DEVELOPMENT OF PHOTOGRAPHIC DYNAMIC MEASUREMENTS APPLICABLE TO EVALUATION OF FLAPPING WING MICRO AIR VEHICLES

THESIS

Jeremy C. Murray

AFIT/GAE/ENY/11-D02

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GAE/ENY/11-D02

DEVELOPMENT OF PHOTOGRAPHIC DYNAMIC MEASUREMENTS APPLICABLE TO EVALUATION OF FLAPPING WING MICRO AIR VEHICLES

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 Aeronautical Engineering

Jeremy C. Murray, BS

December 2011

AFIT/GAE/ENY/11-D02

DEVELOPMENT OF PHOTOGRAPHIC DYNAMIC MEASUREMENTS APPLICABLE TO EVALUATION OF FLAPPING WING MICRO AIR VEHICLES

Jeremy C. Murray, BS

Approved:

_____	_____
Dr. Anthony Palazotto, PhD (Chairman)	Date
_____	_____
Dr. Richard Cobb, PhD (Member)	Date
_____	_____
Dr. Mark Reeder, PhD (Member)	Date

AFIT/GAE/ENY/11-D02

Abstract

Developments in the area of flapping wing micro air vehicles (FWMAVs) of a small size and with limited range. This has lead to a great deal of interest in biomimetric designs based on flapping wing flyers, including the North American Hawkmoth (*Manduca Sexta*). Methods of characterizing wings experimentally are currently limited in scope and capability. By utilizing high speed photography and photogrammetry the dynamic flapping of the wing can be characterized for comparison with mathematical models, namely computational fluid dynamics (CFD) and finite element analysis (FEA).

To successfully utilize high speed image capture, a method of successfully digitizing many data points in a short period of time needs to be developed. The effort coordinated several photographically oriented MATLAB functions in a sequential order to develop the eventual three dimensional output. The key function to track two dimensional points was developed in this research utilizing MATLAB code, and was an important requirement to achieving the eventual results. The increased stroke angle will be used to characterize the wing displacement at the ends of the stroke and near zero where still image photogrammetry cannot capture enough images. Specific concerns with photogrammetry in this test situation have been considered and The utilization of photogrammetry was further improved by a mechanical device to automate calibration image collection, which reduces calibration errors.

Acknowledgments

I would like to thank Dr. Anthony Palazotto for giving me this opportunity to demonstrate what I can accomplish.

Table of Contents

Page

List of Figures

Page

List of Tables

Page

List of Abbreviations

Page

Chapter 1 : Introduction of the Research Topic

1.1 Biomimetic Micro Air Vehicles

The purpose of this research is to investigate a method of characterizing biologically inspired engineered flapping micro air vehicle (MAV) wings in motion using a photographical technique. The basis for these biologically inspired wings is the *Manduca Sexta* or the North American Hawkmoth, seen in Figure 1.

Figure 1: Adult female hawkmoth.

This research is being performed to achieve the Defense Advanced Research Projects Agency's (DARPA's) vision of an optimal MAV, less than a 6 inch wingspan and a range of 1km at a speed of 1m/s[3]. The work herein will build on research performed by Nathanial DeLeón to understand the structural aspects of wings and their response to dynamic flapping[1]. Hawkmoth wing deformation is passive, because there are no control muscles to actively change the wing shape[2].

1.2 The Problem

The overall objective of this thesis is to improve on the method developed by DeLeón to capture wing surface shapes in three dimensions for comparison of biological and engineered wings in air and in vacuum[1]. These can then be used to compare with an FEA (finite element analysis) model or compared directly to CFD (computational fluid dynamics) code output without FEA preprocessing. In Deleón's work, images were taken at 12 different angles in the stroke using a strobe to "freeze" the images for photography[1]. This is very coarse and leads to issues characterizing the wing around zero degrees and at the ends of the stroke as seen in some of DeLeón's output data.

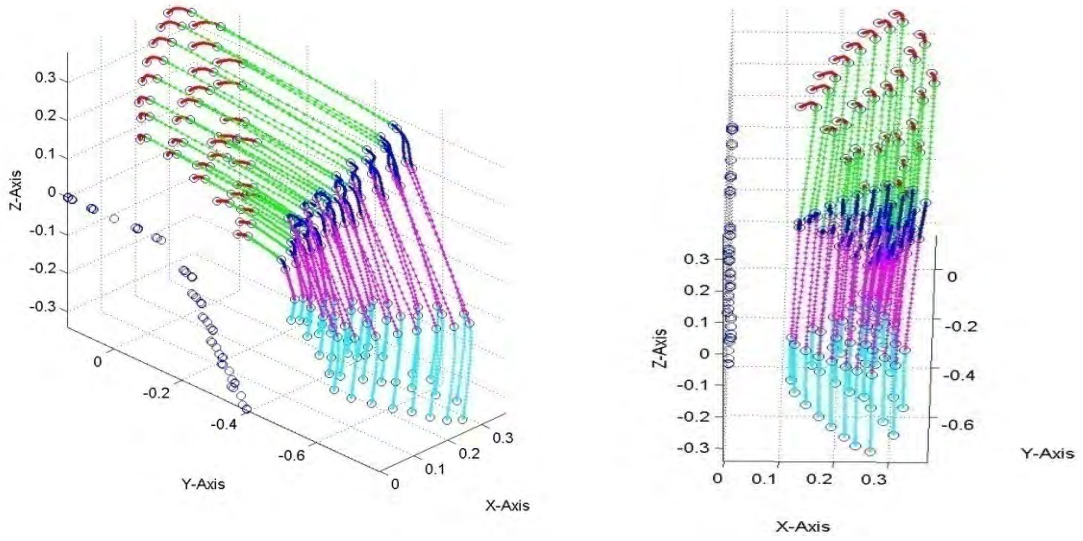

Figure 2: Cardinal spline data: Downstroke engineered wing in air, two views[1].

The intervening points are estimated using a cardinal spline method[1]. By attempting to predict the location of the wing points between measured data, the technique may be losing important details about the dynamic response of the flapped wing. This technique is incapable of capturing data at either end of the stroke where the wing changes direction, which provide important information about vortex shedding[1,15].

Using high-speed cameras, the image count per stroke in this research is 100 images per full stroke. The increased stroke angle resolution compared to the present research will capture more of the nonlinear motion of the wing. The increased resolution makes it more reasonable to use a linear interpolation across the time step between data points.

The interaction between wing structure and fluid are also necessary to characterize the wing response to improve design and manufacture for MAV uses. Therefore, testing will be performed in air and in a medium vacuum at 2.5 torr. This also allows for the data to be compared to FEA in the case of a flapping wing in medium vacuum, or a CFD code in the case of a flapping wing in open air. The goal is to compare an engineered wing with a similar response and performance to freshly removed, less than 2 hours since removal, *Manduca Sexta* wings.

Based on information gathered from previous research and goals of the MAV team, the final system should:

- Utilize high-speed camera input
- Have automated image digitization to reduce processing time
- Produce three dimensional point clouds tracking points similar to DeLeón
- Be capable of handling wing stroke of up to 110 degrees

1.3 Photogrammetry

Stereophotogrammetry is a well understood tool for producing three dimensional locations from pairs of two dimensional images. The system is based in essence on binocular vision providing depth perception. When this is combined with the ability to calibrate a lens to

adjust for the distortions the ultimate system produces very accurate data. Furthermore, Stereophotogrammetry can be applied in situations with little to no effect on the subject being measured. This capability is particularly useful when measuring lightweight systems such as MAV wings.

The Photomodeler package was used by DeLeón [1] to build the point clouds. It was found to be time consuming as images had to be digitized by hand without coded targets[1].

In order to make these measurements, one must first calibrate the images to a known three dimensional object. This provides a relationship between distance measured in an image recorded by the cameras and the distance that is measured in the three dimensional frame. A planar pattern of circles or a checkerboard pattern is frequently used as the input to this system. This allows the operator to pull out several intrinsic parameters needed to perform photogrammetry at a set focal distance [6].

Camera calibration originally required a very complicated system involving an optical bench and stellar calibration. This was found to be very cumbersome and in the case of a mounted lens impossible. Therefore, in Brown's paper, he develops a method of calibrating cameras without using stellar calibration, which is difficult and requires very precise measurements and also an optical bench[6]. Brown used a grid of plumblines suspended in oil and a lens with known high distortion and decentering to determine the calibrations at multiple focal lengths[6]. The use of an intentionally poor lens was to make the distortion easier to measure and to test the equations against a poor case. In this experiment, movement of the principle point was separated from radial distortions, allowing these to be determined separately, and in many cases decentering is a non-issue with modern lenses and can be ignored[6,7]. The

most important conclusion from Brown's work is that most lenses, even cheaply produced ones, can be used to produce quality photogrammetry measurements with proper calibration[6].

Newer systems have been developed using digital cameras and computer systems to determine these intrinsic parameters. One method involves using lumped parameters to transform from the three dimensional positions to the two dimensional coordinates of the point in the image frame[7]. These are set up in a series of linear equations for which the same set of 12 parameters solve the equations[7]. One necessity of direct linear transformation (DLT) is to overdetermine the 12 parameters and solve for them iteratively [8]. For this purpose, a minimum of 9 images is recommended and as many as 20 may be used without excessive runtime[9]. The twelve estimated parameters do not in themselves represent physical quantities. However, with a priori knowledge, the terms can be adjusted into physical lens and camera parameters[7]. This is not generally necessary for photogrammetric measurements[7].

There are other methods aside from DLT for determining the transformation parameters for photogrammetry, namely the Tsai Method[12] and the Zhang Method[11]. All three methods have been shown to produce reliable results. The Tsai method ignores linear distortion, decentering, and also frequently ignores skew and any lack of orthogonality of projection. These simplifications to the camera model make this method somewhat faster and do not require an initial guess to process[8,11]. Tsai was also able to verify that a solid state sensor such as a charge-coupled device (CCD) is valid for high accuracy three-dimensional machine vision[12]. The decision whether or not to ignore decentering should be based on another calibration method, as many lenses do not have decentering issues[7,12]. The residuals calculated using this method are not useful for determining system error[8]. The Zhang method uses a minimum of

three different projections of a planar calibration target to determine the camera calibration[8]. Zhang recommends 4-5 projections for improved accuracy[11]. It is also recommended that the user avoid placing the calibration plane parallel to the image plane when trying to limit the number of required projections[11]. The plane should also not be moved via simple translation for new images[11].

1.4 Image Tracking

One major goal of this research was an automated way to digitize the images, specifically the marker locations in the image reference frame for input to the stereo triangulation program. The simplest method is for an operator to select the points by hand, which is how DeLeón performed his digitization[4]. Individual tracking performed by DeLeón required selecting 42 points in each camera view, three times per angle and 24 total angles which is 3024 points input per test set[1]. These were also performed with three separate image sets per angle to handle potential image issues as well. This is then repeated for each of 4 tests. The total count is 36288 points selected, in careful order, manually by an operator[1]. This process was deemed undesirable for rapid testing as it could take several days to complete the image digitization. It was also found to be error prone as the points needed to be in the same order and the procedure needed to be followed exactly. Automatic tracking was a large focus of the improvements to the current process. Ultimately different methods have been developed in many arenas based on meeting specific challenges encountered in a limited scenario[4]. Photomodeler 6 requires the use of coded targets to use the automated search feature, and while this does make the processing easier and faster, DeLeón determined the coded targets to are too heavy for use on the wings without changing the results of the experiment[1]. The marking system developed by DeLeón uses a low

mass white marker which is volatile enough to flash off[1]. This does preclude the ability to draw specific shapes on the wings. As such, Photomodeler 6 was no longer considered for our current experiments. It should be noted that the latest version of Photomodeler has added a feature to use points on an object for automated marking without coded targets. This feature may be useful for further evaluations of the wing.

Other available methods of point digitization that have been examined are normalized cross correlation and peak tracking. Normalized cross correlation attempts to match an image of a target, smaller than the overall image, to a section of the overall image[4]. This method continues as long as a suitable match can be found in each subsequent frame without any user intervention. An important thing to realize with this method is that it works over an array of pixels rather than an actual marker[4]. This means that any distinctive array of pixels may be used for a search option, such as the natural pattern on an animal. The converse being that if the array of pixels is not distinguishable from other portions within the image, they may produce erroneous results if there is another match. The data must be normalized, because cross correlation assumes a stochastic, stationary process. Potential issues that must be contended with include objects leaving the image frame and lighting changes[4,5]. One possible improvement to cross correlation include the addition of multiple passes which produce smaller subsections to then search for the point in question[4]. An alternative improvement is to use a trajectory system to predict the locations of points in subsequent frames[4]. This does require a priori information from a few frames to determine the trajectory. Often a two dimensional prediction is inadequate to describe a three dimensional motion captured on a two dimensional medium[4]. Normalized cross correlation may be used as a stand-alone tool or as a portion of a more complex system to address issues of scale, feature updating and other issues[5].

Peak tracking on the other hand converts the image into a binary array and then calculates the centroids of the marker points. This requires regular targets such as circles or squares[4]. All points in this case need to be equally visible and requires a sufficiently different background for removal[4].

1.5 Micro Air Vehicles

In MAV research, it has become of utmost importance to develop a wing which deforms properly under flapping conditions to generate lift. This is key because the insect lacks muscles to actively change the wing shape[2]. For a biomimetic system to be efficient its wings will need to deform without additional controllers which would add excessive weight and interfere further with wing flapping. Angle of attack is also largely passive during hovering flight[2]. It has been noted that for similar wing flapping motions and planforms, a rigid wing can produce greater lift, however the effect of wing flexibility on drag and overall efficiency requires further investigation[2].

Nathanial DeLeón found that mimicking weight, vein geometry, and modal properties did not yield a successful MAV wing, due to incorrect inertial loading. A greater range of materials would have to be considered[1]. He also was able to test a previous theory that modal analysis would be key to matching flapping performance of the biological specimen, and found that the primary difference was the distinct lack of ‘scoop’ shape formed by the biological wing on the downstroke[1]. Thus, indicating that modal analysis at the first few modes is insufficient to predict flapping behavior. The scoop shape is believed to be necessary to generate sufficient lift. The case for aerodynamics playing a significant part is backed up in the difference between the biological wing in air as opposed to vacuum. In a vacuum, the wing exhibits a symmetric

passive rotation, which would generate zero lift in air[1]. However, in air the symmetry is gone, and the aforementioned ‘scoop’ is present[1].

1.6 Final comments in the Introduction

The system under consideration requires two dimensional point tracking to develop the three dimensional model as the points in each camera must be correlated in the two dimensional frames. It also requires camera calibration parameters to turn the two dimensional points into three dimensional points. The photogrammetry program that is to be considered is the Stereo Vision Toolbox built in MATLAB by Jean-Yves Bouguet of the California Institute of Technology[14]. This system allows the user to supply digitized points from calibrated cameras and produces the three dimensional point cloud. The bulk of this work was then performed in developing a tracking program to speed the processing of data. Thus, allowing a much larger number of data points and frames to be processed to provide further wing information as well as data for comparison to CFD and FEA models. The current system does not capture key portions of the wing stroke and may not adequately model the motion between capture points. To improve the capability of the system this work will develop a system utilizing a high-speed cameras to capture more images in time to fill out the data set, without extending the processing time beyond the current method.

Chapter 2: Theory

2.1 Introduction Stereophotogrammetry

Stereophotogrammetry is a post test image analysis method, based on digital images, which can determine the location of points in a three dimensional rectilinear coordinate system. This method is very flexible in terms of equipment and applications. It can be implemented to have zero impact on certain test articles, as key points on the article can often be used as tracking locations.

Stereophotogrammetry uses a pair of cameras calibrated to determine the imaging parameters of each camera and lens system (intrinsic parameter) and the orientation of the cameras to one another (extrinsic parameter) to calculate the location of object points in three dimensional rectilinear coordinates relative to either camera.

2.2 Pinhole Camera Model

Stereophotogrammetry starts with a basic pinhole model of a camera to convert points from a two dimensional location in each image to a ray from each camera. The pinhole model of the camera treats the lens as a single point with rays passing through it to the image as seen in Figure 3.

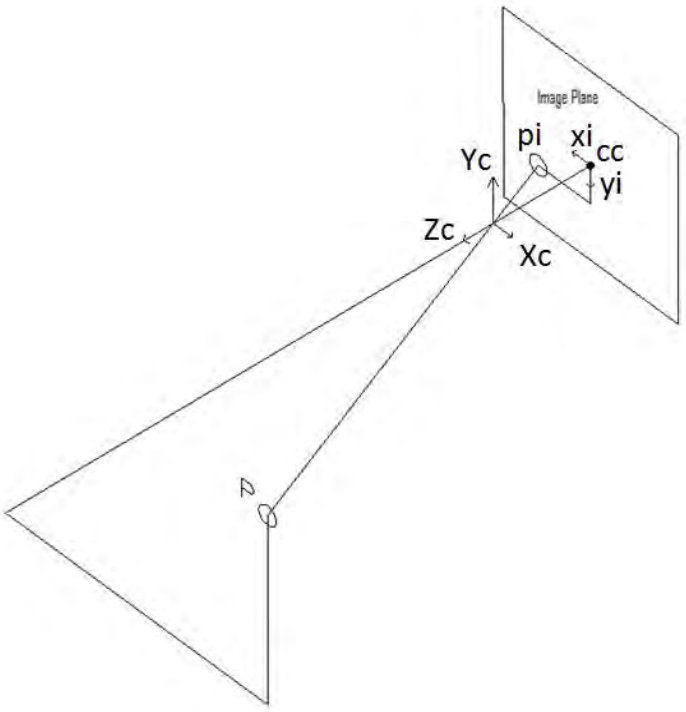

Figure 3: Pinhole camera model with a point P in the image and in space.

In Figure 3, the camera based frame of reference is marked by X_c, Y_c, and Z_c with the lens aimed in the direction of Z_c. The image has a two dimensional coordinate system x_i and y_i. These coordinates use a consistent set of units. This coordinate system is centered about the intersection of Z_c and the image plane designated cc. Any point along the line through the origin to P will be mapped to the same location on the image plane. This model does produce an inverted image, correcting the inversion is a trivial concern handled automatically within the camera. The location of a point in the three dimensional, rectilinear coordinates of the camera relates to the two dimensional image coordinates by Eq. (1) [7].

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = f \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \\ 1 \end{bmatrix} \quad (1)$$

Here f is the distance from the image plane to the origin of the lens model along the Z_c axis. This model makes several assumptions; all of the rays pass through an infinitesimal point, all of the rays are affected in the same way by the lens, and that the image is centered on the Z_c axis. It also assumes in this equation that the pixels are square. These assumptions prevent reliable measurements. Therefore, two forms of distortion are applied based on the design of lenses, decentering and radial distortion. Decentering accounts for the location of the principle point, cc. Radial distortion accounts for locations shifting as the point moves towards the edges of the image. Another frequent simplification is to set f equal to one and account for the scale information elsewhere in the equations developed below.

