ABSTRACT

Title of dissertation: MOTION RECONSTRUCTION OF ANIMAL GROUPS: FROM SCHOOLING FISH TO SWARMING MOSQUITOES

Sachit Butail

Dissertation directed by: Dr. Derek A. Paley
Department of Aerospace Engineering

The long-term goal of this research is to provide kinematic data for the design and validation of spatial models of collective behavior in animal groups. The specific research objective of this dissertation is to apply methods from nonlinear estimation and computer vision to construct multi-target tracking systems that process multi-view calibrated video to reconstruct the three-dimensional movement of animals in a group. We adapt the tracking systems for the study of two animal species: *Danio aequipinnatus*, a common species of schooling fish, and *Anopheles gambiae*, the most important vector of malaria in sub-Saharan Africa. Together these tracking systems span variability in target size on image, density, and movement. For tracking fish, we automatically initialize, predict, and reconstruct shape trajectories of multiple fish through occlusions. For mosquitoes, which appear appear as faded streaks on in-field footage, we provide methods to extract velocity information from the streaks, adaptively seek missing measurements, and resolve occlusions within a multi-hypothesis framework. In each case the research has yielded an unprecedented volume of trajectory data for subsequent analysis. We present kinematic data of fast-start response in fish schools and first-ever trajectories of wild mosquito swarming and mating events. The broader impact of this work is to advance the understanding of animal groups for the design of bio-inspired robotic systems, where, similar to the animal groups we study, the collective is able to perform tasks far beyond the capabilities of a single inexpensive robot.

MOTION RECONSTRUCTION OF ANIMAL GROUPS: FROM SCHOOLING FISH TO SWARMING MOSQUITOES

by

Sachit Butail

Dissertation 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
Doctor of Philosophy
2012

Dedication

To Suba and Insha

Acknowledgments

I sincerely thank my advisor Derek Paley for his guidance and support during these last five years. Among the many things that I have learnt from him are the importance of good writing and accessibility in communication—advice that I will follow in years to come. I would also like to thank Dr. Paley for the tremendous opportunities he has given me. I came into the PhD program hoping to work on things that move together; didn't know it would be fish schools and mosquito swarms. In studying these biological systems I have gained a unique interdisciplinary exposure and I stay excited as ever to explore further.

Thanks to my collaborators who were always enthusiastic about the tools I developed even though sometimes they didn't work as expected. In our fish work, I am grateful to Prof. Sheryl Coombs and her student Joe Coleman for hosting me at Bowling Green and teaching me much of what I know about fish. In our own lab, I have thoroughly enjoyed working with Amanda Chicoli and Jennifer Lun in multiple projects. In our mosquito work, I have found a friend in Nick Manoukis with whom I spent a memorable three weeks in Mali, Africa. José Ribeiro and Nick gave feedback on the tracking system and churned out huge amounts of data that we are all proud to present today. Moussa Diallo helped me in field work both in Mali and then in College Park. Thanks to Tovi Lehmann for reviewing our manuscripts, to Bob Gwadz for his generous support, and to Dick Sakai for hosting me in Mali.

For their time and feedback I am grateful to my dissertation committee members, Professors Rama Chellappa, Rob Sanner, Dave Akin, and Sean Humbert. Professor Chellappa was always approachable and gave important references and directions at various stages of my research.

To graduate students in our lab, Cammy Peterson with whom I completed every milestone in this program, you have been ever helpful. To Sonia Hernandez, Levi Devries, Seth Napora, Tracie Severson, and Nitin Sydney thanks for enriching my graduate life experience—I take with me happy memories. Thanks also to undergraduates in our lab, Julia Ashkanazy, Hongyi Xia, Christian Aller, and Aaron Sassoon for their efforts.

I have used several facilities and equipment for my research without which I couldn't have made any progress. I gratefully acknowledge Prof. Art Popper who let me use his lab for our fish experiments. Thanks to Prof. Avis Cohen and Eric Tytell for loaning us the camera equipment and recording software. I am also thankful to Space Systems Lab, especially Kate McBryan for her help with getting the Neutral

Buoyancy camera system to work. Thanks to the administrative staff at Aerospace engineering, especially Lavita Williams for placing countless equipment orders.

Several people outside my professional life have helped me for which I am ever grateful. I deeply appreciate William Shen's understanding and Lingyi Zhang's timely help on several occasions. To Anand, Byung, Praveen, Abhishek, Kedar, and Cindy, thanks for being patient listeners and showing excitement about my research.

My family has shown unwavering commitment to my work. I cannot ever thank enough my parents Sushma and Jagmohan Butail for encouraging me to continue my studies and for showing genuine interest in my work. To my sister-in-law Suja, and sister Gorun, who recently earned their doctorates, thanks for your help and for sharing your own experiences with me. I am grateful to my parents-in-law Pushkala and Narayanan Ramakrishnan for their optimism and belief in me.

I wouldn't have been able to start this journey let alone finish it without the support and love of two people who make me proud every day: my wife Suba and my daughter Insha. I dedicate this dissertation to them.

Table of Contents

List of Tables

List of Figures

Nomenclature

k	Time index, $k = 1, 2, \dots$
t	Target index, $t = 1, 2, \dots$
m	Measurement index, $t = 1, 2, \dots$
$\boldsymbol{X}_t[k]$	State of target t at time k
$\boldsymbol{Z}_m[k]$	Measurement m at time k
$\boldsymbol{r}$	Three-dimensional position
F	Motion model
H	Measurement model
c	Camera index, c=1,2,3
${}^c\boldsymbol{u}$	Location Measurement in pixels in cth camera image
$\mathcal{C}$	Camera reference frame
$\mathcal{W}$	World reference frame
T	4×4 transformation matrix
$\boldsymbol{g}$	Vertical axis in world frame
${}^cP$	3×4 camera projection matrix
$\mathcal{B}$	Body frame fixed to head
s	Midline coordinate, $s \in [0, 1]$
$\mathcal{E}(s)$	Elliptical cross section of fish body at s
$a(s)$	Semi-major axis of cross section at s
$b(s)$	Semi-minor axis of cross section at s
$d(s)$	Displacement of cross section along normal axis at s
$\boldsymbol{f}(s)$	Midline at s in body-fixed reference frame
v	Camera orientation dependent angle of a point on the surface
$\mathcal{S}$	The surface of a fish body
$\boldsymbol{h}$	Heading vector (orientation of head)
L	A line in three dimensions
$\boldsymbol{m}(s)$	Midline in world reference frame at s
${}^cO$	Occluding contour in camera c
$\boldsymbol{p}$	Vector of polynomial coefficients
$\boldsymbol{t}(s)$	Tangent vector to the midline at s
$\boldsymbol{x}(s)$	Normal vector to the midline at s
$\boldsymbol{y}(s)$	Binormal vector to the midline at s
$\boldsymbol{e}$	End point of a streak
S	Covariance of predicted measurement

Chapter 1

Introduction

Generating three-dimensional trajectory of every individual in an animal group from video is a multi-target tracking problem [1]. With targets that appear similar and have no distinguishable features, the challenge lies in data association, namely the matching of measurements and targets. This ambiguity exists across sensors and through time [2]. Occlusions, target maneuverability, target variability, and the presence of clutter and noise make the task of tracking every target through time non-trivial—a nonlinear combinatorial optimization problem [3]. In this dissertation we present methods to reconstruct three-dimensional trajectories of individual animals in a group using multi-view calibrated video. Using these methods, we design and validate tracking systems for two animal species *Danio aequipinnatus*, a common species of schooling fish, and *Anopheles gambiae*, the most important vector of malaria in sub-Saharan Africa. Fig. 1.1 shows sample outputs generated using our tracking systems.

1.1 Motivation

In the past two decades, mathematical models have been proposed that replicate collective behavior in nature. These have a few common characteristics [4, 1, 5]: the interactions between neighboring individuals is a function of metric distance, and an individual moves in an average agreement with each of its neighbors. These models form the first steps in replicating the vast array of collective motion patterns seen in animal groups in multi-robotic systems. Indeed, the nearest-neighbor interactions implicit in these models have inspired engineers to design distributed control laws

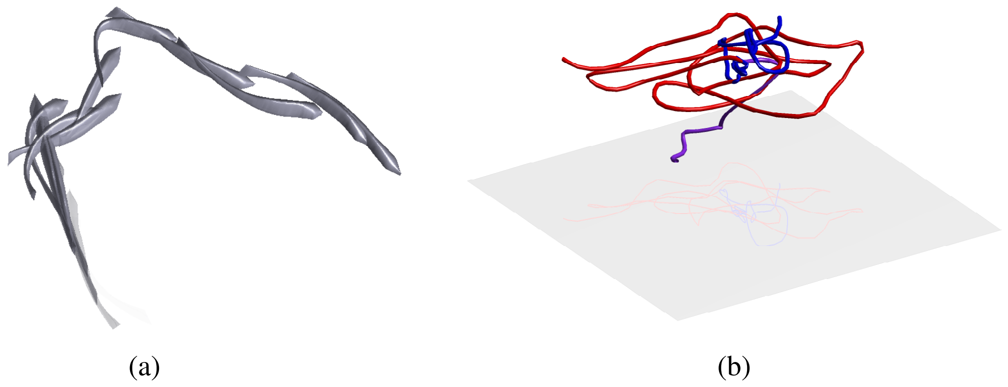


Figure 1.1: Tracking Results. Tracking systems described in this dissertation are used to reconstruct (a) full three-dimensional shape of startled fish in a school and (b) three-dimensional position and velocity of mating mosquitoes (female=red, male=blue, couple=purple) in a wild swarm. Projection on the horizontal plane is shown in lighter colors.

for possible implementation in mobile sensor networks [6, 7, 8]. Going forward, a goal for both the engineer and the biologist interested in collective behavior is to validate these and newer models against empirical data. While the engineer would like to see adaptability in robotic systems akin to social animal behavior [9, 10], the biologist would like to gain deeper insights by performing experiments that highlight the role of an individual in the group [11, 12, 13].

