# ABSTRACT

Title of Thesis: Automated Kinematic Extraction of Wing and Body Motions of Free Flying Diptera

Nicholas I. Kostreski, Master of Science, 2012

Thesis directed by: Dr. Sean Humbert
Department of Aerospace Engineering

In the quest to understand the forces generated by micro aerial systems powered by oscillating appendages, it is necessary to study the kinematics that generate those forces. Automated and manual tracking techniques were developed to extract the complex wing and body motions of dipteran insects, ideal micro aerial systems, in free flight. Video sequences were captured by three high speed cameras (7500 fps) oriented orthogonally around a clear flight test chamber. Synchronization and image-based triggering were made possible by an automated triggering circuit. A multi-camera calibration was implemented using image-based tracking techniques. Three-dimensional reconstructions of the insect were generated from the 2-D images by shape from silhouette (SFS) methods. An intensity based segmentation of the wings and body was performed using a mixture of Gaussians. In addition to geometric and cost based filtering, spectral clustering was also used to refine the reconstruction and Principal Component Analysis (PCA) was performed to find the body roll axis and wing-span axes. The unobservable roll state of the cylindrically shaped body was successfully estimated by combining observations of the wing kinematics with a wing symmetry assumption. Wing pitch was determined by a ray tracing technique to compute and minimize a point-to-line cost function. Linear estimation with assumed motion models was accomplished by discrete Kalman filtering the measured body states. Generative models were developed for different species of diptera for model based tracking, simulation, and extraction of inertial properties. Manual and automated tracking results were analyzed and insect flight simulation videos were developed to quantify ground truth errors for an assumed model. The results demonstrated the automated tracker to have comparable performance to a human digitizer, though manual techniques displayed superiority during aggressive maneuvers and image blur. Both techniques demonstrated non-intrusive methods for establishing reference flight kinematics, which are being used to develop flight dynamics models in future work.

# Automated Kinematic Extraction of Wing and Body Motions of Free Flying Diptera

by

Nicholas I. Kostreski

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Masters of Science
2012

## Acknowledgments

I will be forever grateful to those who continually pushed me to make this thesis possible and for believing in me no matter how impossible the task seemed.

First and foremost I'd like to thank my advisor, Dr. Sean Humbert for giving me this once in a lifetime opportunity to work in perhaps the coolest lab in the country on projects that you normally only see on the Discovery Channel. When I first came to Dr. Humbert's door, he sat me down and took the time to explain the whole graduate school philosophy to me. He took me in with open arms and gave me the freedom to explore my curiosity. When I think back to when I just started, two years ago, I realize just how much of a successful growing experience this has been. I wanted to work on flapping dynamics and controls, little did I know it would involve studying actual flapping insects! This research project has been a great adventure with a pile of emotions throughout the journey and in the end I know I'm going to miss it.

I owe a great amount of thanks to Dr. Jason Vance for getting me started on high speed videography and introducing me to manual kinematics extraction. Jason, you are the smartest biologist I know. I want to thank Mac McFarlane for always willing to discuss insect flight dynamics and helping me resolve some of the hardest problems I've come across. Thanks to Dr. Imraan Faruque for his guidance and showing me how to turn every day life into research projects. I particularly want to thank Doug Szczublewski for keeping me sane these past two years, forcing me to have fun outside the lab and showing me how to celebrate a successful day of research. Andrew Roshwalb, thanks for being my best friend and always being there for me. I'm so glad we were lucky enough to work in the same building! To Greg Gremillion, Badri Ranganathan, Joe Conroy, Renee Gardner, Kedar Dimble, and Lina Castano you are all the smartest and nicest group of people I know and it has been a privilege to have met you and call you my friends.

I want to thank my parents for always supporting my dreams and my pursuit of a higher level education. You have guided me since birth and I am everything because of you, my appreciation cannot be measured.

Lastly, I want to thank my lovely wife Olya Kostreski. Not a day goes by where she doesn't give me the strength and courage to tackle any problem. She has been with me every step of the way and so supportive and understanding of the crazy research hours it took to finish. A special thanks to Dr. Robert Sanner for allowing me to take his stochastic final exam early so that I could attend my wedding stress free.

It's over...or is it?

# Table of Contents

# List of Figures

# List of Abbreviations

| | |
|---|---|
| $A$ | Continuous model dynamics matrix |
| $B_k$ | Background image model in $k^{th}$ frame |
| $d$ | Image scaling factor |
| $\vec{d}$ | Distance |
| $\mathbf{C}$ | Single camera coordinate frame |
| $E[]$ | Expectation |
| $\hat{\mathbf{e}}$ | Unit vector |
| $F_k$ | Foreground image in $k^{th}$ frame |
| f | Focal length, frequency |
| $\mathbf{H}$ | Image coordinate frame |
| $\mathbf{I}$ | Inertial coordinate frame |
| $I_k$ | Raw image in $k^{th}$ frame |
| $J$ | Cost function |
| $K$ | Intrinsic camera parameter matrix |
| $K_k$ | Kalman gain in $k^{th}$ frame |
| $L$ | DLT calibration parameter matrix |
| $\vec{m}$ | Moment vector of a line |
| $\vec{n}$ | Pixel ray vector |
| $\mathbf{P}$ | Principal coordinate frame |
| $P_k$ | Error covariance matrix in $k^{th}$ frame |
| $Q_k$ | Process noise covariance matrix in $k^{th}$ frame |
| $R$ | Rotation matrix |
| $R_k$ | Measurement covariance matrix in $k^{th}$ frame |
| $T$ | Explicit camera parameter transformation matrix |
| $\vec{V}$ | Wing span vector |
| $\mathbf{v}(t)$ | Measurement noise |
| $\mathbf{w}(t)$ | Process noise |
| $_{rw}$ | Right wing |
| $_{lw}$ | Left wing |
| $\mathbf{z}$ | Measurement |
| $\alpha_w$ | Wing pitch |
| $\sigma$ | Standard deviation |
| $\theta_b$ | Pitch |
| $\theta_w$ | Stroke deviation |
| $\Phi$ | State transition matrix |
| $\phi_b$ | Roll |
| $\phi_{con}$ | Roll constraint measurement |
| $\phi_w$ | Stroke amplitude |
| $\psi_b$ | Yaw |
| AVL | Autonomous vehicle lab |
| CFD | Computational fluid dynamics |
| DLT | Direct linear transformation |
| PCA | Principal component analysis |
| PDF | Probability density function |
| rms | Root mean square |

# Chapter 1

# Introduction

The increasing demand for smaller, lighter, more agile flying vehicles capable of autonomous navigation in highly dynamic environments requires innovative engineering perspectives, particularly those inspired by the field of biology. Bio-inspired engineering is a growing field and at the micro and nano scale, there is currently an increasing focus towards understanding the complex sensorimotor control and non-linear dynamics of flying insects. Insects are capable of performing incredible maneuvers in addition to recovering from massive environmental perturbations by use of collective and differential wing kinematic inputs. The aerodynamic forces generated by these complex wing motions can be modeled by quasi-steady aerodynamics, computational fluid dynamics (CFD), and experimental methods **??**. Linearized models for longitudinal and lateral flight are being developed for different species of insects and about different reference flight conditions [7], [8], [16]. In order to develop these models, it is necessary to develop accurate representations of the wing kinematics for a particular reference flight condition in addition to the insect's inertial properties. Accurate kinematic extraction is a daunting task however, requiring the use of several high speed cameras to capture the wing and body motions of a free flying insect as unintrusively as possible. Frame by frame analysis by manual digitization of the insect is typically performed, requiring a great deal of time and effort. Therefore, attempts to apply automated marker-less motion capture concepts have emerged, but these lack the accuracy of a human digitzer.

## 1.1 Insect Tracking Literature Review

There exists a wide variety of approaches to solving the problem of body and wing kinematics extraction of maneuvring insects. Tethered flight studies simplify the filming process a great deal by keeping the insect stationary and allow for visual stimulus studies to be conducted. Photo-diode techniques have been used on tethered insects to measure the stroke amplitude signals of the wings; however, for free flying insects, such techniques are not relevant. In free flight studies a multiple camera set-up is generally required since most of the wing and body motion is impossible to track in a single view due to depth ambiguity. The most common technique involves manual digitizing methods where the image coordinates of six or more landmarks on a fly are digitized in multiple camera views [24], [12]. Larger insects and animals can be tracked with visual markers placed over the surface of the wing to visually aid in the manual tracking process and permit the use of auto or semi-auto tracking methods [13]. With smaller insects such as fruit flies, *Drosophila Melanogaster*, this technique is not feasible due to the small scale and the affect that markers have on the aerodynamics. Even with manual digitizing techniques, a geometric model of the wing is generally needed for determining the wing pitch angle due to its poorly observable nature. The body roll angle is distinctively difficult to extract due to the cylindrical shape of the insect. There have been attempts to use image texture to address this issue [26], but generally the image lacks any definite texture because of the low exposure (40 $\mu$s) associated with a high frame rate (7500 fps), which reduces the size of the aperture and thus the amount of light available. To increase the contrast in such low light environments, the focal volume is back-lit, resulting in bimodal silhouettes of the body and wing, which can be represented by a mixture of Gaussians [10]. Human digitizing still remains to be the most accurate approach to kinematic extraction, although it is entirely too time consuming, with a single sequence taking a day to a week depending on the frame rate, video length, and number of landmark features to track. There is therefore strong motivation to move past manual kinematic methods.

### 1.1.1 Manual Tracking Techniques

Steven Fry introduced a MATLAB graphical user interface (GUI) capable of performing digitizations without needing to extract the cameras internal and external parameters [24]. This requires an orthogonal camera set up, treating each of the image axes as coordinate directions of a global frame. Axis scaling between views is achieved by localized regression, which can be performed on the insect in the actual flight sequence to be digitized, or on a calibration object. The program works by manually digitizing 6 landmarks on the insect (head, tail, wing roots, wing tips). A wing model can then be superimposed onto the image and rotated to extract the wing pitch angles.

To permit a more flexible camera set-up, another approach to manual digitizing, developed by Ty Hedrick was introduced [13]. His program, which is freely available works off a direct linear transformation (DLT) camera calibration. The DLT method is described in detail in Chapter 2. The advantages of this method are that it is well set up for any number of cameras and can handle non-orthogonal camera arrangements. The calibration also permits a mapping from 2D coordinates to a 3D global frame, with actual units of length. This method is ideal for extracting not just angles, but also positions and velocities.

### 1.1.2 Automated Tracking Techniques

Leif Ristroph introduced a method known as Hull Reconstruction Motion Tracking, HRMT, which uses an extrusion process to reconstruct the maximally consistent shape of the insect with the images [21]. The effective reconstruction is defined as the visual hull and is represented by a set of 3D pixels called voxels. In this approach the visual hull is constructed by extruding the fly silhouettes from 3 orthogonal cameras into 3D space. Wings and body are separated using a K-means segmentation algorithm and body and wing angles are determined using a combination of Principal Component Analysis (PCA) and geometric information about the insect. To perform the extrusion, the bounding boxes of the 3 silhouettes need to be rescaled to equal sizes. A major constraint with this approach is it requires an orthogonal camera set-up, because it ignores the pin-hole camera model and therefore avoids any camera calibration procedure. While this extrusion process is efficient for generating visual hulls of the insect, the camera alignment process becomes critical, requiring the use of precision rails or other micro alignment tools. Reconstruction artifacts are inherent to this method, more so than if a calibration were used to reconstruct the visual hull. The roll angle of the body is extracted by first clustering the visual hull of the body into 3 smaller clusters (head, thorax, abdomen) and performing a cross product on the 2 vectors formed between the 3 centroid locations to determine the pitching axis of the body. The angle this vector makes with respect to the unit yaw vector is the roll angle. The wing pitch is extracted by an assumption that the reconstructions of the wings result in parallelogram shaped cross sections, which when projected onto a plane normal to the span axis, the two furthest points on the cross section define the chord vector. While PCA works exceptionally well for extracting certain states, the methods presented in HRMT for estimating roll and wing pitch are prone to large errors.

Ebraheem Fontaine is among the few who have attempted model-based automated tracking methods on flying insects [10]. Literature on model-based tracking techniques generally address the problem of markerless motion capture of humans in real time. For insect tracking, the constraint of real time is lifted, which allows more computationally expensive methods; however, the low number of cameras and poor resolution increases the difficulty exponentially. In Fontaine's approach, the tracking is initialized manually, in the first frame a general model of the fly is superimposed onto the images and modified using generative modeling techniques to customize the shape of the fly. The current state of the fly is then updated by projecting the image pixels as rays in 3D space and minimizing a point to line distance function. A sigma point filter acts as a non-linear filter in this procedure and a time history of manually extracted wing angles in the form of quaternions are used as training data for an initial guess. The body roll angle is determined by a constraint that requires the wings to be symmetric about the transverse plane of the body. This constraint will be further discussed in Chapter 4. While this method represents the state of the art in automated tracking of insects, it is sensitive to situations where incorrect wing and body states yield the global minimum of the non-linear cost function. A major reason for this failure mode occurs when the rigid model doesn't correspond well with the true shape of the insect during wing deformations and body flexing.

## 1.2 Scope and Contributions of Current Research

The is currently no automated tracking program capable of outperforming a human digitizer. Also, the automated tracking programs discussed above have only been tested on a single species. The goal of this work is to take a bottom-up approach to developing an automated tracking program, capable of extracting kinematics from a variety of dipteran species in stable or maneuvering free flight. The kinematic libraries established in this research will aid future researchers in understanding how certain wing motions permit dazzling maneuvers, which will serve as a starting block for designing robust micro aerial systems.

In Chapter 2, the camera set-up and recording procedure is discussed. A standard pin-hole model of a camera is used to develop the derivation of the DLT camera calibration. The discrete Kalman filter is introduced and applied to an automated camera calibration procedure. Results of the automated procedure are compared to a manually digitized calibration.

Chapter 3 introduces the insect reference frames and the underlying mathematics necessary to extract kinematics in a manual digitizing program. A modified version of the DLT program [13] is used to extract the wing and body states and the results of a coordinated turn sequence are presented.

Chapter 4 presents the automated tracking program, explained in detail. An advanced image analysis procedure is fused with previously developed tracking concepts and considerations for preferred methods are discussed. The roll and wing pitch estimation and the post processing procedure is presented

Chapter 5 quantifies the differences between the manual and automated tracker for a variety of sequences. Simulations, representing ground truth are developed to quantify errors involved in both procedures.

The original contributions of this research to the state of the art are the following:

- Manual and automated MATLAB programs for extracting wing and body kinematics from any dipteran insect
- Kinematic data suitable for modeling
- Automated multi-view camera triggering and calibration procedure
- Ground truth simulations for quantifying tracking errors
- Advanced clustering and filtering techniques for refining visual hull reconstructions

- Generative modeling procedure for extracting shape and inertial properties of insects

# Chapter 2

# Automated Multiple-View Camera Calibration

Multiple-view camera calibration involves the process of extracting the internal geometric and optical properties (intrinsic parameters) and additionally the 3-D position/orientation of the camera relative to some global reference system (extrinsic parameters) [14]. The end result of a properly calibrated camera system permits a mapping between 3D global coordinates and 2D image coordinates and 2D image coordinates in one camera view to 2D pixel rays in any other camera view. There are many approaches to camera calibration, which makes choosing any particular type a daunting task. Camera calibration strategies can be broken down into 2 categories: photogrammetric calibration and self calibration [30]. Photogrammetric calibration involves observation of a known object with known geometry and using that information to back out the intrinsic and extrinsic camera parameters. Self calibration does not require a calibration object and instead the camera parameters are backed out by using image information alone. Self calibration, although desired, is not considered here due to the difficulty of implementation and unpredictable performance. However, a photogrammetric calibration that can be performed automatically is highly desired. Therefore this chapter will discuss the choice in calibration approach and explain how auto tracking methodologies can be applied effectively to this problem.

## 2.1 Camera Set-up

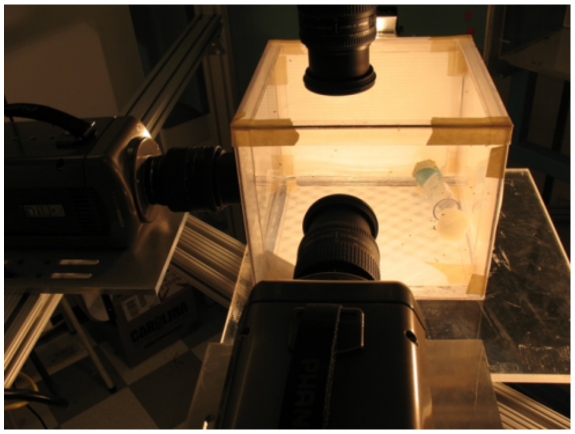


Figure 2.1: Multiple Camera View Set-Up

The camera set up used in this thesis, shown in Figure 2.1, involved three Phantom v710s by Vision Research positioned around a clear lexan flight test chamber. The focal volume was back-lit using three 500 W Lowel V lights. Photography umbrella fabric acted to diffuse the high intensity lighting in addition to dissipating the undesired heat. To achieve synchronous filming, the three cameras were connected via F-sync, with the front camera set as master and the side and top camera set as slaves. Triggering the front camera would then automatically trigger the side and top cameras with a delay of $\Delta T \approx 1\mu$s.

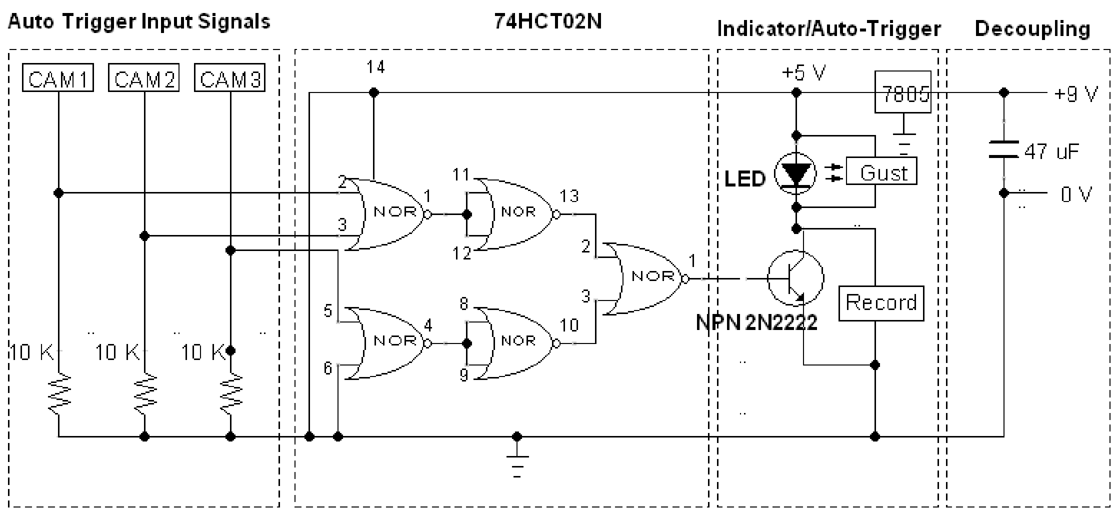


Figure 2.2: Auto-Triggering circuit

Simultaneously capturing events in high speed videography with multiple cameras was a difficult task because of the limited capture volume, further limited to a smaller focal volume region, which the insect passes through in a matter of milliseconds. Therefore an auto triggering system to aid in filming was developed. Previous approaches involving a laser/photo-diode auto-trigger have been used with success [21]; however, the Phantom v710 cameras enabled a real-time image-based triggering system. This meant that a sub-region of the image could be monitored for intensity changes and triggered based on an area percentage and intensity threshold. The automated triggering circuit, shown if Figure 2.2, functioned by using the input triggering signals from the three cameras. The signals were passed through a logic gate composed of two Quad 2-input NOR gate ICs (74HCT02N) . When the image trigger signals went low in all three cameras, the corresponding output signal went

Table 2.1: Auto Triggering Parameters

| | |
|---|---|
| Resolution | 800x800 |
| Calibration Frame Rate (fps) | 100 |
| Filming Frame Rate (fps) | 7500 |
| Exposure Time ($\mu$s) | 40 |
| Threshold | 5 |
| Area % | 2 |
| Interval Update (ms) | 10 |

high, which was sent to the base of an NPN transistor and caused the transistor to conduct. The voltage potential across the transistor acted as a switch, which was fed into the triggering input of the master camera and simultaneously triggered the three cameras. Table 2.1 highlights the typical parameters used for auto-triggering and filming.

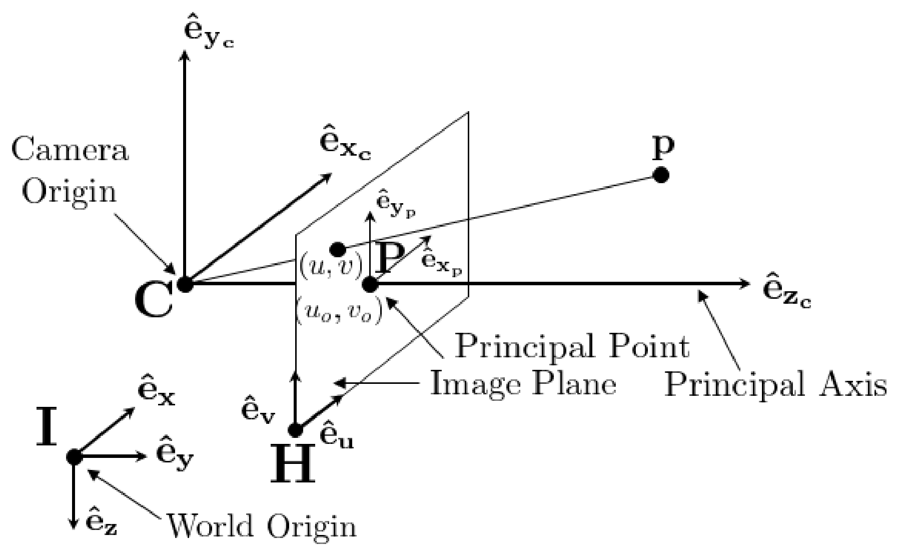
## 2.2 Calibration Approach

Among the more common types of photogrammetric calibration are approaches those that use a static object with known global coordinates [22] [13], a planar object (i.e. checker board pattern) [30] [3] , or a wand with at least 2 markers [19] [28]. Among these methods are those that consider non-linearities such as radial and tangential lens distortion [30] [3], and those that consider the camera model linear which make it acceptable to use a technique know as Direct Linear Transformation (DLT) [13] originally developed by Abdel-Aziz and Karara [1]. A bundle adjustment procedure is a nonlinear least-square algorithm which is typically used as an optional post process to optimize the rough DLT calibration or any other type of calibration for that matter and allows for the prediction of lens distortion.

In this thesis, the calibration approach was chosen to minimize complexity, time, and effort. Lens distortion was a minimal factor in our set-up and therefore ideal for the computationally efficient DLT method. The Autonomous Vehicle Laboratory (AVL), where this research was conducted, uses motion capturing technologies developed by VICON on a daily basis and so the choice of calibration object naturally evolved to that of a calibration wand, similar to the wand used to calibrate a VICON system [28]. The custom built calibration wand, consisting of piano wire and three rapid prototyped calibration markers, was used to calibrate the 3 cameras before all recording sessions. The cameras were positioned roughly orthogonal to one another, though orthogonality was not at all required. The cameras were focused and the capture rate was set to 100 fps with an exposure of $300\mu$s. The wand was waved in a random fashion to fill the capture volume of the three cameras for approximately 5000 frames. A custom MATLAB program was used to automatically initialize and track the wand markers in the three camera views using an Kalman filter and a constant velocity motion model. This automated tracking procedure will be discussed in detail in the next section. The 2D image coordinates of the two outer markers were saved into a spreadsheet, which was then loaded into Ty Hedrick's open source DLT calibration software [13]. This program recursively determined an estimate of the global coordinates of the wand by using the constant wand length constraint, a series of stereo triangulations between pairs of cameras, and a genetic algorithm to determine the intrinsic and extrinsic camera parameters.

### 2.2.1 DLT Theory

In this section, the assumed camera model is developed into a linear system that can be used to extract the parameters necessary to map between object space and image space.