Radial distortion is modeled as moving radially from cc. An extreme example of radial distortion can be seen in Fig 4.

Figure 4: Vertical blinds photographed with a fisheye lens to show radial distortion.

In reality, the blinds are all straight. However in the image, further from the center of the image they begin to appear bent, an artifact of the lens. The potential for this distortion needs to be corrected for in any lens being used to ensure the measurements are accurate. This distortion can be fit by the following Eq. (2) [7].

$$[r_d] = (1 + k_{r1}r^2 + k_{r2}r^4 + k_{r3}r^6 + \cdots)\begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad (2)$$

where

$$r^2 = x_i^2 + y_i^2$$

In Eq. (2) r_d is the corrected radius measured from the principle point of the location that would have projected to x_i and y_i in an ideal pinhole system.

Decentering is an issue as the origin for processing purposes is frequently an extreme corner of the image. Another factor which can introduce decentering is misalignment of the lens in its housing or misalignment to the image plane. This further potential for misalignment prevents the application of a simple predetermined coordinate shift to the midpoint of the image. To account for this distortion, the model applies a tangential distortion which is modeled by Eq. (3)[15].

$$[dt] = \begin{bmatrix} 2k_{t1}x_i y_i + k_{t2}(3x_i^2 + y_i^2) \\ k_{t1}(x_i^2 + 3y_i^2) + 2k_{t2}x_i y_i \end{bmatrix} \quad (3)$$

In Eq(3). dt is the correction on the tangent of a location that would have projected to x_i and y_i in a ideal pinhole system. This handles the second form of distortion introduced by the lens. The two forms of distortion are summed together as Eq. (4).

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = [dt] + [r_d] \quad (4)$$

The lens also changes the size of the object as can be seen in Fig 2. The scale is first accounted for by the value of a measurement at the focal length of the lens and the corresponding

measurement on the image. The ratio of these two numbers gives us the scale factor, s, for the image. The scale factor can be found to be the ratio of f /f_c. Here f_c is the focal length of the lens in millimeters. Also, as these are digital images they typically make measurements in terms of pixels rather than millimeters or inches so this needs to be accounted for as well. Pixels may also be rectangular. Therefore, there can be two length scale conversions from millimeters or inches to pixels, one vertical and one horizontal. This leads to Eq. (5).

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} s * dpdx & 0 & cc_x \\ 0 & s * dpdy & cc_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} \quad (5)$$

Here $dpdx$ is the pixel count per scaled length in x, and similarly $dpdy$ is the pixel count per scaled length in y. The location of the principle point cc is described by (cc_x,cc_y). Now the values are shifted into a pixel coordinate system with an adjustable location of the principle point to account for the location of the origin which can be set arbitrarily by the user. As mentioned above, f is rolled into the scale factors.

These equations provide a method of determining the ray extending out from the camera that a point in space lies on from location in the image, with a consideration of the distortions. Once these rays have been established out of each camera the intersection marks the point as seen in Fig 5.

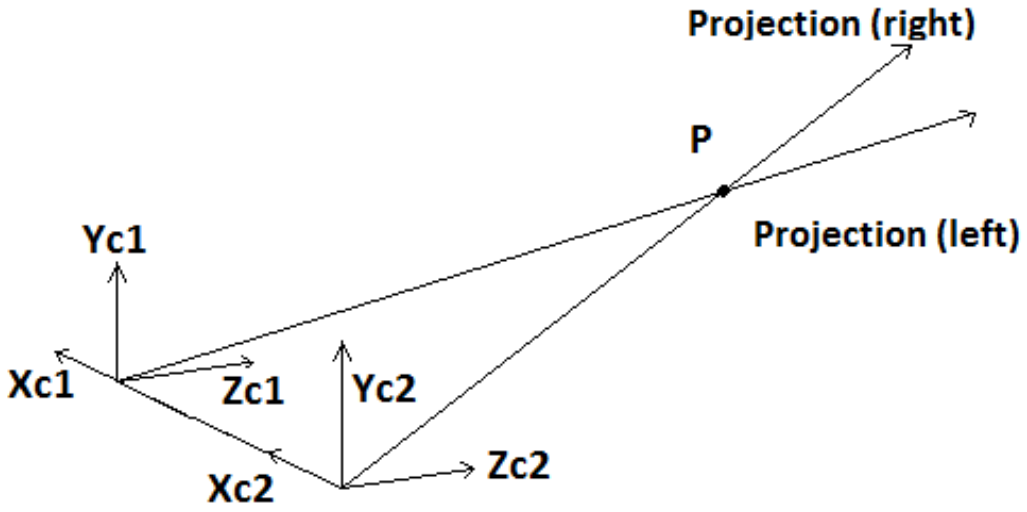

Figure 5: Two camera model with point P projections intersecting.

In Figure 5 each camera is oriented with the lens pointed along the respective Z axes. The projection lines are the lines described by Y/Z and X/Z in each camera's own coordinate system. The intersection marks the actual location of point P. Now the location of point P can be determined in rectilinear coordinates. To determine the three dimensional location the orientation of the two cameras to each other must be established. The camera coordinates are related to each other by Eq. (6).

$$\begin{bmatrix} X_{c1} \\ Y_{c1} \\ Z_{c1} \end{bmatrix} = \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta_y) & 0 & -\sin(\theta_y) \\ 0 & 1 & 0 \\ \sin(\theta_y) & 0 & \cos(\theta_y) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{bmatrix} \left(\begin{bmatrix} X_{c2} \\ Y_{c2} \\ Z_{c2} \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \right) \quad (6)$$

Here the T vector is the location of the origin of camera two from camera one. The angles of rotation, θ, are about the respective axis in order, x, then y, then z. The line segment defined between the second camera and the point, P, are transformed to the coordinates of the first camera. The intersection can then be solved for in the coordinate frame of camera one. This produces the three dimensional location of a point from two images taken at different locations.

Chapter 3 Experimental Setup and Program Application

3.1 Introduction to Setup

In this section the actual system setup will be described with important parameters identified and values used declared. The calibration procedure will be described. Next, the implementation of the automated digitization codes will be developed. Finally the implementation of the Stereo Vision Toolbox will be discussed. The following flowchart is a logical representation of the steps required to produce the three dimensional output, Figure 6.

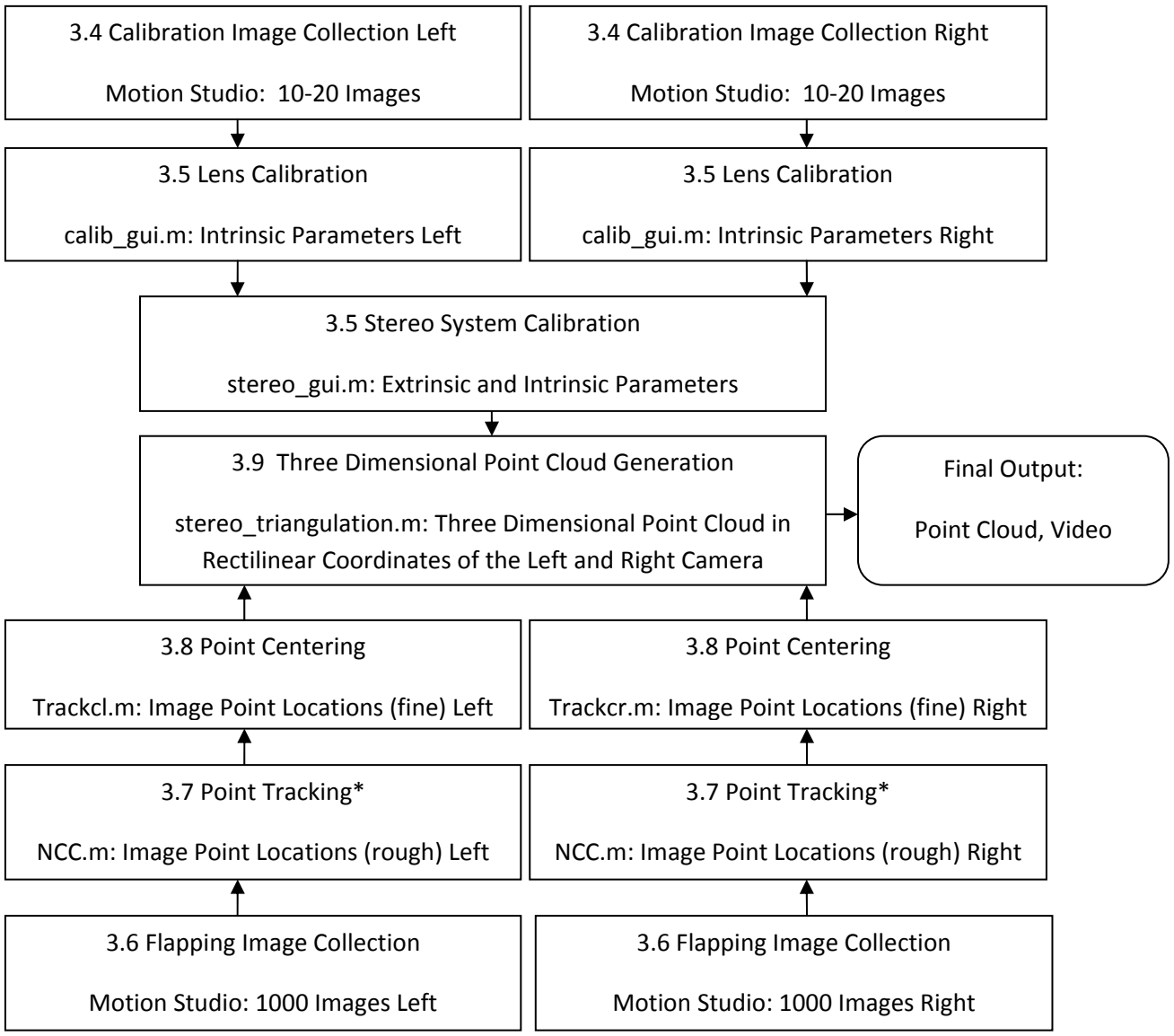

Figure 6: Flowchart of programs used to develop point cloud.

In Figure 6 the steps are listed in order for the particular data stream. The headings contain the section number and name from within this chapter. The "*" denotes a section which uses programs developed during this research. The name of the specific program, either the Motion Studio software, or the MATLAB script file, such as calib_gui.m, is listed followed by the corresponding output to the next program. These programs and steps are each discussed in detail below.

3.2 Wing Preparation

The wing must be marked for tracking. The Pentel White paint marker is used for the high flash off and minimum affect on testing as determined by DeLeón [1]. Flash off reduces the ink added to the wing and the weight added to the wing. The marking is shown in Fig. 7.

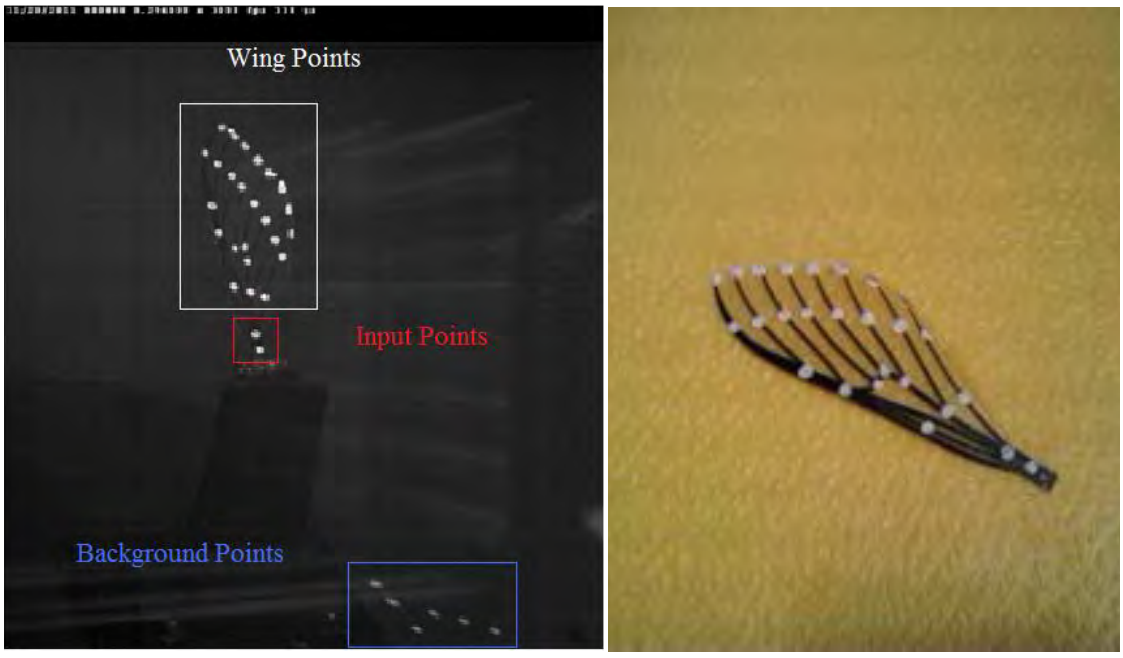

Figure 7: Marked wing mounted with background points and a marked wing with veins.

As can be seen in Figure 7 the veins are marked at the tips, the roots and at the midspan. There are also six points on the black background. These provide stationary points for a reference.

These reference points can be checked for movement at the end of the test. Any significant movement indicates an error, which makes that test invalid. Reference points may also be used to combine multiple tests to build a full stroke. There are also two points near the base that are marked to capture the input for modeling purposes. One point is directly over the flapper mount and the other is placed near the mount. The two points can be considered rigid to one another. These markers are tracked in the images. In order to facilitate two dimensional tracking, the points should not be identical to each other in size and shape. The centroids of these points will be tracked in the images to produce the three dimensional point locations.

3.3 Camera setup

The cameras used were X-Stream XS-4's from IDT. They come bundled with Motion Studio software which was used for data collection. These cameras are capable of 512x512 resolution at up to 5145fps. They use a complementary metal-oxide-semiconductor (CMOS) sensor [14]. The lens was a Nikon AF Nikkor 28mm. The aperture setting was key to allowing in a sufficient quantity of light to properly expose the image. The aperture on a lens determines the size of the hole that light can penetrate. It was also important to set it to allow the greatest depth of field or range of distances for which the images will appear focused. This is key as the wing will be flapping. These two parameters work against each other. The smaller the aperture, the larger the fstop number, the less light can pass through the lens to expose the image. However, a smaller aperture improves the focal characteristics of the lens increasing the depth of field. At an fstop of f/5.8 there is not sufficient light to produce more than a dark image as seen in Figure 8.

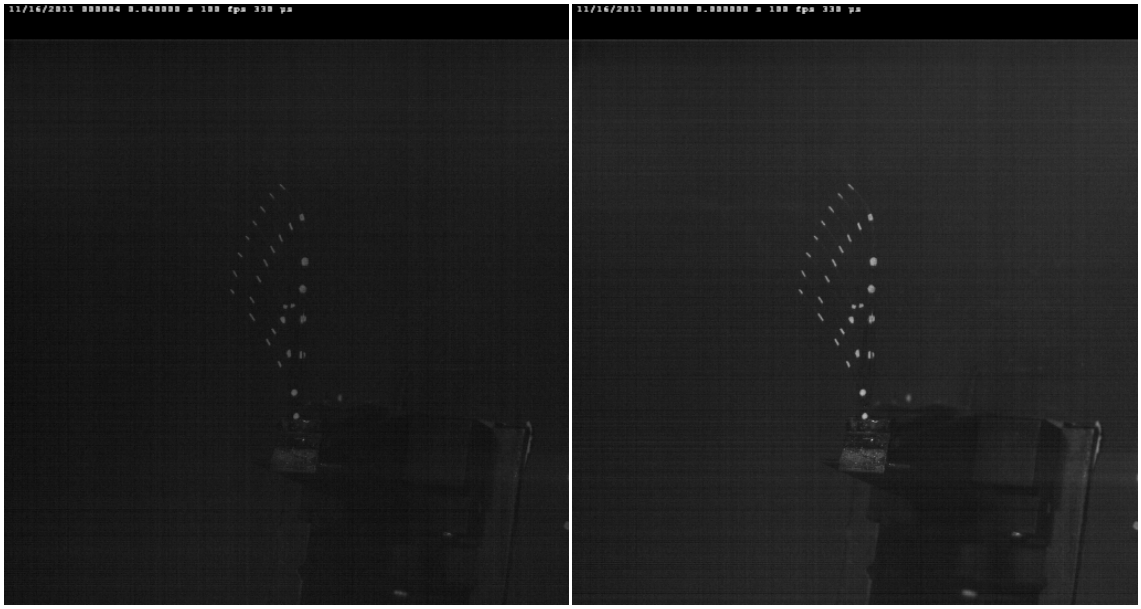

Figure 8: Left f/5.6, right f/4, note increased light and point visibility.

The aperture was set at f/5 for all tests to allow in enough light, and then maximized to increase the depth of field. The points are somewhat difficult to see in Figure 8 left and even more difficult to distinguish with an automated system.

Both cameras were mounted on a single tripod to fix the translation and rotation of the cameras as much as possible. This also provided more options for placing the lights which are aligned directly behind the cameras and just inside them, shown below. For this experiment two common mounted halogen work lights are used as seen in Figure 9.

Figure 9: Cameras on common mount with halogen lamps.

The lighting also required careful placement as the subject as well as the vacuum chamber walls were subject to significant glare, which could obscure the wing. The additional light from the work lamps are required to take data in excess of 1000fps. Above that, frame rate the exposure time is too short to expose the image with ambient room light. The camera tripod has a built in level which is used to keep the lenses level.

3.4 Calibration Image Collection

Calibration for the cameras is very important for producing accurate three dimensional point clouds. Camera calibration is similar to calibrating any other device; known inputs are applied and the output is adjusted to the expected result. Here the process more complicated due to the large number of parameters involved.

To perform the calibration, a grid of squares is held in the view of both cameras. For the stereo calibration both cameras must take the image at the same time. To facilitate this, the Motion Studio software and cameras have a feature where the cameras are synchronized. In Motion Studio there is an option to set one camera as a control, and the other camera's settings are automatically adjusted to match. On the hardware side a BNC connection is made from the SYNC OUT connection on the back of the master camera to the SYNC IN connection on the slave camera. The calibration images are set at 100fps, with the maximum available exposure of 9997μs. The frame rate is lower as the calibration grid does not move very quickly. The longer exposure time allows the images to be taken in ambient light, as the exposure time is not a parameter in stereophotogrammetry.