The impracticality of manually digitizing long videos of large animal groups limits the type of analysis and number of experiments that can be performed to understand collective behavior. The strength of automation in generating such data is evident from recent findings that have led to a better understanding of animal groups. For example, it has been shown that coordinated motion in birds is a function of topological distance [14] and consensus in fish schools is an effect of dynamically changing interaction rules [15]. Other results include validation of models based on consensus [16] and interaction topologies [17, 18], and implementations of multi-robot systems for collective transport [19]. In each case advancements in estimation theory and computer vision have made it possible to use non-invasive visual imaging to extract kinematic data.

Schooling fish and swarming mosquitoes are both instances of collective behavior. While the fish schools responding to an external threat [11] represent heightened coordination towards a single goal—to avoid predation, mosquito swarms are male gatherings (leks) where group members compete to stay in the center [20]—possibly to increase the chance of a successful mating [21]. Tracking individuals in schooling fish presents opportunities to understand the flow of information transmission for possible implementation in underwater robotic systems, and tracking wild swarming malarial mosquitoes is critical to expanding our understanding of the mating behavior for analysis that may inform the first steps towards strategies of vector control [22, 23].

1.2 Related Work

Significant advances in target tracking and computer vision over the past few years have largely been applied towards automation in tracking humans [24, 25]. Most of these techniques aim to track point targets or full body shape without markers. In that respect the challenges faced in tracking humans and animals are alike: to reconstruct non-rigid shapes, automatically initialize models, and track high density aggregations (for example crowds) [24].

Fish schools have been tracked in their natural environment [26] and in laboratories [27, 5]. In [26], Handegard et el. measured sensor orientation, range, and target strength (proportional to the size of the target [1]) using a split-beam echosounder mounted on a moving platform to track individual fish position in a school. Tracking is performed using an extended Kalman filter and targets are matched to the nearest measurement based on Euclidean distance. In [27], Schell et al. implemented a measurement-based tracking method called segmenting track identifier to track positions of up to fourteen fish in two dimensions. They use a modified probabilistic data association method to match measurements to tracks and join broken segments using least-squares. In [5], Viscido et al. track groups of four and eight fish in three dimensions filmed in a 1 m^3 acrylic tank using two cameras arranged orthogonally. Tracking is performed using custom software that joins

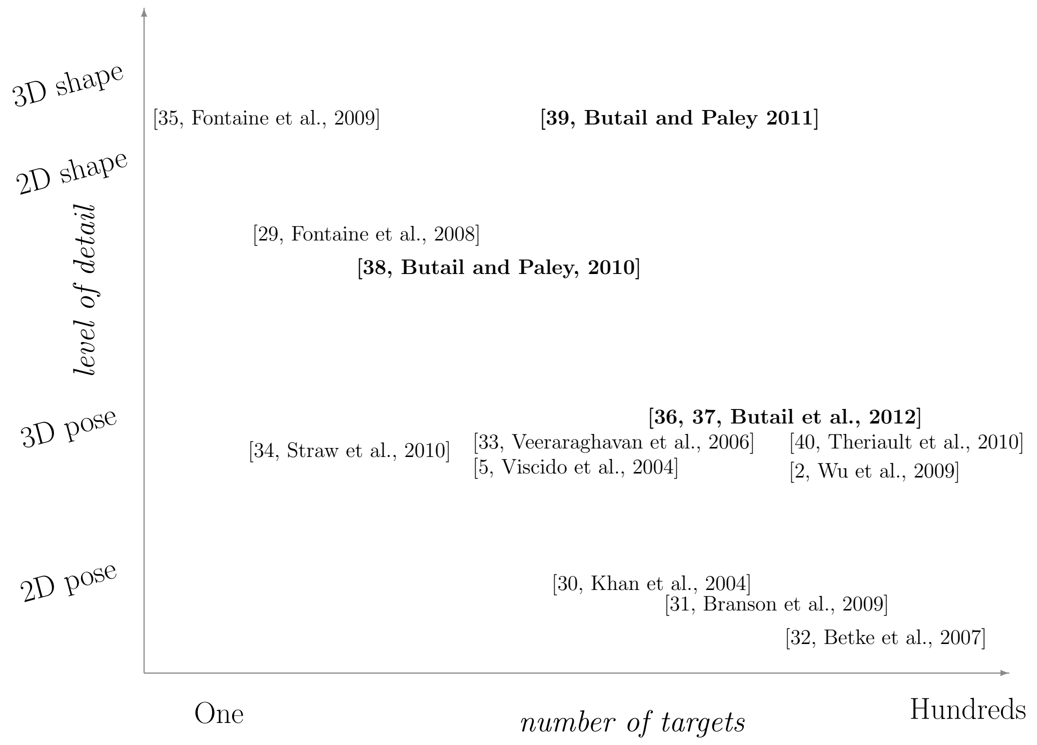


Figure 1.2: Related work. We classify the problem of tracking animal groups in a two dimensional space along number of targets and the size of the state space for each target. Tracking systems described in this dissertation appear in bold.

individual fish tracks in each camera separately using nearest-neighbor distance. Three-dimensional tracks are created by minimizing the least squares distance between projections of two dimensional tracks. In each instance, the fish are modeled as point masses; orientation and shape information is ignored. Shape kinematics for a single fish have been tracked manually [28] and automatically [29] in two dimensions. In [28] Hughes and Kelly use two cameras in a setup to film fish in flowing water. The midline is a series of points marked manually and projected on a plane of orientation from the side view. The plane of orientation projects from the line joining the camera center, snout, and tail of the fish. In [29], Fontaine et al. mark the midline manually in the first frame to fit a series of connected B-splines. A two-dimensional Frenet frame and body outline is used to create a symmetrical fish shape about the midline. Two dimensional shape of single zebrafish filmed at high speed (1500 frames per second) are tracked automatically using an unscented Kalman filter with edge points normal to the midline as measurements.

Multiple flying insects have been tracked in the laboratory [41, 42, 43, 34, 44], as well as in the field [2]. In [34], Straw et al. track multiple fruit flies filmed with up to eleven video cameras in a large arena, in real time. They use an extended Kalman filter with nearest-neighbor data association to match target estimates to measurements. Grover et al. [44] use similar technique on a smaller tracking volume to create visual hull of each fly as it is tracked. In [42], Zou et al. track multiple fruit flies filmed at high speed (200 frames per second) in an acrylic box by setting up the problem of data association across views and time in the form of a global optimization problem solved at every step. In [2], Wu et al. track up to a hundred bats with three cameras using a Kalman filter in conjunction with a multi-dimensional assignment strategy similar to multiple hypothesis tracking. Early attempts at localizing mosquito positions include the work of Gibson et al. [45], where they manual digitize mosquito tracks in two dimensions on film. In [46], Ikawa et al. built a three camera mobile setup that is used to take pictures of a swarm at fixed intervals. They use camera flashlight to illuminate the mosquitoes so that they show as bright spots. Correspondence between views is solved probabilistically to localize three dimensional position. More recently, in [20], Manoukis et al. use a mobile stereo setup with synchronized cameras to film mosquito swarms in the wild. Across-view correspondence is solved manually by marking each mosquito every 15 seconds and three dimensional position is obtained through triangulation. In each case, the mosquito is an identity-less point in space and the analysis have mainly focused on swarm density and structure.

1.3 Contributions

This research advances the current state of the art in tracking animal groups by constructing multi-target tracking systems that process muli-view calibrated video to reconstruct three-dimensional movement of every animal in a group. Together these tracking systems span variability in target size on image, density, and movement. The contributions of this dissertation are:

Automatic model initialization We model fish shape as a series of elliptical cross sections about a flexible midline. To track hundreds of experimental videos without the need to manually initialize every dataset we develop a method to automatically estimate the fish shape using multiple camera views (with at least one overhead view) for eventual use in shape reconstruction.

Shape reconstruction through occlusions We design a tracking system that reconstructs position, orientation, and shape of every fish in a dense school. We solve the problem of resolving sustained occlusions by formulating a variable dimension cost function minimized using simulated annealing—a provably convergent but slow optimization technique. We suggest a candidate generation function that significantly improves the time to convergence.

Generation of high-volume trajectory datasets We evaluate the performance of the shape-reconstruction algorithm and validate the tracking system on schools of up to eight fish. The tracking system is currently being used to investigate fast-start behavior of schooling fish in response to looming stimuli (see Fig. 1.3).

Velocity estimation from motion-blurred silhouettes We provide an end-point based likelihood function to estimate mosquito velocity from motion-blurred silhouettes appearing as streaks on the image plane. During target initialization, we use this likelihood function to reduce uncertainty in predicting position in the next frame.