$\hat{\mathbf{e}}_{\mathbf{y_c}}$
Camera Origin
$\hat{\mathbf{e}}_{\mathbf{x_c}}$
p
$\hat{\mathbf{e}}_{\mathbf{y_p}}$
$(u, v)$
P
$\hat{\mathbf{e}}_{\mathbf{x_p}}$
C
$(u_o, v_o)$
$\hat{\mathbf{e}}_{\mathbf{z_c}}$
Principal Point
Image Plane
Principal Axis
$\hat{\mathbf{e}}_{\mathbf{x}}$
I
$\hat{\mathbf{e}}_{\mathbf{y}}$
$\hat{\mathbf{e}}_{\mathbf{v}}$
$\hat{\mathbf{e}}_{\mathbf{u}}$
H
$\hat{\mathbf{e}}_{\mathbf{z}}$
World Origin

Figure 2.3: Pinhole Camera Model

The DLT method assumes a pinhole camera model, which ignores any radial or tangential distortions. This model can be expressed simply as:

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = -\frac{f}{Z_c} \begin{bmatrix} X_c \\ Y_c \end{bmatrix} \tag{2.1}$$

The image coordinates $(x_p, y_p)$ are represented in a principal coordinate frame $\mathbf{P} = (\hat{\mathbf{e}}_{\mathbf{x_p}}, \hat{\mathbf{e}}_{\mathbf{y_p}})$. The focal length is given by $f$ and $(X_c, Y_c, Z_c)$ are the 3D coordinates of point $\mathbf{p}$ in the camera coordinate frame $\mathbf{C} = (\hat{\mathbf{e}}_{\mathbf{x_c}}, \hat{\mathbf{e}}_{\mathbf{y_c}}), \hat{\mathbf{e}}_{\mathbf{z_c}})$. To convert coordinates $(x_p, y_p)$ from coordinate frame $\mathbf{P}$ to coordinates $(u, v)$ in the image coordinate frame $\mathbf{H} = (\hat{\mathbf{e}}_\mathbf{u}, \hat{\mathbf{e}}_\mathbf{v})$, a scaling factor ($\mathrm{d}_u$,$\mathrm{d}_v$) and translation $(u_o, v_o)$ are introduced to convert the coordinates to pixels and shift them about the desired image origin.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \left( \begin{bmatrix} d_u x_p \\ d_v y_p \\ 1 \end{bmatrix} + \begin{bmatrix} u_o \\ v_o \\ 0 \end{bmatrix} \right) = \begin{pmatrix} d_u f & 0 & u_o \\ 0 & d_v f & v_o \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{2.2}$$

This intrinsic parameter matrix, K, in general is expressed as:

$$K = \begin{pmatrix} d_u f_x & s & u_o \\ 0 & d_v f_y & v_o \\ 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

where s is the skew matrix that takes into account non square pixels and ($\mathrm{f}_x$,$\mathrm{f}_y$) are unique focal lengths that consider radial distortion [23]. For the DLT method, s = 0, and $\mathrm{f}_x$=$\mathrm{f}_y$ = f. When working with multiple cameras, the global reference frame cannot be assumed to have its origin at the camera center. Therefore the external camera parameters must be used to convert 3D global coordinates to 3D camera coordinates.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = c \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_x \\ r_{2,1} & r_{2,2} & r_{2,3} & t_y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.4}$$

$M = [R \quad T]$ is the general transformation matrix which is composed of the rotation matrix R and translation matrix T and c is a scaling parameter in the event that the camera reference frame uses a different unit length compared to that of the global frame. Hence, the conversion of world frame coordinates to image frame pixels can be computed as:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = cK[R \quad T] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.5}$$

The DLT parameters can be expressed by L:

$$L = \begin{pmatrix} L_1 & L_2 & L_3 & L_4 \\ L_5 & L_6 & L_7 & L_8 \\ L_9 & L_{10} & L_{11} & L_{12} \end{pmatrix} = cK[R \quad T] \tag{2.6}$$

Abdel-Aziz and Karara [1] used the constraint $L_{12} = 1$ to avoid trivial solutions while estimating the 11 other parameters. Therefore $\lambda$ in Equation 2.5 can be expressed by

$$\lambda = L_9X + L_{10}Y + L_{11}Z + 1 \tag{2.7}$$

Combining Equation 2.5 and 2.7, the DLT equations are determined to be:

$$u = \frac{L_1X + L_2Y + L_3Z + L_4}{\lambda} \tag{2.8}$$

$$v = \frac{L_5X + L_6Y + L_7Z + L_8}{\lambda} \tag{2.9}$$

Rewriting Equations 2.8 and 2.9 into matrix form for $n$ number of $(X, Y, Z)$ coordinates and their corresponding n number of $(u, v)$ coordinates:

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_nX_n & -v_nY_n & -v_nZ_n \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_{11} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_n \\ v_n \end{bmatrix} \tag{2.10}$$

Rewriting Equation 2.11 as $\mathbf{XL} = \mathbf{Y}$, the solution to the DLT parameters are then backed out using a least squares approach [15]:

$$\mathbf{L} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{2.11}$$

Once the DLT parameters are known, pixel coordinates from at least two cameras are required to reconstruct its 3D coordinate because a single camera coordinate is under-defined. Image coordinate information from 2 or more views results in an over-determined system since there are at least 4 given values (u$^{(1)}$, v$^{(1)}$, u$^{(2)}$, v$^{(2)}$) to recover 3 unknowns (X, Y, Z).

$$\begin{bmatrix} u^{(1)}L_9^{(1)} - L_1^{(1)} & u^{(1)}L_{10}^{(1)} - L_2^{(1)} & u^{(1)}L_{11}^{(1)} - L_3^{(1)} \\ v^{(1)}L_9^{(1)} - L_5^{(1)} & v^{(1)}L_{10}^{(1)} - L_6^{(1)} & v^{(1)}L_{11}^{(1)} - L_7^{(1)} \\ u^{(2)}L_9^{(2)} - L_1^{(2)} & u^{(2)}L_{10}^{(2)} - L_2^{(2)} & u^{(2)}L_{11}^{(2)} - L_3^{(2)} \\ v^{(2)}L_9^{(2)} - L_5^{(2)} & v^{(2)}L_{10}^{(2)} - L_6^{(2)} & v^{(2)}L_{11}^{(2)} - L_7^{(2)} \\ \vdots & \vdots & \vdots \\ u^{(k)}L_9^{(k)} - L_1^{(k)} & u^{(k)}L_{10}^{(k)} - L_2^{(k)} & u^{(k)}L_{11}^{(k)} - L_3^{(k)} \\ v^{(k)}L_9^{(k)} - L_5^{(k)} & v^{(k)}L_{10}^{(k)} - L_6^{(k)} & v^{(k)}L_{11}^{(k)} - L_7^{(k)} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} L_4^{(1)} - u^{(1)} \\ L_8^{(1)} - v^{(1)} \\ L_4^{(2)} - u^{(2)} \\ L_8^{(2)} - v^{(2)} \\ \vdots \\ L_4^{(k)} - u^{(2)} \\ L_8^{(k)} - v^{(2)} \end{bmatrix} \tag{2.12}$$

Expressing Equation 2.13 as $\mathbf{AX} = \mathbf{B}$, this over-determined system can be solved by the weighted least squares method [15].

$$\begin{aligned} \mathbf{AX} &= \mathbf{B} && (2.13) \\ (\mathbf{WA})\mathbf{X} &= \mathbf{WB} \\ (\mathbf{A}^T\mathbf{WA})\mathbf{X} &= \mathbf{A}^T\mathbf{WB} \\ (\mathbf{A}^T\mathbf{WA})^{-1}(\mathbf{A}^T\mathbf{WA})\mathbf{X} &= (\mathbf{A}^T\mathbf{WA})^{-1}(\mathbf{A}^T\mathbf{WB}) \\ \mathbf{X} &= (\mathbf{A}^T\mathbf{WA})^{-1}(\mathbf{A}^T\mathbf{WB}) && (2.14) \end{aligned}$$

Solving equation 2.14 involves an iterative procedure, which iterates and continually updates the diagonal weighting matrix W based on the previous solution of X until X converges below some tolerance.

The framework developed so far in this Chapter, enables a camera calibration to be performed from a set of $(X, Y, Z)$ coordinates and their respective $(u, v)$ coordinates in each camera view. The open source DLT calibration software is capable of performing this calibration, but before it can be performed, the $(u, v)$ coordinates of a calibration object with atleast 2 visual markers must be determined from a set images in which the markers span the focal volume. For 100 images, this corresponds to 600 image coordinates that must be extracted. In the next section, the discrete Kalman filter is formulated to aid in the tedious task of image-based tracking to automatically detect the visual markers of a calibration wand.

## 2.3 Discrete Kalman filtering with Linear Motion Models

The Kalman filter is a recursive process that produces the optimal linear estimate of the state of a system by combining information from a linear dynamical model with actual measurements. The Kalman filter has 2 unique assumptions that seperate it from other recursive Bayesian estimators: 1) the underlying system is linear, 2) all related process and measurment noise is gaussian distributed. Despite the first assumption, Kalman filters can perform well even when the actual process is slightly non-linear. In the Bayesian frame work, the Kalman filter can be thought of as the best probabilistic state update, given a measurement and an a priori estimate. The probability distribution of the a posteriori estimate is proportional to the product of the measurement and predicted state likelihoods.

$$P(\mathbf{x}_k \mid \mathbf{z}_k) = \frac{P(\mathbf{z}_k \mid \mathbf{x}_k)P(\mathbf{x}_k \mid \mathbf{z}_{k-1})}{P(\mathbf{z}_k \mid \mathbf{z}_{k-1})} \tag{2.15}$$

The Probability Density Function (PDF) was assumed to be of the form:

$$P(\mathbf{x}_k \mid \mathbf{z}_k) = \frac{\exp[\frac{1}{2}(\mathbf{x}_k - \mu_k)^T(P_k^-)^{-1}(\mathbf{x}_k - \mu_k)^T]}{2\pi\sqrt{\mid P_k^- \mid}} \tag{2.16}$$

## 2.3.1 State Space Representation

The tracking techniques developed in this research assume linear motion models due to the high frame rates and so the state space representations for these models must be formulated and discretized. A continuous linear time invariant (LTI) system may be written as follows:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) + G\mathbf{w}(t) \tag{2.17}$$

$$\mathbf{z}(t) = C\mathbf{x}(t) + \mathbf{v}(t) \tag{2.18}$$

A and B represent the dynamics and controls matrix, C is the measurement model, G is the process noise matrix considered to be identity, $\mathbf{w}(t)$ and $\mathbf{v}(t)$ are the process and measurement noise, and $\mathbf{x}(t)$ and $\mathbf{z}(t)$ are the system state and measurement respectively. For the purposes of wand tracking, the pixel coordinates (u,v) and radius r of the markers are directly observable states, thus $z(t) = [u(t), v(t), r(t)]^T$.

letting s(t) be the time varying position of a point, the derivative can be represented by a Taylor series

$$s(t) = s(0) + \dot{s}(0)\triangle t + \frac{1}{2}\ddot{s}(0)\triangle t^2 + \ldots + H.O.T \tag{2.19}$$

To capture the wand motion, the calibration videos were filmed at 100 fps and therefore it was appropriate to drop the acceleration term in the Taylor series expansion since $\triangle t^2 = 1e^{-4} \sim 0$. As a result, accelerations take the form of white noise $\sigma_a$ and $\mathbf{w}(t) = [0, \sigma_{\mathbf{a}}]^T$ and the tracked states become $x(t) = [u(t), v(t), r(t), \dot{u}(t), \dot{v}(t), \dot{r}(t)]^T$. The A matrix takes the form of a constant velocity motion model. No other inputs are considered and thus the $\mathbf{u}(t)$ term is dropped from the system.

$$\dot{\mathbf{x}}(t) = \begin{pmatrix} \mathbf{0} & \mathrm{I} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \mathbf{x}(t) + \mathbf{w}(t) \tag{2.20}$$

$$\mathbf{z}(t) = \begin{pmatrix} \mathrm{I} & \mathbf{0} \end{pmatrix} \mathbf{x}(t) + \mathbf{v}(t) \tag{2.21}$$

In order to apply the Kalman Filter to frame by frame analysis, the continuous time model in Equation 2.21 must be converted to discrete time.

$$\mathbf{x}_k = \mathbf{\Phi}_k \mathbf{x}_{k-1} + \mathbf{w}_k \tag{2.22}$$

$$\mathbf{z}_k = H_k \mathbf{x}_k + \mathbf{v}_k \tag{2.23}$$

The subscript k denotes the current frame. $\mathbf{\Phi}_k = \mathbf{\Phi}(t_k, t_{k-1})$ is the state-transition matrix and represents the discrete time motion model. The subscript in $\mathbf{\Phi}_k$ is dropped because it is a constant.

$$\mathbf{\Phi} = e^{(A\Delta t)} = \begin{pmatrix} \mathrm{I} & \Delta t \\ \mathbf{0} & \mathrm{I} \end{pmatrix} \tag{2.24}$$

$H_k$ is the measurement model, which is typically highly non-linear in image processing, but it is considered to be identity here, since the states of the markers are directly observable in the images. The process noise $w_k$ is represented by the following integration.

$$\mathbf{w}_k = \int_{t_{k-1}}^{t_k} \mathbf{\Phi}(t_k, \tau) G(\tau) \mathbf{w}(\tau) d\tau \tag{2.25}$$

This integration cannot be determined since $\mathbf{w}(\tau)$ is a random process and is therefore not integrable. It is thus required to think about the noise from the perspective of the expected mean and variance.

### 2.3.2 Covariance Matrices Q and R

$Q_k$ and $R_k$ represent the covariance in the process noise and measurement noise $N$ respectively.

$$\mathbf{w}_k \sim N(0, Q_k) \tag{2.26}$$

$$\mathbf{v}_k \sim N(0, R_k) \tag{2.27}$$

The Kalman filter assumes that noise in the current time step is completely independent from the noise in the previous time step:

$$E[\mathbf{w}_k\mathbf{w}_j^T] = \begin{cases} Q_k & j = k \\ 0 & j \neq k \end{cases} \tag{2.28}$$

$$E[\mathbf{v}_k\mathbf{v}_j^T] = \begin{cases} R_k & j = k \\ 0 & j \neq k \end{cases} \tag{2.29}$$

$$E[\mathbf{w}_k\mathbf{v}_j^T] = 0 \qquad \forall j, k \tag{2.30}$$

The measurement noise in the case of the wand tracking is very low and so the standard deviations were chosen to be $\sigma_u = \sigma_v = 0.5$ pixels and $\sigma_r = 0.1$ pixels. The 3 measurements are independent and therefore R boils down to a diagonal matrix

$$R_k = \begin{pmatrix} \sigma_u^2 & 0 & 0 \\ 0 & \sigma_v^2 & 0 \\ 0 & 0 & \sigma_r^2 \end{pmatrix} = \begin{pmatrix} 0.25 & 0 & 0 \\ 0 & 0.25 & 0 \\ 0 & 0 & 0.01 \end{pmatrix} \qquad \forall k \tag{2.31}$$

In the continuous case, the noise $\mathbf{w(t)} = [\mathbf{0} \quad \sigma_a]^T$, such that

$$Q = E[\mathbf{w}\mathbf{w}^T] = E\left[\begin{bmatrix} 0 \\ \sigma_a \end{bmatrix}\begin{bmatrix} 0 & \sigma_a \end{bmatrix}^T\right] = \sigma_a^2 \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathrm{I} \end{bmatrix} \tag{2.32}$$

$Q_k$ can now be discretized by integration of the continuous process-noise matrix and the fundamental matrix [29].

$$Q_k = \int_0^{\Delta T} \Phi(\tau) Q \Phi^T(\tau) d\mathbf{t} = \sigma_a^2 \begin{pmatrix} \frac{(\Delta t)^3}{3} \mathrm{I}_{3\times3} & \frac{(\Delta t)^2}{2} \mathrm{I}_{3\times3} \\ \frac{(\Delta t)^2}{2} \mathrm{I}_{3\times3} & \Delta t \mathrm{I}_{3\times3} \end{pmatrix} \tag{2.33}$$

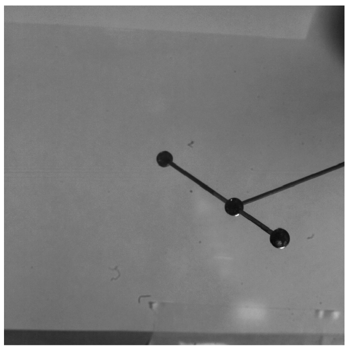
An over approximation of $\sigma_a = 10$ pix/frame $= 1e^5$ pix/s was used as an initial estimate for the acceleration noise intensity for a frame capture rate of 100 fps. The Kalman filter is quite sensitive to the choice in Q and R and these parameters often need to be tuned to achieve optimal performance.

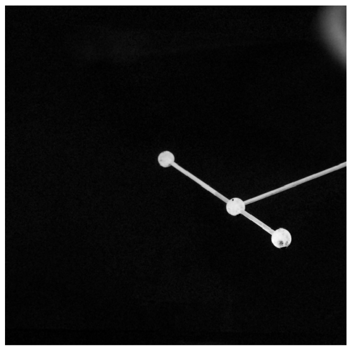
### 2.3.3 Measurements from Images

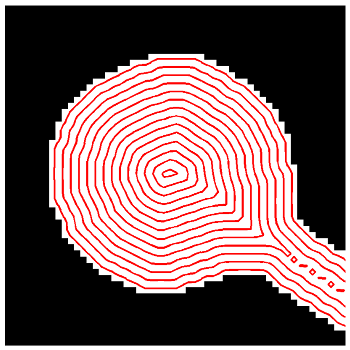
The Kalman filter required some form of measurement using the images $I \in \mathbb{R}^{n,m}$ to track the wand markers frame by frame in the 3 camera views. The resolution of $I$ was set to $n \times m = 800 \times 800$ throughout the thesis. The camera set-up, being in a controlled lab environment, permits a homogeneous background model $B \in \mathbb{R}^{n,m}$ during the calibration recording. This simplified the foreground and background determination during image processing. Since the calibration was performed off-line, a background model for each view was easily initialized by calculating the median intensity of every pixel in the image. This was accomplished by storing a sub-set of images distributed linearly across the full video into a 3D matrix in MATLAB and calculating the median along the time varying dimension of the matrix. A custom function utilizing the *memory* routine in MATLAB to detect the memory settings of the computer was used to establish the number of frames that could be read safely into memory. Maximizing the number of images in this procedure is beneficial for initializing $B$ when there is lots of motion in the video, however good backgrounds have been achieved with as few as 10 frames. Typically, $B$ must be continually updated to account for fluctuations in lighting and background motion. It was determined that for the purposes of calibration this was not necessary; however, in Chapter 4 a method for updating the background model will be introduced. The Foreground $F \in \mathbb{R}^{n,m}$ was found by background subtraction for every k$^{th}$ frame.

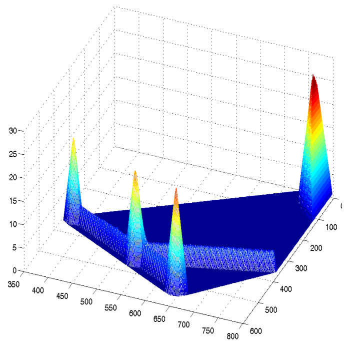
$$F_k = I_k - B \tag{2.34}$$

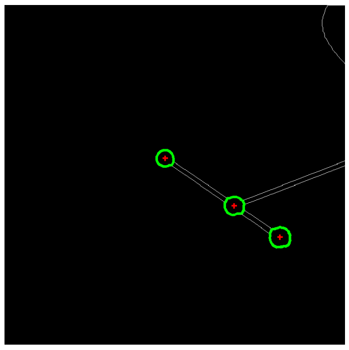
A lowpass adaptive Wiener filter was applied to $F_k$ and a threshold of 3 standard deviations above the mean was used to filter out additional noise. The nearest background pixel to every foreground pixel was calculated using the *bwdist* routine. The result is an effective image of the foreground density $\rho_{F_k}$. Figures 2.4(c) and 2.4(d) illustrate the result as a 2D contour plot and surface plot respectively.


(a)


(b)


(c)


(d)


(e)

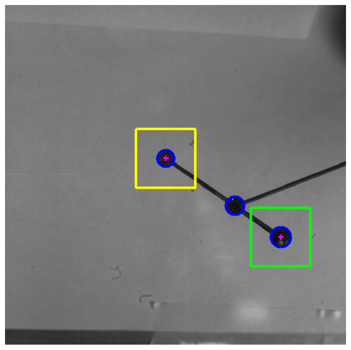

(f)

Figure 2.4: Image analysis procedure: (a) Raw Image, (b) Foreground extraction, (c) Foreground density contour plot, (d) Local Maxima Search, (e) Foreground edge detection, (f) Final Result.

The local maxima tend to reveal the center of the markers, with occasional background noise such as shadows making its way past the foreground threshold. To account for noise, after determining the local maxima using the *imregionalmax* routine, the subspace of maxima are searched such that the 3 predictions lie along the same line. This is accomplished simply by a slope verification procedure in which the slope formed by markers $A$ and $B$, $B$ and $C$, and thus $A$ and $C$ must be approximately equal to within a small threshold. The final position measurement of point $A$ is given by $[u_A, v_A]$ and the radius measurement is given by $r_A = \rho_{F_k}(u_A, v_A)$. The pixel values of the center marker B are always between the other 2 markers, and the distance of marker A is always closer to marker B than the distance between B and C. Therefore the process is auto initialized, requiring only that the user verify that markers are visible in all 3 views at the starting frame.

To increase the robustness to noise and other possible failure modes, adaptive bounding boxes, guided by the Kalman filter are drawn around the marker centers and propagated by the motion model in every frame. Additionally, the size of the bounding box is updated frame by frame based on the velocity states.

### 2.3.4 Kalman Filter Procedure

Once the motion model $\mathbf{\Phi}_k$, measurement model $H_k$, and noise covariance matrices $Q_k$ and $R_k$ are chosen, it is possible to initialize the Kalman filter. The states were initialized automatically, requiring only that the user start on a frame in which the 3 wand markers are visible in all 3 views. The initial velocities were assumed zero and allowed to converge as the Kalman filter progressed.

$$P_k^- = E[(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)(\hat{\mathbf{x}}_k^- - \mathbf{x}_k)^T] \tag{2.35}$$

The initial a priori error covariance matrix, $P_k^-$, given by Equation 2.35 was initialized as a diagonal matrix $P = diag\begin{bmatrix}\mathbf{p}_s & \mathbf{p}_v\end{bmatrix}$ , with $p_s = 1$ and $p_v = 1000$. Higher values of p consider the motion model more uncertain and thus the Kalman filter weights the measurement more at the start. The 2 steps of the Kalman filter are given in Table 2.2. The procedure involves a propagation update, in which previous estimations are used to predict future states. The a priori error covariance matrix, $P_k^-$, is updated using the previous a posterior error covariance $P^k$. The Kalman gain, $K_k$, can then be determined and is used to update the current measurement, $\mathbf{z}_k$. A new $P_k$ is determined for the next frame.

Table 2.2: **Kalman Filter Algorithm**

| Propagation Update | | Measurement Update | |
|---|---|---|---|
| $\hat{\mathbf{x}}_k^- = \mathbf{\Phi}\hat{\mathbf{x}}_k$ | (2.36) | $K_k = P_k^- H^T \left(H^t P_k^- H + R\right)^{-1}$ | (2.38) |
| $P_k^- = \mathbf{\Phi} P_k \mathbf{\Phi}^T + Q$ | (2.37) | $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k(\mathbf{z}_k - H\hat{\mathbf{x}}_k^-)$ | (2.39) |
| | | $P_k = (\mathbb{I} - K_k H) P_k^-$ | (2.40) |

## 2.4 Results

For a wand with an arbitrarily chosen marker length of $\ell = 73.482$ mm, approximately 3-5 minutes of auto tracking resulted in 2D wand data being extracted in 58-133 frames for each of the 3 camera views. The tracking procedure automatically detected when a marker was exiting the view of one of the cameras and would reinitialize, requiring only that the user verify visibility of the wand markers in the next initialization frame. Each viewing angle had 3 separate Kalman filters acting on the 3 wand markers, resulting in 18 data points being saved per frame. The data was loaded into Ty Hedrick's DLT software and the resulting calibration was performed. The same frames were also manually digitized for comparison. The results for 3 different size wands illustrate the increase in calibration performance.

Table 2.3: Calibration Statistics

| | Trial # 1 | | Trial # 2 | | Trial # 3 | |
|---|---|---|---|---|---|---|
| | Auto | Manual | Auto | Manual | Auto | Manual |
| N (frames) | 133 | 133 | 101 | 101 | 58 | 58 |
| $\bar{\ell}$ (mm) | 73.4810 | 73.4815 | 37.9393 | 37.9345 | 17.3799 | 17.385 |
| $\sigma_\ell$ (mm) | 0.0635 | 0.3194 | 0.0496 | 0.2347 | 0.0584 | 0.2955 |
| $q_\ell$ (%) | 0.09 | 0.3235 | 0.13 | 0.62 | 0.34 | 1.7 |
| Time (min) | ∼5 | ∼95 | ∼3 | ∼72 | ∼4 | ∼50 |

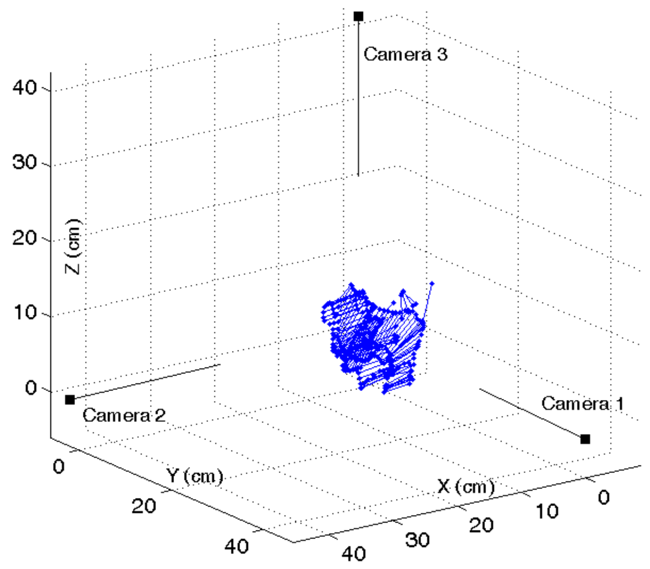


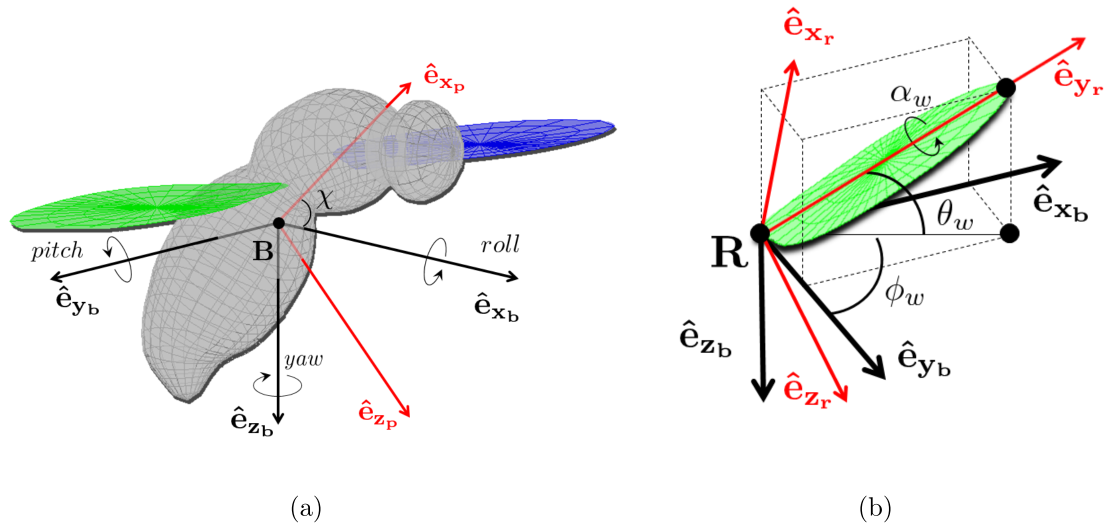
Figure 2.5: camera inertial coordinate frame

The time savings using this auto tracking method is on the order of 50 times faster and the standard deviation is decreased by a factor of 5. In the past, manual calibrations were only performed on up to 50 frames because the increase in quality for adding more frames was negligible. With this new technique however, an abundance of frames can be digitized and poor quality frames can easily be disregarded in an iterative procedure until the calibration quality reaches some steady state. The full calibration procedure, including the filming of calibration video has been reduced down to a $\sim$ 10 min process. Figure 2.5 illustrates the resulting camera inertial coordinate frame, $\mathbf{I_C}$, the origin being the center of the point cloud. This should not be confused with the individual camera reference frame **C** shown in Figure 2.3, in which the origin is the camera. The global reference frame, $\mathbf{I_G}$ shares the same origin as the multi-camera reference frame **C**, however the y and z axes are flipped by an $R_{GC}$ rotation such that z points down, customary to flight dynamics.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_G = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_C = R_{GC} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_C \tag{2.41}$$

# Chapter 3

# Manual Kinematics Extraction



$\hat{\mathbf{e}}_{\mathbf{x_p}}$
$\chi$
pitch
B
roll
$\hat{\mathbf{e}}_{\mathbf{y_b}}$
$\hat{\mathbf{e}}_{\mathbf{x_b}}$
yaw
$\hat{\mathbf{e}}_{\mathbf{z_b}}$
$\hat{\mathbf{e}}_{\mathbf{z_p}}$
(a)
$\hat{\mathbf{e}}_{\mathbf{x_r}}$
$\alpha_w$
$\hat{\mathbf{e}}_{\mathbf{y_r}}$
$\theta_w$
$\hat{\mathbf{e}}_{\mathbf{x_b}}$
R
$\phi_w$
$\hat{\mathbf{e}}_{\mathbf{z_b}}$
$\hat{\mathbf{e}}_{\mathbf{z_r}}$
$\hat{\mathbf{e}}_{\mathbf{y_b}}$
(b)

Figure 3.1: Illustration of the insect reference frame: (a) Body coordinate axis (b) Right wing coordinate axis.