The calibration grid used for these experiments has 441 squares with side lengths of three millimeters. The greater number of squares provides more data points per orientation to ensure the calibration will be accurate. There is an assumption in the software that the points for calibration are all on a plane. To ensure the calibration is accurate the grid is mounted on a piece of foam core with stick glue or another soft glue that can be smoothed. During the mounting, a pair of plastic putty knives are used to smooth the grid to the foam core.

The calibration grid is moved in various orientations to the camera to establish the intrinsic parameters for photogrammetry as discussed in Chapter 2. To improve calibration ease and accuracy an automated system was built using a pan-tilt as shown in Figure 10.

Figure 10: Pan-tilt calibration tool with calibration grid.

The pan-tilt device pictured in Figure 10 rotates about a vertical axis, or pans. Then, it rotates about a horizontal axis, or tilts, through the grid. These rotations are controlled by two servos connected to the computer. They run a program, PanTilt.m, which reorients the grid each second through a pre-determined pattern. This allows the cameras ample time to collect the images required for calibration.

Once the cameras have finished recording, the video is reviewed to ensure that the capture was complete. Next, the user must select ten to twenty images to use for the calibration. The same frame number should be selected from each camera to ensure both images were captured at the same time. This is necessary for the stereo calibration. Proper calibration images should also have all the squares visible and in focus in both views. This can be difficult when the grid is tilted forward because the vertical frame of the pan-tilt device obscures a portion of the calibration grid, as in Figure 11.

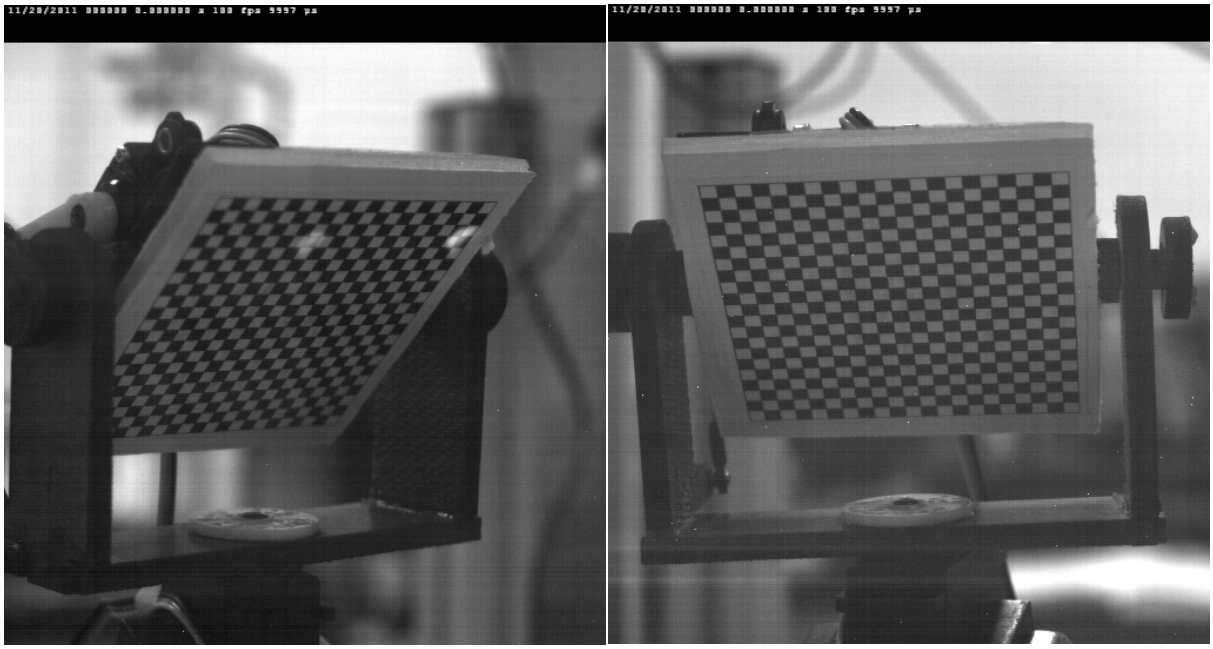

Figure 11: Left and right camera views with obscured calibration squares.

The program requires the same calibration grid area in each view. As seen above the left view has a reduced usable calibration grid area compared with the right camera view. The user can select a portion of the calibration grid in the image that is obscured by the frame. However, they will have to select the same grid area in both views for proper stereo calibration to take place. It is important to note that any discrepancy in the grid sizes will not become apparent until the stereo calibration is performed. Another issue occurs when the grid is tilted aft, the top rows can be beyond the focal depth as seen in Figure 12.

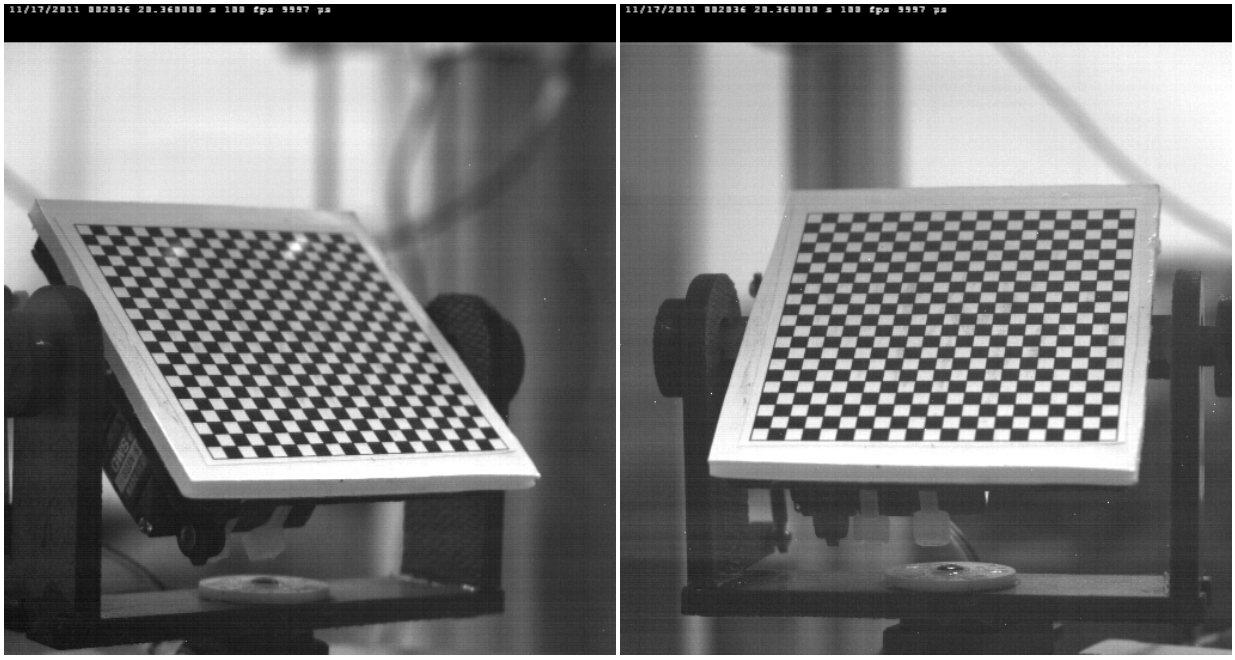

Figure 12: Left and Right views with upper row distortion.

In Figure. 12, the loss of focus leads to problems as the software selects data points automatically based on an expected pattern. The pattern can be broken by the blurriness of the upper right corner. When the images are a great deal out of focus they will not process correctly. The upper right corner of the calibration grid is out of focus in the left camera view. Once again this can be dealt with by selecting only a portion of the grid to be used for calibration so long as it is consistent from camera to camera. The selected portion of the calibration grid does not need to be consistent across different frame numbers.

3.5 Lens Calibration and Stereo System Calibration

The selected images are saved in a single folder separated by left and right camera. With the Stereo Vision Toolbox programs in the MATLAB path the user can simply type calib_gui into the MATLAB command line. This will call the calib_gui.m script file and execute the program[14]. Select the “Standard” option which stores all images in memory. Next, select

“Image Names” and enter the prefix of filenames for left or right images, without the numbers or file extension. This loads the images into the active memory. Now, the user needs to run the “Extract grid corners” procedure. This needs to be performed for all the images, but they can be done with subsets of the images if desired. However, generally it is best to extract all the images at once. The program will display the images one at a time for the user to select the outer four corners clockwise starting with the upper left corner. The corners furthest out that can be selected for calibration are the inside corners of the extreme corners as seen in Fig 13.

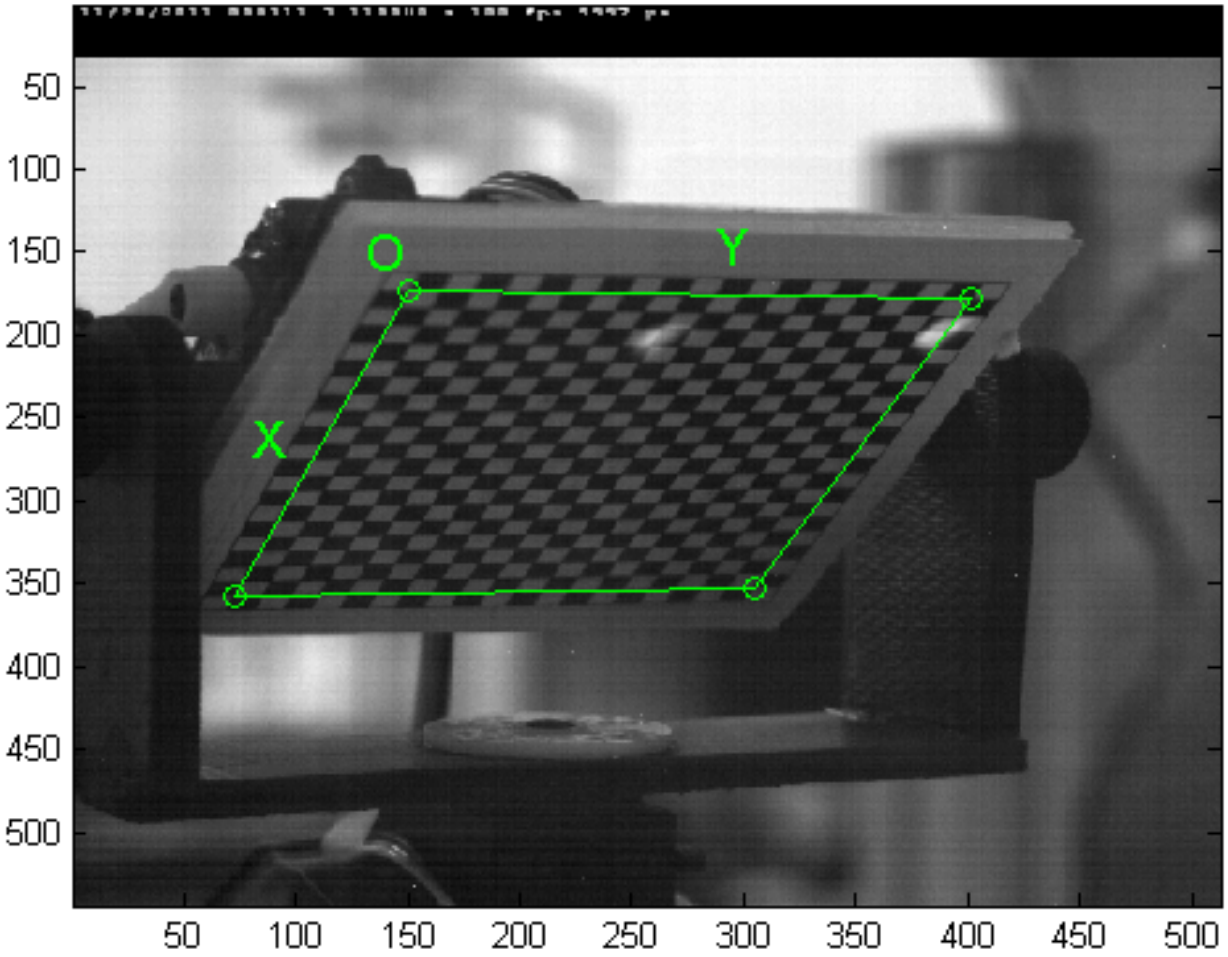

Figure 13. Calibration images with outermost corners marked (in pixels)

Once the corners have been selected the system will request the height and width of the calibration squares. These should be measured on the grid to ensure the printer did not resize the

grid entry. For each image the software will produce a figure with the internal corners marked as in Fig 14.

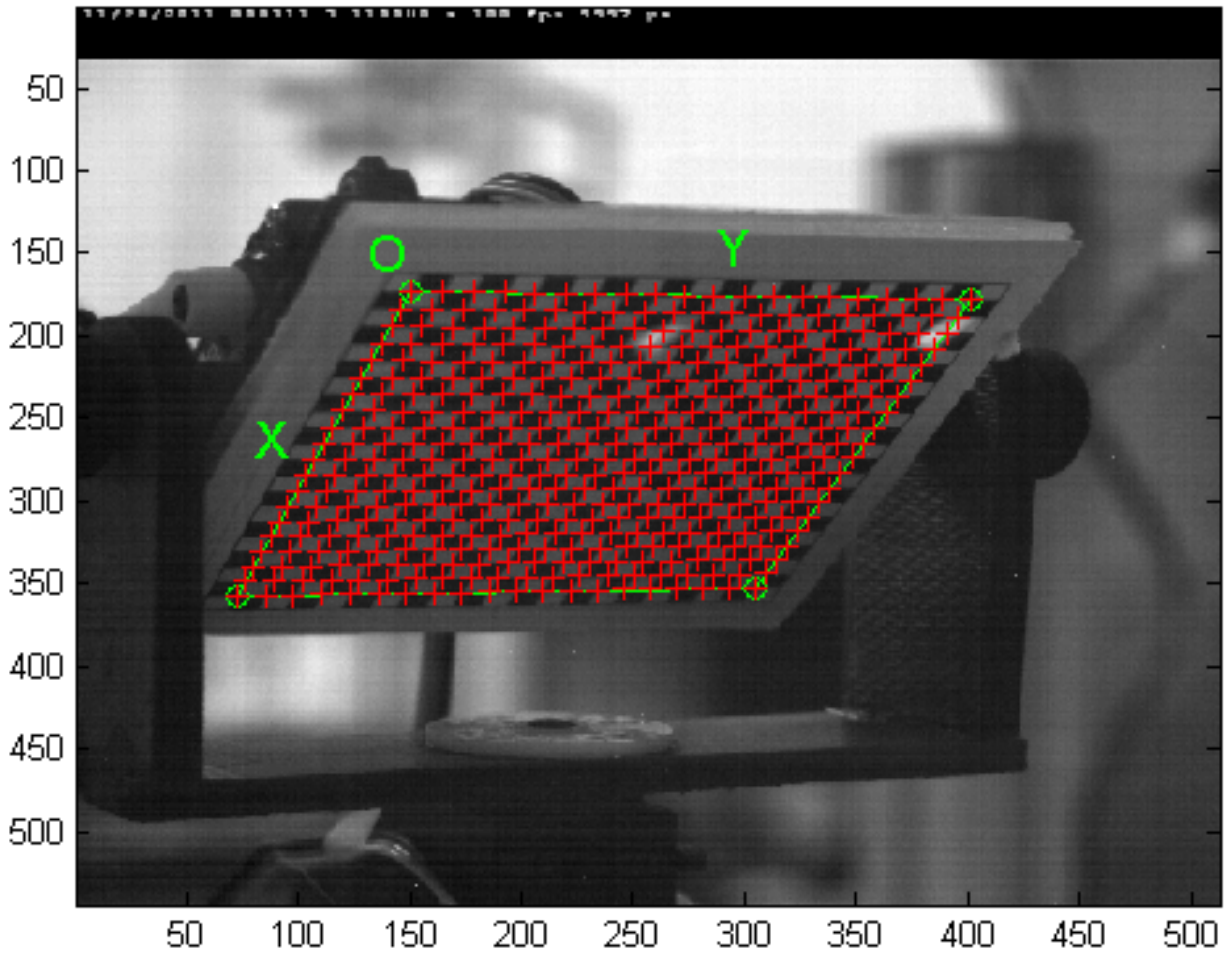

Figure 14: The calibration grid with internal points marked (in pixels).

The software asks the user to confirm that the image is marked properly. If the markers are off by more than a small amount the previous user inputs may be incorrect. Generally it is best to confirm the images as there are further options to correct the errors later. Once all the images have been corner extracted, select the "Calibration" option which determines all the intrinsic parameters of the lens, the lumped focal length scale terms, the radial distortion coefficients, and the decentering distortion coefficients. After this routine has been run it will output the lumped

focal length scale parameters, here fc(1) and fc(2) and the location of the principle point, cc(1) and cc(2). An example of this output is included below in Table 1.

Table 1: Example calibration screen output.

```
Focal Length:        fc = [ 1832.93623   1803.79723 ] ± [ 5.60423   6.20843 ]

Principal point:     cc = [ 174.76929   286.25155 ] ± [ 10.43443   7.97650 ]

Skew: alpha_c =[0.00000] ± [ 0.00000] => angle of pixel axes = 90.00000 ±
0.00000 degrees

Distortion:            kc = [ 0.14842   -9.08626   0.00126   -0.00350
0.00000 ] ± [ 0.06585   3.75282   0.00193   0.00238  0.00000 ]

Pixel error:          err = [ 0.15968   0.15918 ]This also includes error
bars on the fc and cc values.
```

The fc values here are the focal length in pixels in the x image coordinates, followed by the focal length in pixels in y image coordinates. The difference indicate the pixels may not be perfectly square as discussed in Chapter 2. The cc values are the x and y positions of the principle point measured from the upper left corner with y positive down. Note the skew is set to zero indicating the pixels have square corners. The kc vector represents the coefficients of the distortion equations. These are all followed by error bars in units of pixels. These errors are three times their standard deviation which is a conservative estimate. The important error value to take note of is the pixel error value. If this is less than one the calibration can be considered accurate for use. If the error bars are excessive, the user can select “Recomp. corners” using the default setting the software will adjust the marked points on the grid. Select “Calibration” again to update the values. Another option is to use the “Analyze Error” routine. This will plot the errors for all the points under consideration in the model, as in Figure 15.