Tracking through missing measurements and occlusions We design a tracking system that adaptively seeks faded streaks and resolves occlusions by splitting occluded blobs on image into individual measurements. We develop a graphical interface that assists a human for linking track segments output from the automated tracker. We evaluate the performance of the tracking algorithm, and validate the tracking system on stereo image sequences of wild mosquito swarms (see Fig. 1.4).

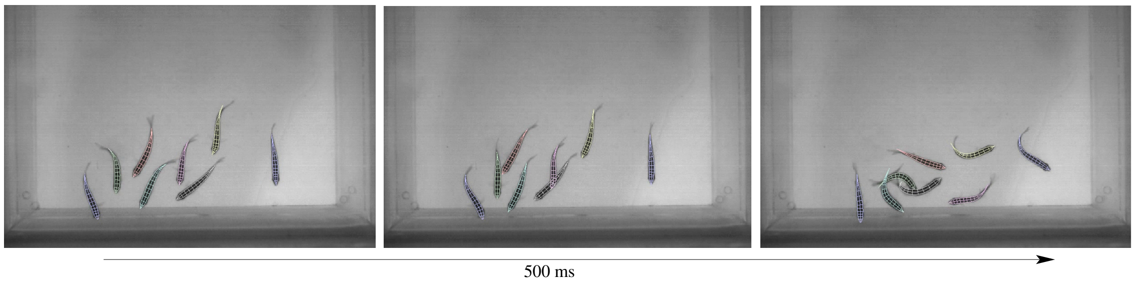


Figure 1.3: Startle response in a school of fish. In order to study information transmission in a fish school we tracked the full body shape of every fish even as they occluded each other. Time series data comprising position and heading quantifies the location of every fish with respect to its neighbor as well as the order of startle.

Figure 1.4: Raw images of mosquito swarms. Three sample images from the Left camera frame from three different days of swarming. A barely visible swarm is in center of each image.

Three dimensional mosquito swarming and mating trajectories We present validated tracking results in the form of three-dimensional trajectories of wild mosquito swarming and mating events filmed in Mali in August 2010. To date, the tracking system has been used to generate trajectory data of all mosquitoes in six swarms and six different mating sequences which is an order of magnitude (97 trajectories and 55,000 position points) larger than the last published result on reconstruction of wild mosquito swarms [20], and the first to contain three-dimensional trajectories rather than only positions.

Versions of the fish tracking system described in this dissertation have been previously published in conference paper [38], where we track multiple fish in a 1135 liters (300 gallon) water tank and journal article [39] which describes the high-speed tracking system detailed in this dissertation. A related model based tracking system for underwater vehicles appeared in conference paper [47], and an application of the tracking system in a virtual reality setup is published in conference paper [48]. An earlier version of mosquito tracking system has appeared briefly in conference paper [36], and in detail in journal article [37].

1.4 Outline

The dissertation is outlined as follows:

In Chapter 2 we formulate the motion reconstruction of animal groups as a multi-target tracking problem. We divide the problem into design components and review existing methods for each component in a separate section. Within each section we put our animal study into context, and justify our choice. A summary of our design choices and the reasons is provided in the end.

In Chapter 3 we apply the methods of nonlinear estimation and deformable shape representation for tracking fish shape in three dimensions. We present methods for estimating shape, data association, and shape reconstruction. Finally, we evaluate the performance of the tracking system on artificial and real datasets.[1]

Chapter 4 details the mosquito tracking system. Following an overview of the tracking algorithm, we describe novel parts of the tracking system, namely the likelihood function and the modified multi-hypothesis tracker for resolving occlusions. We evaluate the performance of the tracking algorithm and present representative three dimensional trajectories of mosquito swarming and mating.

We conclude in Chapter 5 with a summary of the dissertation and suggestions for future research.

Appendix A provides the linearization of nonlinear shape measurement model for use in a Kalman filter; Appendix B describes the camera calibration method for calibrating our multiview systems. Appendix C describes the optimal subpattern assignment metric that we use to evaluate the mosquito tracking algorithm.

[1]All procedures were done according to protocols R08-68 and R11-53 "Quantitative Analysis of Schooling Behavior" approved by the University of Maryland, College Park Institutional Animal Care and Use Committee.

Chapter 2

Tracking Animal Groups

In this chapter we divide the problem of tracking multiple, possibly non-rigid targets, into design components. For each component we review existing methods in computer vision and estimation theory and justify our choice. We begin in Section 2.1 with problem formulation in discrete time state space for a single target and extend the same to multiple targets. Motivated by nonlinearity in our measurement models we reason and describe the use of particle filter and simulated annealing for tracking in Section 2.2. In Section 2.3 we discuss methods to model deformable (non-rigid) shapes including the use of curve framing for generating arbitrary shapes. In Section 2.4 we bring attention to the maneuvering motion of our targets and review dynamic models that approximate the same. Section 2.5 details data association strategies to match measurements and targets and existing techniques to address occlusions. In Section 2.6 we underline the need for automatic initialization in our tracking algorithms to generate high volume datasets and discuss select approaches. We summarize in Section 2.7 with a list of design choices available for each component of the multi-target tracking system and our reasons for picking a particular option as a starting point.

2.1 State Space Representation

A target t at time step k is described by the state vector $\boldsymbol{X}_t[k] \in \mathbb{R}^n$. Examples of target state include the three-dimensional position and velocity in which case $n = 6$. A measurement m at time k is denoted by $\boldsymbol{Z}_m[k] \in \mathbb{R}^m$. In the case of radar systems, the measurements typically consist of range and bearing in which case $m = 2$. In a camera image the measurement may be pixel location of the blob centroid ($m = 2$) or the silhouette of the blob in which case m is variable. In a Bayesian framework, the tracking algorithm recursively iterates through two steps, the update step and the predict step. The update step uses a measurement model to revise the estimate based on new observations. The predict step integrates a motion model to obtain the target state at the time of the next measurement. Assuming motion model F and measurement model H, the state of target t satisfies

$$\begin{aligned} \boldsymbol{X}_t[k] &= F(\boldsymbol{X}_t[k-1], \boldsymbol{w}) \\ \boldsymbol{Z}_m[k] &= H(\boldsymbol{X}_t[k], \boldsymbol{n}), \end{aligned} \tag{2.1}$$

where $\boldsymbol{w}$ and $\boldsymbol{n}$ denote disturbance and noise values respectively, and $\boldsymbol{Z}_m[k]$ is known to have been generated by target t. Because of noise, disturbances, and approximations in F and H, the state $\boldsymbol{X}_t[k]$ is a random quantity and (2.1) describes a random process. The conditional probability density function (pdf) of $\boldsymbol{X}_t[k]$ given all measurements from target t up to k, $\boldsymbol{Z}_m^k$, is called the filtering pdf. The filtering pdf denoted by $p(\boldsymbol{X}_t[k]|\boldsymbol{Z}_m^k)$ is related to the likelihood of $\boldsymbol{Z}_m[k]$ given $\boldsymbol{X}_t[k]$, $p(\boldsymbol{Z}_m[k]|\boldsymbol{X}_t[k])$, by Bayes' rule

$$p(\boldsymbol{X}_t[k]|\boldsymbol{Z}_m^k) = \frac{p(\boldsymbol{Z}_m[k]|\boldsymbol{X}_t[k])p(\boldsymbol{X}_t[k]|\boldsymbol{Z}_m^{k-1})}{p(\boldsymbol{Z}_m^k|\boldsymbol{Z}_m^{k-1})}, \tag{2.2}$$

where $p(\boldsymbol{X}_t[k]|\boldsymbol{Z}_m^{k-1})$ is the prior pdf and $p(\boldsymbol{Z}_m^k|\boldsymbol{Z}_m^{k-1})$ is a normalizing constant. The prior pdf is related to the filtering pdf at $k-1$ by

$$\begin{aligned} p(\boldsymbol{X}_t[k]\big|\boldsymbol{Z}_m^{k-1}) &= \int p(\boldsymbol{X}_t[k]\big|\boldsymbol{X}_t[k-1], \boldsymbol{Z}_m^{k-1})p(\boldsymbol{X}_t[k-1]\big|\boldsymbol{Z}_m^{k-1})d\boldsymbol{X}_t[k-1] \\ &= \int \underbrace{p(\boldsymbol{X}_t[k]\big|\boldsymbol{X}_t[k-1])}_{\text{transition pdf (F)}} p(\boldsymbol{X}_t[k-1]\big|\boldsymbol{Z}_m^{k-1})d\boldsymbol{X}_t[k-1]. \text{ (Markov)} \end{aligned} \tag{2.3}$$

The Markov property (first order) implies that the current state at k depends only on the previous state at $k-1$.

For multiple targets the filtering pdf $p(\boldsymbol{X}[k]|\boldsymbol{Z}^k)$ is the probability of the joint state $\boldsymbol{X}[k]$ of all targets given the set $\boldsymbol{Z}^k$ of all measurements up to k. We make two assumptions:

a1) targets do not interact at short time-scales; and

a2) a target gives rise to one measurement only.