## 3.1 Manual Tracking Methods

A modified version of the open source DLT program [13] was used to track 6 landmarks on the fly: head, tail, left and right wing tips, and left and right wing hinges. The 11 DLT coefficients per camera were loaded into the program along with the desired flight sequence videos. The calibration coefficients map image pixel coordinates $[u, v]$ to three-dimensional coordinates $[X_C, Y_C, Z_C]$ by Equations 2.8-2.9. A 3D coordinate must be defined as pixel coordinates in at least 2 camera views.

The process is made easier by an auto-generated epipolar line projected onto the 2 other camera views, following the pixel coordinate definition in the first view. This method allowed for relatively quick manual digitization and avoided redundancy by requiring a point to be defined by only 2 out of the 3 views. Additional time was saved by interpolation. The body is known to behave linearly over the course of a single wing stroke, therefore the head, and tail points were extracted every 20th frame, which equates to a frequency of 375 hz, or 65-70 % of a wing stroke. The hinges were digitized twice per wing stroke at the maximum tip to tip wing positions. The wing tips were digitized every 3 frames, however care was given to digitize frame by frame during stroke reversal to avoid artificially damping the stroke amplitude peaks ($\phi_{max}$). This interpolation method proved to reduce human induced noise by acting as a temporal filter. Consistency during digitization of landmarks on a biological specimen is known to be a very difficult task and the user error was great enough that on a frame to frame basis, huge accelerations would be induced by failure to pick out the exact pixel locations. Interpolating over larger time intervals acted to smooth and filter some of the user error involved.

The body euler angles $(\phi, \theta, \psi)$ are the 3 angles that make up the 3-2-1 rotation matrix, $R_{bg}$, which maps any vector in the global coordinate system, $\mathbf{g}$, to the body coordinate system, **B**.

$$R_{bg} = R_1(\phi)R_2(\theta)R_3(\psi) \tag{3.1}$$

$$R_1(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & sin(\phi) \\ 0 & -sin(\phi) & cos(\phi) \end{bmatrix} \tag{3.2}$$

$$R_2(\theta) = \begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{3.3}$$

$$R_3(\theta) = \begin{bmatrix} cos(\psi) & sin(\psi) & 0 \\ -sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

These 3 angles were found via the 6 digitized landmark points on the fly. First, the center of mass, $\mathbf{r}_{cg}$, was determined. Predictions of $\mathbf{r}_{cg}$ for *Calliphora* and *Drosophila* were made via known offsets from the mean wing hinge position. Defining $\mathbf{P} = (\hat{e}_{x_p}, \hat{e}_{y_p}, \hat{e}_{z_p})$ to be the principal axis coordinate frame, as shown in Figure 3.1(a), then

$$[\mathbf{r}_{cg}]_p = \frac{([\mathbf{r}_{rwh}]_p + [\mathbf{r}_{lwh}]_p)}{2} - [\mathbf{r}_o]_p \tag{3.5}$$

The x-axis offsets were $[x_o]_p = 0.41$ mm for drosophila and $[x_o]_p = 2.1547$ mm for *Calliphora* **??**. The y-axis offsets for both species were assumed zero because of x-z plane symmetry. The z-axis involved direct computation and was assumed to equal the offset from the hinge to the head-tail line of the insect in the Principal

Frame. Typically, $[z_o]_p = -0.0023$ mm for Drosophila and $[z_o]_p = -0.0075$ mm for *Calliphora*. The roll axis of the fly was defined as the vector that joined the tail to head points, $\vec{\mathbf{V}}_{roll} = \mathbf{X}_h - \mathbf{X}_t$, and it was assumed that $[\mathbf{r}_{cg}]_p$ lies along this vector for extracting global to body euler angles. The global to principal axis yaw angle, $\psi_p$, was found by:

$$\psi_p = \tan_2^{-1}\left(\frac{[\vec{V}^y_{roll}]_g}{[\vec{V}^x_{roll}]_g}\right) \tag{3.6}$$

The global to principal axis pitch angle, $\theta_p$, was found by:

$$\theta_p = -\tan_2^{-1}\left(\frac{[\vec{V}^z_{roll}]_g}{\sqrt{[\vec{V}^x_{roll}]_g^2 + [\vec{V}^y_{roll}]_g)^2}}\right). \tag{3.7}$$

In order to determine the global to principal axis roll angle, $\phi_p$, the coordinate system was first rotated by the yaw and pitch angles to an intermediate reference frame $\mathbf{I}''$:

$$[\mathbf{r}_{cg}]_{i''} = R_2(\theta_p)R_3(\psi_p)[\mathbf{r}_{cg}]_g \tag{3.8}$$

Once rotated to this secondary coordinate system, the roll angle could be extracted by using the pitch axis of the fly $\vec{V}_{pitch} = \mathbf{X}_{rwh} - \mathbf{X}_{lwh}$ :

$$\phi_p = \tan_2^{-1}\left(\frac{[\vec{V}^z_{pitch}]_{i''}}{\sqrt{[\vec{V}^x_{pitch}]_{i''}^2 + [\vec{V}^y_{pitch}]_{i''})^2}}\right) \tag{3.9}$$

Once rotated into the principal coordinate system, a fourth rotation about the pitch axis was performed to define the fly in the body coordinate axis, also known as the stability coordinate axis or the stroke plane coordinate axis. The determination of the second pitch angle, $\chi$ is based on what is considered to be the true body coordinate axis. Biologists typically define $\chi$ as the angle that rotates the fly into the stoke plane frame, whereas flight dynamics engineers define $\chi$ as the trimmed pitch angle about some reference flight condition. Typical definitions range from 45°-60° for Drosophila and 30°-45° for *Calliphora*. The definition is rather arbitrary however because $\chi$ can always be redefined and used to correct the wing and body angles as a post process procedure. Therefore, it is standard to define $\chi$ as a constant, which allows for easier comparison of flight sequences. A result of this fourth rotation is an over-defined rotation matrix, therefore the 3 effective euler angles are recovered from this redundant rotation by:

$$\mathbf{r}_b = R_2(\chi)R_1(\phi_p)R_2(\theta_p)R_3(\psi_p)\mathbf{r}_g = R_2(\chi)R_{pg}\mathbf{r}^g = R_{bg}\mathbf{r}_g \quad (3.10)$$

$$\phi_b = sgn(R_{bg}(2,3))cos^{-1}\left(\frac{R_{bg}(3,3)}{\Delta}\right) \quad (3.11)$$

$$\theta_b = sin^{-1}\left(-R_{bg}(1,3)\right) \quad (3.12)$$

$$\psi_b = sgn(R_{bg}(1,2)cos^{-1}\left(\frac{R_{bg}(1,1)}{\Delta}\right) \quad (3.13)$$

$$\Delta = \sqrt{1 - R_{bg}(1,3)^2} \quad (3.14)$$

In order to extract the wing euler angles $(\theta_w, \alpha_w, \phi_w)$, a similar series of angle extraction and rotations must be performed. Focusing on the right wing, the span vector of the wing is found as the vector joining the wing hinge to the wing tip, $\vec{\mathbf{V}}_r = \mathbf{X}_{rwt} - \mathbf{X}_{rwh}$. The first rotation from the body to wing coordinate system is an $R_3$ about the $z_b$ axis, called the stroke amplitude angle, $\phi_w$.

$$\phi_{rw} = \tan_2^{-1}\left(\frac{-[\vec{V}_r^x]_b}{[\vec{V}_r^y]_b}\right) \tag{3.15}$$

$$\phi_{lw} = -\tan_2^{-1}\left(\frac{-[\vec{V}_l^x]_b}{[\vec{V}_l^y]_b}\right) \tag{3.16}$$

The next rotation,called the stroke elevation angle, $\theta_{rw}$, is an $R_1$ rotation about the $x'$ axis and can be found without having to first rotate the reference frame by $\phi_w$ using the following:

$$\theta_{rw} = \tan_2^{-1}\left(\frac{[\vec{V}_r^z]_b}{\sqrt{[\vec{V}_r^x]_b^2 + [\vec{V}_r^y]_b^2}}\right) \tag{3.17}$$

$$\theta_{lw} = \tan_2^{-1}\left(\frac{[\vec{V}_l^z]_b}{\sqrt{[\vec{V}_l^x]_b^2 + [\vec{V}_r^y]_b^2}}\right) \tag{3.18}$$

The final rotation, called the wing pitch, $\alpha_{rw}$, is a $R_2$ rotation about the $\hat{e}_{y_w}$ axis, but cannot be determined from the 6 digitized landmarks. Instead a wing pitch fitting procedure using a geometric model of the insect wing was performed. Using MATLAB's image processing toolbox, an image of an insect wing was used to define a wire-frame as seen in Figure 3.2 on page 37. The wire-frame was transformed by the previously defined body and wing states, projected directly onto the videos, and $\alpha_w$ was visually adjusted to match the videos frame by frame. This span-wise rotation corresponded to an inverse rotation about the $y_b$ axis, $-R_2(\alpha_w)$, and thus allowed for the rotation to be done last. The angle, $\alpha$ found was applied to the body-wing rotation sequence:

$$\mathbf{r}_W = R_2(\alpha_w)R_1(\theta_w)R_3(\phi_w)\mathbf{r}_b = R_{WB}\mathbf{r}_b$$

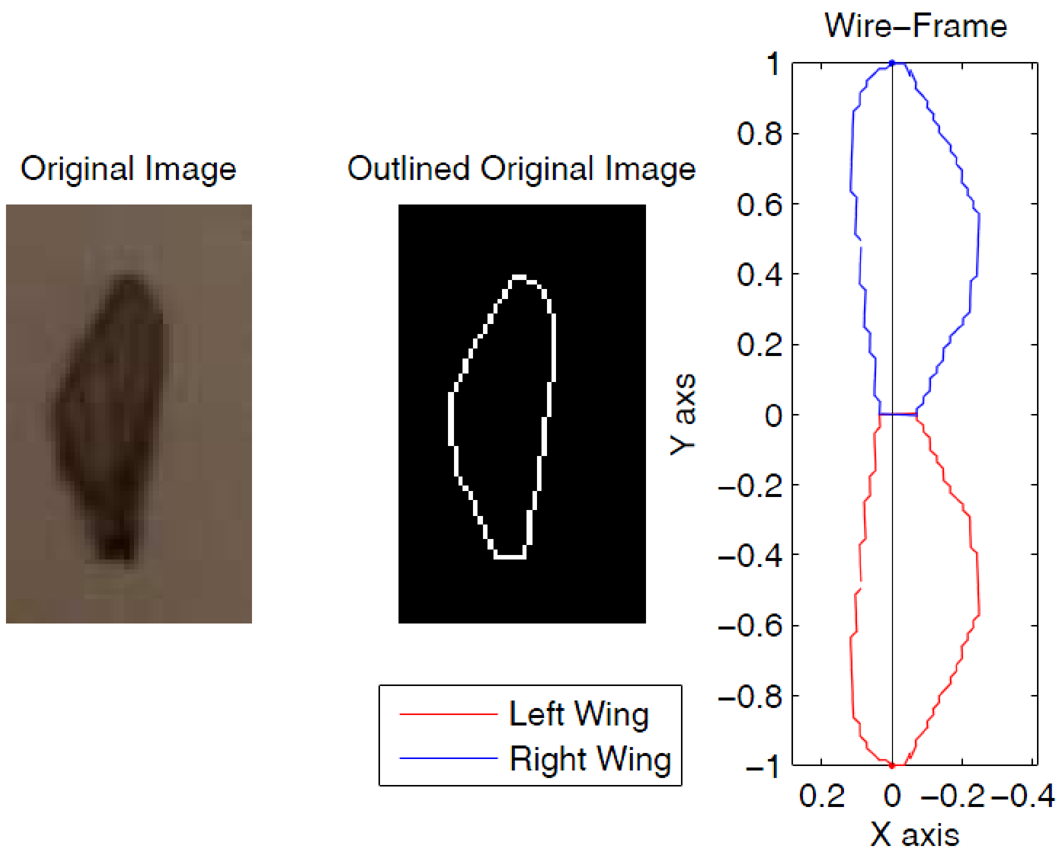


Figure 3.2: Wing wire frame extraction

## 3.2 Kinematic Visualization

Visualization tools were essential for detecting digitizing errors. Therefore upon completion of a digitized sequence, the 6 landmark points were graphically illustrated in the global, principal, and stability coordinate reference frames.

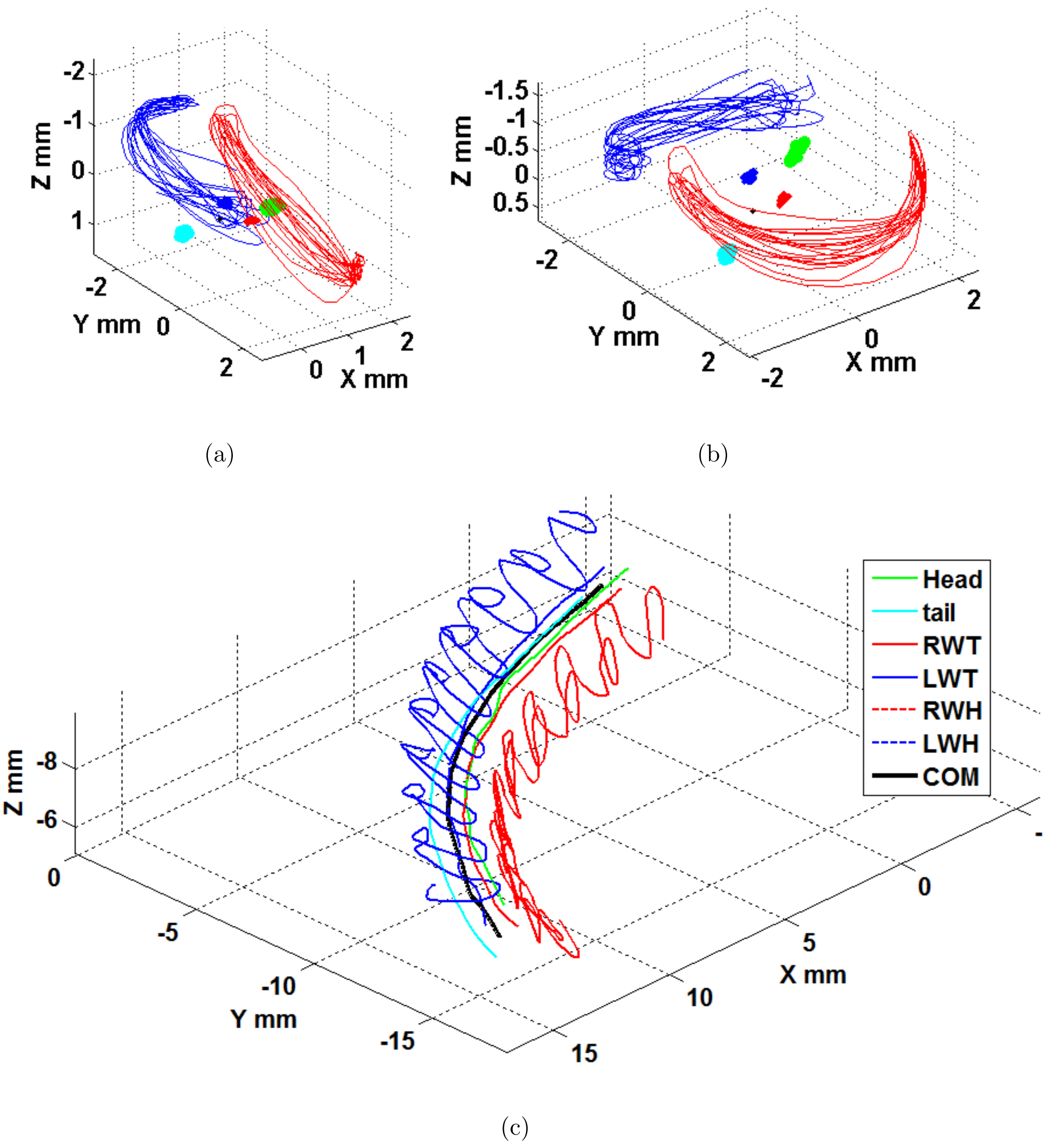


Figure 3.3: Time history for a Drosophila coordinated turn in: (a) Principal Coordinate Frame, (b) Stability/Body Coordinate Frame, (c) Global Coordinate Frame.

Similarly, the wing and body kinematics were plotted as functions of time for detecting digitizing errors and visualizing the wing motions that generate the body maneuvers.

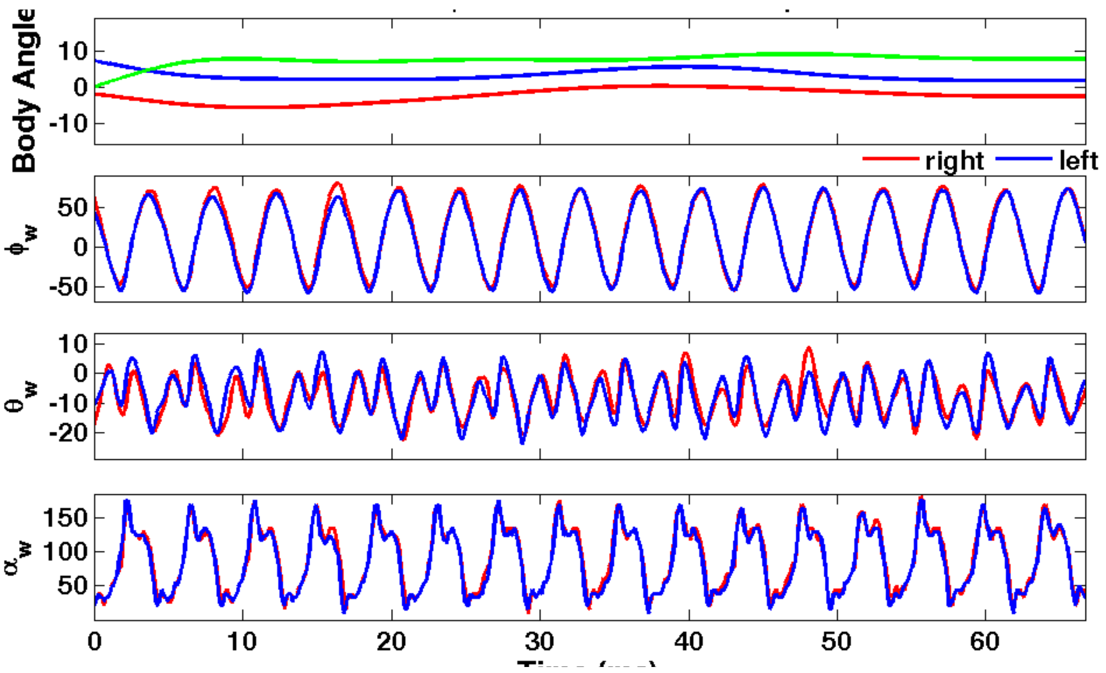


Figure 3.4: Body and wing euler angle time history for the same *Drosophila* coordinated turn sequence shown in Figure 3.3(a)-3.3(c)

Lastly, all the kinematic data was combined into a MATLAB generated GUI to display the actual camera views with the kinematic time history along with a simulated model of the insect. The power of these visualization tools were vastly important for studying the complex wing motions that enable rapid maneuvers in addition to minimizing kinematic errors induced by a human digitizer.

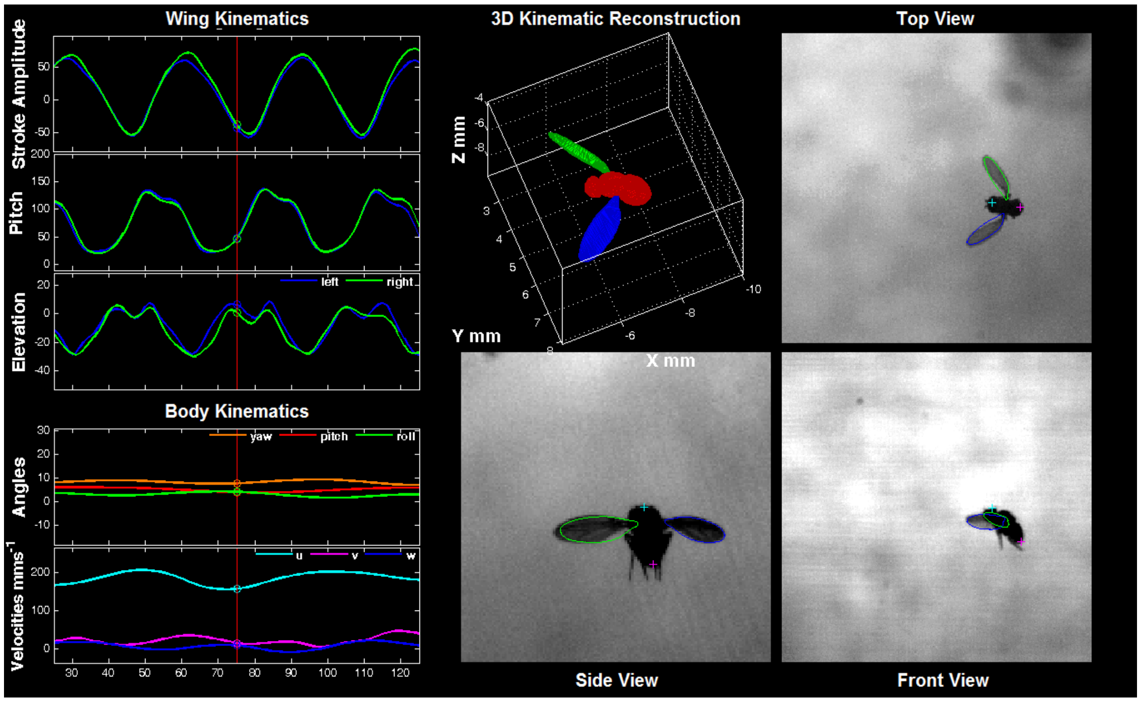
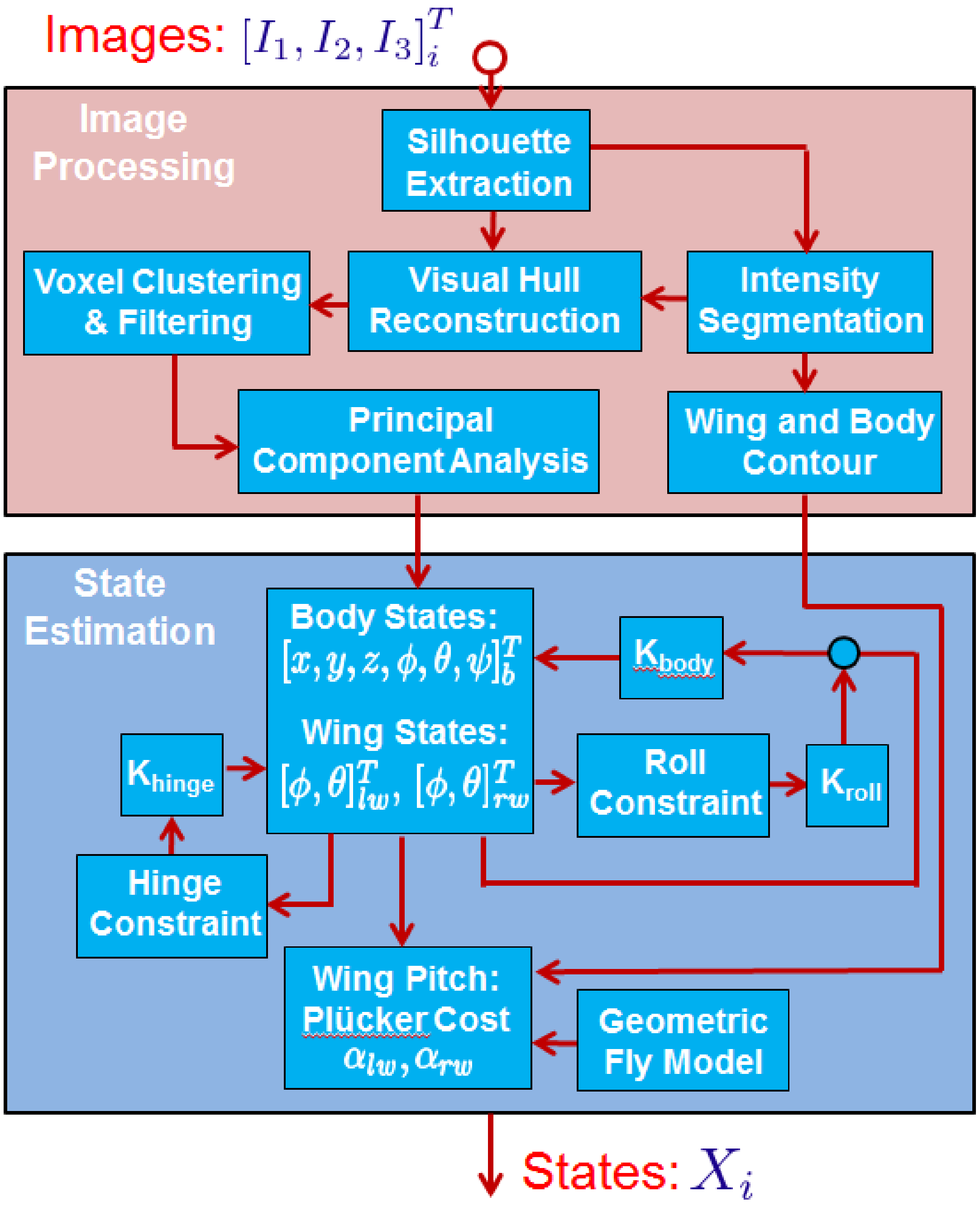


Figure 3.5: Body and wing euler angle time history for a Drosophila coordinated turn

# Chapter 4

# Automated Kinematics Extraction



Images: $[I_1, I_2, I_3]_i^T$
Image Processing
Silhouette Extraction
Voxel Clustering & Filtering
Visual Hull Reconstruction
Intensity Segmentation
Principal Component Analysis
Wing and Body Contour
State Estimation
Body States: $[x, y, z, \phi, \theta, \psi]_b^T$
Wing States: $[\phi, \theta]_{lw}^T, [\phi, \theta]_{rw}^T$
$K_{body}$
$K_{hinge}$
Roll Constraint
$K_{roll}$
Hinge Constraint
Wing Pitch: Plücker Cost $\alpha_{lw}, \alpha_{rw}$
Geometric Fly Model
States: $X_i$

Figure 4.1: Proposed Automated Tracking Algorithm

## 4.1 Silhouette Extraction

As described in Chapter 2, a background model for each view was initialized by calculating the median intensity of every image pixel across a linearly distributed set of images in the video. Unlike the calibration videos however, the flight chamber usually contained multiple flies, on the order of a hundred for *Drosophila* recording. Therefore a background model update was applied using the approximate median method [17]. If the intensity of the image pixel $I_{i,j}$ in the current frame was greater than the corresponding background pixel, the background pixel's intensity was incremented by 1. Similarly the background pixel's intensity was decremented by 1 when the opposite was true.

$$[B_{i,j}]_{k+1} = \begin{cases} [B_{i,j}]_k + 1 & \text{if} \quad [I_{i,j}]_k > [B_{i,j}]_k \\ [B_{i,j}]_k - 1 & \text{if} \quad [I_{i,j}]_k < [B_{i,j}]_k \end{cases} \tag{4.1}$$

The background eventually converges to a state where half the pixels are greater than the input pixels and the other half are less than the input pixels. Upon initialization of the automated tracker, background subtraction was performed to find $F$ and an intensity threshold $= \mu_F + 7\sigma_F$ was applied to convert $F$ to a binary foreground image $F_b$. The MATLAB routines *regionprops* and *bwconncomp* were used for blob detection and analysis of size and shape. A series of morphological filtering was applied to the blobs to fill in holes and remove sparse pixels. Upon initialization of the automated tracker, if more than 1 candidate blob was considered to be a fly, a user interface was opened to allow the user to select which fly to track. The centroid of the blob corresponding to the fly was considered as a Markov process for determining the silhouette of the fly in subsequent frames. This assumption was well suited for the high frame rates used, however in the event that 2 flies crossed paths in the perspective of one of the camera views, the silhouette extracted would contain the effective shape of the 2 flies. The reconstruction of the fly from the 3 silhouettes eliminated this issue.