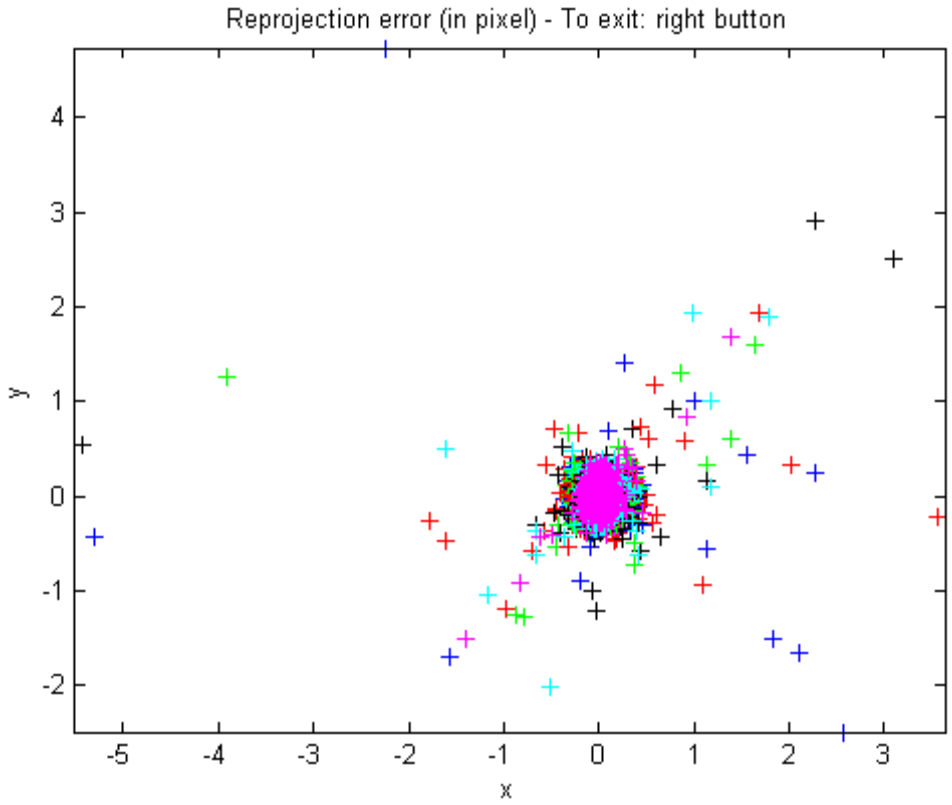

Figure 15: Selectable error plot from calib_gui.m (in pixels).

In Figure 15 the individual points for calibration are divided by image with colors. Each location on this chart represents the error at an individual point on an image. For extreme values the user may select the point by clicking on it and the program will print to the command line, valuable information such as: point number, image number, and error information. The most useful piece of information output is the image number. Images with extreme error points can be removed from the calibration with the "Add/Suppress Image" routine. Once these have been removed the user must repeat the "Calibration" with the new image list. Error values are calculated in pixels.

Once the calibration has been completed for the first camera, one can save the calibration using the "Save" routine. Rename the saved file by adding "_left" or "_right" as a suffix for the corresponding camera. The calibration procedure is repeated for the other camera. Exit calib_gui.m and enter stereo_gui in the command line[14]. Load the left and right calibration

files, and select “Stereo Calibration” routine. This will output a rotation vector and translation vector of the right camera coordinates to the left camera coordinate frame as seen in Table 2.

Table 2: Example output of Stereo Calibration

```
Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ -0.00564   0.72092  -0.00271 ]
± [ 0.03058   0.04279  0.01246 ]

Translation vector:          T = [ -281.80652   3.04792  96.95990 ]
± [ 4.70794   1.63099  8.40220 ]
```

Here the 3x1 rotation vector, om^T, is the eigenvector of the 3x3 rotation matrix, R, for an eigenvalue of one, shown in Eq. (8).

$$[R][om]^T = [om]^T \tag{8}$$

Since the rotation vector, om^T, does not change for the rotation it also describes the axis about which the coordinate rotation of the right camera occurs. The rotation matrix is the product of a one-two-three rotation. The translation vector is a standard position vector locating the right camera from the location of the left camera. Now, save the stereo calibration results for use later in section 3.9. With this step the system calibration is complete.

3.6 Flapping Image Collection

To collect the flapping wing data the calibration tool is removed and the O’Hara flapper is mounted in the vacuum chamber[15]. The O’Hara flapper utilizes a piezoelectric actuator to provide the drive force[15]. The flapper is shown in Figure 16.

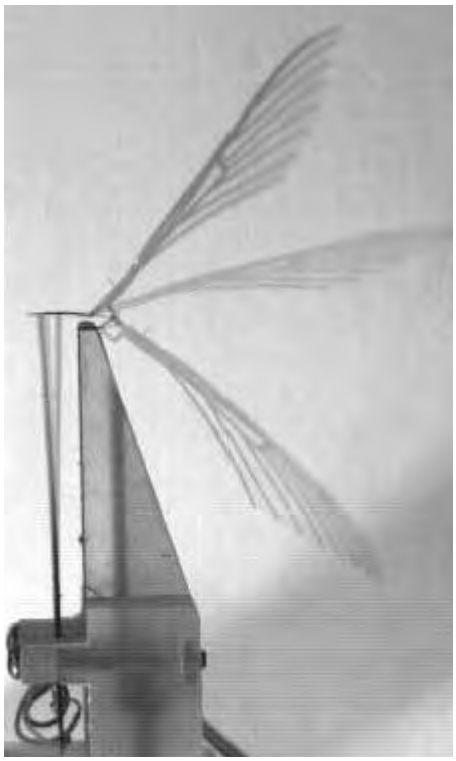

Figure 16: O'Hara flapper.

For collection of the flapping wing images, the cameras are reset to now capture 3000 fps and the exposure is set again to the maximum allowable time, 330 μs. The O'Hara flapper is connected to a MATLAB script and runs for less than one second to minimize wear and tear on parts[15]. This time limitation leads to synchronizing the cameras to the flapper. The flapper is connected to the computer through a National Instruments USB-6229, USB/BNC interface box. Then, the National Instruments box is connected to both cameras through their triggering input. The MATLAB script sends a high signal to the cameras at the time that the program sends the signals to flap the wing. In the Motion Studio software the cameras are set to record one thousand frames. They are also set to use external triggering in the record tab. The key settings for this feature are show in Fig 17.

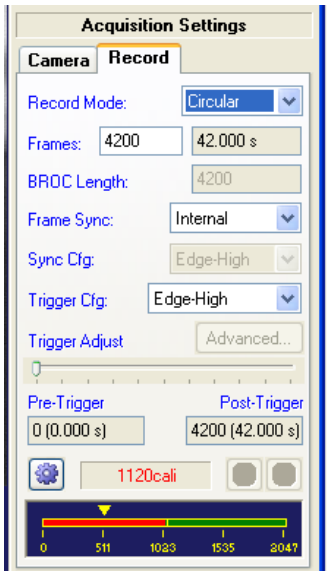

Figure 17: Record tab in Motion Studio to adjust triggering settings [13]

The key settings are: Record Mode, Frames, Frame Sync, and Trigger Cfg. The Record mode sets the program to record the full frame count once the camera has been triggered. The "Frames" setting dictates the number of frames to record this is followed by the length of the recording in seconds. The "Frame Sync" option determines how the cameras will be synchronized. "Internal" indicates the system will use the camera's internal clock. The "Trigger Cfg" sets what sort of signal the camera is expecting to set the trigger. In this case the camera is waiting for the signal voltage to increase when the recording is intended to start. Once the camera is triggered, it will record 4200 frames regardless of further changes to the trigger signal input.

Once the cameras are set within the frame work of the software, the user should adjust the placement of the flapper. The flapper should be centered in both camera views to keep the wing in frame during the entire stroke. This also keeps the wing inside the calibrated section of the images as it is impossible to calibrate to the edges of the frame. All of these adjustments should be made at the flapper, being careful to not move the cameras. Even the slightest

movement of the cameras could affect the focus and ruin the calibration. These two image capture steps can occur in either order so long as the cameras are not moved relative to each other and the lenses are left unchanged between the steps.

Once everything has been connected and the programs are setup, the MATLAB script file, BIAS_CHIRP_CAM_VOL_CUR_DISP_A.m[15] , which drives the flapper and triggers the cameras runs. The captured images are then reviewed to ensure that the markers stay in frame and that the flapper is running as expected. Of the captured images the user must select a range of the images over which points will be tracked and the three dimensional measurements will be taken. These should be taken after the wing stroke has reached a period of 1/30 of a second, or 100 frames to avoid transient effect from startup. Also the entire stroke range will not be visible to the camera. The first image used for this data and the image preceding it are pictured in Figure 18.

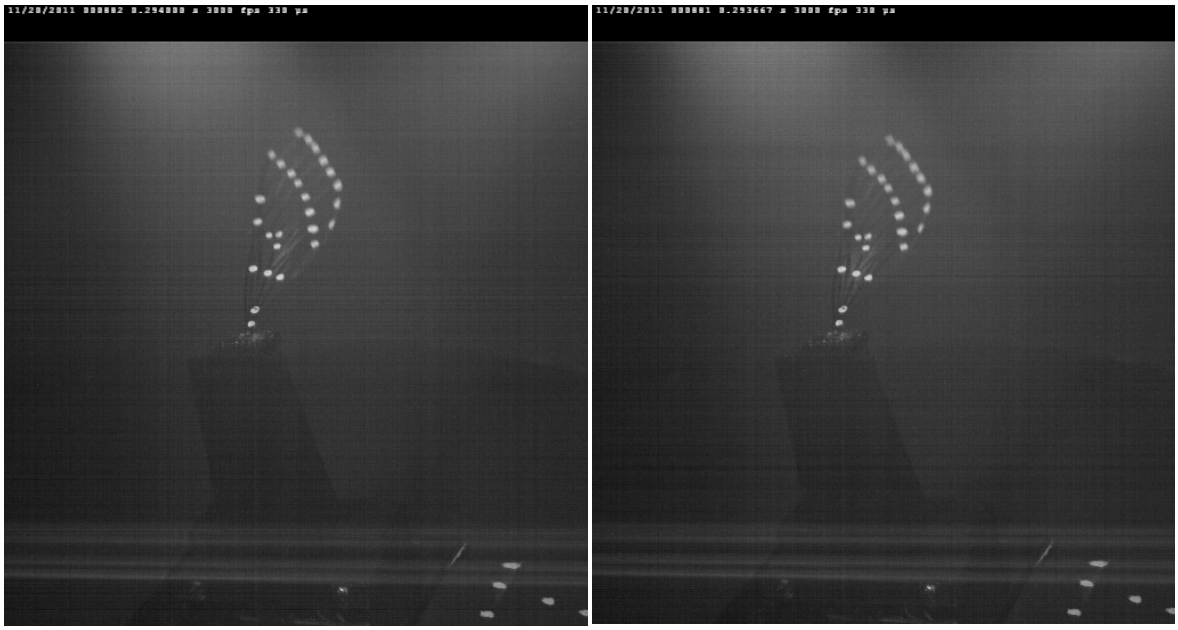

Figure 18: Left: the first image used in the sequence. Right: Preceding image.

In the image on the right several of the tip points run together making it difficult to distinguish them by eye. They will be even more difficult to distinguish with the software. Thus, the image

on the left is the first suitable image for tracking. The user selects a range generally a few strokes into the system to capture the flapper after transient effects are no longer present. The wing strokes 100 degrees which prevents capturing a full stroke in a single run. The user selects all the images from the first image to the next image where the points become unclear again. In this case the extra image is used as a throwaway point at the end of tracking. These images are used in the two dimensional tracking software to develop the digitized points for triangulation. Once again the same sequence needs to be selected from both cameras and the first and last images should have clear distinct points.

3.7 Tracking

Now using the image sequences selected in section 3.6 the user must locate the points of interest in the image frames. This is performed in two steps: the first locates each point to a pixel level accuracy and distinguishes the points from each other, the next step locates the centroid of each point to sub pixel accuracy.

As tracking was key in making the shift from still shots to video, a considerable amount of time was consumed developing this two dimensional tracking system. As an initial a single point was tracked under two dimensional motion, in a plane parallel to the image plane. Once successful out of plane motion was added which introduces the issues inheirent in capturing three dimensional movement in two dimensions. Namely tracking target deformations. With the addition of a correction function this was solved. Then a second point was added which occasionally lead to the points shifting between targets. With the addition of a limit on distance traveled between frames this effect could be limited. Finally, many points were tracked, by

tuning the distance limit and the updated points the system in NCC.m was complete for tracking flapping wing data for entry into the photogrammetry system.

The 18 wing tracking points, combined with the higher frame count made it important to automate the process. For the first step a process known as normalize cross correlation was used to locate each point to pixel level accuracy and to distinguish the points. Normalized cross correlation compares elements of two matrices element by element to determine their similarity. This ensures that the point at the end of the leading edge always has the same point number. These numbers also need to be correlated across views. To ensure this the pattern used in this research was leading edge to trailing edge and tip to root. This is important, as out of order points will cause incorrect point locations in the final three dimensional point cloud.

To accomplish this, normalized cross correlation was used to locate the key points in subsequent frame and to mark their location in the image. In this case the user selects each point to be tracked as seen in Figure 19.

Figure 19: Initial subimage selection for normalized cross correlation.

Figure 18 contains two views of the same subimage selection, the left view is simply zoomed. This feature is available to the user during selection and can be invaluable in centering the marker in the box. The program then compares the matrix defined by the box, or subimage, against all the possible matrices in the subsequent images that it can overlay evenly. This produces a matrix of correlation coefficients, where the highest value marks the upper left corner of the subimage where the selected point is located. The location of the midpoint of the selected subimage relative to the upper left corner is added to the new upper left corner location to locate the point. This process can produce erroneous results so a check that the point has not shifted too far between frames, usually 8-10 pixels in any direction, is used. If the point is located too far from its previous position the user is prompted to reselect the point. When this check fails, it produces a figure with the previous correct point marked by a red cross and the last incorrect location as in Figure 20.

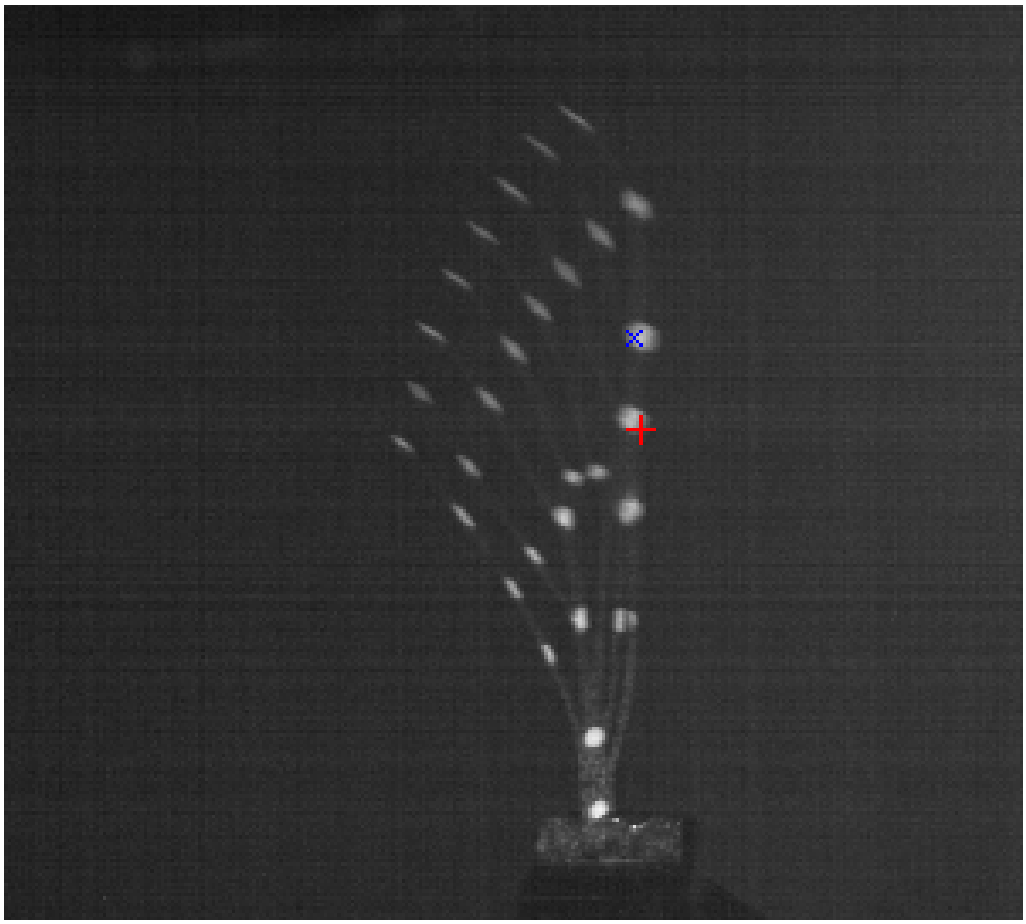

Figure 20: Example of the correction figure generated when normalized cross correlation fails.

It is also important to point out that having other tracking points in the sub image can aid in keeping the points distinguishable. The somewhat generic round points can cause the program to shift if additional points are not in the subimage to provide further reference. Once all the images have processed the last value is dropped as the shift check is not performed on the final data point.

3.8 Point Centering

Now this data is input and the image sequence are fed into another pair of programs, trackcl.m or trackcr.m for left and right images respectively, to refine the location of the tracking points to subpixel accuracy. This uses script files from Georgetown's "Particle location and tracking tutorial" for MATLAB, bpass.m and cntrd.m [10]. The first is an image filter which cuts off the background below a threshold and smoothes the values over the marked points in the images, example output shown below Figure 21.

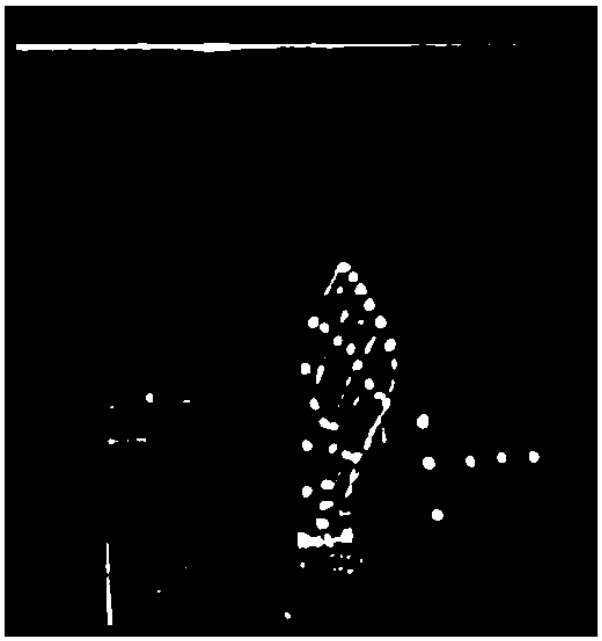

Figure 21: Band passed image to prepare for centroid finder.