Based on these assumptions, the joint filtering pdf is

$$\begin{aligned}
p(\boldsymbol{X}[k]\big|\boldsymbol{Z}^k) &= p(\boldsymbol{X}_1[k], \boldsymbol{X}_2[k], \ldots, \boldsymbol{X}_{n_t}[k]\big|\boldsymbol{Z}^k) \\
&= p(\boldsymbol{X}_1[k]\big|\boldsymbol{Z}^k)p(\boldsymbol{X}_2[k]\big|\boldsymbol{Z}^k)\ldots p(\boldsymbol{X}_{n_t}[k]\big|\boldsymbol{Z}^k) \text{ (non-interacting targets)} \\
&= p(\boldsymbol{X}_1[k]\big|\boldsymbol{Z}[k], \boldsymbol{Z}^{k-1})p(\boldsymbol{X}_2[k]\big|\boldsymbol{Z}[k], \boldsymbol{Z}^{k-1})\ldots p(\boldsymbol{X}_{n_t}[k]\big|\boldsymbol{Z}[k], \boldsymbol{Z}^{k-1}) \\
&= \prod_{t=1}^{n_t[k]} \frac{\overbrace{p(\boldsymbol{Z}[k]\big|\boldsymbol{X}_t[k])}^{\text{likelihood function}}\, p(\boldsymbol{X}_t[k]\big|\boldsymbol{Z}^{k-1})}{p(\boldsymbol{Z}[k]\big|\boldsymbol{Z}^{k-1})} \text{ (Bayes rule, independent observations)} \\
&= \prod_{t=1}^{n_t[k]} \sum_{m}^{n_m[k]} \beta_{tm} \frac{p(\boldsymbol{Z}_m[k]\big|\boldsymbol{X}_t[k])p(\boldsymbol{X}_t[k]\big|\boldsymbol{Z}^{k-1})}{p(\boldsymbol{Z}[k]\big|\boldsymbol{Z}^{k-1})}, \text{ (one measurement per target)}
\end{aligned} \tag{2.4}$$

where β_{tm} is the association probability of target t with measurement m, and $n_t[k]$ and $n_m[k]$ are the number of targets and measurements at k respectively. We also assume that the observation $\boldsymbol{Z}[k]$ is independent of previous observations. Together (a1) and (a2) split the joint multi-target estimation problem (2.4) into multiple single-target estimation problems (2.2).

In the case when (2.1) is linear and $\boldsymbol{w}$ and $\boldsymbol{n}$ are additive white noise Gaussian processes, the filtering pdf is also Gaussian and is fully represented by the mean and the covariance matrix. A Kalman filter gives the optimal (in the sense minimum mean square error) maximum *a posteriori* (MAP) estimate that maximizes the filtering pdf. Nonlinear motion and measurement models make the computation of the pdf (2.2) impractical due to the multi-dimensional integral (2.3). To sample from the filtering pdf we must therefore resort to other methods. In the next section we discuss two methods in detail: particle filtering, which indirectly samples from the filtering pdf, and simulated annealing, a metaheuristic (local search based) optimization technique that converges to the MAP of the filtering pdf with a sufficiently slow annealing schedule [49, 50].

2.2 Nonlinear Estimation

Suboptimal methods to sample from the filtering pdf include the extended Kalman Filter (EKF), unscented Kalman filter (UKF) and the particle filter. The EKF [51] retains the Gaussian assumption on noise and disturbance and performs filtering using a first-order linearization of F and H. EKF performance depends on the linearization error and diverges easily for moderately nonlinear motion and measurement models [52]. The UKF marks a significant improvement over the EKF by representing the random state by weighted sample points that capture the pdf up to a third order linearization [53], but the UKF cannot handle multimodal density functions and non-Gaussian noise and disturbance. Due to ambiguity in orientation and shape of our targets, we encounter multimodal likelihood functions in tracking fish [38] as well as mosquitoes [36]. A particle filter [54, 55] represents the filtering pdf as a point-mass distribution and relaxes most restrictions on the target and measurement models and the disturbance and measurement noise. However, a particle filter can be computationally burdensome for a large state space. Based on the assumption (a1) that the targets do not interact at short time-scales ($<$ 40 ms), a separate filter for each target may be run. The particle filter recursively samples from the filtering distribution by employing importance sampling, a method we describe next.

Importance sampling

Importance sampling is a method to sample from a target distribution $p(\boldsymbol{X}[k]|\boldsymbol{Z}^k)$ indirectly by instead sampling from a known, easy-to-sample, proposal distribution [52, 55].[1] Let $\boldsymbol{X}^i$ be a sample generated from the proposal distribution $\pi(\boldsymbol{X}[k]|\boldsymbol{Z}^k)$. The importance weight w^i is

$$w^i = \frac{p(\boldsymbol{X}^i[k]|\boldsymbol{Z}^k)}{\pi(\boldsymbol{X}^i[k]|\boldsymbol{Z}^k)}. \tag{2.5}$$

[1]The proposal distribution is also referred to as the importance sampling distribution.

The filtering pdf $p(\boldsymbol{X}[k]|\boldsymbol{Z}^k)$ can be approximated by

$$\hat{p}(\boldsymbol{X}[k]|\boldsymbol{Z}^k) = \sum_{i=1}^{N_s} \tilde{w}^i \delta(\boldsymbol{X}^i[k] - \boldsymbol{X}[k]), \tag{2.6}$$

where N_s is the number of samples generated, and $\tilde{w}_i = w^j \left(\sum_{i=1}^{N_s} w^i \right)^{-1}$ is the normalized weight. In order to implement importance sampling in recursive estimation the proposal distribution $\pi(\boldsymbol{X}[k]|\boldsymbol{Z}^k)$ at k must admit the proposal distribution at $k-1$ as the marginal so that the following relation holds

$$\pi(\boldsymbol{X}[k]|\boldsymbol{Z}^k) = \pi(\boldsymbol{X}[k]|\boldsymbol{X}[k-1], \boldsymbol{Z}^k)\pi(\boldsymbol{X}[k-1]|\boldsymbol{Z}^{k-1}). \tag{2.7}$$

This permits us to recompute the weights at k using the weights at $k-1$ and Bayes' rule (2.2)

$$w^i[k] \propto w^i[k-1] \frac{p(\boldsymbol{Z}[k]|\boldsymbol{X}^i[k])p(\boldsymbol{X}^i[k]|\boldsymbol{Z}^{k-1})}{\pi(\boldsymbol{X}^i[k]|\boldsymbol{X}^i[k-1], \boldsymbol{Z}^k)}. \tag{2.8}$$

We recall (2.3) to realize that a choice of proposal distribution that also satisfies the condition (2.7) is the prior pdf $p(\boldsymbol{X}[k]|\boldsymbol{Z}^{k-1})$ at k which can be sampled from by augmenting the filtering pdf at $k-1$ with transition density. Therefore, beginning with a guess for the prior at $k = 0$, the importance sampling algorithm propagates particles and updates their associated weights every time a measurement is received. This sampling strategy, however, is inefficient and quickly degenerates to a single particle [55]. Degeneracy is mitigated by using normalized weights to redistribute the particles according to their weights in a resampling step (also known as bootstrapping) [54]. The resulting filter is known as the sampling importance resampling filter (SIR). The SIR algorithm is listed in Table 2.1.

Other methods that improve the sampling efficiency include Rao-Blackwellization [55, 56], unscented particle filter [57], and MCMC sampling [30, 55]. Rao-Blackwellization is amenable to problems where the state can be partitioned so that a portion can be computed analytically; thus effectively marginalizing out that part of the estimate from the filtering pdf. The unscented particle filter uses the unscented transformation to sample from a proposal distribution. This permits sampling from heavy-tailed proposal distributions that admit the support of the filtering pdf and

Table 2.1: **Sampling Importance Resampling Particle Filter**

```
Input:   Set of particles $\{\boldsymbol{X}_t^i[k]\}_{i=1}^{N_s}$, for each target $t$; an associated measurement
         $\boldsymbol{Z}_m[k]$ for each target, likelihood function $P(\boldsymbol{Z}_m[k]\big|\boldsymbol{X}_t[k])$ and
         motion model $F$.
Output:  Mode or mean of the distribution from the particle set $\{\boldsymbol{X}_t^i[k+1]\}_{i=1}^{N_s}$
1: Compute weights for each particle using the likelihood function $P(\boldsymbol{Z}_m[k]\big|\boldsymbol{X}_t[k])$
   as $w_i = P(\boldsymbol{Z}_m[k]\big|\boldsymbol{X}_t^i[k])$
2: Normalize the weights $\tilde{w}^i = w^i/(\sum_{i=1}^{N_s} w^i)$.
3: Resample the particles using the cumulative distribution of normalized weights
   $\{\tilde{w}^i\}_{i=1}^{N_s}$[54].
4: Predict: Use the motion model $F$ to propagate each particle to the next time-step.
```

include recent observations which may have been missed by the prior distribution [57]. Markov Chain Monte Carlo (MCMC) is a class of methods that sample from the target distribution by simulating a Markov process (offline, as opposed to online as in particle filtering) whose equilibrium density is the same as the target distribution. In order to draw samples from the target distribution the Metropolis Hastings algorithm is used [58], which accepts a sample generated from the proposal distribution $\pi(\boldsymbol{X}^i; \boldsymbol{X}^j)$ with the probability α given by

$$\alpha = \min\left(\frac{p(\boldsymbol{X}^j\big|\boldsymbol{Z}^k)\pi(\boldsymbol{X}^i;\boldsymbol{X}^j)}{p(\boldsymbol{X}^i\big|\boldsymbol{Z}^k)\pi(\boldsymbol{X}^j;\boldsymbol{X}^i)}, 1\right). \tag{2.9}$$

The Metropolis Hastings algorithm is also used in a hill climbing optimization technique called simulated annealing [59].