## 4.2 Visual Hull Reconstruction

Visual Hull Reconstruction, also known as Shape from Silhouette (SFS), refers to the technique of constructing the 3D shape of an object, using the corresponding object contours from multiple views. The resulting reconstruction contains the shape maximally consistent with the silhouettes. As more cameras are added, the view becomes less occluded and thus the visual hull becomes increasingly refined and more consistent with the true shape. Regardless of the number of cameras, concavities in the true object and contour segmentation errors in the camera views, and camera calibration errors lead to reconstruction errors known as artifacts.

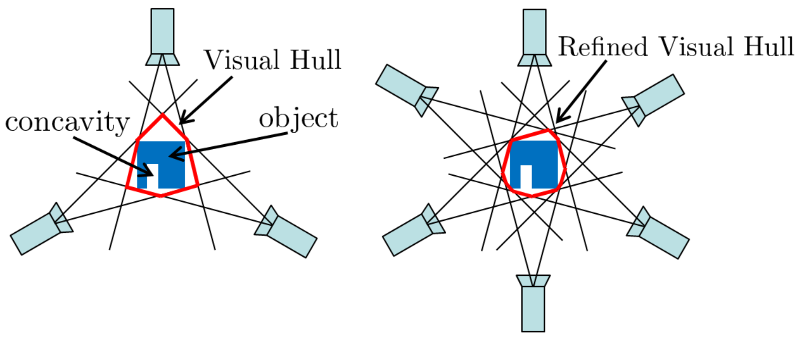
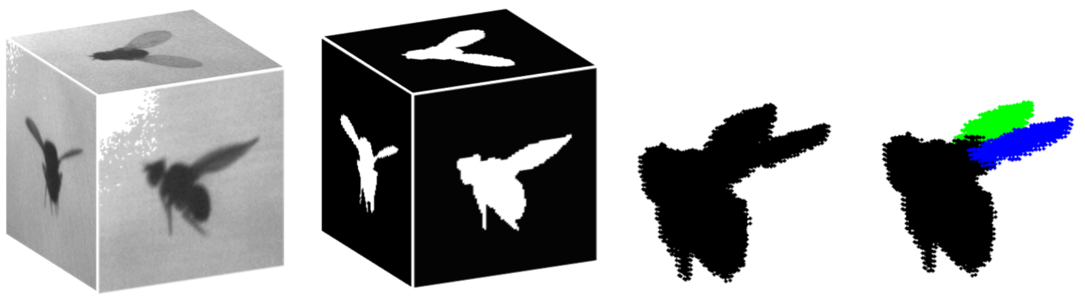


Figure 4.2: Visual hull reconstruction. Increasing number of cameras decreases occlusions, but fails to observe concavities in the object

This approach was implemented by initializing a 3D bounding cube, defined by the 3 bounding boxes from the 3 camera views. The 12 bounding corners were used to estimate the bounding cube, which was then divided into millions of individual voxels, or 3D pixel. Every voxel was back projected onto the images and discarded if it was not inside the silhouette in all 3 views. This approach differs from the HRMT method [21] in that a camera calibration was used, allowing for flexibility in the camera set-up, whereas HRMT assumes an orthogonal camera set up to extrude the images into 3D space. Moreover, no scaling corrections were necessary to conduct the reconstruction, whereas HRMT requires resizing and repositioning of the silhouette bounding boxes before the extrusion process can be conducted.



(a) (b) (c) (d)

Figure 4.3: Visual hull reconstruction steps. (a) Raw Images (b) Image Silhouettes (c) Visual hull reconstruction (d) K-means segmentation

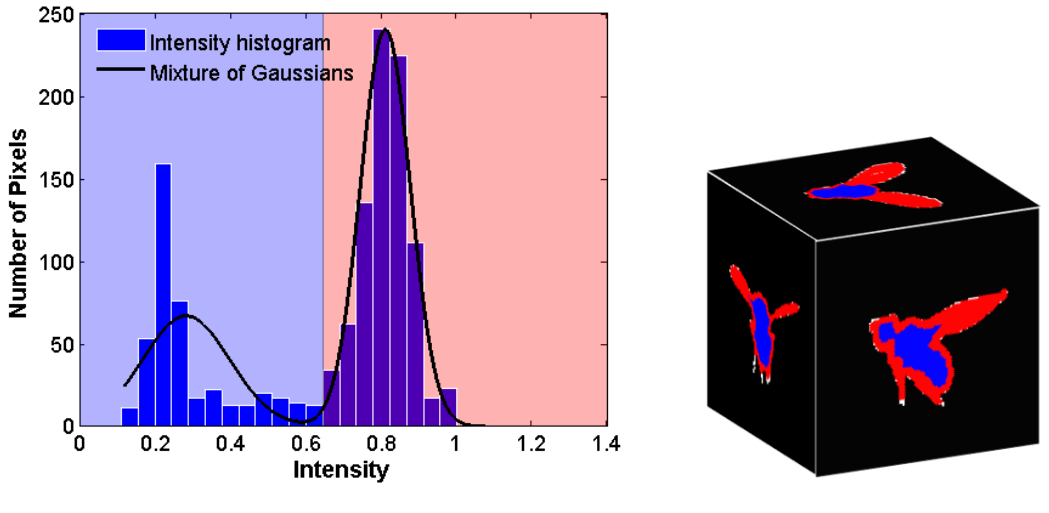
## 4.3 Clustering and Segmentation Techniques

In order to study the wing and body motions separately, the fly was segmented into 3 components: body, right wing, and left wing. One of the simplest voxel segmentation techniques is the k-means clustering algorithm, which simply partitions $n$ observations into $k$ sets $\mathbf{S} = \{S_1, S_2, \ldots, S_k\}$, while attempting to minimize the Within Cluster Sum of Squares (WCSS). This is the method employed by HRMT.

$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{z_j \in S_i} \|z_j - \mu_i\|^2 \tag{4.2}$$

One issue observed with this method is the inherent attempt to cluster voxels into elliptically shaped regions, which does a poor job as artifacts get introduced into the reconstruction or when the wings flap excessively contiguous to one another. In computer vision, to deal with unwanted reconstruction artifacts, photo consistency can be used to further disregard voxels that do not contain similar pixel intensities. In color images, there are 3 intensities present for every pixel: red,blue green. However, high speed videography is typically limited to monochrome video and furthermore, textures are generally not observable due to the intense back lighting required to reduce the exposure and prevent blur. The resulting image results in a shadow of a fly that is generally bimodal in appearance. This permits the silhouette of the fly to be broken down into: body pixels, represented by darker (lower intensity) pixels and wing pixels, which are brighter (higher intensity) pixels due to the transparency of the wings permitting more of the back lighting to pass through.

### 4.3.1 Intensity Based Image Segmentation



(a) (b)

Figure 4.4: Mixture of Gaussians used to determine body/wing segmentation threshold. (a) Intensity histogram and gaussian mixture of the top camera view (b) Resulting segmentation of a typical frame

The cut off threshold between body and wing pixels was accomplished by using the Expectation-Maximization algorithm to solve for a Gaussian Mixture Model. The unknowns parameters are $\theta_k = [\mu_k, \Sigma_k]^T$, where $\mu_k$, $\sigma_k$ are the mean and covariance of the $k^{th}$ Gaussian mixture respectively. In the case of fly silhouette segmentation, there are k=2 Gaussian mixtures to determine. Each intensity observation, $z_i$, is represented by a density function $p(z_i|\theta)$.

$$p(z_i|\theta) = \sum_{k=1}^{K} \alpha_k f(z_i|\theta_k) \tag{4.3}$$

The weighting parameters are given by $\alpha_k$, where $\sum_{i=1}^{k} \alpha_i = 1$. Each component density, $f(z_i|\theta_k)$, is a Gaussian probability distribution with mean $\mu_k$ and covariance $\Sigma_k$.

$$f(z_i|\theta_k) = \frac{exp[-\frac{1}{2}(z_i - \mu_k)^T \Sigma_k^{-1}((z_i - \mu_k)]}{\sqrt{(2\pi)^n det(\Sigma_k)}} \tag{4.4}$$

The Expectation-Maximization algorthim is used to solve the complete set of unknown Gaussian mixture parameters, $\Theta = [\alpha_1, \ldots, \alpha_k, \theta_1, \ldots, \theta_k]^T$. The EM algorithm is a general iterative method for determining the maximum-likelihood of unobserved latent variables, consisting of an E-step and an M-step. The expectation (E) step involves creating a function, $Q(\Theta, \Theta^{(t)})$, for the expectation of the log-likelihood evaluated at the current estimate of the mixture parameters, $\Theta^{(t)}$ [25].

$$Q(\Theta, \Theta^{(t)}) = \sum_{i=1}^{N} \sum_{k=1}^{K} log[\alpha_k f(z_i|\theta_k] E(m_k^i|z_i; \Theta^{(t)}) \tag{4.5}$$

The posterior probability $E(m_k^i|z_i;\Theta^{(t)})$ is given by:

$$E(m_k^i|z_i;\Theta^{(t)}) = \frac{\alpha_k^{(t)} f(z_i|\theta_k^{(t)})}{\sum_{j=1}^{K} \alpha_j^{(t)} f(z_i|\theta_j^{(t)}} \tag{4.6}$$

The maximization (M) step, involves solving for the parameters that maximize $Q(\Theta, \Theta^{(t)})$. These updated parameters are used as inputs to the E-step in the next iteration.

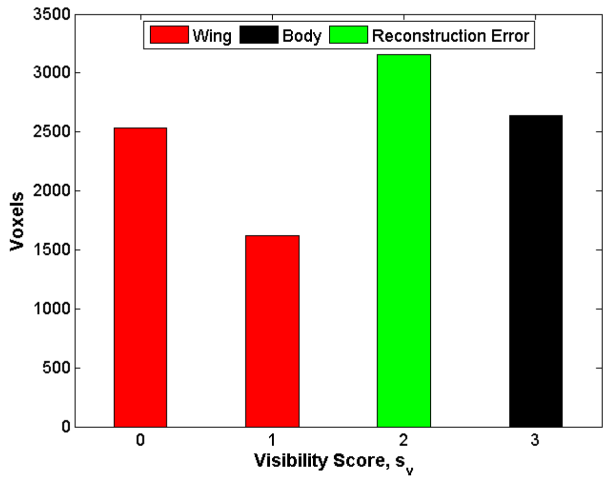
$$\alpha_k^{(t+1)} = \frac{E(m_k^i|z_i;\Theta^{(t)})}{\sum_{k=1}^{K} E(m_k^i|z_i;\Theta^{(t)})} \tag{4.7}$$

$$\mu_k^{(t+1)} = \frac{\sum_{i=1}^{N} z_i E(m_k^i|z_i;\Theta^{(t)})}{\sum_{i=1}^{N} E(m_k^i|z_i;\Theta^{(t)})} \tag{4.8}$$

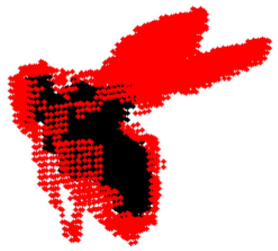
$$\Sigma_k^{2(t+1)} = \frac{\sum_{i=1}^{N} E(m_k^i|z_i;\Theta^{(t)})(z_i - \mu_k^{(t+1)})(z_i - \mu_k^{(t+1)})^T}{\sum_{i=1}^{N} E(m_k^i|z_i;\Theta^{(t)})} \tag{4.9}$$

It is not always cut and dry to segment the images based solely on image intensity. Sometimes the body appears brighter in some regions due to light reflections. A secondary filter to check the number of 2D body clusters present was implemented using *regionprops* and *bwconncomp* routines in MATLAB, keeping only the largest cluster and filling in any holes. Advanced filtering and clustering strategies were developed to deal with the wing pixels.

### 4.3.2 Visibility Score 3D Segmentation



(a)



(b)

Figure 4.5: Intensity Based Voxel Segmentation. (a) Voxel visibility score (b) Resulting segmentation

To segment the visual hull by intensity without the use of k-means clustering, the voxels were back projected onto the original image and assigned a visibility score, $s_v$, based on how many views the projection was considered a body pixel. it was assumed that the body could occlude a wing, but a wing, being mostly transparent could not occlude the body. Therefore a true body voxel would have an $s_v = 3$, meaning the projected voxel corresponds to a body segmented pixel in 3 views. An $s_v = 2$ was considered reconstruction error and an $s_v \leq 1$ corresponded to wing voxels.

$$\mathbf{X}_i = \begin{cases} \text{body} & s_{v_i} = 3 \\ \text{reconstruction error} & s_{v_i} = 2 \\ \text{wing} & s_{v_i} \leq 1 \end{cases} \tag{4.10}$$

### 4.3.3 Geometric Wing Filtering

An unfortunate consequence of this segmentation technique is it requires additional processing to clean up the wing voxels. This was accomplished by a geometric filtering and custom cost analysis in both 2D and 3D space. The geometric filtering was set up to easily filter bad choices for wing voxels based on knowledge of the wing length of the insect. All wing voxels greater than 0.75 $L_w$ from the k-means estimated wing centroids were filtered. Additionally, and wing voxels greater than $0.25L_w$ from the estimated span axis of the wings were filtered.

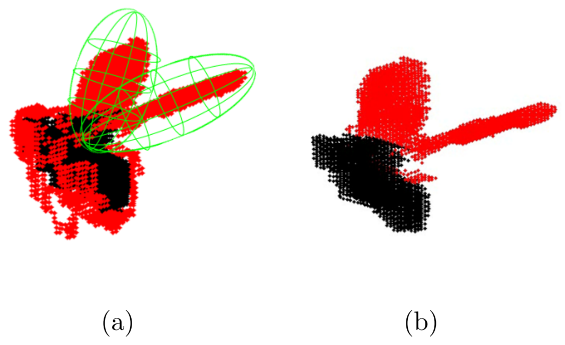


Figure 4.6: Geometric filtering was used for filtering poor wing voxels. (a) Geometric constraint (b) Filtered result

### 4.3.4 Cost Function Filtering

The cost function filtering method involves a much more elegant process and was capable of properly filtering poor pixels without the aid of the initial geometric filtering. This method was preferred when higher computational resources were available. In 2D space, every wing pixel was assigned a value based on several factors. The higher the value, the more likely a pixel corresponds to a true wing pixel.

$$J_{2D} = 4\rho + 4 \parallel d_B \parallel^2 + 2 \parallel d_b \parallel^2 \tag{4.11}$$

The resulting 2 dimensional cost, $J_{2D}$, was summed across the 3 views for every voxel and combined with a similar cost analysis in 3D space.

$$J = 4\sum_{i=1}^{3} J_{2D,i} + 3\rho + 2 \parallel d_B \parallel^2 + \parallel d_{cg} \parallel^2 \tag{4.12}$$

A new term, $d_{cg}$, was introduced as the distance to the nearest estimated wing centroid.

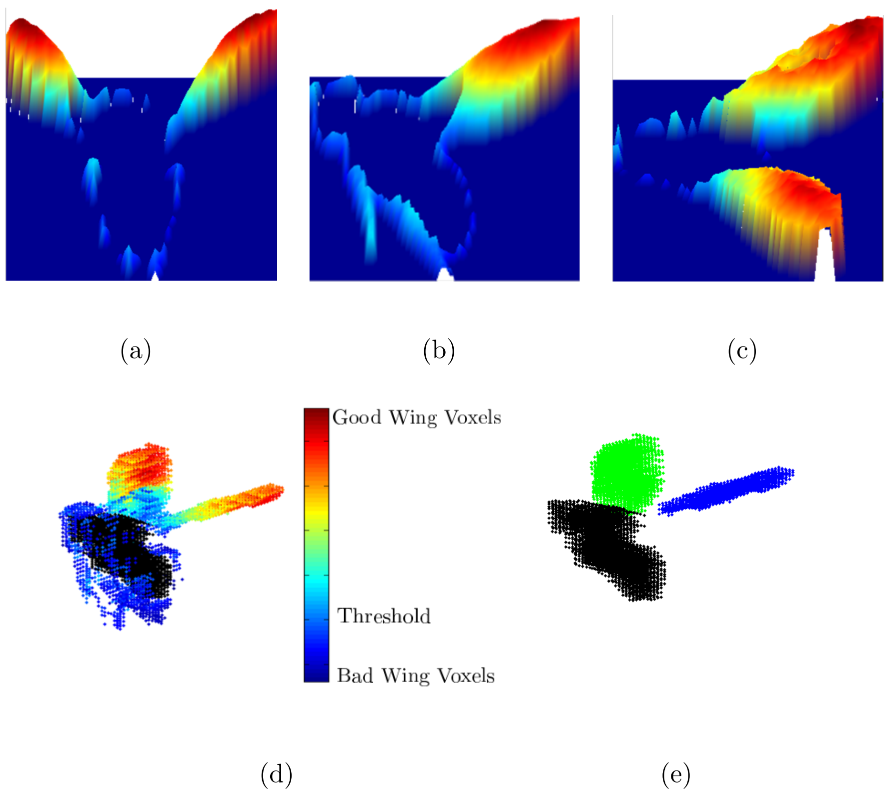


Figure 4.7: Cost analysis of wing pixels used to filter poorly segmented pixels. (a)-(c) 2D cost analysis of each view (d) 3D cost analysis (e) Final segmented result

The final cost function, $J$ was sorted in ascending order and the y intercept of a linear fit was used as the cut off threshold for filtering poor wing voxels. Lastly, the filtered wing voxels were segmented into left and right wings by spectral clustering.

### 4.3.5 Spectral Clustering

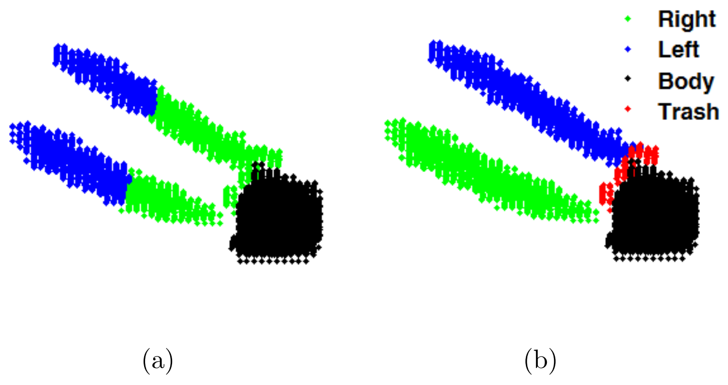


Figure 4.8: Comparison of kmeans clustering and spectral clustering in the case of excessively contiguous wings. (a) kmeans clustering (b) spectral clustering

Spectral clustering refers to a class of advanced clustering techniques, which works off the eigenstructure of a similarity matrix. The particular algorithm tested here was developed by Jordan and Weiss of Berkeley [27]. Typically, a Gaussian similarity function is chosen to build the similarity matrix, or affinity matrix $A$:

$$A_{ij} = \begin{cases} \exp(-\|x_i - x_j\|^2/2\sigma^2) & i \neq j \\ 0 \quad i = j \end{cases} \tag{4.13}$$

The parameter $\sigma$ controls the size of the neighborhoods and can be chosen automatically by iteratively calculating the clusters to minimize the euclidean metric. This method is computationally expensive for large datasets. An alternative method to build a sparse similarity matrix was implemented to deal with the high number of voxels that must be clustered [4]. The mutual k-nearest neighbor similarity graph was used to efficiently build a sparse similarity matrix in which a pair of $x_i$ and $x_j$ data points were considered similar, $s(x_i, x_j) = 1$, only if $x_j$ was among the k-nearest points to $x_i$ and $x_i$ was among the k-nearest points to $x_j$. D was defined as a diagonal matrix, where $D_{ii}$ is the row sum of the $i^{\text{th}}$ row of $A$. The Laplacian matrix was then defined as:

$$L = D^{-1/2} A D^{-1/2}. \tag{4.14}$$

The k-largest eigenvectors were determined and used to define the eigenvector matrix $X$. Each row of $X$ is normalized to have unit length

$$Y_{ij} = \frac{X_{ij}}{\sqrt{\sum_j X_{ij}^2}} \tag{4.15}$$

Each row of $Y$ was then clustered into the desired number of clusters using k-means. The original data $x_i$ was clustered into the j$^{th}$ cluster only if row $i$ of $Y$ was assigned to cluster $j$ [27].

### 4.3.6 Results of Advanced Segmentation Procedure

The resulting 3D reconstruction was found to better describe the true volume of the insect. Filtering poorly defined voxels based on an intensity based visibility metric, geometric constraints, and multi-variable cost functions successfully eliminated unwanted features, such as leg reconstruction in addition to other reconstruction artifacts, which greatly affected the next step in kinematic extraction. Spectral clustering was used as a shape based clustering technique, which was proven to be superior in situations where k-means failed, such as in Figure 4.8. Spectral clustering offered potential to simplify the clustering process by taking the original bimodal intensity segmentation and using the calculated cost function as a fourth data dimension to directly cluster wing voxels into 3 groups: right wing, left wing, and trash. This method required further testing and was left for future work.

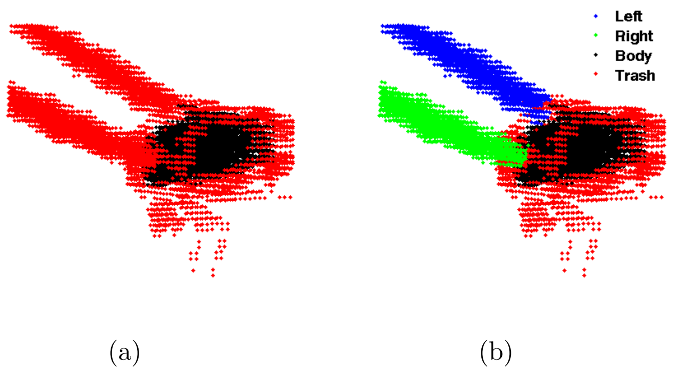


Figure 4.9: Preliminary intensity based segmentation can be directly applied to spectral clustering to filter the poorly resolved wing voxels (a) Intensity segmentation (b) Spectral clustering

To conclude, for off-line kinematic extraction, this multi-step segmentation technique was preferred over the standard k-means clustering technique used in previous studies.

## 4.4 Principal component Analysis

The body roll axis and the wing-span axes were determined using principal component analysis (PCA). The first principal component corresponded to the axis of greatest variance by any projection of the voxel point cloud onto that axis. The inherent shape of the body and wings were ideal for consistently estimating the first principal components to be the body roll axis the wing-span axes.

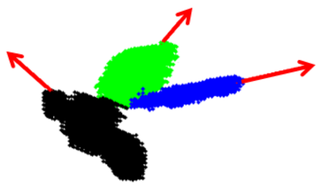


Figure 4.10: First principal component axes of body and wings

The result of PCA on the voxel point clouds replicated the tail to head and hinge to tip vectors determined from manual tracking. The roll axis vector permitted body yaw and pitch measurements. Similarly, the stroke amplitude and elevation angles were extracted by the wing span axis. Upon initialization, the direction of the vector was manually determined and used to predict the vector directions in subsequent frames. Wing bending has been a known issue in all automated insect tracking, however there have been no attempts to study this issue in detail. An innovative approach was developed to study the nature of these errors in addition to improvements upon the PCA-derived wing-span.