Note the marked points are highlighted and the background is subtracted off. The newly filtered image and the location of a pixel level point are then input to cntrd.m which calculates the centroid of the marked point [10]. The band pass creates a cut edge for the location of the marked point. If the NCC point falls on the white dot then the centroid of the round white area is calculated. This centroid is recorded as the center of the marker for use in the next step. The program loops over all the points and records all the relevant points. If the output from NCC.m

does not fall within a white mark the system will register the background and not record the erroneous point. The data are formatted for the stereo_triangulation.m from the Stereo Vision Toolbox and are saved[14].

3.9 Three Dimensional Point Cloud Generation

To build the three dimensional point cloud, load the Stereo_calib_results.mat and the centered points locations from the previous section. In the Stereo Vision Toolbox [14], stereo_triangulation.m is built as a function, and will only one marker point at a time for the sequence being considered. Thus, a script was built to call stereo_triangulation.m and loop through each marker point[14]. The two inputs from the left and right image are processed through the calibrated system as discussed in chapter 2, which builds the three dimensional location. The final output from this system is the location of all the points in the coordinate system of the left camera and a set in the coordinate system of the right camera coordinate system.

Chapter 4: Results and Discussion

4.1 Introduction to Results

During the course of this research four major tests were performed from beginning to end to verify the system performs as intended and is applicable to flapping wing MAV testing. The tests performed were a rigid body under motion using a printed target on the pan-tilt device, a calibration grid test, a leading edge tracking of a wing without the Mylar film, and a final test of a wing with Mylar film characterizing the wing reversal at the top of the stroke. The rigid body test was intended to test specifics of calibration as well as the tracking limits on NCC.m. In terms of calibration the test was to determine whether the vacuum chamber wall introduced any significant distortion into the model, particularly under medium vacuum, 2 torr. Normalized cross correlation has a tendency to fail to locate the marker in subsequent frames when the test marker moves out of plane which distorts the markers shape. The second test was used to determine how sensitive calibration is to grid size selection, as there was a lack of guidance within documentation of the Stereo Vision Toolbox on grid sizing. The third test was used to test the limits of the system to handle wing stroke range. The final test was to produce a portion of the desired of data, a flapping wing recorded at 3000fps, while flapping at 30Hz.

4.2 Rigid Body Test

This was performed using a pair of coded targets mounted on the calibration pan-tilt tool, in air with the chamber open, in air with the chamber closed, and at 2 torr with the chamber closed. The Plexiglas vacuum chamber walls are known to flex under vacuum from a previous experiment performed with the Polytec Scanning Laser Vibrometer. In that experiment when the chamber was brought under vacuum, the laser array had to be adjusted from the aim prior to

vacuum, because of the distortion of the walls of the chamber affected the path of the lasers. Thus, there was some concern that stereophotogrammetry testing would require calibration under exact test conditions to produce meaningful results.

For the rigid body under consideration, a planar test object was printed off similar to the calibration grids. The coded targets are shown below mounted on the pan-tilt Figure 22.

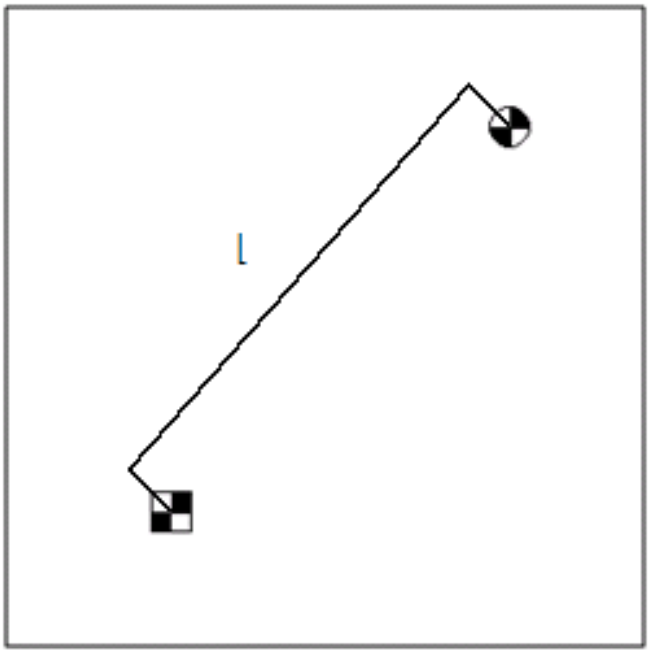

Figure 22: Coded targets drawing with length “L”.

The targets are a circle and a square with a cross on them. The value used for comparison was the length between the center points of the square to the circle denoted by the L in Figure 21. The distinct shape provides the normalized cross correlation function with distinctly different targets. The cross hair provides a centering feature to normalized cross correlation. Thus, preventing the need to run the point centering function discussed in section 3.8. The difficulty in applying an effective crosshair and sufficient number of distinct shapes hand drawn on the wing prevent the use of normalized cross correlation alone to locate points. Also the use of only

normalized cross correlation produces tracking of points accurate to a pixel which is ultimately not ideal as the shift induced can be severe.

To perform this test a standard calibration was performed in open air, without the chamber door between the camera and the calibration grid. It should be noted this calibration was performed with a coarser grid than was used in later tests, as it preceded the calibration sensitivity testing. Once the calibration images were collected the rigid body image was affixed over the calibration grid. Open air images were collected next, followed by the closed chamber images, followed finally by the vacuum images. Between each test movement of the cameras was avoided and after initial setting the focus rings were taped down to prevent changes to the lens parameters. All images were captured using standard calibration camera settings, 100fps, 9997μs exposure for 4200 frames.

Once the images were captured a standard calibration was performed which produced the results in the following Tables 3-5.

Table 3: Left camera calibration for rigid body test.

```
Focal Length:      fc = [ 1829.89506   1816.11601 ] ± [ 33.52322   39.87115 ]

Principal point:     cc = [ 163.95846   316.98437 ] ± [ 76.91556   54.90092 ]

Skew: alpha_c = [ 0.00000 ] ± [ 0.00000]   => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:    kc = [ -0.18328   0.33282   -0.00387   -0.00940  0.00000 ]
± [ 0.35564   4.55324   0.01620   0.01954  0.00000 ]

Pixel error:          err = [ 0.26072   0.22603 ]
```

Table 4: Right camera calibration for rigid body test.

```
Focal Length:      fc = [ 1825.25605   1822.47898 ] ± [ 32.30069   34.88583 ]

Principal point:     cc = [ 259.95541   299.06578 ] ± [ 22.25418   34.90884 ]

Skew:  alpha_c = [ 0.00000 ] ± [ 0.00000]   => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:      kc = [ 0.87731   -49.32273   0.00783   0.00499  0.00000 ]
± [ 0.52000   45.00693   0.01689   0.00810  0.00000 ]

Pixel error:          err = [ 0.24994   0.23810]
```

Table 5: Stereo camera calibration for rigid body test.

```
Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ -0.00564   0.72092  -0.00271 ]
± [ 0.03058   0.04279  0.01246 ]

Translation vector:          T = [ -281.80652   3.04792  96.95990 ]
± [ 4.70794   1.63099  8.40220 ]
```

In Tables 3 and 4, the pixel error is once again less than one. The increased error on the focal length and principle points are artifacts of the coarse calibration grid. These will be shown to improve with a finer calibration grids in the next experiment.

The images of the rigid body moving on the calibration grid were tracked next using NCC.m to locate the center of the images to a pixel level accuracy. The left and right images from frames with an extreme angle are shown below for each test case in Figure 23-25.

Figure 23: Open air tracked rigid body, coded targets test.

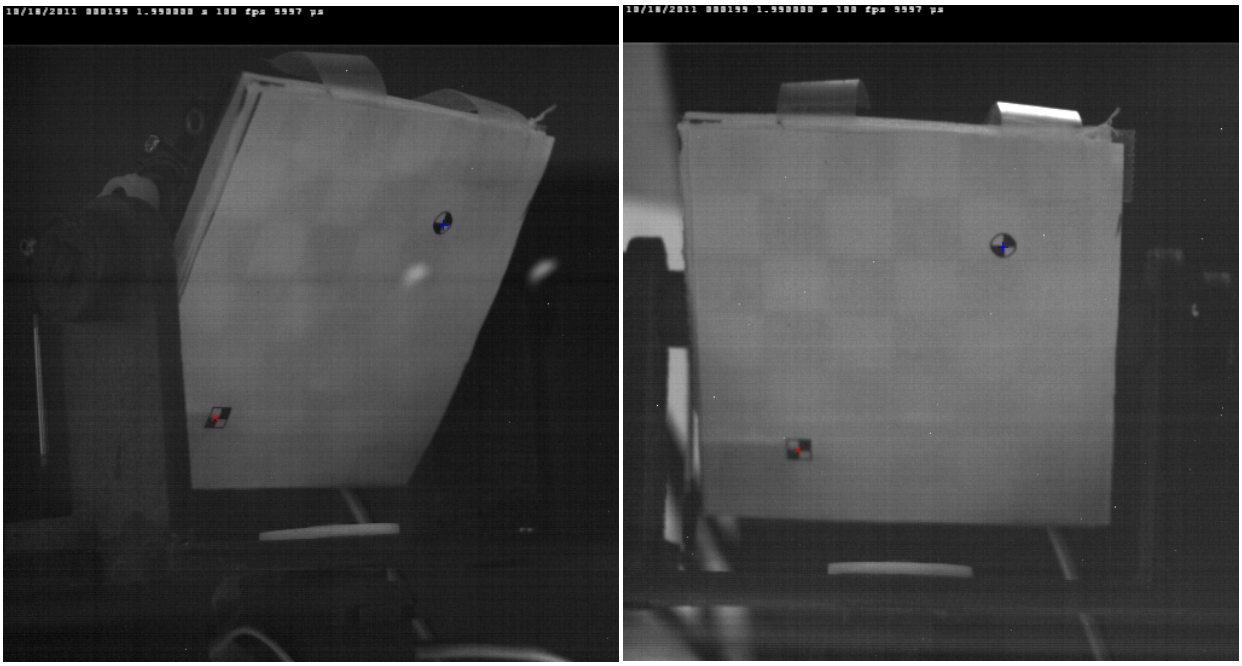

Figure 24: Closed chamber tracked rigid body, 750 torr.

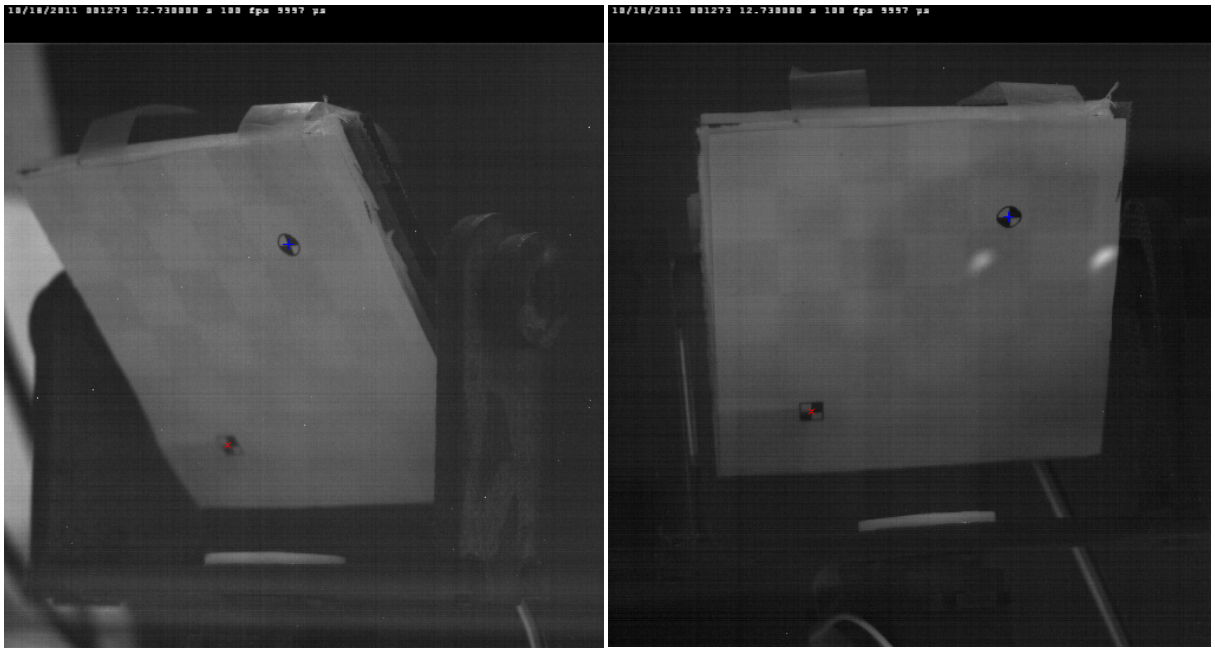

Figure 25: Vacuum test, rigid body at 2 torr.

These values demonstrate the degree of out of plane shift the new tracking system can handle. They also show the limits of the system in terms of accuracy. The square is marked above by the red x and the circle is marked by the blue +. Note they are not perfectly centered over the crosshair, this is due to the pixel level accuracy of normalized cross correlation which may induce a slight shift. These data location were input to the stereo_triangulation.m, described in section 3.10 of the flow chart along with the calibration data generated above. The program then triangulated the position of the square and circle in the three dimensional rectilinear coordinates centered at the left camera. These output use units of millimeters. The three dimensional plots generated for each are shown below as Figure 26.

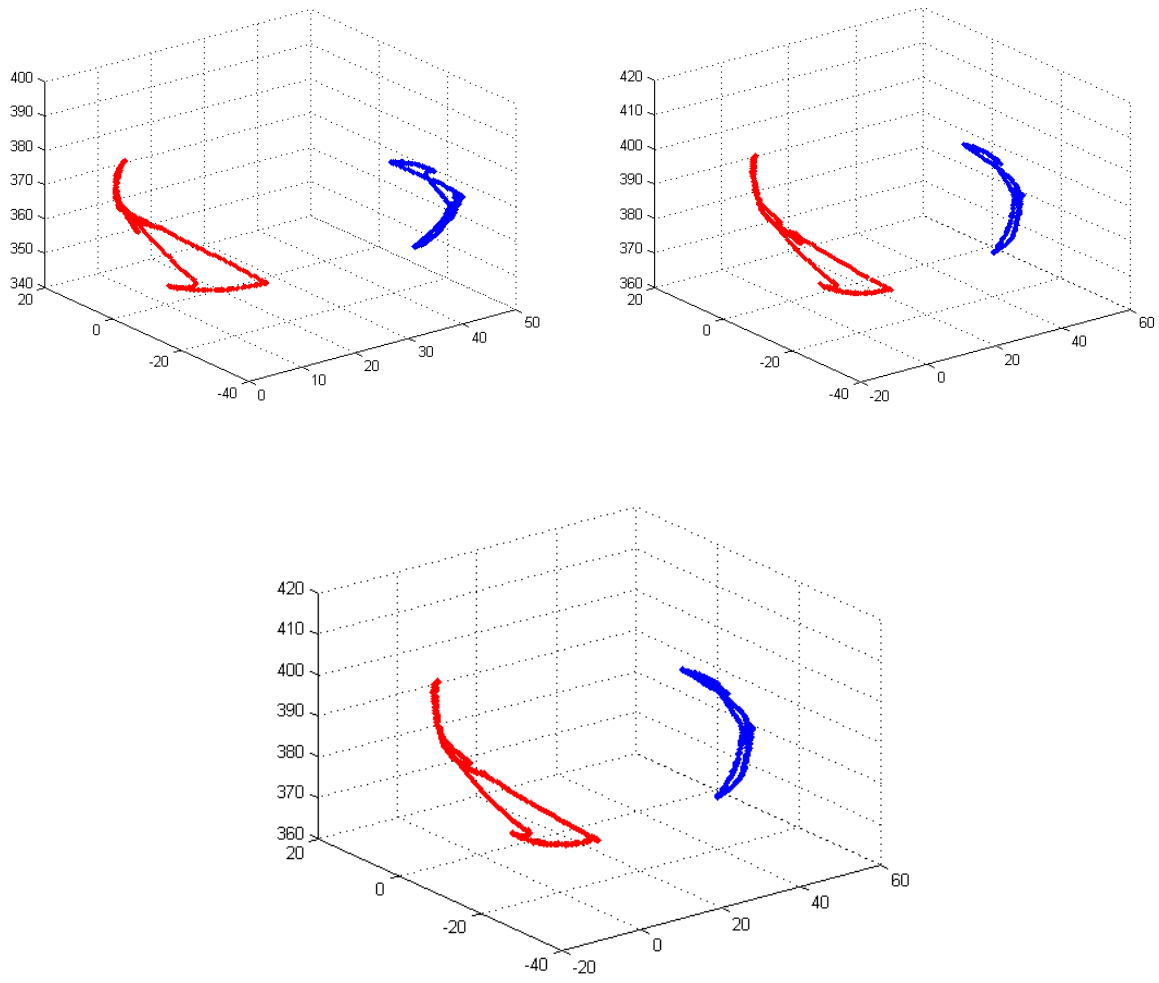

Figure 26: Three dimensional plot for the open air (upper left), closed chamber (upper right) and vacuum (2 torr) (in pixels)

Here the red represents the path of the square and the blue represents the path of the circle. It should also be noted that due to the pinhole model the output is flipped vertically, across the horizontal axis.

Further testing was performed on these points to determine how the three dimensional length between them varied over the length of the test. The length should remain constant as the points are undergoing rigid body motion. The three dimensional length was determined using the root sum of squares method in Eq.(9).

$$L = \sqrt{(x_s - x_c)^2 + (y_s - y_c)^2 + (z_s - z_c)^2} \quad (9)$$

In Eq. (8) the subscript "s" indicates the location of the square described by x, y, and z and the subscript "c" indicates the coordinate location of the circle. This equation was repeated for each frame in each case. The lengths over the frames are displayed below as Figure 27.