Simulated Annealing

Simulated annealing (SA) mimics the thermodynamic annealing process by accepting a jump out of a local minimum with a probability that decreases as the temperature cools down according to a cooling schedule [59]. The probability of acceptance is computed using the Metropolis Hastings algorithm described in MCMC sampling (2.9). The main difference is that the SA algorithm simulates an inhomogeneous Markov chain (in contrast to a homogeneous Markov chain in MCMC sampling) where the transition probabilities change as the temperature is lowered. Instead of sampling from the target distribution, we extract the MAP estimate of the distribution by casting the system (2.1) into a numerical optimization problem. We minimize an objective function $C(\boldsymbol{X}) = \|\boldsymbol{Z} - H(\boldsymbol{X}, \boldsymbol{n})\|$, which evaluates the match between measurements and the estimate. The main components of the SA algorithm are the perturb function, which generates candidate solutions, the cost function that evaluates the solutions, a cooling schedule that lowers the temperature thereby reducing the probability of a jump, and a termination criteria. The general algorithm is summarized in Table 2.2.

Table 2.2: **Simulated Annealing Algorithm**

Input: Cost function $C : \mathbb{R}^n \to \mathbb{R}$, perturb function $\boldsymbol{r} : \mathbb{R}^n \to \mathbb{R}^n$, and a non-increasing cooling schedule $T : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$

Initialize: State estimate at current time-step, $\boldsymbol{X}^1 = \boldsymbol{X}[k]$

Until a *termination criteria* is reached, iterate for $j = 1, 2, \dots$

1: Perturb $\tilde{\boldsymbol{X}}^j = \boldsymbol{r}(\boldsymbol{X}^j)$. Let $\delta C = C(\boldsymbol{X}^j) - C(\tilde{\boldsymbol{X}}^j)$ be the change in cost.

2: Sample from a uniform distribution $\rho \sim \mathbb{U}(0,1)$ and update the state:

$$\boldsymbol{X}^{j+1} = \begin{cases} \tilde{\boldsymbol{X}}^j & \text{if } \rho \leq \min(1, \exp(-\delta C/\tau^j)) \\ \boldsymbol{X}^j & \text{otherwise,} \end{cases}$$

where τ^j is the temperature.

3: Update the temperature $\tau^{j+1} = T(\tau^j, j)$.

One or more termination criteria may be used such as reaching a freezing temperature, exceeding a maximum number of evaluations, or attaining a minimum cost value. Note that we can simulate a homogeneous Markov chain by enforcing an *inner-loop criterion* on steps 1 and 2, running them in a loop until additional criteria are satisfied (the temperature stays constant) [60].

While convergence results are proven for both homogeneous and inhomogeneous Markov chains [61], the time taken in each case is impractically slow. In [62] convergence results are shown for a logarithmic cooling schedule $T = \tau^0/\log(1+j)$ with Gibbs sampling [63]. In [60] finite time behavior has been analyzed for inhomogeneous Markov chains. They suggest a cooling schedule $T = rL/\log(1+j+j_0)$ that depends on r, the radius of the underlying graph, and L a bound on the local slope of the cost function; j_0 is a design parameter. Simulated annealing algorithm has been successfully applied to discrete optimization problems in image restoration [49, 62]. A parallelized version known as the annealing particle filter has been implemented in markerless motion capture [64].

In tracking fish shape, where the task is to fit a polynomial midline to a three-dimensional region, the resulting optimization problem has multiple minima. This is because similar curves can be attained with different parameter sets. We therefore use simulated annealing to solve it. We adopt the following methods to improve finite time behavior and verify convergence: (a) we downsample the search space for the initial run and approach the actual resolution in successive runs; (b) we modify the proposal distribution to generate solutions arising from approximate fish motion; and (c) we verify convergence using synthetic images.

2.3 Deformable Shape Representation

Point mass representation of extended objects that deform on video leads to loss of accuracy in position information because the location of centroid on the image blob depends on instantaneous shape.

Deformable objects may be tracked using model-based methods that enforce a pre-determined shape geometry on image blob features. An exoskeleton model may be generated using cylinders and cones joined at their ends, where as a midline skeleton may be constructed using lines [65]. Instead of using cylinders or cones, shapes that can deform based on a parameter may also be used. These include quadrics [66, 67] and superquadrics [68].

Once a model is determined, two-dimensional features such as silhouettes [44] and occluding contours [66, 29, 35] may be extracted using background subtraction followed by contour tracing. Correspondingly, three-dimensional convex volume that encloses the target may be extracted from volumetric pixels (voxels) in the filming volume [69].

The last step in tracking deformable objects is to fit the shape model on to the raw features. This can be done by computing the principal components of the feature set as is done for tracking fruit fly in [69], or using a measurement model within a filter to minimize the combined distance of the features from the model as is done in tracking vehicles [70], hands [66], mice [71], and a fly [35]. Instead of directly using raw features to enforce a model, intermediate steps that track deformable shapes locally through a set of control points may be used. These include active contours, which wrap a pre-defined contour based on a decreasing energy function around the edges of regions of high-contrast [72, 73], mesh modeling, which constructs a 2D triangular mesh model of the target [74], and medial axis transform, which finds the set of points that are centers of circles tangent to two points on the occluding contour [65]. In order to extract meaningful pose and shape data one must again enforce a geometric model on the output of these methods. In [75] an unscented Kalman filter is used to deform an active contour along specific control points to track single targets through occlusions. The contours are then projected on to PCA base generated from a training set of images.

Generative modeling provides a framework for modeling the shape of asymmetrical objects. A generative model may be produced by rotating and translating a shape along a trajectory [76]. Formally, a continuous set of transformations are applied on a shape (also called the generator) to build a generative model. A generator of the form $\gamma(v) : \mathbb{R} \rightarrow \mathbb{R}^3$ is transformed through a parameterized transformation, $\delta(\gamma(v), s) : \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$, to form a surface. For example, a cylinder with radius r

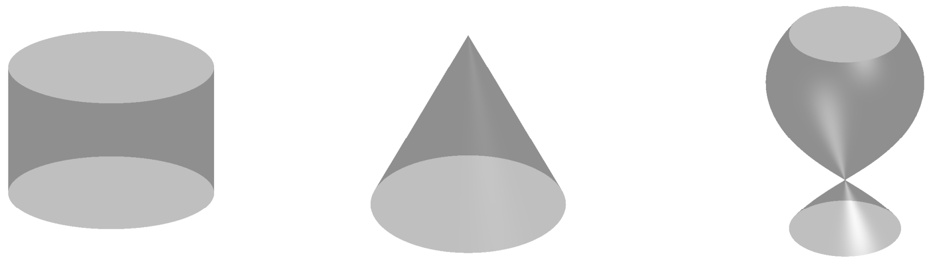


Figure 2.1: Generative modeling. A cylinder, cone, and a goblet shape produced using generative modeling.

is produced by choosing

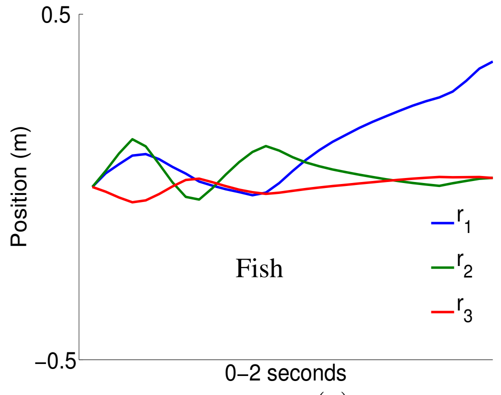
$$\gamma(v) = \begin{bmatrix} \cos v \\ \sin v \\ 0 \end{bmatrix} \text{ and } \delta(\gamma(v), s) = \begin{bmatrix} r\gamma_1 \\ r\gamma_2 \\ s \end{bmatrix}, \tag{2.10}$$

where $s \in [0, 1]$ and $v \in [0, 2\pi]$. Similarly a cone is produced by decreasing $r = 1 - s$ linearly along the trajectory (see Fig. 2.1). Complex shapes about a three-dimensional curve require a coordinate frame at each point on the curve for projecting the generator.

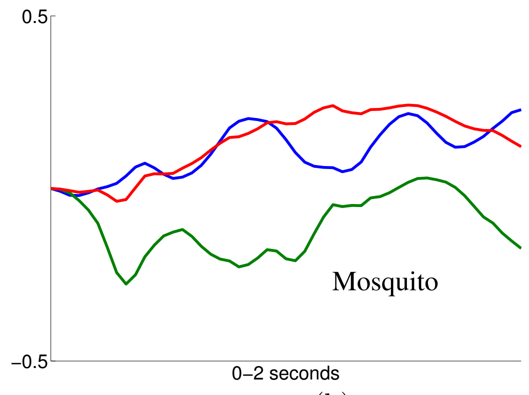
In a curve framing setting [77] under the assumption that the curve $\boldsymbol{f}(s) \in \mathbb{R}^3$ is twice differentiable, an orthogonal frame may be used to transform the generator $\gamma(v)$ onto a point on the curve. For tracking fish shapes we model each fish shape as a series of elliptical cross sections about a flexible midline. Similar approximation of fish cross section as an ellipse was done in [78]. Let $\boldsymbol{N}(s)$ and $\boldsymbol{B}(s)$ be the normal and binormal vector to the curve $\boldsymbol{f}(s)$ at s. Beginning with a simple circle for the generator $\gamma(v)$ in (2.10), we add functions $a(s), b(s)$ of scaling parameter s as ellipse axes lengths to generate a surface

$$\delta(\gamma, s) = \begin{bmatrix} \boldsymbol{N}(s)a(s) & \boldsymbol{B}(s)b(s) & \mathbf{0}_{3\times 1} \end{bmatrix} \gamma + \boldsymbol{f}(s), \tag{2.11}$$

where the $\mathbf{0}_{3\times 1}$ entry denotes that the cross section is planar (zero thickness in the direction tangent to the curve).