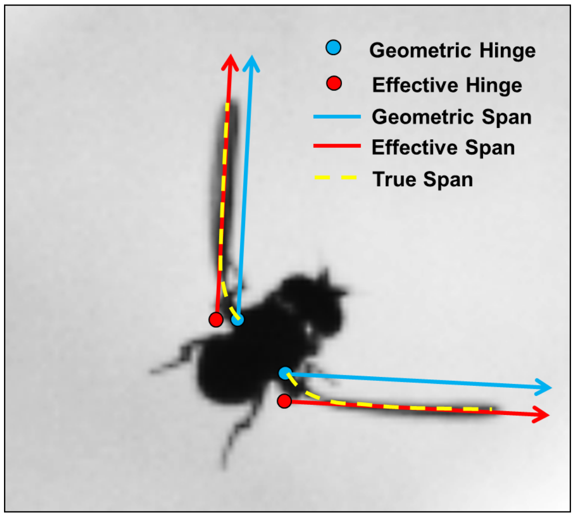


Figure 4.11: The span bending due to high inertias during stroke reversal resulted in effective wing hinges far from the geometric hinge

### 4.4.1 Wing Bending and PCA correction

PCA, although an efficient method to predict the wing-span axes without estimation of the wing hinge or tips, was prone to error during certain phases of the wing stroke. There were two explanations for why this occurred. First, high inertias during stroke reversal in the transversal direction led to effective wing hinges $X_{H_{\text{eff}}}$ far from the geometrically defined hinge. Second, poorly resolved wing reconstructions resulted in the PCA-derived wing-span vectors to inconsistently match with the geometric hinge even when no bending was present. A orthogonal projection of the geometric hinge onto the effective span was used to determine the location of $X_{H_{\text{eff}}}$. Transforming $X_{H_{\text{eff}}}$ to the body axis permitted a method to study the nature of the error. It was clear that the errors induced in the body roll angle due to the roll constraint occurred when $[X_{H_{\text{eff}}}]_b$ deviated in the $y$ and $z$ directions. Although the $x$ component of $[X_{H_{\text{eff}}}]_b$ also appeared noisy, further analysis confirmed this noise was a result of wing bending during stroke reversal. The amount of bending varied from species to species, for *Drosophila* it was observed to occur approximately about the first 10% of the span. Therefore, the $y$ and $z$ directed hinge data was more heavily filtered than the $x$ direction, in order to preserve the bending motion. The dynamics of the hinge states were modeled as a constant for Kalman filtering.

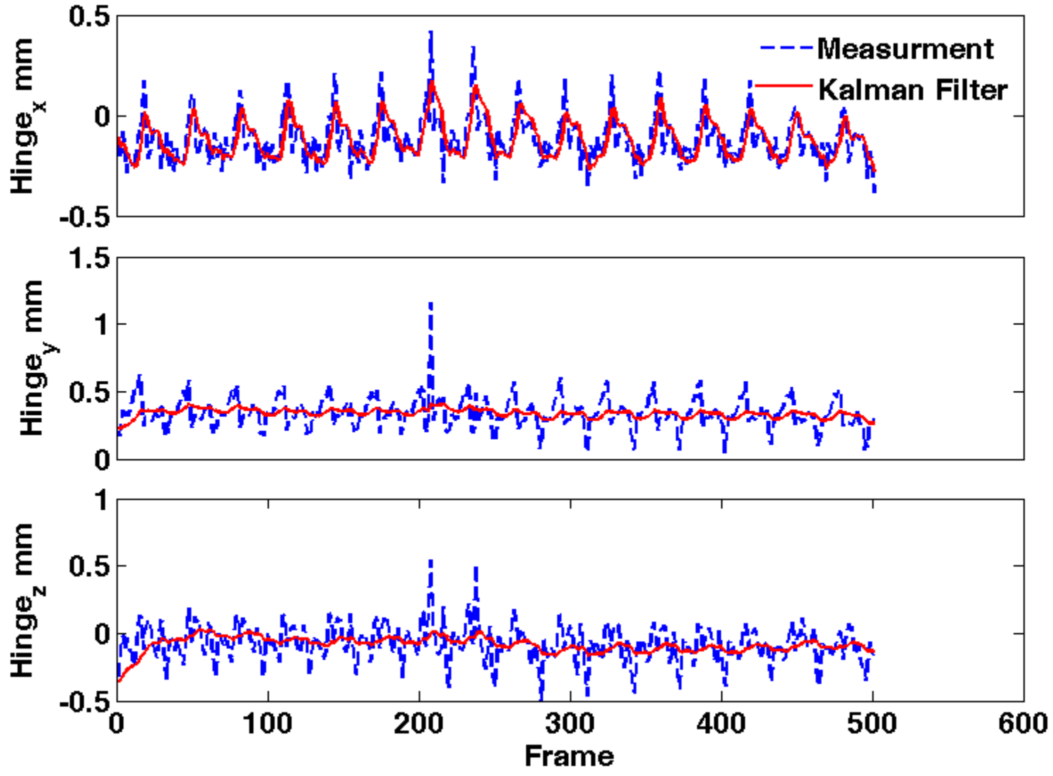


Figure 4.12: The effective hinge $[X_{H_{\text{eff}}}]_b$ measurements before and after filtering

As a correction to the PCA-derived wing-span axis, the vector formed by $X_{H_{\text{eff}}}$ and the effective wing tip $X_{T_{\text{eff}}}$ was used to define a revised wing-span axis. $X_{T_{\text{eff}}}$ was estimated as the furthest voxel along the wing-span from $X_{H_{\text{eff}}}$. Large improvements were made in the wing deviation angle $\theta_w$, which led to much better body roll estimates through the roll constraint explained next.

## 4.5 Body Roll Constraint

The unobservability of the body roll angle is primarily due to the cylindrical shape of the body. Voxel noise induced by shape from methods introduces even more ambiguity. Backlighting and low image resolution prevent the use of textures for roll angle determination. The problem of accurate roll angle estimation as it turns out is one the largest challenges faced in automated motion tracking. Even in manual approaches, the roll angle is the most poorly defined. This is due to the short wing hinge distance from the body centroid, which causes the roll angle to be highly sensitive to manually digitized hinge points. In a semi-automated approach, the other wing and body angles can be determined in an automated fashion and the roll angle can be predetermined by manually extracting the two hinge locations. Tackling this problem in an automated fashion however, has led to some creative approaches. The easiest solution can be to restrict studies to longitudinal flight, with the assumption that the roll angle is equal to zero. This assumption is valid, however flies depend heavily on visual feedback to navigate. By zig-zagging around, a fly can gain a high level of understanding of its environment from the constantly changing optic flow as compared to straight and level flight. Related work on motion tracking of fish revealed that fish don't roll when they turn and so the assumption of zero roll angle is valid in that case [5]. Unfortunately, flies take advantage of thrusting their legs and changing their wing kinematics in such a way as to result in highly banked turns at yaw rates on the order of $5000^o$. Another approach is to break the 3D reconstruction of the body into 3 smaller clusters (head, thorax, abdomen) and perform a cross product on the 2 vectors formed between the 3 centroid locations of these smaller components to determine the pitching axis of the body. The angle this vector makes with respect to the unit yaw vector is the roll angle [21]. This method is very noisy due to the constant fluctuation in the body reconstruction which involves k-means segmentation of body from wings and then body into 3 separate components.

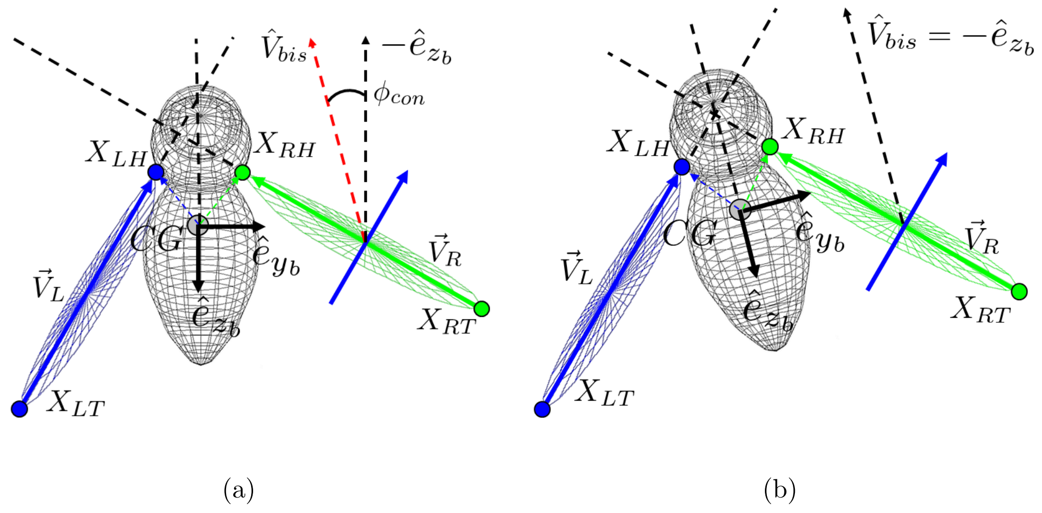


Figure 4.13: Roll Constraint illustration. (a) Prior to constraint. (b) Constraint Applied.

A more carefully defined approach as will be described here was originally seen in work by Fontaine. In his approach, the assumption made was that the wings tend to maintain symmetry with respect to the transversal plane of the body. This assumption is poorly made on a frame to frame basis because exact symmetry is too restrictive, however when applied to a longer time scale (i.e.. over the course of a full wing beat), the constraint is relaxed, but still enables an extraction of the roll angle. The stroke deviation angles of the 2 wings will remain unique from frame to frame, requiring only that symmetry be maintained over the course of the full wing stroke. Such a constraint will cause more of a vertical shift in the wing deviation angles. The wing tip to hinge vectors $\vec{V}_L$ and $\vec{V}_R$ are defined in the body coordinate frame, $\mathbf{B} = (\hat{e}_{x_b}, \hat{e}_{y_b}, \hat{e}_{z_b})$, such that they can easily be projected onto the y-z transverse plane.

$$\vec{V}_L = (\mathbf{X}_{LH} - \mathbf{X}_{LT})_b = R_{bg}(\mathbf{X}_{LH} - \mathbf{X}_{LT})_g \tag{4.16}$$

$$\vec{V}_R = (\mathbf{X}_{RH} - \mathbf{X}_{RT})_b = R_{bg}(\mathbf{X}_{RH} - \mathbf{X}_{RT})_g \tag{4.17}$$

The projection onto the Y-Z plane of the body is performed by inner product operations and the bisecting vector, $\hat{V}_{bis}$, is found by averaging the 2 projections $\hat{V}_L$ and $\hat{V}_R$.

$$\hat{V}_L = \langle V_L, \hat{e}_{z_b}\rangle \hat{e}_{z_b} + \langle V_L, \hat{e}_{y_b}\rangle \hat{e}_{y_b} = 0\hat{\imath} + V_L^y \hat{\jmath} + V_L^z \hat{k} \tag{4.18}$$

$$\hat{V}_R = \langle V_R, \hat{e}_{z_b}\rangle \hat{e}_{z_b} + \langle V_R, \hat{e}_{y_b}\rangle \hat{e}_{y_b} = 0\hat{\imath} + V_R^y \hat{\jmath} + V_R^z \hat{k} \tag{4.19}$$

The sgn$(\hat{V}_{bis}^z)$ term is added to prevent $\hat{V}_{bis}$ from flipping directions.

$$\hat{V}_{bis} = \frac{1}{2}\left(\frac{\hat{V}_L}{\|\hat{V}_L\|} + \frac{\hat{V}_R}{\|\hat{V}_R\|}\right) \tag{4.20}$$

$$\hat{V}_{bis} = \operatorname{sgn}(\hat{V}_{bis}^z)\hat{V}_{bis} \tag{4.21}$$

The constrained roll angle offset is defined to be the angle between $\hat{V}_{bis}$ and $Z_b$. The calculation of this angle is always positive, therefore to determine the direction of roll rotation, the sgn($\phi_{\text{con}}$) term is added. This term will make the rotation clockwise when the right wing span is larger in the Z direction than the left wing span and counter-clockwise when the reverse is true.

$$\phi_{\text{con}} = \text{sgn}(\phi_{\text{con}}) cos^{-1}\left(\langle \hat{V}_{bis}, -Z_b\rangle\right) \tag{4.22}$$

$$\text{sgn}(\phi_{\text{con}}) = \begin{cases} 1 & \hat{V}_R^z \geq \hat{V}_L^z \\ -1 & \hat{V}_R^z < \hat{V}_L^z \end{cases} \tag{4.23}$$

Redefining the roll angle directly affects the wing angle definitions, since they are the euler angles that relate the wing frame to the body frame. Therefore it is easiest to deal with the roll constraint problem as a subsequent procedure to finding body yaw and pitch and wing stroke amplitude and stroke deviation. The updated roll is then used to redefine stroke amplitude and stroke deviation prior to the last step of finding the wing pitch angle. The stroke amplitude is almost entirely unaffected by the updated roll, whereas the stroke deviation angles see pronounced affects. It should be noted that even if the roll constraint were applied frame to frame, the stroke deviation angles would not be forced to be completely equivalent. The simplest explanation for why this is true can be made by first looking at the equation used to pull out the deviation angle.

$$\theta_L = \tan_2^{-1}(-V_L^z, \sqrt{(-V_L^x)^2 + (-V_L^y)^2}) \tag{4.24}$$

The denominator contains a vector component in the x-direction, which was ignored during the projection of $V_L$ and $V_R$ onto the transversal plane of the body. In the Linear Filtering and Estimation Method section near the end of this chapter, the approach used to low pass filter the noisy roll constraint estimates is described.

$$\hat{\theta}_L = \tan_2^{-1}(-V_L^z, \sqrt{(-V_L^y)^2}) \tag{4.25}$$

It is therefore $\hat{\theta}_L = \hat{\theta}_R$, which must be true by the roll constraint. Only when $V_L^x = V_R^x$, will $\theta_L = \theta_R$.

## 4.6 Wing Pitch Estimation

The visual hull reconstruction unfortunately results in wings that resemble ellipsoids, which makes the chord vector and hence the wing pitch estimation nearly impossible. The task of wing pitch measurement, as it turns out, is the most difficult task in insect tracking. Attempts to use PCA on cross sectional blade elements near the wing centroid were unsuccessful. HRMT assumes a parallelogram shaped cross section, which when projected onto a plane normal to the span axis, the two furthest points on the cross section define the chord vector [21]. This method is constrained to a non-calibrated orthogonal camera set-up, which is not appropriate for the automated tracker developed here. The best voxel based approach for wing pitch estimation was a planar fit method solved by a least squares method [9]. Only a fraction of the wing stroke resulted in good wing pitch estimates due to certain wing positions yielding more planar-like reconstructions. It was concluded that wing pitch was an unobservable state using voxels as observations. Instead, a ray tracing technique was used to calculate a Plücker coordinate point-to-line cost function.

### 4.6.1 Ray Tracing with DLT Parameters

An image pixel can be thought of as a 3D ray that crosses through the pin-hole origin of the camera. The DLT calibration parameters can be used to determine the location of the camera origin $C_o$ as well as the direction of view, which is dependent on the camera transformation matrix, $R$ and the pixel location relative to the image plane principal point.

$$C_o = -\begin{bmatrix} L_1 & L_2 & L_3 \\ L_5 & L_6 & L_7 \\ L_9 & L_{10} & L_{11} \end{bmatrix}^{-1} \begin{bmatrix} L_4 \\ L_8 \\ 1 \end{bmatrix} \tag{4.26}$$

A pixel ray vector $\vec{n}_i$ is given by

$$\vec{n}_i = \operatorname{sgn}(|R|)R^{-1} \begin{bmatrix} \frac{u-u_o}{d_u} \\ \frac{v-v_o}{d_v} \\ -1 \end{bmatrix} \tag{4.27}$$

The image plane coordinates of the principal point $[u_o, v_o, 0]^T$ are determined by:

$$u_o = D^2(L_1L_9 + L_2L_{10} + L_3L_{11}) \tag{4.28}$$

$$v_o = D^2(L_5L_9 + L_6L_{10} + L_7L_{11}) \tag{4.29}$$

$$D = \frac{1}{\sqrt{L_9^2 + L_{10}^2 + L_{11}^2}} \tag{4.30}$$

The scaling parameters $[d_u, d_v]$ can then be calculated.

$$d_u = \sqrt{D^2\left[(u_o L_9 - L_1)^2 + (u_o L_{10} - L_2)^2 + (u_o L_{11} - L_3)^2\right]} \tag{4.31}$$

$$d_v = \sqrt{D^2\left[(v_o L_9 - L_5)^2 + (v_o L_{10} - L_6)^2 + (v_o L_{11} - L_7)^2\right]} \tag{4.32}$$

The camera transformation matrix R is given by:

$$R = D \begin{bmatrix} \frac{u_o L_9 - L_1}{d_u} & \frac{u_o L_{10} - L_2}{d_u} & \frac{u_o L_{11} - L_3}{d_u} \\ \frac{v_o L_9 - L_5}{d_v} & \frac{v_o L_{10} - L_6}{d_v} & \frac{v_o L_{11} - L_7}{d_v} \\ L_9 & L_{10} & L_{11} \end{bmatrix} \tag{4.33}$$

Equation 4.27 can now be solved and the 3D pixel ray $L$ can be fully defined.

$$L = C_o + t\vec{n} \tag{4.34}$$

### 4.6.2 Plücker Coordinate Analysis

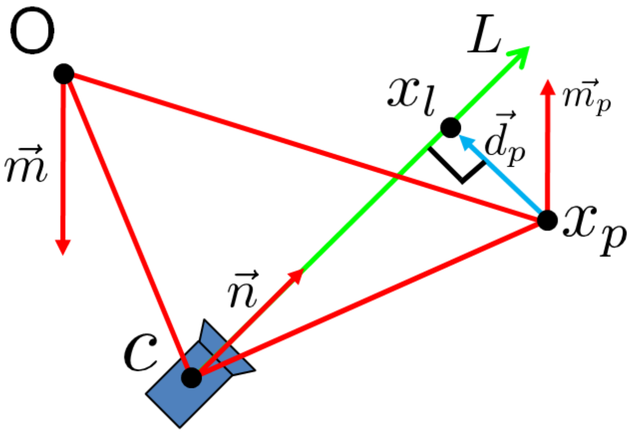


Figure 4.14: Illustration of a line L with the corresponding unit vector $\vec{n}$ and moment $\vec{m}$. The distance to an arbitrary point $x_p$ is determined efficiently by the Plücker definition of the line.

The Plücker coordinate of a line is given by a 6 coordinate, 4 degree of freedom, representation.

$$L = (\vec{n}, \vec{m}) \tag{4.35}$$

$$\vec{m} = \vec{n} \times \vec{C}_o \tag{4.36}$$

The moment of the line $m$ is given by a cross product operation and results in a vector direction perpendicular $L$. Plücker coordinates inherently are well suited for finding nearest point-to-line or line-to-line distances. The euclidean norm of $m_i$ is the nearest distance from the global camera origin to the i$^{th}$ pixel ray. The point-to-line distance $\| \vec{d_p} \|$ of an arbitrary point $x_p$ can similarly be determined and the location of the point on the line $x_l$ nearest to $x_p$ can also be calculated.

$$\| \vec{d_p} \| = \| m - \vec{n} \times x_p \| = \| \vec{m}_p \| \tag{4.37}$$

$$\vec{d_p} = \vec{m}_p \times \vec{n} \tag{4.38}$$

$$x_l = x_p + \vec{d_p} \tag{4.39}$$

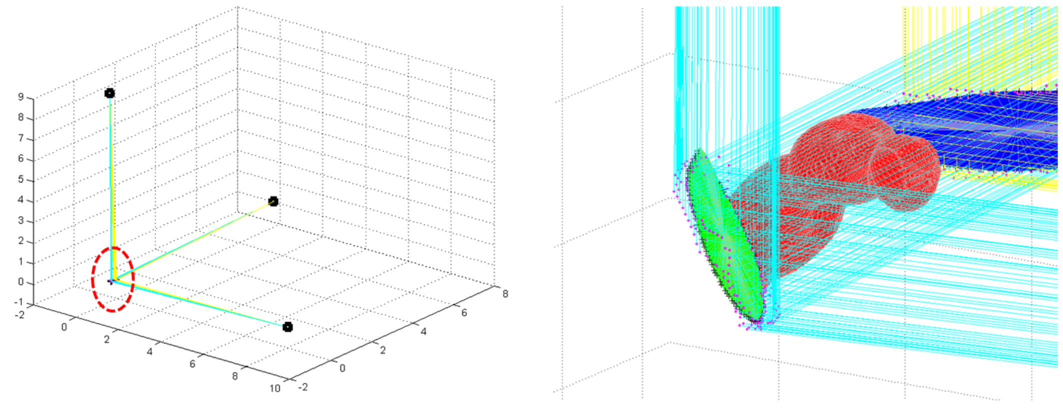


Figure 4.15: Ray tracing illustration. (a) Illustration of the wing contour pixel rays emitted by the 3camera views. (b) Zoomed in view, illustrating the ray to wing point correspondences for the optimal $\alpha$ estimate.

For the purposes of wing pitch estimation, the pixels corresponding to the wing contour in the 3 views were projected as rays into the global coordinate space. A model of the fly was transformed by the body and wing states into the global coordinate axis, the only state not known at this point was the wing pitch. The Plücker distance of the i$^{th}$ contour ray in the k$^{th}$ camera view to the set of wing model points $x_p(\alpha)$ at the current pitch estimate $\alpha$ was given by the the model point that minimized the following function.

$$\| \vec{d}_p(\alpha)_{k_i} \| = \underset{x_p(\alpha)}{\operatorname{argmin}} \| \vec{m}_{k_i} - \vec{n}_{k_i} \times (x_p(\alpha)) \| \tag{4.40}$$

The Plücker cost function was solved by summing all $\vec{d}_p(\alpha)_{k_i}$, where $N_k$ corresponds to the number of contour rays in camera view $k$, and $K$ is the number of camera views.

$$J(\alpha) = \sum_{k=1}^{K} \sum_{i=1}^{N_k} \| \vec{d}_p(\alpha)_{k_i} \|^2 \tag{4.41}$$

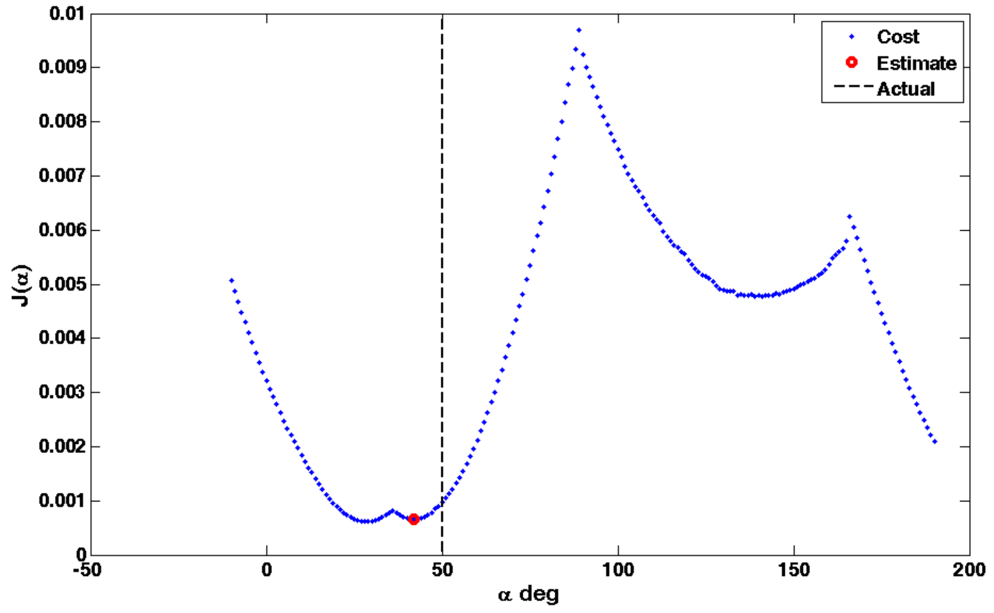


Figure 4.16: The cost $J(\alpha)$ across a full $\alpha$ sweep is highly non-linear and contains several local minima.

For an $\alpha$ sweep of -10 to 190 degrees, corresponding to all the possible wing pitch angles for a dipteran insect, there existed several local minima. Unfortunately, it was not guaranteed that the $\alpha$ that produces the global minima was the optimal wing pitch estimate. Due to wing morphing, the true wing contour differs from the rigid wing model. Also, the shape of the wing model used was generic for the species, resized based on the manually initialized frame, but not fine tuned to fit the shape of the particular fly being tracked. Non-linear estimation techniques, such as simulated annealing, capable of resolving a large number of states was not necessary because the wing pitch estimation required only varying 2 states. The wing pitch sweep was an efficient brute force method used to calculate $J(\alpha)$. The local minima of $J(\alpha)$ were kept as potential candidates for the optimal wing pitch estimate.

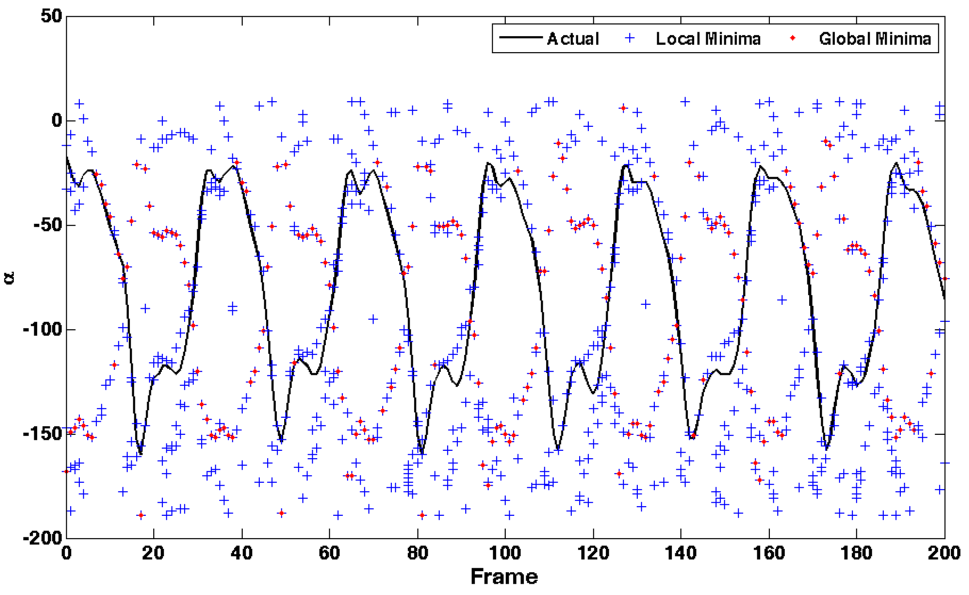


Figure 4.17: The $\alpha$'s that produce local minima in $J(\alpha)$ contain the optimal estimate for wing pitch, however the $\alpha$ which produces the global minimum is not necessarily the optimal estimate.