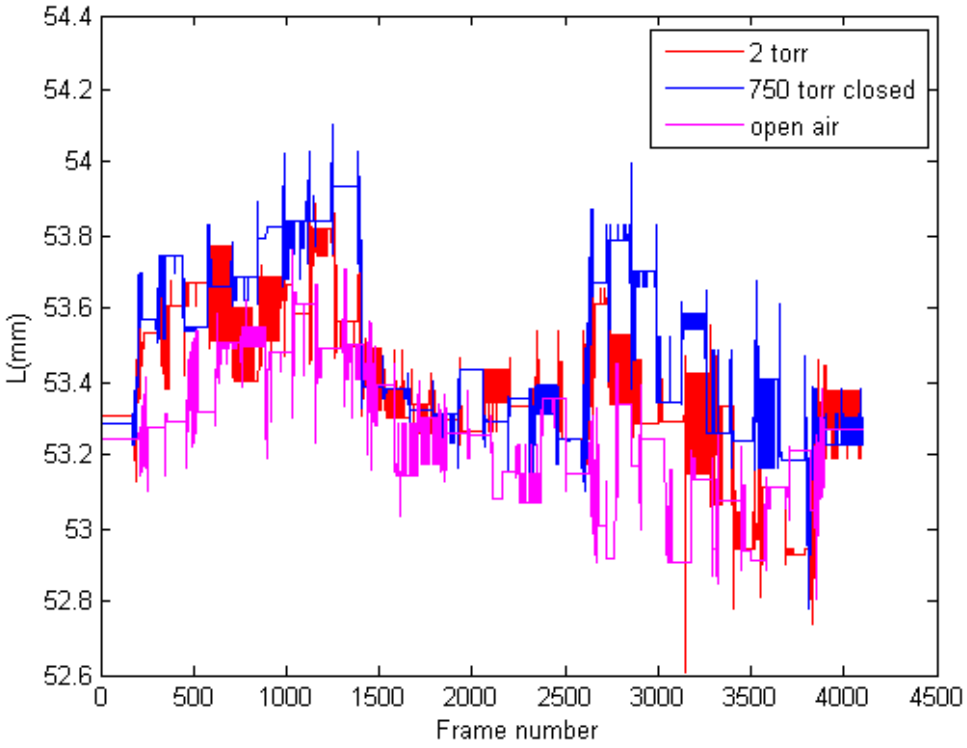

Figure 27: The length vs. frame for the open case, the closed air case, and the vacuum case.

As can be seen in the figures all three shifted some over the frames. However, they all remain around the measured distance of 53.2mm. For further comparison the maximum length, minimum length and the standard deviation of the lengths are displayed in Table 6.

Table 6: Statistics for the length L in the tested cases.

Test	Max L(mm)	Min L(mm)	Mean L(mm)	Std. Dev. (mm)
Open Air	53.9952	52.8223	53.2548	0.1636
Closed Air (750 torr)	54.1026	52.7796	53.4792	0.2270
Vacuum (2 torr)	54.0264	52.6063	53.3806	0.2014

Here it can be seen that all cases measured around the 53 mm and the standard deviations are low. This indicates that calibration is not largely affected by the Plexiglas bowing under pressure when the vacuum pump is on. This is a reasonable result as the cameras are not using coherent light as is the case with the scanning laser vibrometer. This means the vacuum chamber condition during calibration image collection does not necessarily need to match the vacuum chamber conditions during flapping wing image collection.

4.3 Calibration Grid Comparison

There was a shortage of information regarding recommended calibration grids. The previous test was performed using a truncated grid supplied with the Stereo Vision Toolbox. The squares measured 10 mm. Three finer calibration grids were tested using squares with side lengths of 7mm, 4.5mm and 3mm. These are displayed in Figure 28 below.

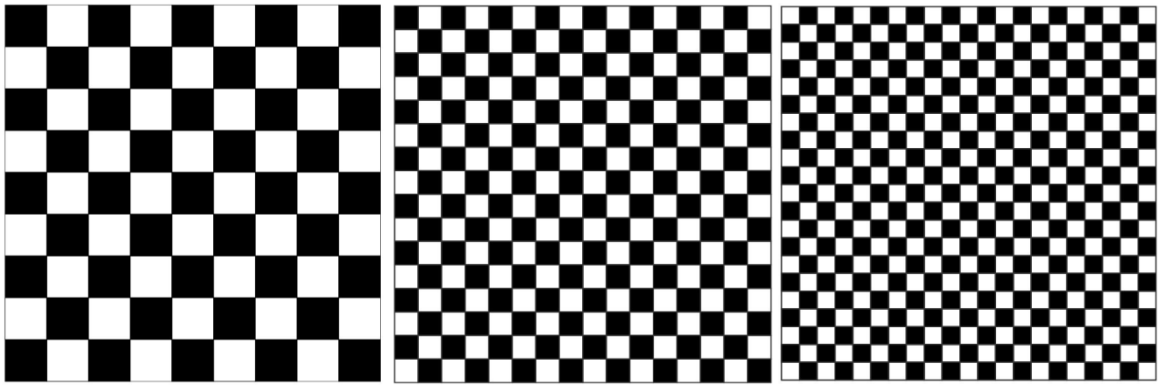

Figure 28: Calibration grids from left to right 7mm, 4.5mm, and 3 mm.

These used the same overall grid size of 2.5 inches. Thus, a finer grid provides more points to determine intrinsic camera parameters. The calibration outputs for the test grids are listed in the tables 7-9.

Table 7: Calibration outputs left camera for 7mm calibration grid.

```
Focal Length:      fc = [ 1944.09263   1918.96670 ] ± [ 24.55654   20.12724 ]

Principal point:     cc = [ 487.30161   255.91294 ] ± [ 39.50286   16.46281 ]

Skew: alpha_c = [ 0.00000 ] ± [ 0.00000  ]  => angle of pixel axes = 90.00000
± 0.00000 degrees
Distortion:         kc = [ -0.28164   2.67154   -0.00250   0.02976  0.00000 ]
± [ 0.17450   1.30196   0.00135   0.01665  0.00000 ]

Pixel error:          err = [ 0.15387   0.17915 ]
```

Table 8: Calibration outputs left and right for 4.5mm calibration grid.

```
Focal Length:      fc = [ 1858.70075   1860.48207 ] ± [ 12.97954   13.59486 ]

Principal point:     cc = [ 236.32464   408.37714 ] ± [ 12.46572   18.19379 ]

Skew: alpha_c = [ 0.00000 ] ± [ 0.00000  ]  => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:         kc = [ -0.00000   -1.57548   0.01958   0.00205  0.00000 ]
± [ 0.00000   1.03498   0.00436   0.00174  0.00000 ]

Pixel error:          err = [ 0.15823   0.25594 ]
```

Table 9: Calibration outputs left and right for 3mm calibration grid.

```
Focal Length:        fc = [ 1938.14436   1887.70878 ] ± [ 9.25062   7.93624 ]

Principal point:       cc = [ 413.45587   245.33176 ] ± [ 9.88625   4.24255 ]

Skew: alpha_c = [ 0.00000 ] ± [ 0.00000  ]  => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:        kc = [ 0.29752   -3.20161   -0.00629   0.06053  0.00000 ]
± [ 0.05678   0.77473   0.00058   0.00399  0.00000 ]

Pixel error:          err = [ 0.13945   0.13823 ]
```

As the grid becomes finer the error bars associated with the principle point and the focal length shrink, which indicates a better fit. Thus, a finer grid is needed for accurate calibration of the lenses. The pixel error also minimizes with the finest grid size.

4.4 Leading Edge Tracking Without Mylar Film

The initial intent was to track a flapping wing through its whole stroke. However, the Mylar coating tended to produce a glare that obscures the tracking markers. An example of the glare is in the following Figure 29.

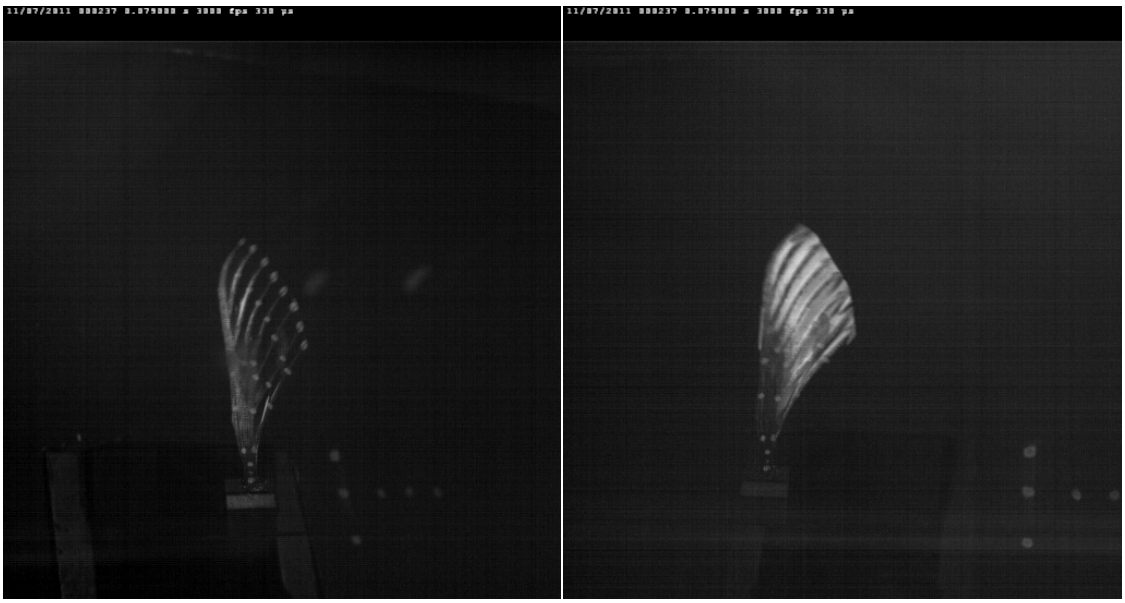

Figure 29: Images with glare obscuring the markers.

As can be seen in the above images it is difficult to distinguish the tracking points by eye. This makes it impossible to track them automatically. In order to test the system without the glare so that one can determine if there are other issues, a wing was flapped without the Mylar film. As this was intended to be a quick test, only the leading edge of the wing was tracked. The calibration was performed as described in Chapter 3. The calibration output is listed below Tables 10-12.

Table 10: The left calibration output for no Mylar tests.

```
Focal Length:        fc = [ 1856.90162   1826.98820 ] ± [ 5.44163   5.46507 ]

Principal point:       cc = [ 219.47333   273.55310 ] ± [ 5.64123   6.98456 ]

Skew:  alpha_c = [ 0.00000 ] ± [ 0.00000]   => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:       kc = [ -0.40754   30.41544   -0.00671   0.00193  0.00000 ]
± [ 0.09045   8.25989   0.00126   0.00116  0.00000 ]

Pixel error:          err = [ 0.14672   0.15003 ]
```

Table 11: The right calibration output for no Mylar tests.

```
Focal Length:        fc = [ 1865.66380   1853.38961 ] ± [ 4.77087   4.71813 ]

Principal point:       cc = [ 205.00636   316.46705 ] ± [ 8.19910   6.61466 ]

Skew: alpha_c = [ 0.00000 ] ± [ 0.00000  ]  => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:         kc = [ 0.20859   -6.52705   0.00309   -0.02198  0.00000 ]
± [ 0.05079   2.35492   0.00150   0.00225  0.00000 ]

Pixel error:          err = [ 0.12243   0.12580 ]
```

Table 12: The stereo calibration output for no Mylar tests.

```
Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ 0.03216   0.78600  -0.00154 ]
Translation vector:           T = [ -272.02342   -0.64709  118.07268 ]
```

The wing was captured for 1000 frames and tracked for 18 frames for the upstroke and 25 frames for the downstroke. Sample normalized cross correlation marked the points to pixel level accuracy using NCC.m, are shown below, Figure 30.

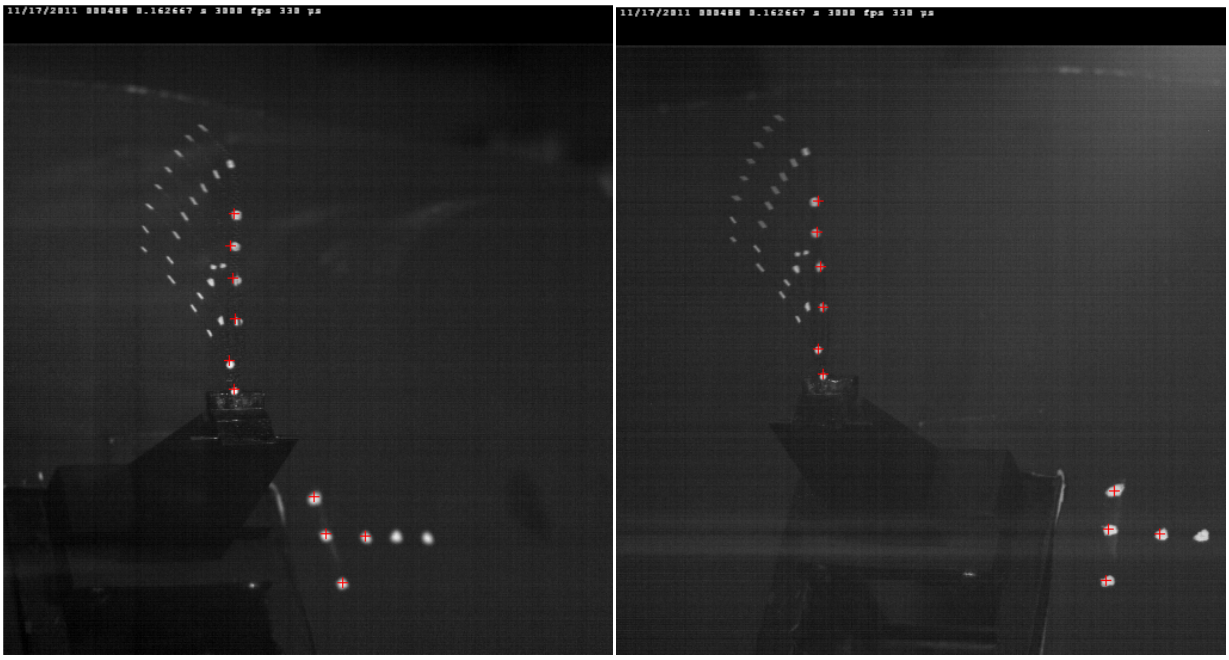

Figure 30: Normalized cross correlation for leading edge test.

The smaller tracking points lead to a problem maintaining tracked points over the light images. The tip points could not be tracked successfully. This can cause dropped points when the tracked points are centered using trackcl.m and trackcr.m. These points with the same images are processed through trackcl.m and trackcr.m to improve the marker locations for input to stereo_triangulation.m[14]. Examples of this improved data are displayed below in Figure 31.

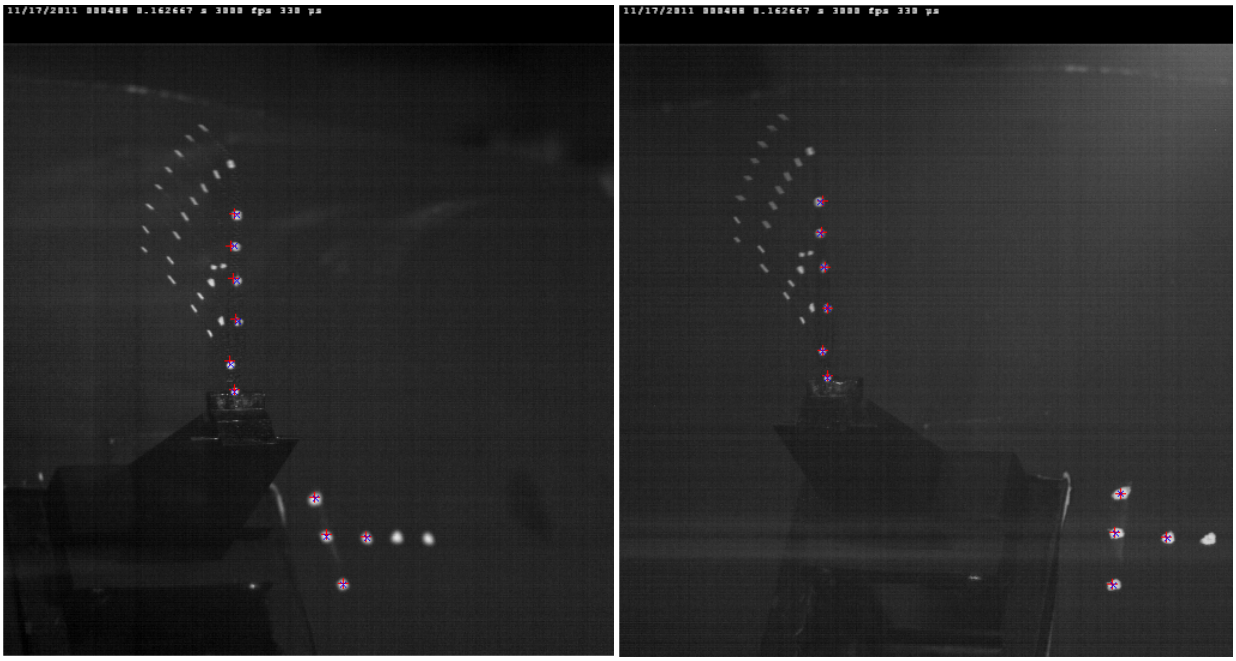

Figure 31: Centered data with normalized cross correlation data for reference.

The normalized cross correlation data are plotted with the red +'s and the centered data are plotted using the blue x's. Note the centered data better represent the center point of the target marker. The centered data are entered into the stereo_triangulation.m function with the calibration data to build the three dimensional point cloud. The three dimensional output displayed below with the points differentiated by color, Figure 32.

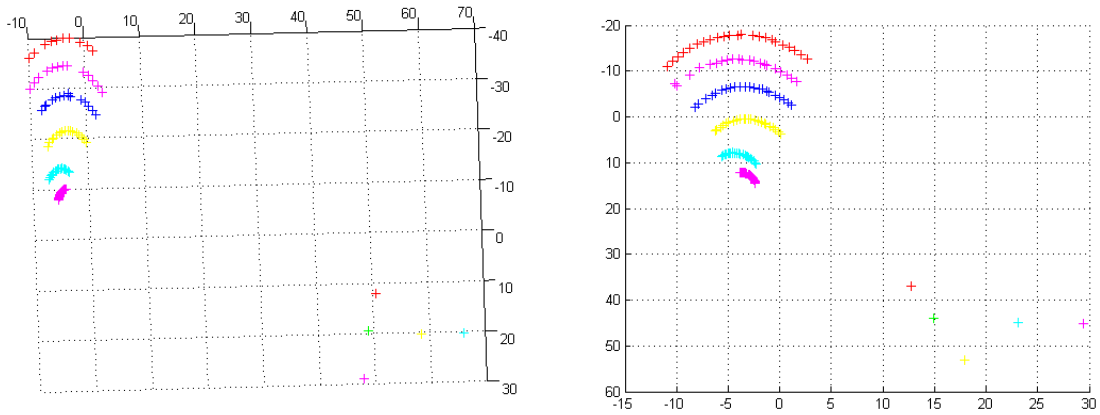

Figure 32: The three dimensional plot of the upstroke and downstroke, respectively without film (in pixels)

These outputs are in millimeters with the origin at the center of the left camera. During these tests, there was still glare off the carbon fiber, although it did not obscure the target markers. However, at extreme angles the target markers begin to run together, shown here in Figure 33.