(a)



(b)

Figure 2.2: Maneuvering targets. Position trajectories in each of the three dimensions for (a) fish and (b) mosquitoes are shown for a randomly selected 2 second window of tracked data.Both fish and mosquitoes exhibit maneuvering target motion that can hardly be approximated by a simple motion model.

2.4 Maneuvering Targets

Individual animal motion within a group is characterized by rapid maneuvers comprising quick turns and speed-ups to avoid collisions and predators, and to keep up with the rest of the group. These sudden changes in kinematics are distinctly different from regular motion and hard to model with a simple motion model. For example, fish startles comprise quick turns and speed-ups up to ten times the regular motion [79]. Three-dimensional reconstruction of mosquitoes reveal speeds ranging between 1–4 m/s and accelerations up to 10 m/s^2 [37]. Acceleration up to 2 m/s^2 have been recorded previously for midges [80].

Maneuver models for tracking maneuvering targets include acceleration models that include acceleration in the state space, and turning models that incorporate turn rates in the dynamics [81]. Examples of acceleration models include [51, 81] white noise acceleration, and Singer acceleration models. Compared to the constant velocity model, $d\dot{\boldsymbol{r}} = d\boldsymbol{w}$, the white noise acceleration model where target acceleration is a white noise process $d\ddot{\boldsymbol{r}} = d\boldsymbol{w}$ represents greater ambiguity in acceleration. Another maneuver model called the Singer acceleration model approximates target acceleration as a first order Markov process with correlation $R_a(\tau) = \sigma^2 \exp(-\alpha|\tau|)$, where α is inversely proportional to the maneuver time constant: α is 1/60 for a lazy turn and 1 for sudden jerks.

In tracking mosquitoes where a maneuver typically lasts for a fraction of a second, the Singer model reduces to constant velocity. We evaluate a constant velocity model perturbed by a white noise Gaussian process whose variance was computed empirically with white noise and Singer models and observe that the constant velocity model produces less variance in the predicted estimates, which in our case leads to better data association.

Acceleration maneuver models perform poorly in predicting turns [81] especially because turns are typically an effect of torque inputs in the body frame [81]. We model fish movement by modeling unknown turn rates in the body frame, as a function of curvature in [38], and as independent disturbance processes that reflect higher yaw movement compared to pitch in this dissertation.

2.5 Data Association

A multi-target tracking system must associate measurements and targets. A target-based method associates each target to a measurement [51], whereas a measurement-based method associates each measurement to a target [82]. The difference is that a measurement-based method can inherently handle a variable number of targets, which may appear and disappear from the field of view. The reliability of the association depends on the proximity of the actual measurement to the predicted measurement, which is produced from the target estimate using the motion model and the measurement model; measurement proximity is determined using a distance function.

2.5.1 Gating Measurements

A data association strategy begins with evaluating candidate associations prior to making a match. The number of possible associations can be significantly reduced by gating the measurements. The gating volume or validation region is generated from the difference $\nu = \boldsymbol{Z}_m - f(\boldsymbol{X}_t)$ between the position measurement $\boldsymbol{Z}_m \in \mathbb{R}^m$ and

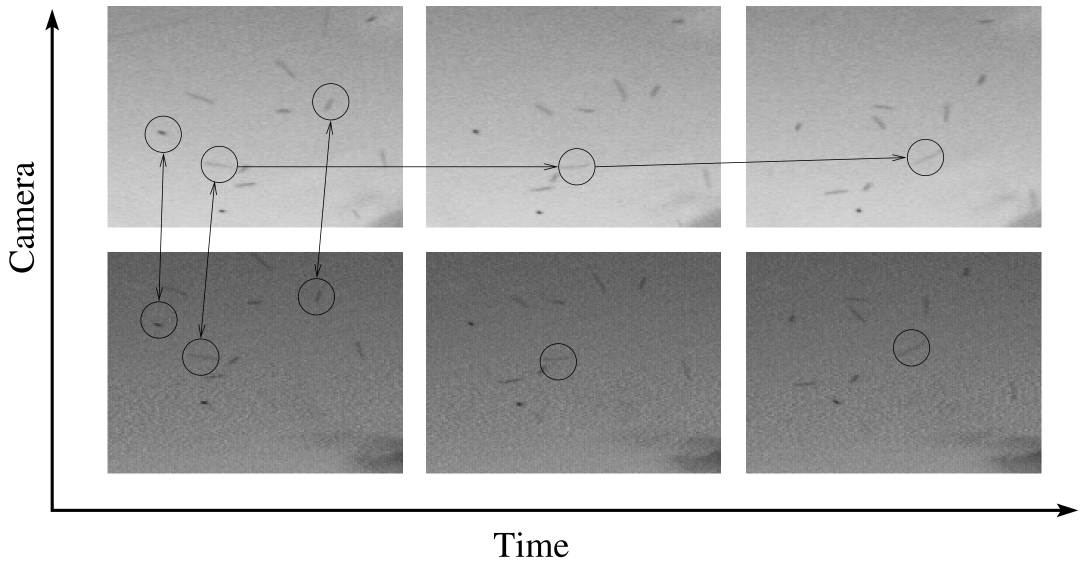


Figure 2.3: Data association. To generate meaningful tracks, the identity of each target must be maintained across camera views and through time.

the predicted position measurement $f(\boldsymbol{X}_t) \in \mathbb{R}^n$. Let S be the covariance of ν, which is also called the innovation. If a measurement that is normally distributed about the true value lies within the validation region, the weighted norm $\|\nu\|^2 = \nu^T S^{-1} \nu$ satisfies $\|\nu\| < t_{gate}$. (The quantity $\|\nu\|$ is also known as the Mahalanobis distance.) For example, a threshold value $t_{gate} = 16$ defines a region around a predicted two-dimensional measurement (m=2) with 99.97% probability of containing the actual measurement [51]. A large gating volume ensures a high probability that the true association will be evaluated, but increases computational load and the possibility of track switching. A small gating volume that misses the true measurement results in track termination. The aim therefore is to gate true measurements as tightly as possible. Methods to reduce the size of gating volume include weighted correlation matching [83] and window warping [84].

We use a simple gating volume for tracking fish, but then associate targets to measurements based on a midline distance function. For tracking mosquitoes, we use the gating volume for two purposes: to evaluate possible associations, and to seek missing measurements.

Across-view measurement associations can be gated based on the epipolar constraint described as follows. Let ${}^{c}\boldsymbol{u} \in \mathbb{R}^2$ be the location of centroid of a blob in camera c. Let $\tilde{\boldsymbol{u}}^c = [(\boldsymbol{u}^c)^T, 1]^T$ be the homogeneous representation of $\boldsymbol{u}^c$. Assuming without loss of generality that camera 1 is the inertial frame, a pair of measurements with midpoints $\boldsymbol{u}^1$ and $\boldsymbol{u}^2$, one from each camera, must satisfy epipolar constraint [85]:

$$\left|(\tilde{\boldsymbol{u}}^2)^T F \tilde{\boldsymbol{u}}^1\right| < t_e, \tag{2.12}$$

where $F \in \mathbb{R}^{3\times 3}$ is the fundamental matrix for the stereo camera calibration and $t_e \ll 1$ is a value that depends on calibration accuracy. Only measurement pairs from a true target should satisfy the constraint (2.12), noise and clutter should not.

2.5.2 Matching Measurements and Targets

Our choice of a data-association strategy is based on speed, variability, and density of targets in the image. A nearest-neighbor association is target-based and associates a target to the closest measurement based on Euclidean distance. Typically the distance is computed between the predicted measurement from the target and the actual measurement. It works well in low-target densities with high frame rates [33, 34], but results in duplicate tracks and incorrect associations at high target densities. A global nearest-neighbor (GNN) association avoids duplicate assignments by minimizing a global assignment of all targets to measurements with the constraint that a target be associated to only one measurement [86]. GNN has been successful in tracking dense aggregations [31, 87] in which the number of targets are fixed and move in two dimensions (so that target overlap is rare), however, the possibility of a variable number of targets and frequent occlusions make it difficult to use GNN without additional heuristics.

For tracking systems that track more than just position of the target, a nearest-neighbor association may be improved by defining the distance function on shapes. This makes a nearest-neighbor association reliable, even when the targets are close to one another. Therefore, for tracking three-dimensional fish shapes we define the shortest distance between predicted midline and silhouette as the distance function to associate targets to measurements.