Due to the high number of minima, a loose constraint was placed on the range of $\alpha$ angles to consider.

| **Wing Stroke Phase** | $\phi_w(t)$ | $\dot{\phi_w}(t)$ | $\alpha$ **Range** |
|---|---|---|---|
| Downstroke | - | $> 5$ | $70^o - 190^o$ |
| Upstroke | - | $< 5$ | $-10^o - 100^o$ |
| Pronation | $> 0$ | $\approx 0$ | $40^o - 160^o$ |
| Supination | $< 0$ | $\approx 0$ | $30^o - 150^o$ |

Table 4.1: Assuming $\phi$ is known, $\alpha$ search window can be decreased

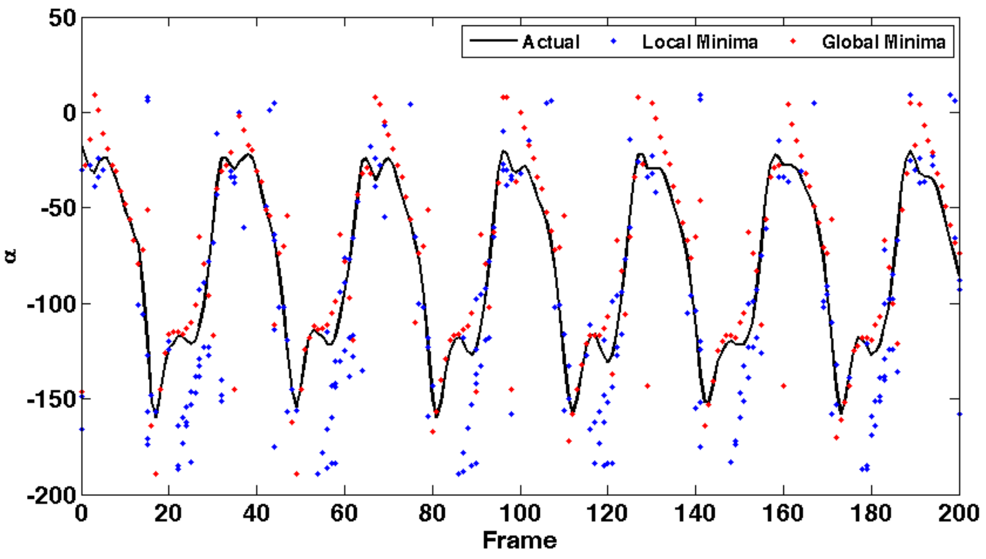


Figure 4.18: Constrained $\alpha$ search window resulted in global minimum of $J(\alpha)$ occurring at the optimal $\alpha$ more consistently

The constrained $\alpha$ search window decreased the possibility of wing flipping, a phenomena due to the inability of distinguishing the leading edge from the trailing edge. Estimation of $\alpha$ during stroke reversal encountered the most trouble due to this depth ambiguity in combination with a rapidly flipping wing causing bending and twisting. This was anticipated because even during manual tracking, $\alpha$ estimation during stroke reversal is a daunting task and often required human interpolation of the image data across several frames.

### 4.6.3 Training Data to resolve Wing Pitch

The Plücker coordinate cost analysis generated several local minima as estimates. The choice of the optimal estimate was made by comparison with a set of training data. The automated tracker estimated stroke amplitude $\phi_w$ almost identical to manual tracking, thus the current $\phi_w$ and $\dot{\phi}_w$ were compared with a 500 frame training data sequence. The wing pitch corresponding to the best matching frame in the training data was compared with all the $\alpha$ estimates, the best matching one was considered the optimal $\alpha$ if the difference between the two angles was less than 10 degrees. If no local minima satisfied this condition, then the global minima was considered the optimal estimate. The performance of this conditional method outperformed extended Kalman filtering and pure global minima searching.

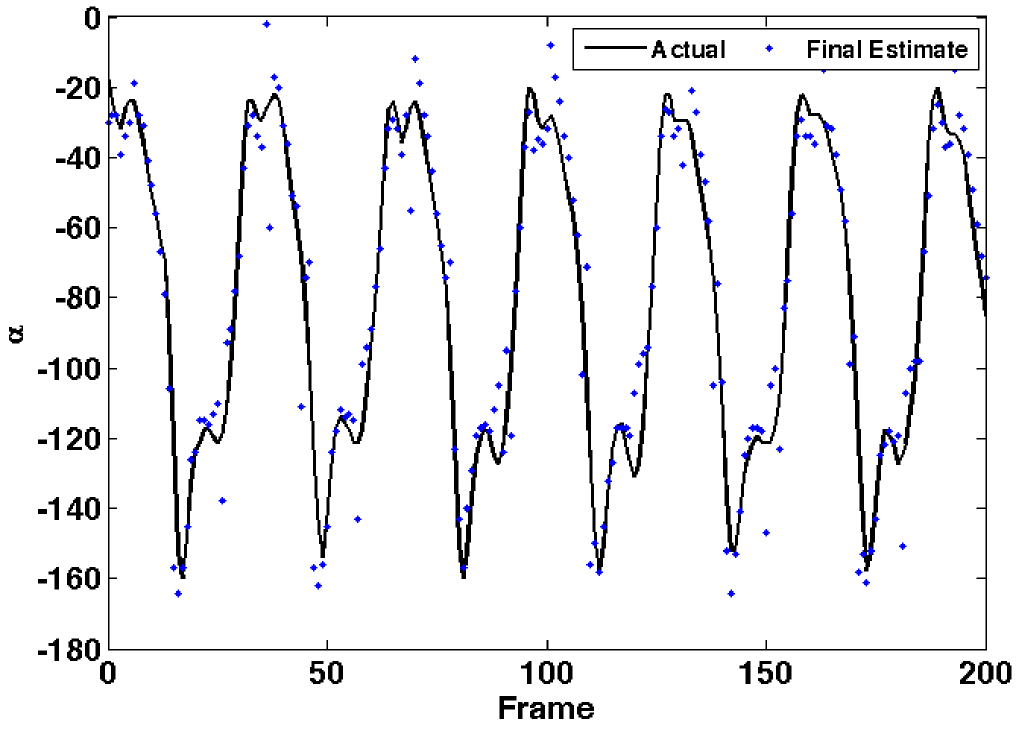


Figure 4.19: Final $\alpha$ estimate after the conditional training data and global minima filter

## 4.7 Linearized Estimation and Filtering Methods

So far in this chapter, the Kalman filter has been mentioned several times. In Chapter 2, the a continuous constant velocity dynamical model was discretized and the discrete Kalman filter was introduced to track image coordinates and the radius of three wand markers. Insect tracking, although a much more involved process, can utilize similar linear filtering strategies. The motion models assumed for the insect varied depending on which state was being tracked.

### 4.7.1 Body states

The frame rate of the cameras (7500 fps) was set up to approximately record 30 frames per wing stroke. As a result of such a high frame rate, the slower moving body behaved in an almost linear fashion. To filter the center of mass estimates, which were measured by computing the mean of the body voxel reconstruction, a constant velocity motion model $\mathbf{\Phi}_v$ was assumed. The euler angles of the body were measured by a combination of PCA and a roll constraint and was expected to behave less linearly, thus a constant acceleration motion model $\mathbf{\Phi}_a$ was assumed.

The motion models for one measured state are given by:

$$\mathbf{\Phi}_v = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{4.42}$$

$$\mathbf{\Phi}_a = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{4.43}$$

$$\tag{4.44}$$

The corresponding process noise covariance matrices $Q_v$ and $Q_a$ are given by:

$$Q_v = \sigma_a^2 \begin{bmatrix} \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} \\ \frac{\Delta t^2}{2} & \Delta t \end{bmatrix} \tag{4.45}$$

$$Q_a = \sigma_J^2 \begin{bmatrix} \frac{\Delta t^5}{20} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{6} \\ \frac{\Delta t^4}{8} & \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} \\ \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} & \Delta t \end{bmatrix} \tag{4.46}$$

The process noise is given by $\sigma_a$ and $\sigma_J$ and represent the acceleration and jerk noise uncertainty in the motion model.

### 4.7.2 Roll Constraint Filters

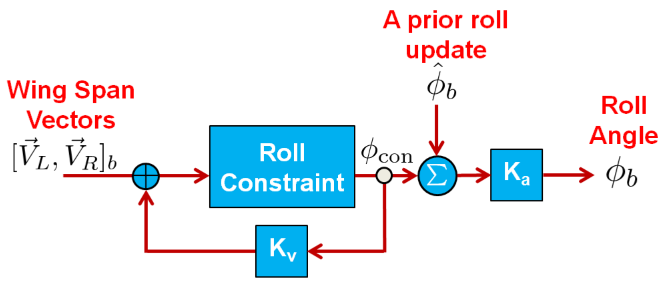


Figure 4.20: Roll constraint involved two independently acting Kalman Filters

Additional filters were indirectly applied to the measured states to increase the robustness of the tracking program. The roll constraint produced $\phi_{con}$, which acted as updates to the a priori roll estimate $\hat{\phi}_b$. The outputs from the roll constraint were very noisy, thus a constant velocity Kalman filter was used to update the defined wing-span vectors in the body axis. The feedback resulted in a more refined $\phi_{con}$, which was added to $\hat{\phi}_b$ to serve as the roll estimate in the constant acceleration Kalman filter described above. While it may seem like the same state was filtered twice, $\phi_{con}$ and $\phi_b$ acted as two independent states and therefore two different filters were implemented.

### 4.7.3 Effective Hinge Filtering

The effective hinge estimation acted as a correction factor for the wing-span estimation. PCA produced wing-span vectors, which were assumed to pass through the center of the wing reconstruction. By projecting the geometrically defined hinge onto the wing-span vector, the effective hinge was determined. Transforming the effective hinge into the body frame provided an estimate that was known to have some cyclic motion in the transversal direction, due to wing bending during stroke reversal. However, the hinge, acting as a ball joint, should in general remain constant in the body frame and was therefore modeled as such. The discrete constant position motion model $\Phi_p$ and its corresponding process noise covariance $Q_p$ were defined by:

$$\Phi_p = 1 \tag{4.47}$$

$$Q_p = \sigma_v^2 \Delta t \tag{4.48}$$

The process noise level is given by $\sigma_v$, where the noise represents motion in the state. For the transversal direction $\mathbf{e}_{\mathbf{x}b}$, $Q_p$ was set very high compared to the other directions to allow effective hinge motion to occur due to wing bending.

### 4.7.4 Wing Angle Filtering

Linear filters were not appropriate for the wing angle filtering. It was possible to apply the extended Kalman filter by treating the periodic wing angle states as harmonic oscillators. The discrete model would then have to be linearized on a frame by frame basis and the frequency and vertical offset would have to be estimated. The effect of the extended Kalman filter was very sensitive to initialization and the noise covariance matrices. During high maneuvering sequences, the wing angles produced time varying sinusoidal signals, which were distorted by an extended Kalman filter. Therefore, to deal with poor wing angle measurements, a post process procedure was developed.

## 4.8 Post Processing

It was unlikely that the results of an automated tracker would be free from post processing. However, the results were well suited enough to apply a simple automated post processing script to detect poorly resolved frames, delete them, and re-interpolate the frames using least-squares. This method was specifically chosen as an alternative to extended Kalman filtering the periodic wing states. Rather than distort the true sinusoidal signals by attempting to track frequency, amplitude, phase shift, and a mean offset, by differentiating the raw wing state data, it was very clear where something went wrong during a particular frame. It should be noted that this method was acceptable due to the high level of accuracy of the raw wing state measurements, which was only achieved by the extensive image processing and filtering of body states.

Estimation of the maximum wing beat frequency for a particular species, f $\approx$ 260 Hz for *Drosophila*, and a max peak-to-peak amplitude of 150 degrees, it was possible to predict an average stroke amplitude rate $\dot{\phi}_w$. However, because half of the wing stroke was spent in the stroke reversal phase, a better estimate of $\dot{\phi}_w$ was made by considering the wing to move at twice the frequency such that $\dot{\phi}_w \approx 2\phi f$. Frames were filtered based upon a threshold of $\dot{\phi}_{w,\text{thresh}} = 1.5\dot{\phi}_w$. A similar procedure was conducted for $\theta_w$ and $\alpha_w$. To remove unwanted peaks, the local maxima and local minima of $\phi_w$ were found and any data points exceeding one standard deviation above the local maximum mean or one standard deviation below the local minimum mean were filtered.

Typically, only $1 - 2\%$ of the $\phi_w$ data was filtered, whereas $5 - 10\%$ of the $\theta_w$ and $\alpha_w$ data were filtered. In manual tracking, 66% of the wing state data was interpolated since only every third frame was digitized. It was therefore appropriate to remove and re-interpolate unwanted kinematic data in the auto tracker.

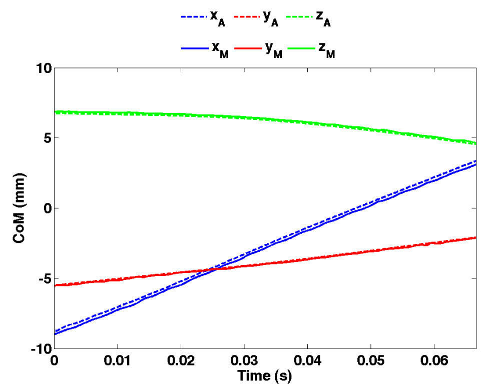
# Chapter 5

# Analysis and Results

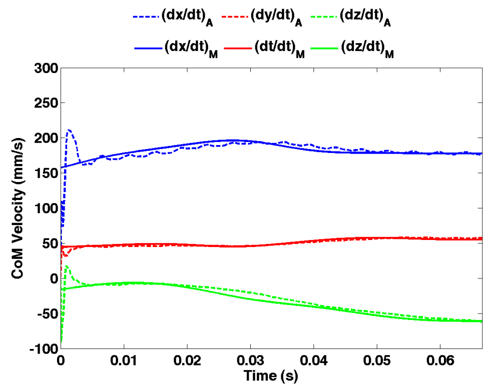
The development of a manual and automated tracking program have been described in Chapters 3 and 4. In this Chapter, the differences between the two methods are qualitatively and quantitatively compared. Additionally, an insect flight simulation video with ground truth kinematics is used to quantify the errors involved in the two tracking techniques.

## 5.1 Forward Flight: *Drosophila melanogaster*

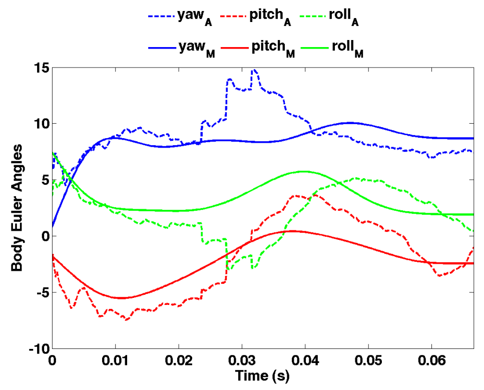
Forward flight represents the simplest flight sequence to track because the assumed motion models accurately portray the body kinematics. Presented here is a 500 frame *Drosophila* sequence in stable forward flight with a slight transition into climb. The manual and automated tracking techniques were implemented for comparison in these ideal flight conditions. The rms differences relative to the manually digitized results are presented as Flight 1 in Table 5.7 near the end of this chapter.



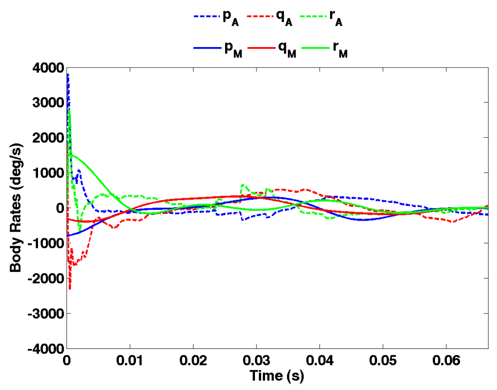
(a)



(b)



(c)



(d)

Figure 5.1: Body states: (a) Global Position (b) Global Velocity (c) Euler angles (d) Body rates.

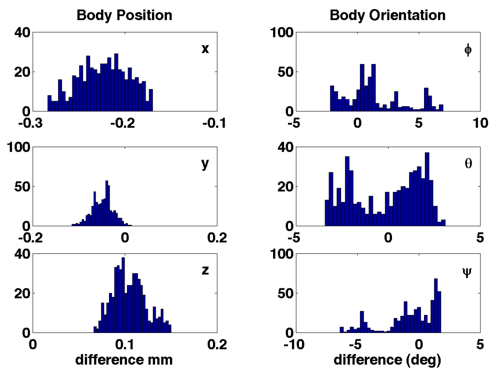
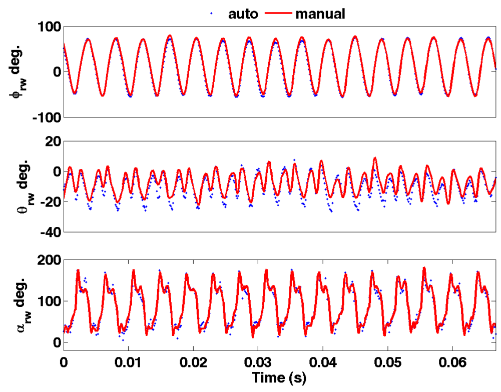
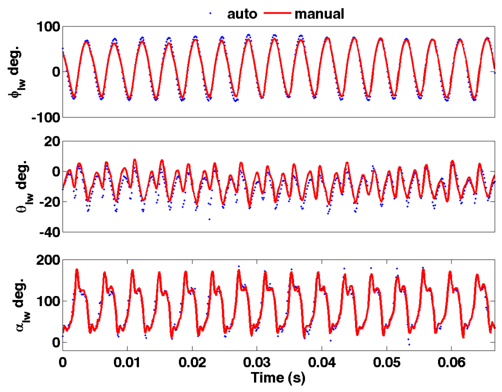


Figure 5.2: Histogram of differences in Body states between manual and automated tracking over the course of 500 frames in *Drosophila* forward flight.

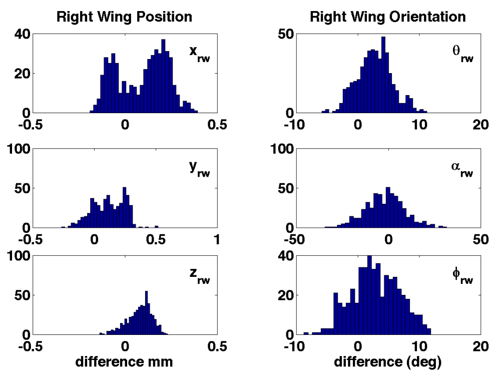


(a)

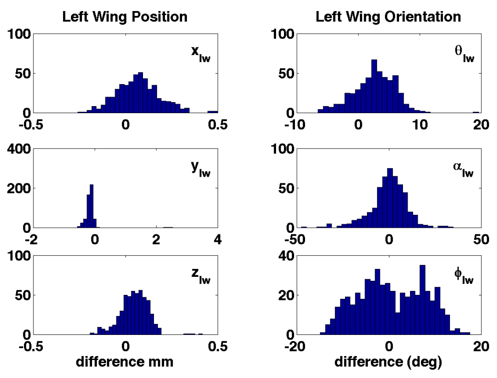


(b)

Figure 5.3: Wing States: (a) Right wing (b) Left wing.



(a)



(b)

Figure 5.4: Histogram of differences in wing states between manual and automated tracking over the course of 500 frames in *Drosophila* forward flight: (a) Right wing (b) Left wing.

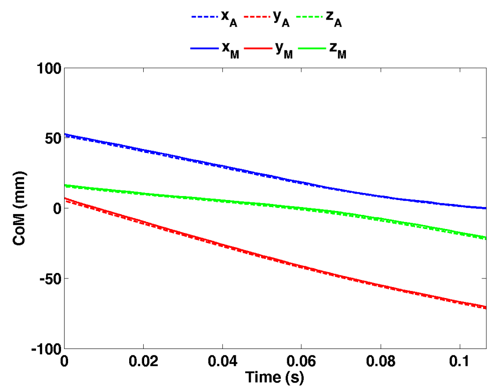
| States | \| Mean\| | RMS | Standard dev. |
|---|---|---|---|
| CG (mm) | 0.25 | 0.25 | 0.03 |
| Yaw (deg) | 1.67 | 2.23 | 2.14 |
| Pitch (deg) | 1.0 | 1.90 | 1.90 |
| Roll (deg) | 2.00 | 2.71 | 2.37 |
| $\theta_w$ (deg) | 3.40 | 4.10 | 3.08 |
| $\alpha_w$ (deg) | 7.83 | 10.34 | 10.26 |
| $\phi_w$ (deg) | 5.00 | 5.97 | 5.48 |

Table 5.1: Statistical differences between automated and manual tracking for the case of *Drosophila* in forward flight.

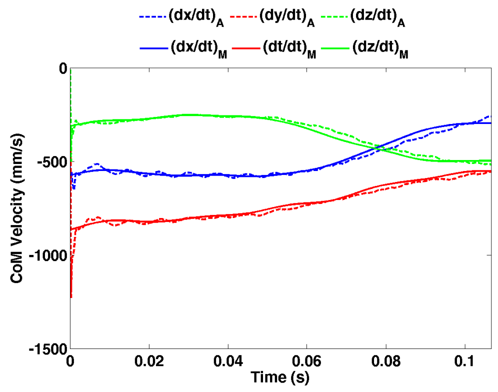
To summarize, the center of mass had an rms difference of 0.25 mm, which is approximately 12% of the body length. The manually defined center of mass is defined based on the tail to head vector and a constant offset from the digitized wing hinges. The mean location of the body reconstruction forms the automated approach to determining the center of mass, which unsurprisingly varies from the manual approach. The yaw, pitch, and roll rms values were all under $3^o$ and the mean wing rms values for stroke amplitude $\phi_w$, deviation $\theta_w$, and wing pitch $\alpha_w$ was $5.97^o$, $4.10^o$,and $10.34^o$ respectively. While some of the differences can be attributed to error in the automated tracking, a large explanation for any discrepancy in the wing angles in this case was attributed to differences in the body coordinate frame definition, particularly the definition of the roll axis of the body. Regardless, the results in this simple case demonstrate the performance capabilities of the automated tracker.

## 5.2 Climbing Maneuver: *Calliphoridae*

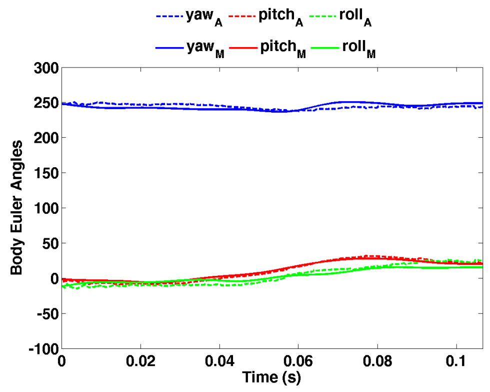
The sequence presented here represents an 800 frame transition from forward flight into an aggressive climb maneuver of a *Calliphora*. Both the manual and automated tracking techniques clearly observed an increase in wing stroke frequency in addition to a collective increase in the stroke amplitude $\phi_w$ of the wings, which are both expected to result in higher lift production. The automated tracking program was not modified to track this species, which resulted in larger errors due to increased reconstruction artifacts. The rms differences compare to the manually digitized results are included as Flight 2 in Table 5.7.



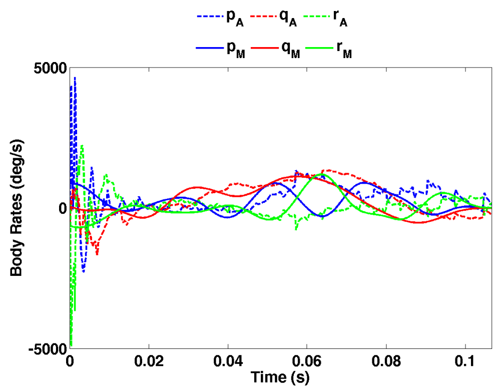
(a)



(b)



(c)



(d)

Figure 5.5: Body states: (a) Global Position (b) Global Velocity (c) Euler angles (d) Body rates.

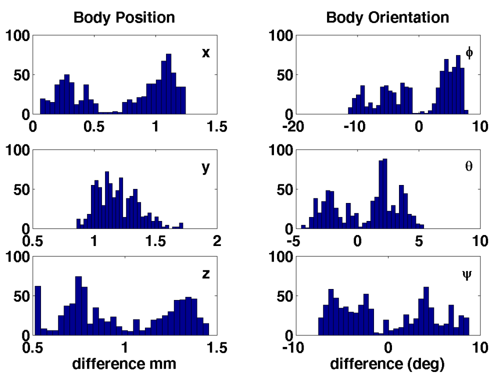
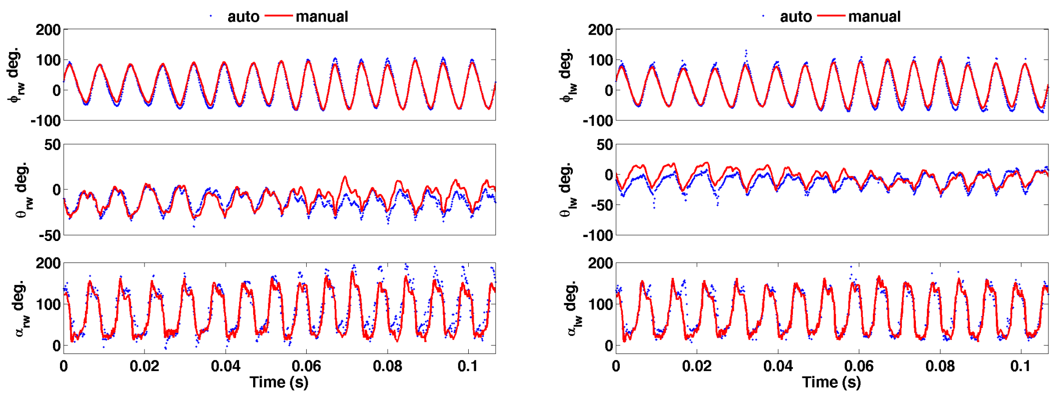
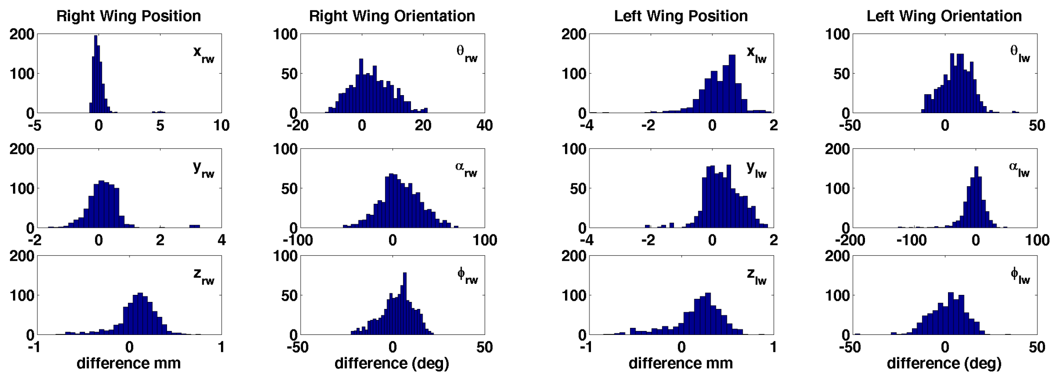


Figure 5.6: Histogram of differences in Body states between manual and automated tracking over the course of 800 frames in *Calliphora* climbing flight.