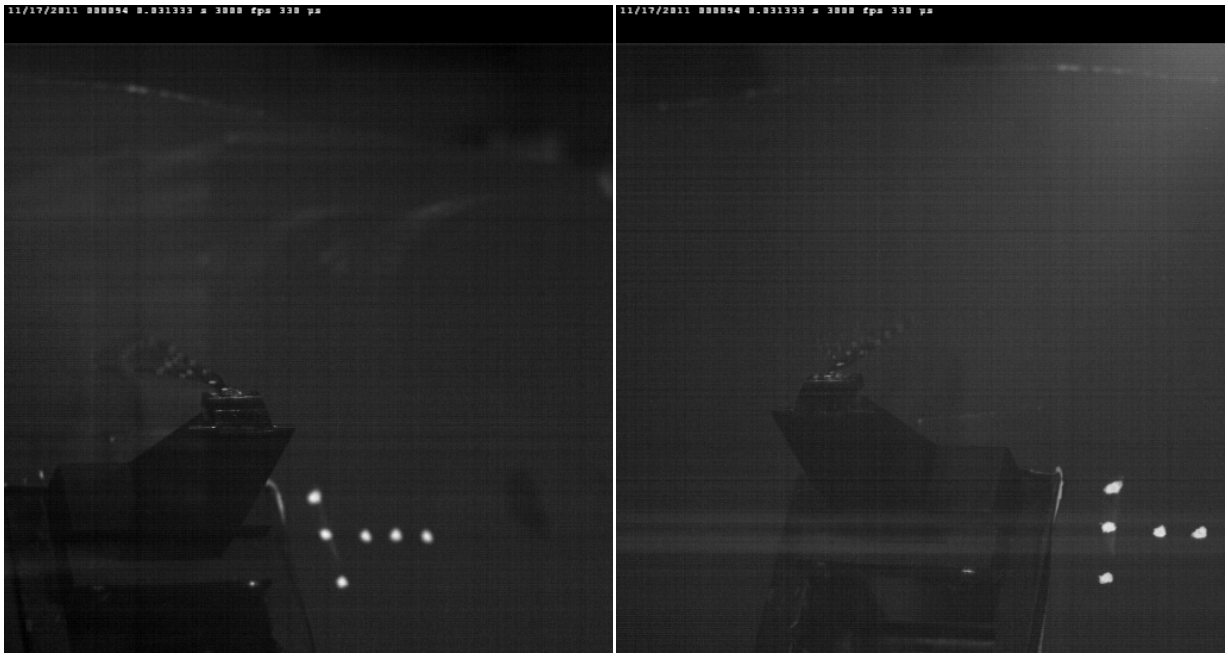

Figure 33: Flapping wing without Mylar with points running together.

In Figure 33, it can be seen that as the wing plane aligns with the Z axis of the camera the points become too small to distinguish for tracking even by hand. This limitation is due to the camera being locked in place. This difficulty is introduced in this system and was not seen in Deleón's work with Photomodeler, because the camera there was hand held and the stereo calibration was based on reference points in each image[1]. Based on this test it was determined that a full stroke from a single capture is not going to be possible with this two camera setup. The intended stroke of 110 degrees prevents the cameras from viewing the wing clearly at all stroke angles.

4.5 Wing Reversal at the Top of the Stroke

This final test was performed using the current bio-inspired wing, developed by Ryan O'Hara, attached to the O'Hara flapper[15]. This test is performed using the information culled from the previous experiments and the process described in Chapter 3. The calibration was performed behind the Plexiglas in air using the 3mm square calibration grid. The calibration results are listed below in Tables 13-15.

Table 13: Calibration output for left camera for reversal test.

```
Focal Length:        fc = [ 1831.60863   1802.18198 ] ± [ 6.28440   6.96830 ]

Principal point:     cc = [ 174.48862   283.85977 ] ± [ 12.35307   9.22361 ]

Skew: alpha_c = [ 0.00000 ] ± [ 0.00000  ]  => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:            kc = [ 0.12993   -8.25456   0.00051   -0.00325
0.00000 ] ± [ 0.07617   4.28169   0.00219   0.00278  0.00000 ]

Pixel error:          err = [ 0.20722   0.18335 ]
```

Table 14: Calibration output for right camera for reversal test.

```
Focal Length:        fc = [ 1838.88826   1832.69629 ] ± [ 5.17937   5.42973 ]

Principal point:       cc = [ 259.61691   319.08517 ] ± [ 7.92413   7.75864 ]

Skew: alpha_c = [ 0.00000] ± [ 0.00000  ]   => angle of pixel axes = 90.00000
± 0.00000 degrees

Distortion:        kc = [ 0.22737   -9.09422   0.00839   -0.00551  0.00000 ]
± [ 0.06166   2.89796   0.00236   0.00170  0.00000 ]

Pixel error:          err = [ 0.15746   0.15820 ]
```

Table 15: Stereo calibration output for reversal test.

```
Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ 0.02600   0.73277  0.00259 ]
Translation vector:          T = [ -272.68292   -1.49531  108.67954 ]
```

The calibration pixel errors all come in under one pixel. Therefore, the testing proceeds with processing the flapping wing images. Based on the results of the experiment described in section

4.4 the frame range is limited to 38 frames, in order to avoid glare and points running over one another. It was determined that the strength of this new system is characterizing areas where shifts occur, which was very difficult using the strobe image system from Deleón's work[1]. The greater frame rate makes it easier to determine the wing deflections through the reversal at the top of the stroke.

One benefit of the wing with the Mylar film in place, is that the target markers can be slightly larger, making them easier to track and identify successfully. A sample of the normalized cross correlation marked and the center marked wing is displayed for the maximum upstroke of the leading edge in Figure 34.

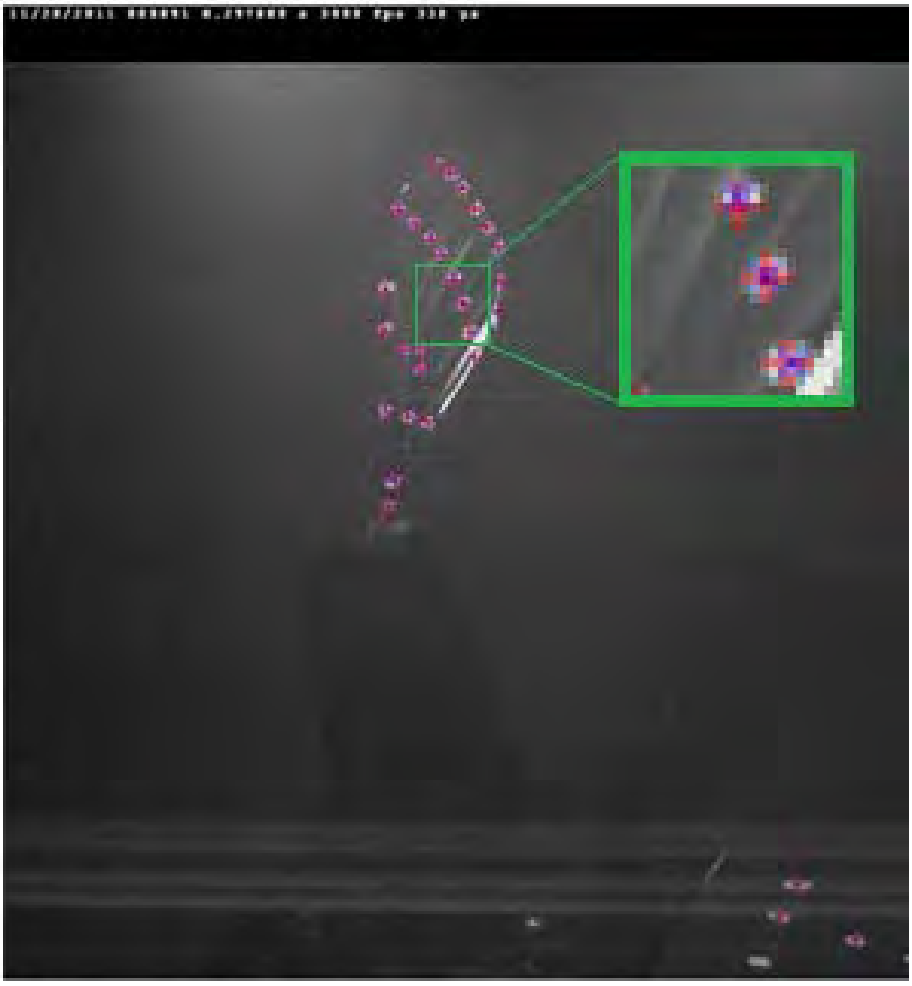

Figure 34: Top of the leading edge stroke with pixel and subpixel markings.

The red +'s mark the pixel level tracking and the blue x's mark the subpixel level tracking. These images are captured within the framework of the high-speed camera system and would have been very difficult to capture using a still camera system. During the pixel level tracking the number of points that required relocation was recorded for the right camera 90 points had to be relocated for tracking. In the left camera 72 points had to be relocated for the 38 frames. Including the initial 30 points that had to be located in each camera frame this comes to 120 and 102 points selected for the right and left cameras respectively. This is relatively few points to select compared to the 1140 points tracked for each camera that would have been selected by hand without the automated process. The centered data with the calibration data was loaded into the triangulation function to produce the final point cloud. The point cloud is displayed with the upstroke, stroke reversal and downstroke displayed in Figure 35-37.

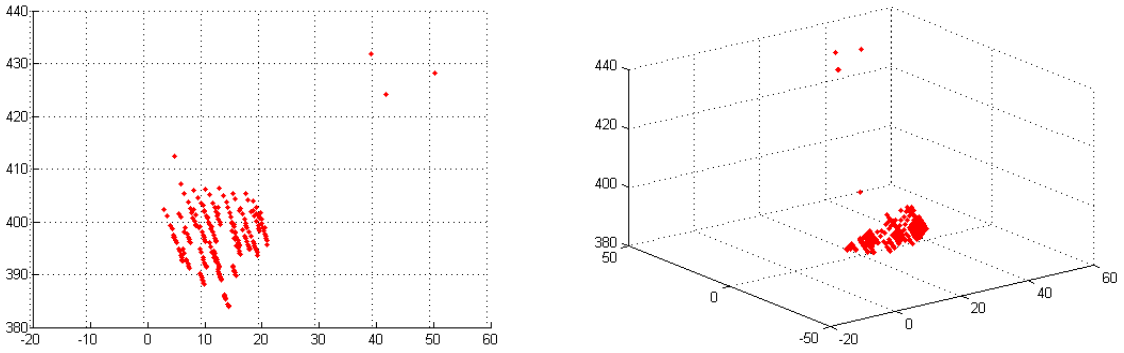

Figure 35: Wing upstroke three dimensional point cloud (in pixels).

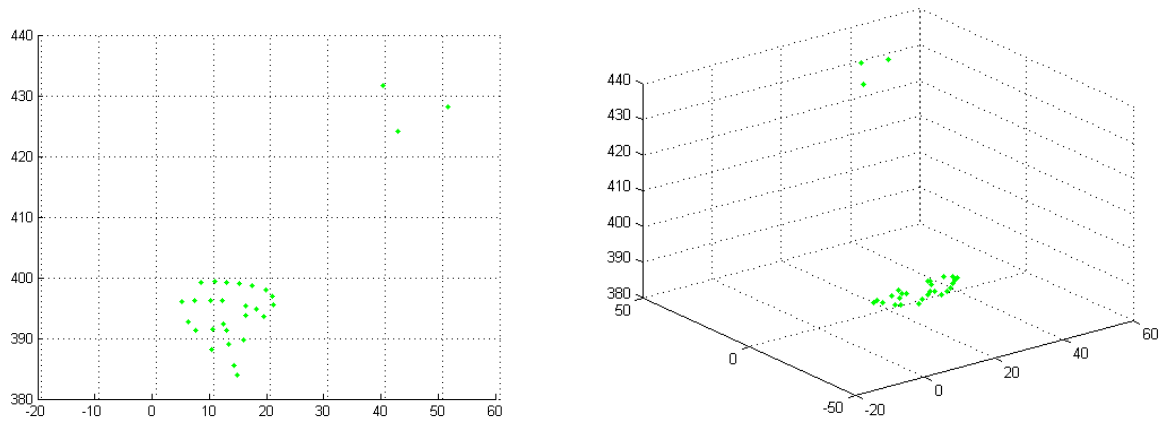

Figure 36: Wing at maximum upstroke at leading edge, stroke reversal (in pixels).

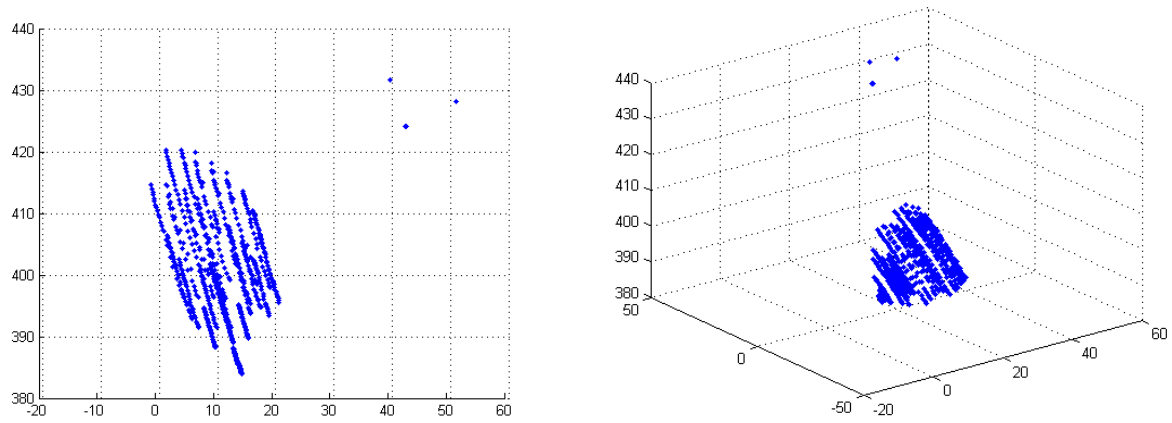

Figure 37: Wing downstroke three dimensional point cloud, from the top of the wing stroke (in pixels).

Down the left column are the point clouds rotated to show the end view. Down the right column are the isometric views of the point clouds. In the upper right corner are the three dimensional locations of the reference points, which remain relatively still. The point of stroke reversal in this case is defined by the maximum stroke of the leading edge. The trailing edge does continue to stroke upwards beyond this point. However, the input function has begun to actuate the wing downwards. The maximum displacement of any of these points is 0.0632 mm. This indicates

the system outside system is remaining still and the stereophotogrammetry system is not introducing artificial motion.

Chapter 5: Conclusions and Recommendations

5.1 Conclusions

The literature review provided several key pieces of information going forward to successfully develop the stereophotogrammetry system and to utilize high-speed video input. Nathanial Deleón's[1] description of the wing shapes made it clear that imposing constraints on the three dimensional output, such as the wing remaining planar, or limiting the expected deflection would not produce the desired information. This combined with the assertion that anything applied to the wings should be kept to a minimum leads to the use of stereophotogrammetry[1]. Hedrick's[4] assertion that prebuilt tracking programs tend to have a limited usable range beyond their original intended use further indicated that a system such as the Stereo Vision Toolbox[14] would be an ideal choice. The Stereo Vision Toolbox contains the components for setting up the camera system, but does not dictate how the image points are selected. This allowed for the development of an automated tracking system, combining a software tool with a marking scheme to produce reliable results. This additional flexibility makes the Stereo Vision Toolbox more versatile than Photomodeler which requires using built in image point location systems.

The use of high-speed cameras with synchronization provides a considerable advantage over still images captured using a stroboscope. The images are captured at the same time, so both views will capture the same wing in the case of shifts in the data. The flapper being linked to the camera also shortens the test runs significantly. The benefit of this is twofold. First, the wing only needs to be flapped for one second or less, which reduces the risk of wing failure. Second, the tests can be performed more rapidly. The high speed cameras also capture many more frames in a stroke improving the fit and characterizing areas of the stroke that were difficult to capture accurately using Deleón's method.

Another important tool developed during this research was pan-tilt device for calibrations. Previously, the calibration grid was manipulated by hand to produce the various angles. While this provided more potential angles, the grid had a tendency to drift off frame and move in and out of focus. Automating this reduced the time required to collect the images and produced more quality images to select from to calibrate the lenses.

The results of the experiments confirm that the Stereo Vision Toolbox[14] produces reliable data and can be applied to the flapping wing tests. By analyzing the potential distortion effects of the vacuum chamber walls, under vacuum, this research was able to remove that as a potential source of error. Again this serves to reduce testing time. Once set, the cameras should not be moved. Therefore, to perform vacuum tests the pan-tilt device does not need to be placed in the chamber.

5.2 Recommendations for Future Development

The goal of this system is to record a full wing stroke similar to Deleón[1], using the high speed cameras. To compile a full wing stroke in three dimensions will require piecing together the stroke from multiple tests similar to the method employed by Deleón[1]. To provide common points to link the tests, a reference grid was built into the Deleón flapper, pictured below Figure 38.

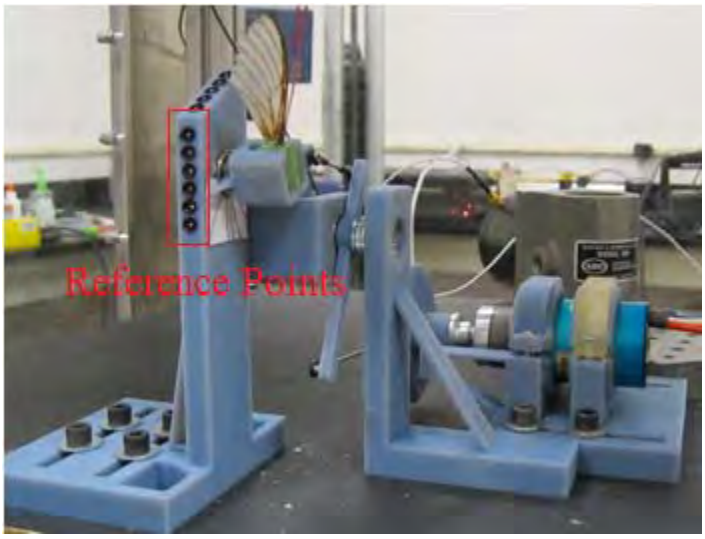

Figure 38: Deleón flapper with highlighted reference points.