Among data association methods that make target-measurement assignments probabilistically are the target-based joint probabilistic data association (JPDA) [51] method and the measurement-based multiple-hypothesis tracker. The JPDA algorithm assigns probability values to measurement-target associations based on current measurements and state estimates. These values are then used to assign a weight to each association. The final update to a target estimate during a time-step is a weighted sum of all possible measurement updates. At any time-step k, the set of all valid target-measurement associations, $\boldsymbol{\theta}$, is generated based on a gating volume. A feasible event $\theta \in \boldsymbol{\theta}$ is created such that (a) each measurement has only one source and (b) each target (excluding clutter) produces exactly one measurement or no measurements at all. The joint measurement-target association probability β_{tm} between measurement m and target t is [51]

$$\beta_{tm} = \sum_{\theta \in \boldsymbol{\theta}} P(\theta | \boldsymbol{Z}^k), \tag{2.13}$$

where $\boldsymbol{Z}^k$ is again the set of all measurements up to time k. $P(\theta|\boldsymbol{Z}^k)$ as per Bayes theorem is the product of the measurement-association likelihood function $P(Z[k]|\theta, \boldsymbol{Z}^{k-1})$ and the association prior $P(\theta)$. All unassigned measurements are assumed to be uniformly distributed across the entire observation region. The probability of each association in a feasible event is computed using the measurement likelihood function. Like GNN, JPDA also assumes fixed number of targets [51]. JPDA has been previously used to track multiple fish [27] in 2D. We use a particle filtering version of JPDA to track multiple fish in 3D in an underwater 1135 liters (300 gallon) tank filmed at 30 frames per second [38].

The multiple hypothesis tracker (MHT) looks into future assignment probabilities before making a decision on the current assignment [82]. A hypothesis in MHT is a combination of measurement-target assignments that satisfy the following two rules [82]: (i) a target is not associated to more than one measurement and (i) a target is only associated to a measurement that lies within its gating volume. A measurement may be assigned to an existing target, a new target, or a false alarm. As time progresses each hypothesis gives rise to successive hypotheses resulting in an exponential growth in time. Hypotheses reduction strategies include applying a threshold on track probability, choosing a few best hypothesis [88], and clustering the targets [82]. We describe salient features of the multiple hypothesis tracker [82]. For a detailed description of this algorithm, see [89, 82, 88].

Hypothesis generation: Based on the gating threshold, a gating volume is generated for each target. A validation matrix $V \in \mathbb{R}^{N_m[k] \times (N_t[k]+1)}$ is then constructed, where $N_m[k]$ is the number of measurement pairs and $N_t[k]$ is the number of targets at k. For each measurement m and target t Set $V(t, m+1) = 1$ if the measurement lies within the gating volume of target i. For example, a validation matrix with 3 targets and 3 measurements may look like (see Fig. 2.5)

$$V = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

The first column is all 1 which implies that a measurement can always be generated from a false target. The hypotheses are generated based on the rules that a target is not associated with more than one measurement. Fig. 2.4 shows two out of thirty three possible hypotheses associating the measurements to targets based on validation matrix (2.14). The number of possible hypotheses grow with the number of measurements and targets raising the need to divide the measurement-target groups into independent sets called clusters. Each cluster then generates its own hypotheses based on associated measurements that fall within the gating volume of any target within that cluster.

Clustering: We use the following strategies to cluster targets and measurements [82]: (1) a measurement that does not lie within the gating volume of an existing cluster is assigned its own cluster; (2) two clusters with a common measurement are combined to form one cluster; and (3) a target that is assigned to a single measurement in the past hypotheses is split to form its own cluster. A validation matrix is created for clusters and measurements based on the gating volume. For clusters that are combined, the resulting hypotheses are a product of the number

2	**1**	**3**
0	1	3

3	**1**	**2**
2	1	3
0	1	2
5	1	2
3	1	0
3	1	4

6	2	3	0	1
0	2	3	8	1
0	2	3	0	1
6	2	3	8	1
6	3	2	0	1
0	3	2	8	1
0	3	2	0	1
6	3	2	8	1
7	3	2	0	1
0	3	2	9	1

Figure 2.4: Hypotheses generation. Possible hypotheses for the validation matrix (2.14) with two time-steps into the future. Each column represents a measurement and each row is a hypothesis denoting which target is assigned to that measurement. The highlighted hypothesis is branched into child hypotheses in the next step. Target indices 1-3 are existing and index 0 denotes a false target.

of hypotheses in each cluster. The validation matrix (2.14) shows that we can form two clusters with targets and measurements (t_1, t_2, z_1, z_2) and (t_3, z_3), where t denotes a target and z denotes a measurement with their respective subscripts.

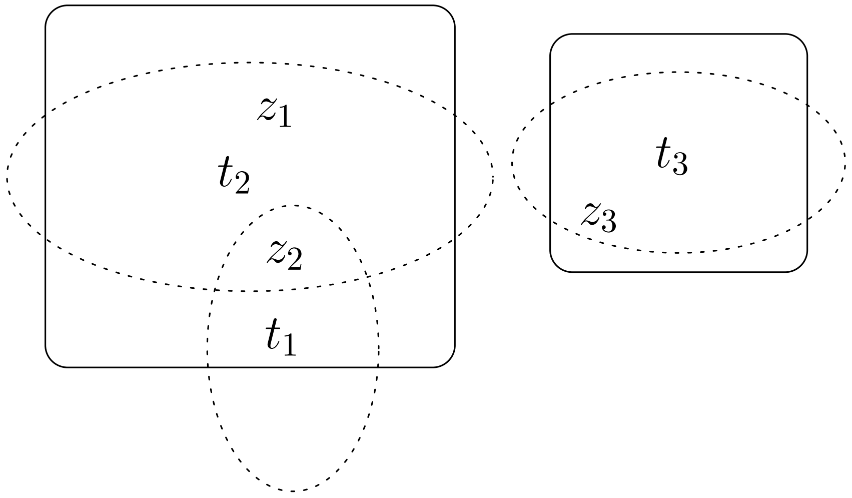


Figure 2.5: Clustering measurements and targets. Targets t_1 and t_2 have common measurements and can therefore be clustered together. Target t_3 can form its own cluster.

Probability of a hypothesis: For each hypothesis, the following values must also be computed: N_d, denoting the number of existing targets detected, N_t, the number of previously known targets, N_f, the number of false targets, and N_n, the number of new targets. The total number of measurements are $N_m[k] = N_d + N_f + N_n$ The probability of a hypothesis $\Omega_i[k]$ that arises from the parent hypothesis $\Omega_j[k-1]$ is

$$\begin{aligned} P(\Omega_i[k] \big| \boldsymbol{Z}^k) = \frac{1}{C} P_D^{N_d} (1 - P_D)^{(N_t - N_d)} \beta_f^{N_f} \beta_n^{N_n} \times \\ \prod_{m=1}^{N_d} P(\boldsymbol{Z}_m \big| \boldsymbol{X}_{(m)}) P(\Omega_j[k-1] \big| \boldsymbol{Z}^{k-1}), \end{aligned} \tag{2.15}$$

where C is the normalization constant and P_D is the probability of detection. β_f is the density of false targets and β_n is the density of new targets both of which are set to $1/V$ where V is the volume of the sensor (width $\times$ height of the image frames), and the subscript (m) denotes the target associated to a given measurement m. For associating measurements to track mosquitoes, we use the position likelihood function with the measurement $\boldsymbol{u}_m^c$, predicted measurement $f^c(\boldsymbol{r}_{(m)})$ of the associated target, and the covariance S.

$$P(\boldsymbol{Z}_m \big| \boldsymbol{X}_{(m)}) = P_{pos}(\boldsymbol{u}_m^{1,2} \big| \boldsymbol{r}_{(m)}) = \prod_{c=1,2} \mathbb{N}(\boldsymbol{u}_m^c; f^c(\boldsymbol{r}_{(m)}), S), \tag{2.16}$$

where the covariance $S = \text{cov}(f^c(\boldsymbol{r}_{(m)}))$ computed over all samples in the particle filter distribution.

Hypotheses reduction: The number of hypotheses in each step grow exponentially and must be reduced. In addition to clustering, other methods for reducing the number of hypotheses include N-scanback, and selecting few best hypotheses at each step. The N-scanback method uses the probability of current hypotheses (2.15) to resolve ambiguity at $k - N_s$ step where N_s is the scanback value. A set of hypotheses are then formed for each cluster which denote the collective measurement-target association. Finally, the hypotheses may also be reduced based on a numerical sort on probability values and selecting a few best.