(a) (b)

Figure 5.7: Wing States: (a) Right wing (b) Left wing.



(a) (b)

Figure 5.8: Histogram of differences in wing states between manual and automated tracking over the course of 800 frames in *Calliphora* climbing flight: (a) Right wing (b) Left wing.

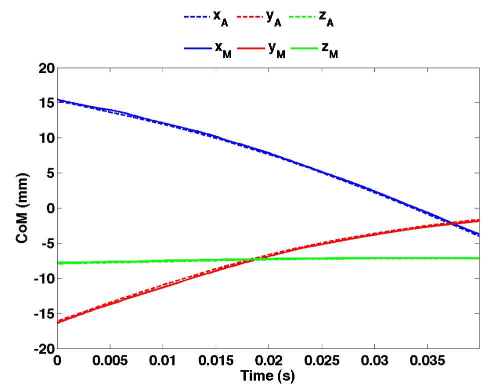
| States | \| Mean\| | RMS | Standard dev. |
|---|---|---|---|
| CG (mm) | 1.78 | 1.78 | 0.13 |
| Yaw (deg) | 4.51 | 4.94 | 4.94 |
| Pitch (deg) | 2.43 | 2.66 | 2.56 |
| Roll (deg) | 5.33 | 5.81 | 5.81 |
| $\theta_w$ (deg) | 7.12 | 8.74 | 7.27 |
| $\alpha_w$ (deg) | 15.04 | 20.62 | 19.63 |
| $\phi_w$ (deg) | 7.66 | 9.39 | 9.13 |

Table 5.2: Statistical differences between automated and manual tracking for the case of *Calliphora* in climbing flight.

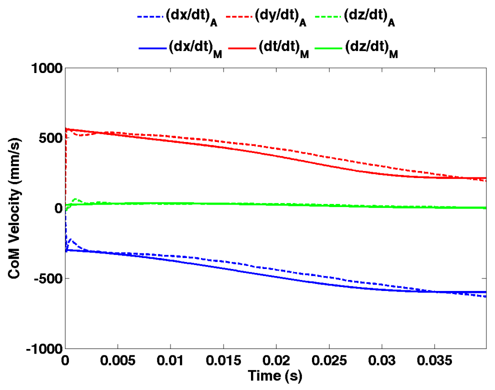
The results of this pitch up sequence of a *Calliphora* demonstrate the automated tracker's ability to function with different dipteran species of totally different body types and sizes. The tracker was designed for this capability and the rms differences in the yaw, pitch, and roll were only $4.94^o$, $2.66^o$, and $5.81^o$ respectively. The estimated center of mass demonstrated a clear discrepancy in the two methods, with a $1.78mm$ rms difference, however this discrepancy was consistently applied as can be seen by the $0.13mm$ standard deviation. Larger variations were induced in the wing kinematics because of their dependence on the body states, which resulted in rms differences in $\phi_w, \theta_w, \alpha_w$ of $9.39^o$, $8.74^o$, and $20.62^o$ respectively. A major reason for the higher differences was attributed to an unaccounted failure mode in which the position of the hind legs were visible in the top view as protruding out the back of the body and mistaken as wing segments during the reconstruction. Due to occlusions during the pronation phase of the wing, these legs resulted in reconstruction artifacts that highly affected the wing angle estimates. The overall results can still be classified as a success, though future work must address this failure mode.

## 5.3 Coordinated Turn Flight: *Drosophila melanogaster*

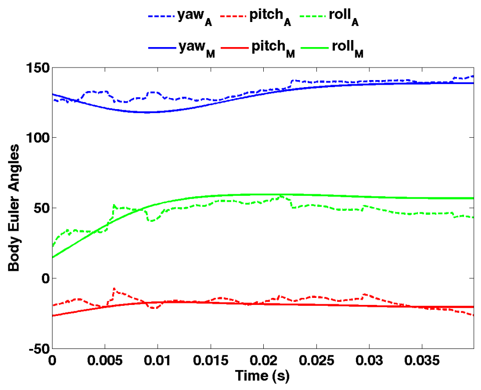
In Chapter 3, a coordinated turn flight was presented to illustrate the time histories of the states in different reference frames. The same sequence is presented here to compare with the automated tracker and demonstrates 300 frames of *Drosophila* coordinated turn sequence. The rms statistics are presented as (Flight 3 in Table 5.7)



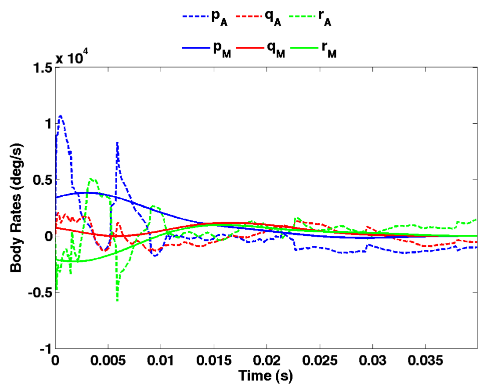
(a)



(b)



(c)



(d)

Figure 5.9: Body states: (a) Global Position (b) Global Velocity (c) Euler angles (d) Body rates.

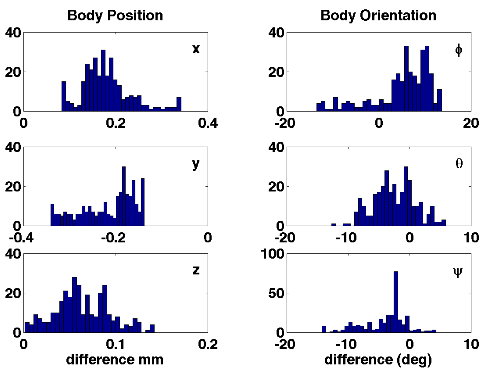
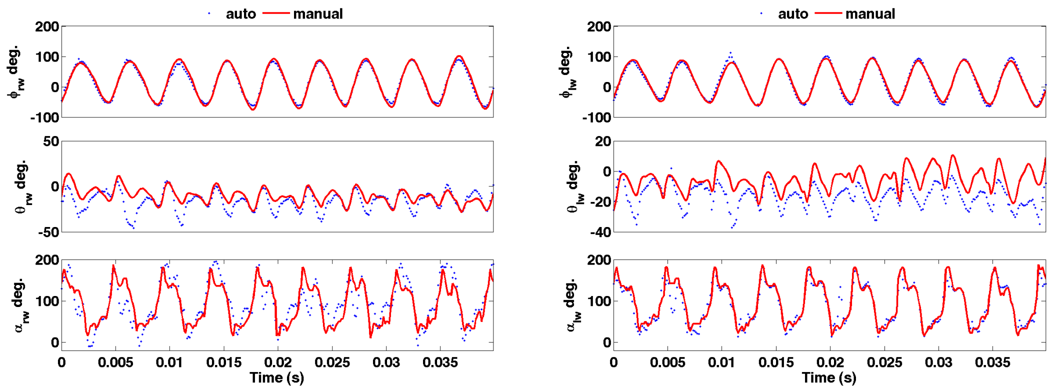
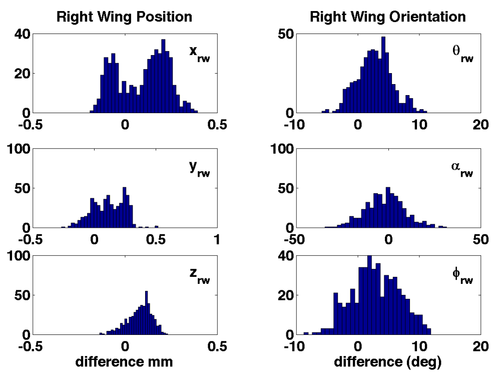


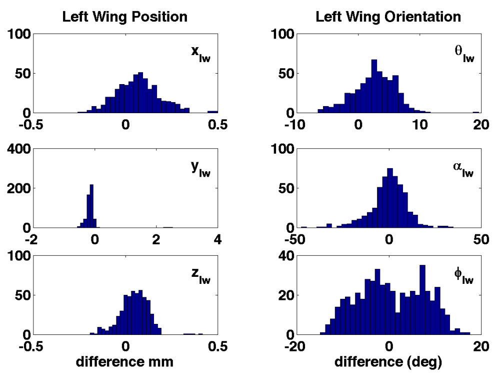
Figure 5.10: Histogram of differences in Body states between manual and automated tracking over the course of 300 frames in *Drosophila* turning flight.



(a) (b)

Figure 5.11: Wing States: (a) Right wing (b) Left wing.





(a) (b)

Figure 5.12: Histogram of differences in wing states between manual and automated tracking over the course of 300 frames in *Drosophila* turning flight.: (a) Right wing (b) Left wing.

| States | \| Mean\| | RMS | Standard dev. |
|---|---|---|---|
| CG (mm) | 0.29 | 0.30 | 0.07 |
| Yaw (deg) | 4.23 | 5.60 | 4.36 |
| Pitch (deg) | 4.56 | 5.00 | 2.81 |
| Roll (deg) | 3.67 | 4.69 | 4.35 |
| $\theta_w$ (deg) | 7.24 | 8.94 | 6.72 |
| $\alpha_w$ (deg) | 12.11 | 16.16 | 15.52 |
| $\phi_w$ (deg) | 5.55 | 6.83 | 6.52 |

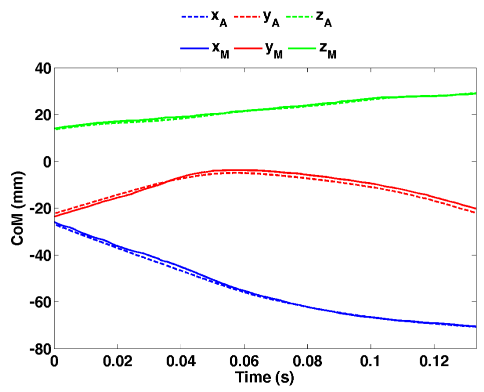
Table 5.3: Statistical differences between automated and manual tracking for the case of *Drosophila* in turning flight.

The importance of this test validated the use of the roll constraint as being applicable to high roll maneuvers. In this sequence, roll angles as high as $62^o$ were reached, initialized in the first frame as $11.4^o$. The automated tracker succesfully estimated the roll angle by observation of the wing angles, resulting in an rms difference of $4.69^o$. The rms differences in yaw and pitch were $5.60^o$ and $5.00^o$. The rms differences in $\phi_w$, $\theta_w$, and $\alpha_w$ were $6.83^o$, $8.94^o$, and $16.16^o$. The higher $\alpha_w$

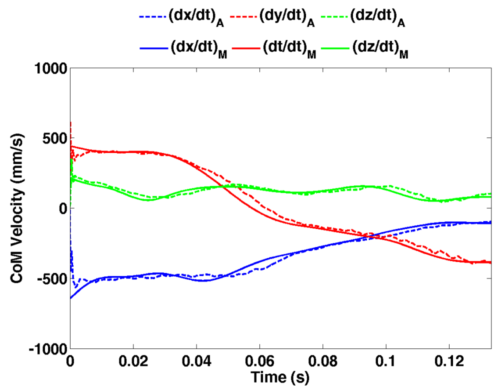
errors, especially in the right wing were a result of poor visibility due to one of the images being out of focus and a poorly resolved choice of the optimal local minima of the wing pitch cost function. Modifications to the wing pitch estimation are clearly required in future work.

## 5.4 Aggressive Turn Maneuver (Saccade): *Calliphoridae*

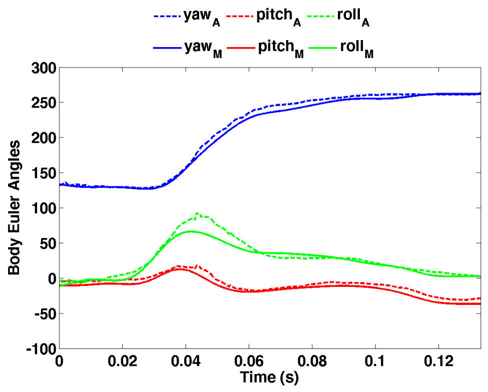
The maneuvers studied so far have been fairly linear and well modeled by linear motion models. Presented here is a 1000 frame sequence of a nearly 180 degree turning maneuver (Flight 4 in Table 5.7). An unfortunate failure mode in the automated tracker was discovered, in which the long legs of the *Calliphora* were confused as wings mid way through the turn. This was devastating to the wing angle and roll angle measurements and demonstrated the need to develop a method of filtering the leg reconstructions. The manual tracking program revealed differential wing kinematic inputs to generate the torques necessary to perform the turn. Following the turn, the fly recovered by entering a near hovering mode followed by a transition to accelerating forward flight in which the body pitch clearly decreased to make itself more aerodynamic.



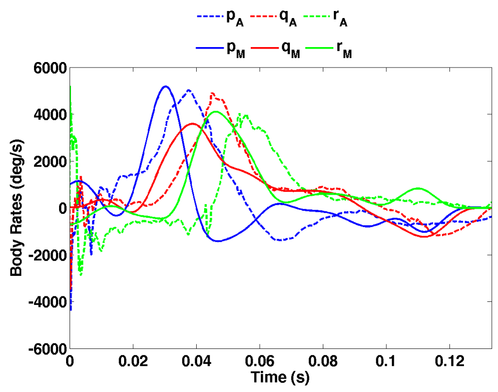
(a)



(b)



(c)



(d)

Figure 5.13: Body states: (a) Global Position (b) Global Velocity (c) Euler angles (d) Body rates.

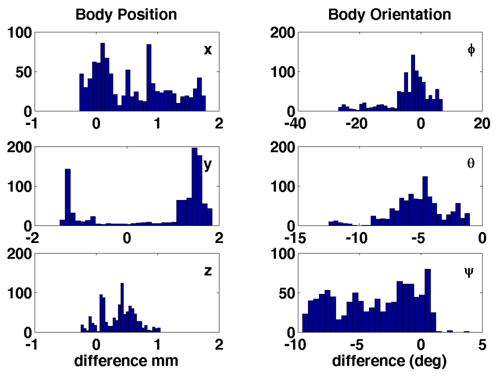
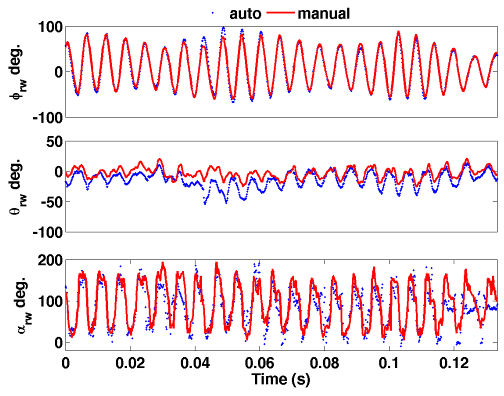
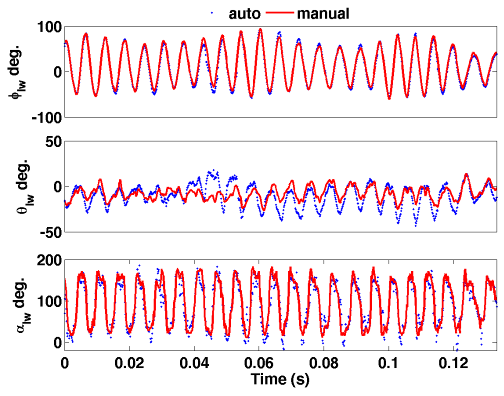


Figure 5.14: Histogram of differences in Body states between manual and automated tracking over the course of 1000 frames in a *Calliphora* saccade maneuver.

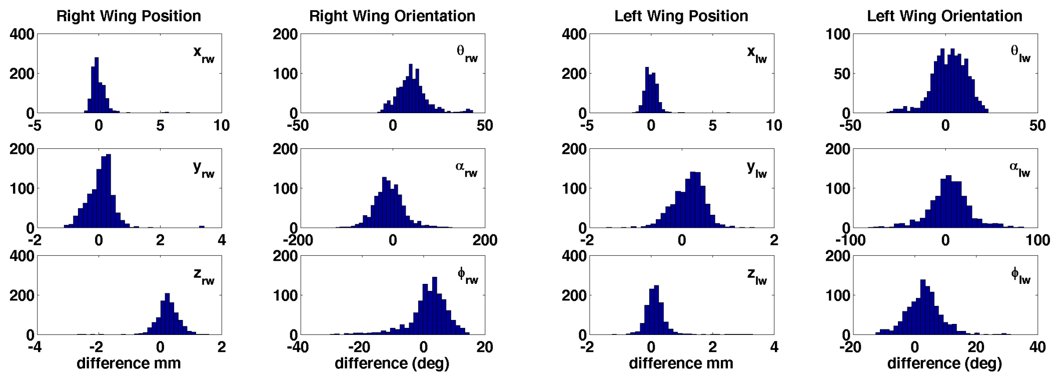


(a)



(b)

Figure 5.15: Wing States: (a) Right wing (b) Left wing.



(a) (b)

Figure 5.16: Histogram of differences in wing states between manual and automated tracking over the course of 1000 frames in a *Calliphora* saccade maneuver: (a) Right wing (b) Left wing.

| States | \| Mean\| | RMS | Standard dev. |
|---|---|---|---|
| CG (mm) | 1.66 | 1.67 | 0.16 |
| Yaw (deg) | 3.92 | 4.95 | 3.10 |
| Pitch (deg) | 5.41 | 5.93 | 2.44 |
| Roll (deg) | 5.74 | 8.35 | 7.33 |
| $\theta_w$ (deg) | 9.30 | 11.45 | 8.79 |
| $\alpha_w$ (deg) | 20.93 | 27.55 | 26.79 |
| $\phi_w$ (deg) | 5.01 | 6.56 | 6.09 |

Table 5.4: Statistical differences between automated and manual tracking for the case of *Calliphora* in a saccade maneuver.

The automated tracker was put to the test in this aggressive maneuvering sequence. The observed saccade maneuver, despite its non-linear body motions was capable of being tracked well without a need to alter the control knobs that form the Kalman filtering gains. The CG was tracked with an rms difference of 1.67 mm and standard deviation of 0.16 mm, which matches well with the statistics in the *Calliphora* climbing sequence. The yaw, pitch, and roll rms differences were $4.95^o$,

$5.93^o$, $8.35^o$ respectively. The peak roll angle showed the most deviation due to the Kalman filter damping the roll estimates to maintain the linear dynamic model prediction and possibly wing stroke asymmetry that may have provided poor roll estimates via the roll constraint assumption. The majority of the dynamics were captured during the maneuver however, demonstrating the automated tracking capabilities to handle aggressive maneuvers. The rms differences in $\phi_w$, $\theta_w$, and $\alpha_w$ were $6.56^o$, $11.45^o$, and $27.55^o$. Earlier versions of the automated tracking program did not differentiate legs from wings, which caused this particular sequence to have catastrophic results. During the saccade, the long legs of the *Calliphora* were extended to change its inertial properties and in the image processing, the legs appear no different than wings. it was necessary to add the geometric filtering to delete the legs in the visual hull reconstruction as to not cause problems when computing the span axes of the wings. The rms values are included as Flight 4 in Table 5.7.

## 5.5 Quantifying Error

A difficult task in quantifying the errors involved in tracking techniques was the establishment of some "ground truth" that could be used for comparison. Traditionally, automated tracking techniques were compared to manually tracking methods with the assumption that a human digitizer was much more accurate than any automated tracking program. However, manual tracking techniques are not free of error. User error is introduced due to imperfect landmark digitization, rigid wing model fitting, depth ambiguity, and camera calibration error. It was therefore beneficial to quantify not only the differences between automated and manual tracking, but also estimate the error involved in both of these techniques relative to a ”ground truth”.

### 5.5.1 Insect Flight Simulation

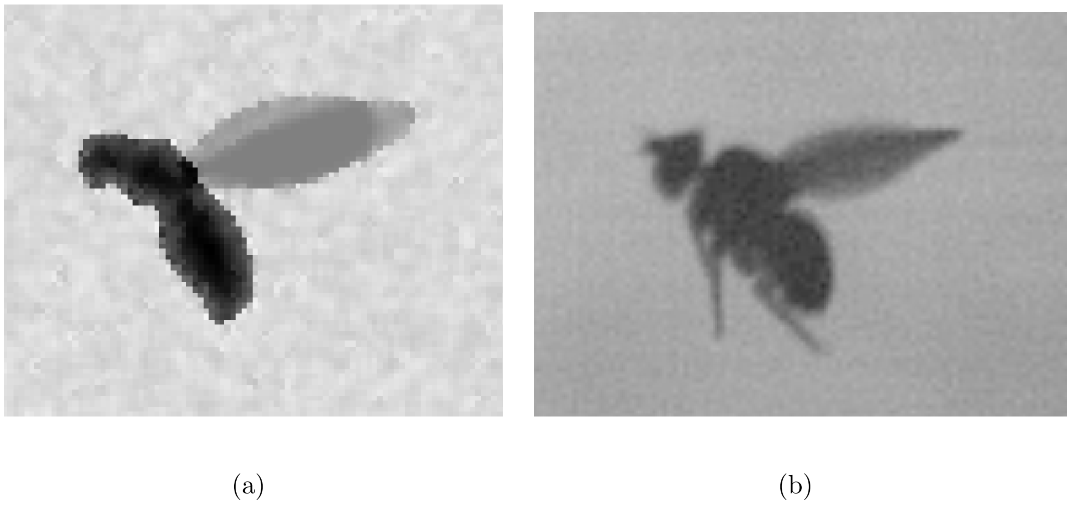


Figure 5.17: Simulation of a fly using geometric fly model. (a) Simulation (b) Actual Insect

The simulations required generative models of the particular insect, which were then transformed by a set of wing and body kinematics and projected onto a set of three equivalent views by used of the camera calibration parameters. The detailed process for the simulation development can be found in Appendix A and the generative modeling techniques are described in Appendix B, which follow similar methodologies to previous studies [10], [5]. A video simulation of a fly was generated using a forward flight kinematic model. The wing motions were given (in radians) by:

$$\phi_w = \phi_a \cos(wt) + \phi_{off} \tag{5.1}$$

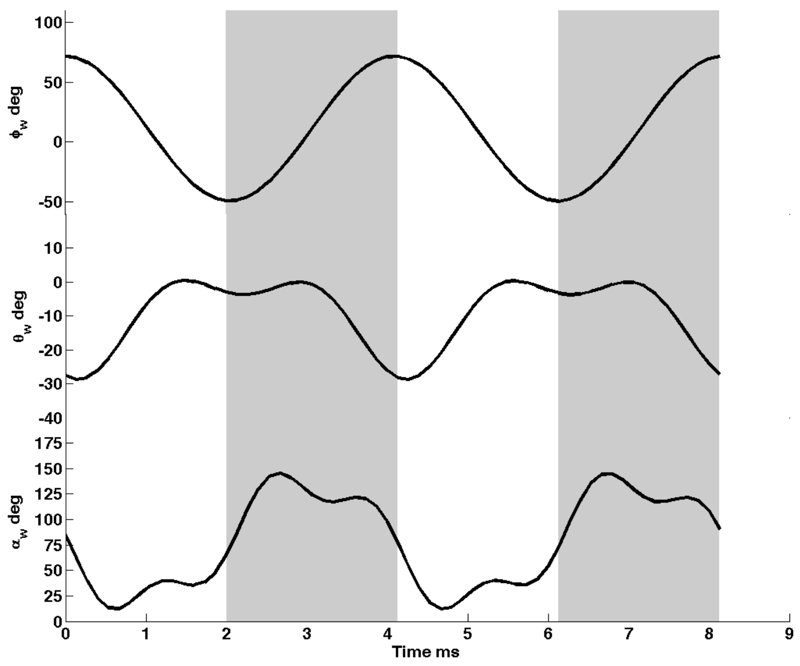
$$\theta_w = \theta_{a_1} \cos(wt + \theta_{ph_1}) + \theta_{a_2} \cos(2wt + \theta_{ph_2}) + \theta_{off} \tag{5.2}$$

$$\alpha_w = \alpha_{a_1} \cos(wt + \alpha_{ph_1}) + \alpha_{a_2} \cos(3wt + \alpha_{ph_2}) + \alpha_{off} \tag{5.3}$$

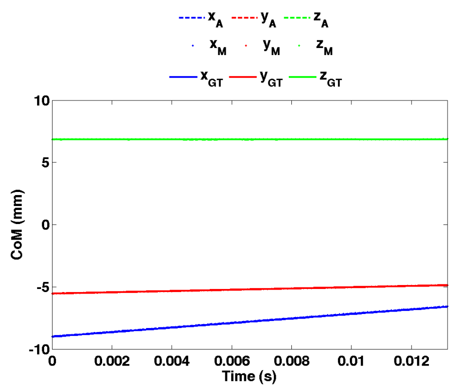
| Stroke Elevation | | Wing Pitch | | Stroke Amplitude | |
|---|---|---|---|---|---|
| $\theta_{a_1}$ | 0.218 | $\alpha_{a_1}$ | 1.063 | $\phi_{amp}$ | 1.056 |
| $\theta_{a_2}$ | 0.119 | $\alpha_{a_2}$ | 0.324 | - | - |
| $\theta_{ph_1}$ | 2.917 | $\alpha_{ph_1}$ | 1.644 | - | - |
| $\theta_{ph_2}$ | 2.653 | $\alpha_{ph_2}$ | 7.224 | - | - |
| $\theta_{off}$ | -0.166 | $\alpha_{off}$ | 1.372 | $\phi_{off}$ | 0.194 |

Table 5.5: Forward flight wing stroke parameters

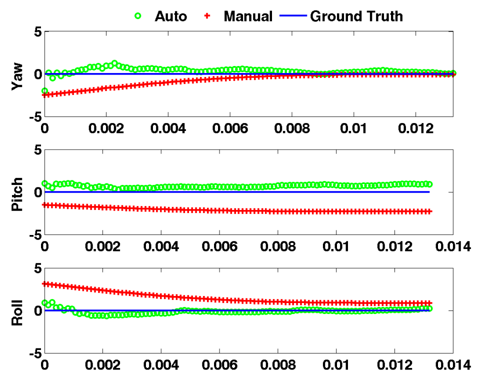
The angular frequency was given by $w = 2\pi f$, where the flapping frequency was set to $f = 245$ hz. This frequency was representative of a typical forward flight sequence for *Drosophila*. Body angles were set to $\phi_b = \theta_b = \psi_b = 0$. The forward flight speed was set to $u = 182.3$ mm/s. A side slip velocity of $v = 50.9$ mm/s was set arbitrarily to vary the y position of the cg. The heave velocity was kept constant, $w = 0$ mm/s.