A similar set of reference points would need to be incorparated into the O'Hara flapper[15]. The reference points used in this research were attached to a background sheet, which was not directly attached to the flapper. They serve the purpose for a single test, but keeping the background at the same orientation to the wing mounted in the flapper is not realistic. By building the points into the flapper as seen above they can provide a common point for data captured at the multiple angles required to capture the full stroke.

The tracking programs are currently two separate scripts. Thus, the data points are centered after the normalized cross correlation has taken place. If the data were centered the centered point rather than a pixel level point could be used to limit the acceptable search range.

This would improve the tracking and prevent lost points in the data. Another possible improvement once the two scripts are blended would be to automatically update the subimage for each frame. Then, there would only be a single frame between a subimage and the appropriate target, this should decrease the number of excessively shifted points. Alternatively this could be combined with an image mask to remove extraneous points, similar to the bpass.m output[10]. In this case the search image could be blacked out beyond a certain distance from the previous frame's centered tracking point. Thus, severely limiting the possible matches to a few marked points rather than the whole set.

Finally, no tests were performed using the biological wing, due to issues with mounting the wing to the O'Hara flapper, and getting the cameras positioned so that both can detect the tracking markers. These issues will require further consideration. With respect to biological wing testing it may be possible to track the natural pattern on the hawkmoth's wings preventing the need to remove the scale and paint on the markers.

5.3 Summary

In the framework of this research, a set of experimental tools have been developed and collected to utilize a photographical method to perform three dimensional measurements of a flapping wing. It has been demonstrated that these tools can be used to produce high speed dynamic measurements at up to 3000 fps. Furthermore, these tools have been applied to another object to perform three dimensional dynamic measurements successfully.

Works Cited

1. DeLeón, Nathanial E., *Manufacturing and Evalution of a Biologically Inspired Engineered MAV Wing Compared to the Manduca Sexta Wing Under Simulated Flapping Conditions,* Master's thesis, AFIT/GAE/ENY/11-M07, Air Force Institute of Technology, May 2011.

2. Tanaka H. J.P. Whitney and R. J. Wood. "Effect of Flexural and Torsional Wing Flexibility on Lift Generation in Hoverfly Flight". Integrative and Comparative Biology, Volume 51, Issue 1 pgs.142-150, 2011.

3. Tobias, A., *Experimental Methods to Characterize Nonlinear Vibration of Flapping Wing Micro Air Vehicles,* Master's thesis, AFIT/GAE/ENY/07-M23, Air Force Institute of Technology, March 2007.

4. Hedrick, Tyson L, "Software Techniques for two- and three-dimensional kinematic measurements of biological and biomimetric systems." Bioinspiration & Biomimetics, Vol:3 pgs:1-6,2008.

5. Lewis, J. P. "Fast Normalized Cross-Correlation" *Vision Interface,* pg. 120-123, Expanded Paper, 1995

6. Brown, Duane C. "Close-Range Camera Calibration." *Photogrammetric Engineering,* pgs: 855-866, 1971.

7. Heikillä, Janne, Olli Silvėn. "A Four-step Camera Calibration Procedure with Implicit Image Correction", *Computer Vision and Patter Recognition,* pgs: 1106-1112, 1997.

8. Zollner, H. and R. Sablatnig. "Comparison of Methods for Geometric Camera Calibration Using Planar Calibration Targets", in *Proceedings of the Austrian Association of Pattern Recognition,* pgs: 234-244, 2004.

9. Sturm, Peter F. and Stephen J. Maybank. "On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications", Proceedings IEEE Third International Confrence Automatic Face and Gesture Recognition, pgs: 100-105, Apr. 1998.

10. http://physics.georgetown.edu/matlab/tutorial.html

11. Zhang, Zhengyou. "A Flexible New Technique for Camera Calibration". Transactions on Pattern Analysis and Machine Intelligence, Vol. 22 No 11, pgs: 1330-1334, Nov 2000.

12. Tsai, Roger Y. "A Versitile Camera Calibration Technique for High-Accuracy 3d Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses". IEEE Journal of Robotics and Automation. Vol RA-3, No. 4, pgs: 323-344, Aug 1987.

13. X-Stream XS-4 brocure, IDT.

14. Bouguet, Jean-Yves, "Visual methods for three-dimensional modeling." Ph.D Thesis, California Institute of Technology, 1999.

15. O'Hara, Ryan, "The Material Property and Structural Dynamic Characterization of the Manduca Sexta Forewing for Application to Flapping Micro Air Vehicle Design", Dissertation Prospectus, AFIT/DS/ENY/DSY-12S-XX, Air Force Institute of Technology, 2011.

Appendix A: NCC.m

```
%% ********************************************************************
%% NCC.m
%%
%%  Program builds pixel level tracking locations from 2-d images.
%%  Has error check with a set radius that the subsequent point must fall
%%  in to continue tracking.  Number of points, frames, and which camera
%%  are all user defined.  Produces tracking video for verification at the
%% the end of the program.
%%
%%      By: Jeremy Murray     12/09/2011
%%*****************************************************

clc,clear all ,close all

%% Get .tif files
Files=dir('*.tif');
NumPoints=input('How many points are you tracking?\n');
NumFrames=input('How many frames are you using?\n');
cam=input('Which camera are you using (l or r)?\n','s');
savefile='XandYpos';

maxtravel=8;  %% maxtravel limits the distance a point can shift between
frames (pixels)

l=1;
k=1;
n=2;
m=1;
err=0;
for o=1:NumPoints
     Point(o).x=zeros(2,NumFrames-1);
end


while m<=NumFrames-1;

    I(1).I = imread(Files(m).name);
    I(2).I = imread(Files(m+1).name);
    for p=1:NumPoints

        if m == 1

            fprintf('Select point %d.\n',p);
            [I(1).Point(p).Sub,I(1).Point(p).SubRect] = imcrop(I(1).I);
% Choose the Subimage of square from I1
            [I(2).Point(p).Sub,I(2).Point(p).SubRect] =
imcrop(I(2).I,I(1).Point(p).SubRect);    % Choose the Subimage from I2
            if p==NumPoints
                fprintf('Go grab coffee.\n');
            end
        end
        if m>3
```

```
            checks =(abs(Point(p).x(1,m-1)-Point(p).x(1,m-
2))>9||abs(Point(p).x(2,m-1)-Point(p).x(2,m-2))>maxtravel);
            if checks == 1
                fprintf('Error at frame %d. Reselect point %d.\n',m,p)
                figure(1)
                imshow(I(1).I);
                hold on
                plot(Point(p).x(1,m-2),Point(p).x(2,m-2),'r+');
                plot(Point(p).x(1,m-1),Point(p).x(2,m-1),'bx');
                figure(2)
                [I(1).Point(p).Sub,I(1).Point(p).SubRect] = imcrop(I(1).I);
% Choose the Subimage of square from I1
                [I(2).Point(p).Sub,I(2).Point(p).SubRect] =
imcrop(I(2).I,I(1).Point(p).SubRect);   % Choose the Subimage from I2
                fprintf('Go back to your coffee.\n');
                close all
                err=err+1;
                m=m-1;
            end
        end
    % Compute Normalized Xcorr2 Sub-Image 1 to Image 2
    c = normxcorr2(I(n-1).Point(p).Sub,I(n).I);
    %figure, surf(I(n).c(:,:,1)), shading flat
    [MaxC,Imax] = max(abs(c(:)));
    [y_peak, x_peak] = ind2sub(size(c),Imax);
    corr_offset_xs = x_peak-size(I(n-1).Point(p).Sub,2);
    corr_offset_ys = y_peak-size(I(n-1).Point(p).Sub,1);
    Point(p).x(1,m) = corr_offset_xs+I(n-1).Point(p).SubRect(3)/2;
    Point(p).x(2,m) = corr_offset_ys+I(n-1).Point(p).SubRect(4)/2;

    if mod(m,20)
        save crash.mat     %save every 20 data sets in case of a crash
    end

    end
    m=m+1;
end

if cam=='l'
    Pointl=Point;
    save('xl.mat','Pointl')
end


if cam=='r'
    Pointr=Point;
    save('xr.mat','Pointr')
end


%% Make Avi of Tracking progress

% Create Video Object
vidObj = VideoWriter('TrackingDesktopimpIRoot.avi');
```

```
% Set and view the frame rate.
vidObj.FrameRate = 30;

% Open Video Object
open(vidObj);
figure
for m=2:NumFrames-1

   imshow(imread(Files(m).name))
   hold on
   for p=1:NumPoints
   plot(Point(p).x(1,m),Point(p).x(2,m),'r+')
   end
   hold off
   currFrame = getframe;
   writeVideo(vidObj,currFrame);
end



% Close the file.
close(vidObj);
```

Appendix B: trackcl.m

```
%*********************************************************************
%
%   trackcl.m
%
%   Program loads output from NCC.m, xl.mat and image set process in
%   previous step.  The images are processed using the bpass.m function
%   (Georgetown, Location and tracking tutorial) to subtract background and
%   determine tracking points.  Then the cntrd.m function (Georgetown,
%   Location and tracking tutorial is applied which locates the centroid of
%   the area where the xl.mat marker is located.  This program also
%   produces video to check tracking quality.
%   Note trackcr.m is essentially the same program with r's swapped in for
%   l's to process right camera images.
%
%          Jeremy Murray                          12/09/2011
%*********************************************************************

clc, clear all, close all
%% Get .tif files
tic
Files=dir('*.tif');
load xl.mat;
NumFrames=999;
NumPoints=2;
%% Load 2 Images
for i=1:NumFrames

%% Load NCC Data and Ensure input points to cntrd are Integers
    for p=1:NumPoints

        Img(i).pt(p).xl=[ceil(Pointl(p).x(1,i)),ceil(Pointl(p).x(2,i))];

    end
m=1;
I(1).I = imread(Files(i).name);


%% Use a special bandpass filter to smooth the image and subtract the
background
I(1).b = bpass(I(1).I,1,10,20); % (image, noise size(pixels, marker
size(pixels),threshold (below set to zero))

for p=1:NumPoints

%% Find Centroid to subpixel accuracy

    Img(i).cnt(p).xl = cntrd(I(1).b,Img(i).pt(p).xl,9); % (images, peak
locations, window for centroids)
end

end
toc
```

```
save('cntrdl.mat','Img');
tic

%% Make Avi of Tracking progress

% Create Video Object
vidObj = VideoWriter('TrackingDesktopBlobDmod.avi');

% Set and view the frame rate.
vidObj.FrameRate = 30;

% Open Video Object
open(vidObj);
figure


for m=1:NumFrames

   imshow(imread(Files(m).name))
   hold on
    for p=1:NumPoints
   plot(Img(m).cnt(p).xl(1,1),Img(m).cnt(p).xl(1,2),'r+')
   plot(Img(m).pt(p).xl(1,1),Img(m).pt(p).xl(1,2),'bx')
    end

   hold off
   currFrame = getframe;
   writeVideo(vidObj,currFrame);
end




% Close the file.
close(vidObj);
toc
```

Appendix C: triang.m

```
%*****************************************************************
%
%   triang.m
%
%   Program calls data from previous centering operations, cntrdl.mat and
%   cntrdr.mat and sets data for input into stereo_triangulation.m
%   (Stereo Vision Toolbox, Jean-Yves Bourguet, CalTech).  This program
%   loops over all the frames and data points to create 3-d point clouds.
%
%            By: Jeremy Murray                      12/09/2011
%*****************************************************************

clc, clear all, close all
%%[XL,XR] =
stereo_triangulation(xL,xR,om,T,fc_left,cc_left,kc_left,alpha_c_left,fc_right
,cc_right,kc_right,alpha_c_right)
% convert last output to new input

load cntrdl.mat
NumFrame=max(size(Img))-1;
NumPoint=max(size(Img(1).cnt))-3;
for i=1:NumFrame
for p=1:NumPoint
PT(p).xl(1,i)=Img(i).cnt(p).xl(1,1);
PT(p).xl(2,i)=Img(i).cnt(p).xl(1,2);
end
end


load cntrdr.mat
for i=1:NumFrame
for p=1:NumPoint
PT(p).xr(1,i)=Img(i).cnt(p).xr(1,1);
PT(p).xr(2,i)=Img(i).cnt(p).xr(1,2);
end
end




load Calib_Results_stereo.mat

for p=1:29

[PT(p).XL,PT(p).XR]=stereo_triangulation(PT(p).xl,PT(p).xr,om,T,fc_left,cc_le
ft,kc_left,alpha_c_left,fc_right,cc_right,kc_right,alpha_c_right);

hold on
plot3(PT(p).XR(1,:),PT(p).XR(2,:),PT(p).XR(3,:),'r')
end

% plot3(PT(2).XR(1,:),PT(2).XR(2,:),PT(2).XR(3,:),'m+')
```

```
% plot3(PT(3).XR(1,:),PT(3).XR(2,:),PT(3).XR(3,:),'b+')
% plot3(PT(4).XR(1,:),PT(4).XR(2,:),PT(4).XR(3,:),'y+')
% plot3(PT(5).XR(1,:),PT(5).XR(2,:),PT(5).XR(3,:),'c+')
% plot3(PT(7).XR(1,:),PT(7).XR(2,:),PT(7).XR(3,:),'r+')
% plot3(PT(6).XR(1,:),PT(6).XR(2,:),PT(6).XR(3,:),'m+')
% plot3(PT(8).XR(1,:),PT(8).XR(2,:),PT(8).XR(3,:),'g+')
% plot3(PT(9).XR(1,:),PT(9).XR(2,:),PT(9).XR(3,:),'m+')
% plot3(PT(10).XR(1,:),PT(10).XR(2,:),PT(10).XR(3,:),'y+')
% plot3(PT(11).XR(1,:),PT(11).XR(2,:),PT(11).XR(3,:),'c+')
grid on
save('3dall.mat','PT')
```

Appendix D: PanTilt.m

```
%*****************************************************************
%
%   PanTilt.m
%
%   This program controls Pan-Tilt calibration device.  Servo 1 is the
%   pan axis and servo 2 is the tilt axis.  3 distinct tilts are output as
%   well as 7 pan positions.  This provides the set locations for camera
%   calibration.
%
%   By: Ryan O'Hara and Jeremy Murray       12/09/2011
%*****************************************************************


clc,clear,close all


%% Note Matlab 2011b 32 bit required to make this counter example run
NI=daqhwinfo('nidaq');

%% Create Device object
gv = daq.getVendors();
s = daq.createSession(gv.ID);
s.addCounterOutputChannel(NI.InstalledBoardIds{1},'ctr0', 'PulseGeneration')
%PFI12 Screw Terminal
s.addCounterOutputChannel(NI.InstalledBoardIds{1},'ctr1', 'PulseGeneration')
%PFI13 Screw Terminal
%% Set Frequency Choices

% HS-311 Servo Specs at http://www.servocity.com/html/hs-311_standard.html

t     = 1; % Hold Time (s)
tPer  = 20; % Period (ms)
tFrq  = 1/(tPer*1E-3); % Frequency in Hz
%Position setpoints for 1 HS311
tLow  = 1.3;  % -90 Period Setpoint (ms)
tNeu  = 1.5; %   0 Period Setpoint (ms)
tHigh = 1.7; %  90 Period Setpoint (ms)
%Position setpoints for 2 S03N These are matched
% tLow  = .6;  % -90 Period Setpoint (ms)
% tNeu  = 1.5; %   0 Period Setpoint (ms)
% tHigh = 2.4; %  90 Period Setpoint (ms)
%view setpoints
tLow2 = .6;
tNeu2 = 1.5;
tHigh2 = 2.0;

idx=7;
DC2h=linspace(tHigh2,tHigh2,idx); % Duty Cycle (%)
DC2l=linspace(tLow2,tLow2,idx);
DC2n=linspace(tNeu2,tNeu2,idx);
DC2=[DC2h DC2n DC2l];
DC2=[DC2 tNeu2]/tPer;
DC1=linspace(tLow,tHigh,idx);
```

```
DC1in=linspace(tHigh,tLow,idx);
DC1=[DC1 DC1in DC1];
DC1=[DC1 tNeu]/tPer;
% Return to Neutral and Divide by period
%DC=ones(1,idx)

for n=1:length(DC1)
    % Spec Channel 1
    ch1 = s.Channels(1);
    ch1.Frequency = tFrq;
    ch1.InitialDelay = 16E-3;
    ch1.DutyCycle = DC1(n);

    % Spec Channel 2
    ch2 = s.Channels(2);
    ch2.Frequency = tFrq;
    ch2.InitialDelay = 0;
    ch2.DutyCycle = DC2(n);

    % Output Channel Settings and Run
    s.Rate = 1000;
    s.DurationInSeconds = t;
    s.startForeground; % Actually Start the Program
end
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE (DD–MM–YYYY)	2. REPORT TYPE	3. DATES COVERED *(From — To)*
20-12-2011	Master's Thesis	Oct 2010 – Dec 2011

4. TITLE AND SUBTITLE
Development of Photographic Dynamic Measurements Applicable to Evaluation of Flapping Wing Micro Air Vehicles

5a. CONTRACT NUMBER
5b. GRANT NUMBER
5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Jeremy C. Murray

5d. PROJECT NUMBER
5e. TASK NUMBER
5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/ENY)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GAE/ENY/11-D02

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research
Attn: Dr. Douglas Smith
875 Randolph St. Suite 325
Arlington, VA 22203
douglas.smith@afosr.af.mil
(703) 696-6219

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR/RSA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES
This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

Developments in the area of flapping wing micro air vehicles (FWMAVs) of a small size and with limited range. This has lead to a great deal of interest in biomimetric designs based on flapping wing flyers, including the North American Hawkmoth (*Manduca Sexta*). By utilizing high speed photography and photogrammetry the dynamic flapping of the wing can be characterized for comparison with mathematical models, namely computational fluid dynamics (CFD) and finite element analysis (FEA).To successfully utilize high speed image capture, a method of successfully digitizing many data points in a short period of time needs to be developed. The effort coordinated several photographically oriented MATLAB functions in a sequential order to develop the eventual three dimensional output. The key function to track two dimensional points was developed in this research utilizing MATLAB code, and was an important requirement to achieving the eventual results. The increased stroke angle will be used to characterize the wing displacement at the ends of the stroke and near zero where still image photogrammetry cannot capture enough images. Specific concerns with photogrammetry in this test situation have been considered and The utilization of photogrammetry was further improved by a mechanical device to automate calibration image collection, which reduces calibration errors.

15. SUBJECT TERMS
FWMAV, Photogrammetry, Hawkmoth, MATLAB,

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Anthony N. Palazotto
U	U	U	UU	87	19b. TELEPHONE NUMBER (Include Area Code) (937)255-3636, ext 4599