
</tbody>
</table>

### Noise= 10, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.96475</td>
<td>5.00000</td>
<td>-0.03525</td>
<td>0.08052</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.03227</td>
<td>0.00000</td>
<td>-0.03227</td>
<td>0.07526</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.09482</td>
<td>-20.00000</td>
<td>-0.09482</td>
<td>0.26348</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.15507</td>
<td>15.00000</td>
<td>-3.94493</td>
<td>1.38878</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.79952</td>
<td>10.00000</td>
<td>-0.20048</td>
<td>0.75951</td>
</tr>
<tr>
<td>Phi_k</td>
<td>5.62229</td>
<td>5.00000</td>
<td>0.62229</td>
<td>3.84175</td>
</tr>
<tr>
<td>BC</td>
<td>0.95123</td>
<td>1.00000</td>
<td>-0.04877</td>
<td>0.44302</td>
</tr>
<tr>
<td>TC</td>
<td>0.41820</td>
<td>0.50000</td>
<td>-0.06180</td>
<td>0.09578</td>
</tr>
</tbody>
</table>

### Noise= 10, Rigid Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.14665</td>
<td>5.00000</td>
<td>0.14665</td>
<td>0.08725</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>0.48419</td>
<td>0.00000</td>
<td>0.48419</td>
<td>0.24086</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.98572</td>
<td>-20.00000</td>
<td>0.01428</td>
<td>0.12007</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>11.34189</td>
<td>15.00000</td>
<td>-3.65811</td>
<td>0.83407</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.85707</td>
<td>10.00000</td>
<td>-0.14293</td>
<td>0.83201</td>
</tr>
<tr>
<td>Phi_k</td>
<td>-1.39215</td>
<td>5.00000</td>
<td>-6.39215</td>
<td>0.91254</td>
</tr>
</tbody>
</table>

### Noise= 25, Deformation Model

<table>
<thead>
<tr>
<th></th>
<th>FIT</th>
<th>EXACT</th>
<th>ERROR</th>
<th>PRECISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>4.98367</td>
<td>5.00000</td>
<td>-0.01633</td>
<td>0.04130</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.02711</td>
<td>0.00000</td>
<td>-0.02711</td>
<td>0.03803</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.03678</td>
<td>-20.00000</td>
<td>-0.03678</td>
<td>0.13375</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>14.45712</td>
<td>15.00000</td>
<td>-0.54289</td>
<td>0.69989</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.89562</td>
<td>10.00000</td>
<td>-0.10438</td>
<td>0.38438</td>
</tr>
<tr>
<td>Phi_k</td>
<td>4.75190</td>
<td>5.00000</td>
<td>-0.24810</td>
<td>1.96557</td>
</tr>
<tr>
<td>BC</td>
<td>0.84682</td>
<td>1.00000</td>
<td>-0.15318</td>
<td>0.22666</td>
</tr>
<tr>
<td>TC</td>
<td>0.42655</td>
<td>0.50000</td>
<td>-0.07345</td>
<td>0.04838</td>
</tr>
<tr>
<td>Variable</td>
<td>Value</td>
<td>Exact Value</td>
<td>Error</td>
<td>Precision</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-------------</td>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>DeltaX_k</td>
<td>4.91756</td>
<td>5.00000</td>
<td>-0.08244</td>
<td>0.19254</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>-0.04685</td>
<td>0.00000</td>
<td>-0.04685</td>
<td>0.18663</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-20.23545</td>
<td>-20.00000</td>
<td>-0.23545</td>
<td>0.64453</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>13.31776</td>
<td>15.00000</td>
<td>-1.68224</td>
<td>3.46643</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.53157</td>
<td>10.00000</td>
<td>-0.46843</td>
<td>1.87462</td>
</tr>
<tr>
<td>Phi_k</td>
<td>7.69098</td>
<td>5.00000</td>
<td>2.69098</td>
<td>9.20866</td>
</tr>
<tr>
<td>BC</td>
<td>1.20212</td>
<td>1.00000</td>
<td>0.20212</td>
<td>1.06343</td>
</tr>
<tr>
<td>TC</td>
<td>0.39737</td>
<td>0.50000</td>
<td>-0.10263</td>
<td>0.23788</td>
</tr>
</tbody>
</table>

Noise= 25, Rigid Model

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Exact Value</th>
<th>Error</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeltaX_k</td>
<td>5.33253</td>
<td>5.00000</td>
<td>0.33253</td>
<td>0.17191</td>
</tr>
<tr>
<td>DeltaY_k</td>
<td>1.03234</td>
<td>0.00000</td>
<td>1.03234</td>
<td>0.46285</td>
</tr>
<tr>
<td>DeltaZ_k</td>
<td>-19.93685</td>
<td>-20.00000</td>
<td>0.06315</td>
<td>0.23884</td>
</tr>
<tr>
<td>Alpha_k</td>
<td>12.26844</td>
<td>15.00000</td>
<td>-2.73156</td>
<td>1.65980</td>
</tr>
<tr>
<td>Beta_k</td>
<td>9.59825</td>
<td>10.00000</td>
<td>-0.40175</td>
<td>1.66612</td>
</tr>
<tr>
<td>Phi_k</td>
<td>-1.06885</td>
<td>5.00000</td>
<td>-6.06986</td>
<td>1.81670</td>
</tr>
</tbody>
</table>
Bibliography


4. Ruyten, Wim. Correspondence from Dr. Ruyten.


Vita

Lt. Sean Krolikowski was born in Chicago, IL. After his parents divorce at the age of 3, he moved to Michigan with his mother, where they continued to move around quite a bit.

Sean graduated from Tecumseh High School in 1993, and immediately left for the United States Air Force Academy. In 1997 he graduated from the Academy with a Bachelor's of Science in Astronautical Engineering.

After his immediate commissioning, Sean received his first assignment at Wright-Patterson AFB in the Aeronautical Systems Center. He was assigned to the Air Superiority TPIPT of ASC/XR, development planning. There he assisted in the production of long range planning documents.

Sean received his Master's of Astronautical Engineering from AFIT in 2001. Upon graduation, he was assigned to the Space and Missile Center at Los Angeles AFB. There he will work in the Evolved Expendable Launch Vehicle (EELV) office.
This study improved the current method of position and attitude determination to account for structural deformation of the wind tunnel test article due to aerodynamic loading. To account for deformation, parabolic bending and linear twisting coefficients were added into the Levenberg-Marquardt multi-parameter solver. By accounting for deformation, the accuracy of position and attitude determination was greatly improved. This study also takes a qualitative look at the optimum number of wind tunnel cameras and model targets. Optimal configuration was found to be around 50 targets and 2 cameras offset by 90 degrees.

### Subject Terms
Structural Deformation, Levenberg-Marquardt, Position and Attitude Determination