It should be noted that through previous studies, the manually defined roll axis of a *Drosophila* inherently differed from the natural principal axis of a fly by approximately 10 degrees. This offset was considered by defining the principal to stability coordinate frame rotation angle $\chi_m = \chi + 10\text{deg}$. $\chi_m$ was the modified rotation, used only in manual kinematics extraction.

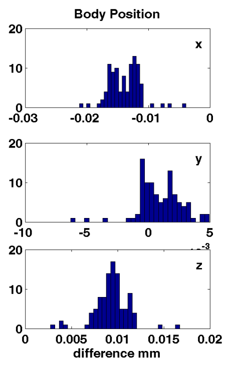


(a)



(b)

Figure 5.18: Body states: (a) Global Position (b) Euler angles



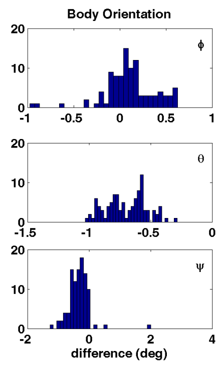
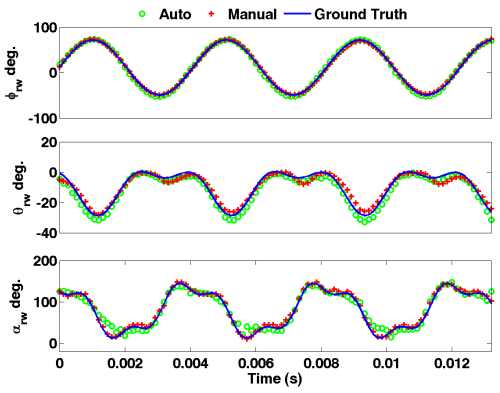
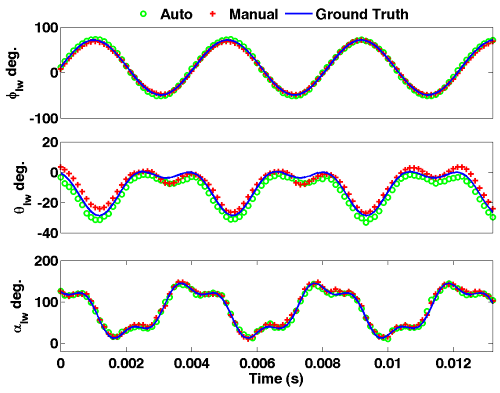


Figure 5.19: Histogram of differences in Body states between automated tracking and ground truth.

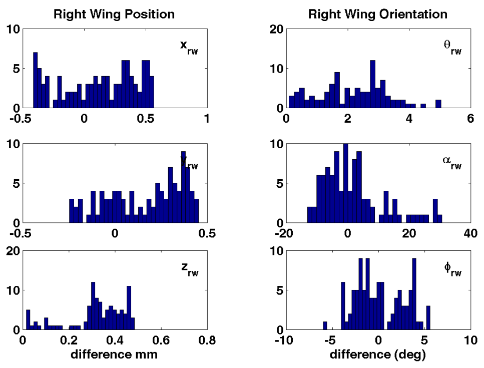


(a)

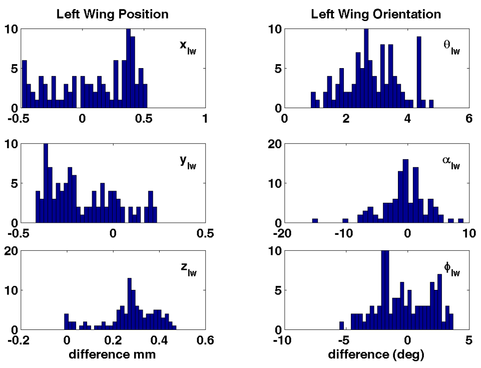


(b)

Figure 5.20: Wing States: (a) Right wing (b) Left wing.



(a)



(b)

Figure 5.21: Histogram of differences in wing states between automated tracking and ground truth: (a) Right wing (b) Left wing.

## 5.6 Future Work

The results of this simulation demonstrated that in the ideal case of non-deforming wings, rigid bodies, and no legs to cause reconstruction artifacts, the automated tracker was capable of outperforming a human digitizer. The rms errors are given in Table 5.6

| Parameters | Auto | Manual |
|---|---|---|
| CG$_b$mm | 0.02 | 0.05 |
| CG$_{rw}$mm | 0.54 | 0.38 |
| CG$_{lw}$mm | 0.50 | 0.37 |
| Yaw deg | 0.49 | 1.04 |
| Pitch deg | 0.72 | 2.13 |
| Roll deg | 0.30 | 1.61 |
| $\theta_{rw}$ deg | 2.44 | 2.40 |
| $\alpha_{rw}$ deg | 10.18 | 4.26 |
| $\phi_{rw}$ deg | 2.65 | 1.50 |
| $\theta_{lw}$ deg | 3.00 | 2.84 |
| $\alpha_{lw}$ deg | 3.72 | 4.26 |
| $\phi_{lw}$ deg | 2.19 | 2.14 |

Table 5.6: Ground truth rms error in automated and manual tracking of a forward flight simulation video

## 5.7 Manual vs. Automated Tracking

The inherent differences between the two tracking techniques caused performance variations that were not necessarily due to errors. The body roll axis, as described earlier, was defined uniquely in the two methods resulting in an estimated correction factor of ten degrees. The insect was modeled as a kinematic chain in which the wing angles were defined with respect to the body coordinate system. Any differences in the body coordinate frame definition were coupled as differences in the wing angle definitions. For low roll angles, this had a direct impact on the peak-to-peak wing pitch $\alpha_w$ and stroke amplitude $\phi_w$. During high roll angles, additional affects were visible in the yaw angle and wing deviation $\theta_w$.

In manual tracking, the data is collected by landmark tracking, whereas the auto tracker considered all the insect silhouette pixels as individual measurements, which were then formulated into state estimates. The significant errors in manual tracking were a result of human error and interpolation strategies. The visualization tools qualitatively illustrated the accuracy of human digitizations, which varied depending on level of experience. Additionally, oscillatory affects due to the high dependency on frame interpolation in the body states were revealed, which required low-pass butterworth filtering with a cut off frequency of 50 Hz. The frame by frame analysis in the automated tracker prevented the oscillatory behaviour by Kalman filtering the high frequency noise in the measurements. The automated tracker offered more consistency in it's strategy to extract kinematics, but was prone to much different issues. The significant errors discovered in the auto tracker were a result of reconstruction artifacts, which were mistakenly considered as belonging to the insect. An unfortunate consequence of the visual hull reconstruction, particular in larger insects, were leg reconstruction artifacts that were confused as wings, especially during pronation of the wings. This affected wing angle estimation severely, which then caused the roll constraint to produce wildly noisy estimates. This failure mode was a result of the inability to differentiate wing pixels from leg pixels, something a human can easily handle.

| Parameters | Flight 1 | Flight 2 | Flight 3 | Flight 4 | Simulation |
|---|---|---|---|---|---|
| $CG_b$ mm | 0.25 | 1.78 | 0.30 | 1.67 | 0.02 |
| $CG_{rw}$ mm | 0.26 | 0.93 | 0.29 | 0.92 | 0.54 |
| $CG_{lw}$ mm | 0.33 | 0.98 | 0.32 | 0.73 | 0.50 |
| Yaw deg | 2.23 | 4.94 | 5.60 | 3.10 | 0.49 |
| Pitch deg | 1.90 | 2.66 | 5.00 | 2.44 | 0.71 |
| Roll deg | 2.71 | 5.81 | 4.69 | 7.33 | 0.30 |
| $\theta_w$ deg | 4.10 | 8.74 | 8.94 | 11.45 | 2.44 |
| $\alpha_w$ deg | 10.34 | 20.62 | 16.16 | 27.55 | 10.18 |
| $\phi_w$ deg | 5.97 | 9.39 | 6.83 | 6.56 | 2.65 |

Table 5.7: RMS differences between automated and manual tracking for the four flight sequences discussed earlier in addition to the forward flight simulation video.

## 5.8 Conclusions

In this thesis, a new approach to kinematic extraction of wing and body motions of dipteran insects was presented. The development of this method began by applying high speed videography techniques for a multiple camera set-up. An automated triggering system was designed to assist in synchronized recording and an image-based wand tracking program was developed to enable fast and accurate camera calibrations. Direct Linear Transformation (DLT) methods were derived from the pin-hole camera model and used to establish mappings between global space and image space. Manual techniques to extract wing and body kinematics were implemented with a modified and freely available DLT tracking program [13]. Computer vision techniques to process and analyze high speed imagery, in addition to markerless motion capture concepts were fused into a successful tracking program with comparable performance to a human digitizer. Previously devised tracking methods were expanded upon and new concepts that addressed wing deformation increased robustness to failure modes. Linear estimation with discrete

Kalman filters not only acted to provide good estimates of the 12 major wing and body states, but also permitted estimation of body velocities and rates. Generative models of the body and wings were constructed to extract inertial properties for different insects and to develop ground truth simulation videos to provide insight into the errors involved in manual and automated tracking in an ideal rigid body world. This work represents the first to present automated tracking results for more than one species of diptera. The simulation results demonstrated that yaw, pitch, and roll states can be estimated with rms errors less than one degree in simulated forward flight, whereas manual tracking had rms errors of approximately two degrees. Stroke amplitude $\phi_w$ and deviation $\theta_w$ were approximately $2 - 3^o$ in both tracking methods. Wing pitch $\alpha_w$ showed higher rms errors of up to $10.2^o$ in automated tracking and $4.25^o$ in human digitizing. In actual flight sequences and non-ideal flight conditions, yaw and pitch were still consistent in both methods with average rms differences of $5^o$ and $3^o$ respectively. Roll showed higher average rms differences of $6^o$, however the roll constraint used to estimate roll showed good performance in banked turns, where roll angles exceed $60^o$. The wing angles $\phi_w, \theta_w$, and $\alpha_w$ showed average rms differences of $9^o$,$8^o$,and $19^o$ between the two methods. These rms differences in the wing states are competitive with the current state of the art, whereas the body state estimates actually exceed previous performance reports. With the framework of the auto tracking program in place, it is only a matter of analyzing the failure modes that decrease the performance compared to manual digitizing. Over a hundred sequences have already been recorded with more than 20 sequences having been manually digitized. Digitization of those sequences required analyzing over 14,000 frames and extracting nearly 200,000 image coordinates. The time saving opportunities offered by automated camera triggering, calibration, and insect tracking are critical to building the kinematic libraries necessary to understand how flapping appendages of a mico-aerial system are controlled to produce lift and perform highly dynamic maneuvers. Lastly, the techniques presented here are not only limited to insect flight dynamics, but can be expanded to many different fields of research such as medical image processing and markerless motion capturing of other micro or larger scale systems.

## 5.9 Future Work

While the automated tracker has been shown to perform well in ideal conditions, there were certain scenarios where the tracker easily failed. The most common failure mode was a result of the visual hull reconstructions creating artifacts due to occlusions. The worst case scenario was when the wings were pointed backwards, transitioning from up-stroke to down-stroke, because any visibility of the legs in the images, especially from the top camera, led to leg reconstruction artifacts jutting out from the back of the fly and being mistaken as part of the wings. The high accuracy of the automated tracker in the simulation videos, which were free of any legs, suggests that a correction of this problem would greatly increase the performance of the tracker. Also, only two dipteran species were tested in this work, however the manual and automated tracking programs were devised to work on any dipteran insect. Future work to extract kinematics from different species would provide insight to the different approaches insects utilize to perform certain maneuvers.

The kinematic library that was started here can be used to develop flight dynamics models for different reference flight conditions by using quasi-steady aerodynamics, CFD, and experimental methods such as Robo-Fly [2]. The automated triggering circuit is capable of triggering a solenoid valve for performing gust disturbance studies. Disturbance rejection is a critical performance requirement in micro-aerial system development for achieving robustness to environmental uncertainties. Insects will hopefully show us how to get there.

# Appendix A

# Simulation

The following explains the step by step process involved in generating an insect flight simulation in MATLAB:

## Simulation Steps

1. Transform a geometric model of a fly.

$$[X_B]_g = M_{gb} \begin{bmatrix} [X_B]_b \\ 1 \end{bmatrix} \tag{A.1}$$

$$[X_W]_g = M_{gb} M_{bw} \begin{bmatrix} [X_W]_w \\ 1 \end{bmatrix} \tag{A.2}$$

2. Project the transformed model onto the equivalent of the 3 views using the DLT calibration parameters from a calibrated set-up.

$$u = \frac{L_1 X + L_2 Y + L_3 Z + L_4}{L_9 X + L_{10} Y + L_{11} Z + 1} \tag{A.3}$$

$$v = \frac{L_5 X + L_6 Y + L_7 Z + L_8}{L_9 X + L_{10} Y + L_{11} Z + 1} \tag{A.4}$$

3. Initialize 3 background image matrices, $B$, with maximum white noise intensity $w_n \sim N(0.125, 0.125^2)$.

4. Initialize a body image matrix , I$_b$, and add $I_b = 0.75$ to the pixel locations occupied by the body in the image.

5. Calculate the foreground density, $\rho$, and subtract an intensity $I_i = 0.2\frac{\rho}{\rho_{max}}$ from the i$^{th}$ body pixel. The goal is to recreate the halo of brighter intensity that appears around the body as a result of light diffusion around an object.

6. Initialize a right wing image matrix, I$_{rw}$ and add $I_w = 0.3$ to the pixel locations occupied by the right wing pixels in the image. Repeat for the left wing to get I$_{lw}$.

7. Add the wing images together to get I$_w$ = I$_{rw}$ + I$_{lw}$. Any pixels occupied by both wings will have a value of 0.6. Reassign all overlapping pixels a value of 0.5, for a more realistic contrast.

8. Add the background, body, and wing images together to get a temporary image of the fly I$_{temp}$ = $B$ + I$_b$ + I$_w$.

9. Set any pixels greater than $I = 1$ to $w_{max} \sim N(0.9, 0.1^2)$ .

10. Calculate the inverse of the image I$_{fly}$ = I$_{temp}^{-1}$. Low intensities now correspond to body pixels and higher intensities correspond to the wings.

11. Repeat the process for the entire kinematic sequence, saving the generated image matrices as images for each time step.

12. Convert images to movies.

13. Use the manual and automated techniques to track the simulated sequence.

# Appendix B

## Generative Modeling

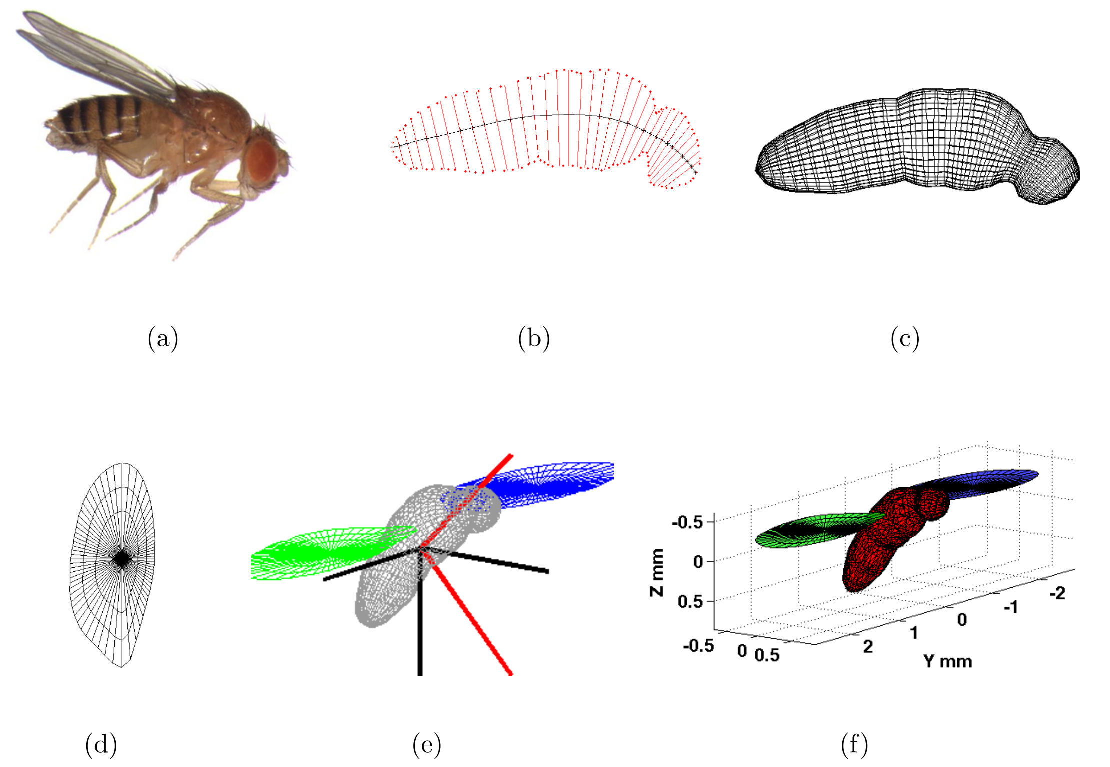


Figure B.1: Generative Model Process (a) Raw photo (b) Contour and b-spline analysis (c) Generative Body (d) Generative Wing (e) Completed Generative Model (f) Polyhedral Conversion

Generative Modeling is a method of describing shape by a set of elementary shape operators [11]. It is highly efficient and it can be used to generate simple models, such as an insect body from a single image. It was assumed that the shape of an insect is cylindrical about a body spline curve $s$. The generative modeling procedure was performed as follows:

1. Extract the body contour $c_b$ of a profile image of an insect.

2. Determine the head to tail center spline curve $s = [s_u 0 s_v]^T$.

3. The set of vectors $\vec{n}(s)$ normal to every point in $s$ determines a set of rotation angles $\theta(s)$.

4. The mean distance of the 2 intersections of every $\vec{n}(s)$ with $c_b$ to the corresponding $s$ is used to define a set of radii $r(s)$.

5. Transform the unit circle operator $\gamma(u)$ by $s$,$d(s)$,$\theta(s)$

$$\gamma(u) = \begin{bmatrix} \cos(2\pi u) \\ \sin(2\pi u) \\ 0 \end{bmatrix} \qquad u = [0 : du : 1]^T \tag{B.1}$$

$$[X_{\text{gen}}]_p = [R_2(\theta(s))r(s)\gamma(u) + s] \tag{B.2}$$

6. Rotate the body from the principal coordinate axis to the stability axis by a $\chi$ pitch rotation.

$$[X_{\text{gen}}]_b = R_2^{-1}(\chi)[X_{\text{gen}}]_p \tag{B.3}$$

7. Export the xyz point data as a *.asc file into Meshlab to create a refined polyhedral mesh using marching cubes algorithm.

8. Save the mesh file as a *.ply, which can be read into MATLAB using the *plyread* routine [20].

9. Upon initialization of the automated tracker, resize the model to the appropriate head-tail length. Similarly, a generative model of the wing is resized to the appropriate span length.

Upon generation of a polyhedral model, it was possible to perform approximations of the center of mass, volume, and moments and products of inertia using Mirtichs algorithm [18]. It was assumed the mean uniform body density of an insect was $\rho_b = 1100$ kgm$^{-3}$ [6]. The importance in estimating these quantities is crucial for any dynamic simulation or physically based modeling. The modeling process described here is a quick and simple procedure, which allows for quick model development to be used in model based automated tracking, quasi-steady/CFD system identification. The mass properties for a Drosophila Melanogaster with inertias given about the principal axis are given in Table B.1.

Table B.1: Mass properties for varying length Drosophila models

| Body Length | Mass | Ixx 1e-13 | Iyy e-13 | Izz 1e-13 | Ixy 1e-13 | Iyz 1e-13 | Ixz 1e-13 |
|---|---|---|---|---|---|---|---|
| mm | mg | kgm$^2$ | kgm$^2$ | kgm$^2$ | kgm$^2$ | kgm$^2$ | kgm$^2$ |
| 1.75 | 0.2664 | 0.0824 | 0.4727 | 0.4640 | 0.0000 | -0.0000 | 0.0034 |
| 2.0 | 0.3976 | 0.1606 | 0.9215 | 0.9047 | 0.0000 | -0.0000 | 0.0066 |
| 2.25 | 0.5661 | 0.2894 | 1.6606 | 1.6303 | 0.0000 | -0.0000 | 0.0119 |
| 2.5 | 0.7765 | 0.4902 | 2.8123 | 2.7609 | 0.0000 | -0.0000 | 0.0202 |
| 2.75 | 1.0337 | 0.7894 | 4.5292 | 4.4464 | 0.0000 | -0.0000 | 0.0325 |
| 3.0 | 1.3421 | 1.2197 | 6.9979 | 6.8699 | 0.0000 | -0.0000 | 0.0502 |

# Bibliography

[1] Y. I. Abdel-Aziz and H. M. Karara. Direct linear transformation into object space coordinates in close-range photogrammetry. In , Urbana, Illionis. Symposium on Close-Range Photogrammetry.

[2] J. M. Birch and M. H. Dickinson. The influence of wingwake interactions on the production of aerodynamic forces in flapping flight. *Journal of Experimental Biology*, (206):2257–2272, 2003.

[3] Jean-Yves Bouget. Camera calibration toolbox for matlab, July 2010.

[4] Ingo Bürk. Spectral clustering. Universität Stuttgart, 2012.

[5] Sachit Butail. *Motion Reconstruction of Animal Groups: From Schooling Fish to Swarming Mosquitoes*. PhD thesis, University of Maryland, 2012.

[6] C.P. Ellington. The aerodynamics of hovering insect flight ii. morphological parameters. In *Philosophical Transactions of the Royal Society of London*, volume 305 of *B*, pages 17–40. Biological Sciences, Royal Society of London, Feb 1984.

[7] I Faruque and JS. Humbert. Dipteran insect flight dynamics: Part 1: Longitudinal motions about hover. *Journal of Theoretical Biology*, 264(2):538–552, 2010.

[8] I Faruque and JS. Humbert. Dipteran insect flight dynamics: Part 2: Lateral-directional motions about hover. *Journal of Theoretical Biology*, 265(3):306–313, 2010.

[9] Imraan Faruque. *Control-Oriented Reduced Order Modeling of Dipteran Flapping Flight*. PhD thesis, University of Maryland, 2011.

[10] Zabala F. Dickinson M.H. Fontaine, E.I. and J.W. Burdick. Wing and body motion during flight initialization in *Drosophila* revealed by atuomated visual tracking. *The Journal of Experimental Biology*, 212:1307–1323, 2009.

[11] Tu Graz. Generative-modeling.org, July 2010.

[12] Hao Liu Hao Wang, Lijiang Zeng and Chunyong Yin. Measuring wing kinematics, flight rajectory and body attitude during forward flight and turning maneuvers in dragonflies. *The Journal of Experimental Biology*, 206:745–757, 2003.

[13] T. Hedrick. Software techniques for two and three-dimensional kinematic measurements of biological and biomimetic systems. *Journal of Bioinspiration and Biomimetics*, 2008.

[14] Janne Heikkilä and Olli Silvén. A four-step camera calibration procedure with implicit image correction. Technical report, Infotech Oulu and Department of Electrical Engineering, University of Oulu, 1997.

[15] Young-Hoo Kwon. Dlt method. 1998.

[16] Faruque I. Humbert J.S. Macfarlane K., Bush B. and Baeder J. Quasi-steady and computational aerodynamics applied to hovering drosophila dynamics. In *29th Applied Aerodynamics Conference*, Honolulu, Hawaii, 2011.

[17] N. J. B. McFarlane and C. P. Schofield.

[18] Brian Mirtich. Fast and accurate computation of polyhedral mass properties. *Journal of Graphics Tools*, 1(1), 1996.

[19] Hilton Adrian Mitchelson, Joel. Wand-based multiple camera studio calibration. Technical report, CVSSP, 2003.

[20] Gabriel Peyr. Matlab file exchange, 2003.

[21] Leif Ristroph. Automated hull reconstruction motion tracking(hrmt) applied to sideways maneuvers of free-flying insects. *The Journal of Experimental Biology*, 2009.

[22] Tsai R.Y. An efficient and accurate camera calibration technique for 3d machine vision. Miami Beach, Florida, 1986. IEEE Conference on Computer Vision and Pattern Recognition.

[23] Ki-Young Shin and Joung Hwan Mun. A multi-camera calibration method using a 3-axis frame and wand. Technical report, Korea Electrotechnology Research Institute, 2010.

[24] Michael H. Dickinson Steven N. Fry, Rosalyn Sayaman. The aerodynamics of free-flight maneuvers in drosophila. *Science Magazine*, 300:495–498, April 2003.

[25] M. Sujaritha and S. Annadurai. Color image segmentation using adaptive spatial gaussian mixture model. *International Journal of Information and Communication Engineering*, 2010.

[26] Kyrki V. and D. Kragic. Tracking unobservable rotations by cue integration. In *IEEE International Conference on Robotics and Automation*, pages 2744–2750, Orlando, FL. ICRA.

[27] Jordan Michael Weiss, Yair and Andrew Ng. On spectral clustering: Analysis and an algorithm. Technical report, U.C. Berkeley, 2002.

[28] Adrian Woolard. *Vicon 512 Manual.* Vicon Motion Systems.

[29] Paul Zarchan and Howard Musoff. *Fundamentals of Kalman Filtering*, volume 232. American Institute of Aeronautics and Astronautics, 3 edition, 2009.

[30] Zhengyou Zhang. A flexible new technique for camera calibration. volume 22, pages 1330–1334. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000.