In videos of mosquito swarms, the number of mosquitoes on the image vary because new mosquitoes join the swarm and swarming mosquitoes appear and disappear from the camera field of view. In addition birds, and other insects also appear in the field of view. We therefore use the multi-hypothesis tracker (MHT) as a data

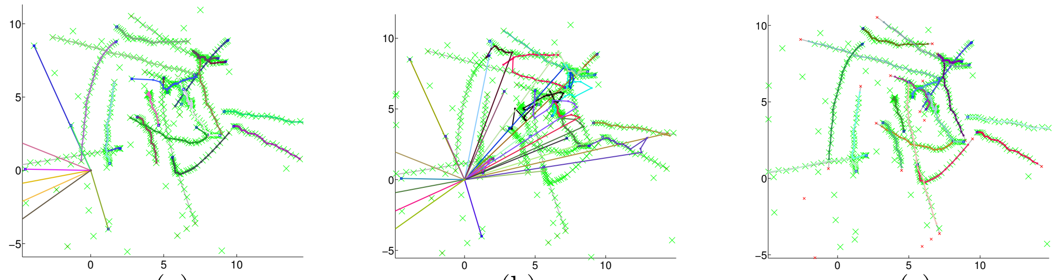


Figure 2.6: Comparison of data association techniques. A test dataset in two-dimensions was created with 20 targets moving randomly with noise equal to one-hundredth of area of measurement. The final snap shot after fifty time steps is shown. Measurements (green crosses) include those from real as well as false targets. Ground truth is shown in light grey and estimated tracks are shown in color. The gating volume for all three methods, (a) nearest neighbor, (b) global nearest neighbor and (c) MHT is set to 16. Only MHT is able to track all targets (light grey) without switching or early terminations.

association strategy. Furthermore, because MHT runs online it also permits us to use a motion model at each step to search for missing measurements.

2.5.3 Occlusions

Except the nearest-neighbor matching every data-association strategy discussed in this section assumes that a measurement is generated by a single target, and that motion coherence will automatically associate the right tracks in a future time step [90, 91, 32]. For multiple target tracking with occlusions, a nearest-neighbor matching invariably results in duplicate tracks.

We encounter sustained and frequent occlusions in our datasets both of which prove challenging for a motion-coherence based approach. For example, a single fish may successively occlude with other fish as well as self-occlude during the course of the video. The fish may turn or startle while being occluded. Similarly, in mosquito swarms, a mosquito may undergo frequent occlusions with multiple mosquitoes as it passes through the center of the swarm resulting in track switches. Strategies that address occlusions include increasing the number of views [34], model based tracking [70, 29], and track linking of pre- and post-occlusion segmented tracks [92, 93].

The simplest strategy to reduce the number of occlusions in an animal group is to have a large number of views of the tracking volume. This increases the probability of a target unoccluded in at least two camera views. Although multiple cameras can be set up in the laboratory [34], it may not be practical for filming in the wild.

For targets that appear extended on the image, model-based tracking mitigates some of the occlusion effects by enforcing geometric constraints on the occluded observations. In [70] vehicles filmed on a highway are modeled as rigid bodies. In the case of an occlusion, overlapping regions within occluded blobs are identified using shape estimates and constraints on motion of the vehicle. In [94] multiple objects are tracked through occlusions by enforcing a rigid body motion constraint on grouped features. In [71] multiple mice are tracked by modeling each mouse's contour as an ellipse that best fits a template library of B-spline contours, and in [87] occlusions between fish are resolved by modeling the occluded blob as a Gaussian mixture and finding the best fit using an Expectation minimization algorithm.

Long-duration occlusions last for more than several frames and can affect track integrity. Depending on target movement, such occlusions can be overcome by methods that minimize a global cost function over all measurements in a sliding window [95]. The implicit assumption, however, is that the probability of detection is close to one; it is unclear how a global optimization method can adaptively seek missing measurements. Such track linking approaches [92, 93] work in an offline fashion with the assumption that the probability of detection is close to one. Instead MHT, which is an online implementation (as opposed to an offline method) allows specialized methods to be incorporated for extracting missed measurements.

We use model based tracking to resolve occlusions in both fish and mosquitoes. For occluded fish we perform optimization over a larger state-space that consists of individual state of all fish in an occlusion. The silhouette of the occluded blob is used as a measurement. For mosquitoes we model occlusions has a mixture of Gaussians that are hard clustered based on predicted measurements. The clustered blobs are then used as individual measurements.

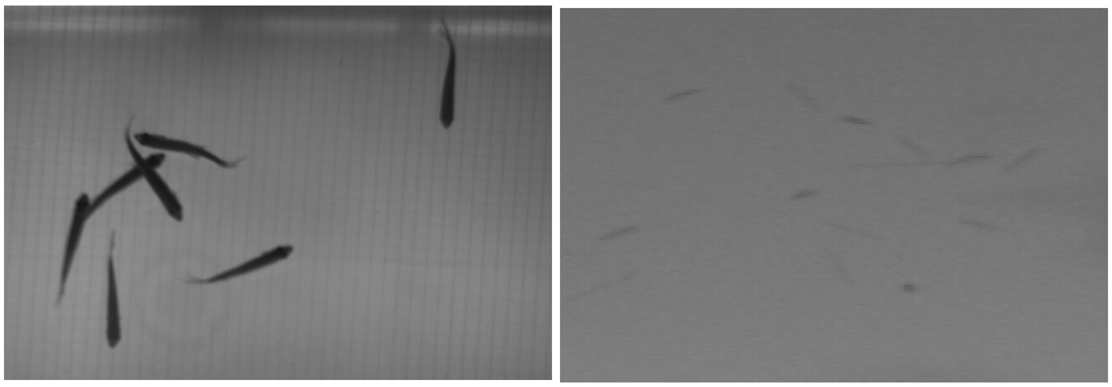


Figure 2.7: Sample occlusions in fish and mosquitoes. All data association methods assume that each measurement corresponds to a single target, which is violated in the case of occlusions.

2.6 Initialization

A target track must be initialized before the first time a measurement is received. For point targets, a high variance may be set around each measurement representing the lack of certainty in a possible track [88, 34]. However, high-dimensional states such as target orientation and shape are difficult to initialize in this way, mainly because a model must be generated to map the measurements to the state. Existing methods ascertain shape geometries manually by selecting the target contour by hand [29, 96]. In the event of a track termination, the target must again be selected by hand for the tracking to continue. To track large datasets, manual shape initialization presents a challenge that must be overcome. In human tracking, for example, shape initialization is performed by fitting an existing model to the target in view [24]. In [70], vehicles modeled as rigid objects, are tracked by running the shape estimation and tracking algorithm in parallel. We apply the same general framework for tracking non-rigid fish shapes. We parameterize fish shape surface using a generative model and estimate the model parameters within an extended Kalman filter that runs in parallel to the shape reconstruction algorithm.

In tracking swarming mosquitoes, due to appearance of individual mosquito as faded streak, we are faced with a low signal-to-noise environment. One approach is to run track-before-detect using raw sensor data [97]. The success for track-before-detect relies on the low target density and relatively straight movement of targets in the measurement space [89]. Furthermore, using raw sensor data is impractical with multiple sensors because they generate more false targets than observed in a single noisy image. Setting a high variance on each detected measurement as before [88] introduces large ambiguities regarding measurement origin in high target densities and additional techniques must be used to reduce the same [83]. The multiple hypothesis tracker uses a similar approach by deferring the decision of a confirmed track until the probability is above a certain threshold [82]. We use the epipolar constraint and velocity likelihood function to reduce the uncertainty in our initial estimates.

2.7 Summary of Design Choices

Each component of a multi-target tracking system requires a design choice for a starting point. The design choice is typically made on the basis of target appearance and movement in the video sequence. Fig. 2.8 lists the different components of the tracking system along with the criteria that were used to pick a particular approach.

In the design of fish tracking system, we picked generative modeling around a polynomial curve representing the midline to approximate the fish body shape. We assume that the fish bend laterally only and therefore a planar midline was sufficient. Since we track in three-dimensions, the measurement model is nonlinear, and therefore we resort to nonlinear estimation methods. Furthermore, our measurement model is multimodal since there are multiple ways to obtain the same approximate midline shape. We pick optimization over particle filtering due to (a) large state space of $n = 11$ per fish that increases with occlusions, and (b) small time difference between successive measurements; tracking therefore primarily entails processing the measurements, and does not require an accurate motion model. Because the number of targets are fixed, and occlusions are handled as a joint state estimation problem, we use nearest-neighbor data association. We evaluate target-measurement matches by measuring midline distance, which makes the process reliable. We implement an

State Space
Target rigidity
Target extent
Position
Pose
Shape

Estimation
Measurement Model
State-space size
Time-step length
Kalman filter
Extended Kalman filter
Unscented Kalman filter
Particle filter
Optimization

Data association
Target variability
Target density
Nearest neighbor
Global nearest neighbor
JPDA
MHT

Motion model
Interaction
Maneuverability
Constant velocity
Constant acceleration
Learned
MRF

Initialization
Target density
State-space representation
Manual
Large variance
Automatic

Multi-view arrangement
Tracking volume
Target extent on image
Target density
Stereo parallel
Stereo orthogonal
Camera and mirror
Multiple cameras

Figure 2.8: Design choice. Factors such as target density, its extent on the image, and the size of the tracking volume among other factors determine which data association method, estimation tool and experimental setup must be used. Similar looking targets and absence of distinguishable features preclude the use of appearance based models.

turning motion model to efficiently generate candidate solutions within the optimization routine. To initialize fish shapes we run an extended Kalman filter that uses automatic head, nose, and tail detection. Finally, to obtain accurate shape reconstruction, and utilize the flexibility of a laboratory environment we place our cameras orthogonally with one camera overhead.

In the design of the mosquito tracking system we use a stereo camera setup that is calibrated onsite. Tracking challenges in the form of missing measurements, variable number of targets, and elongated streak-like appearance on image motivate our choice of particle filtering as the estimation method and multiple hypothesis tracking for data association. Position is initialized with a low variance using triangulation from the matched pair of measurements and uncertainty in velocity is reduced by using a streak endpoint likelihood function.

Chapter 3

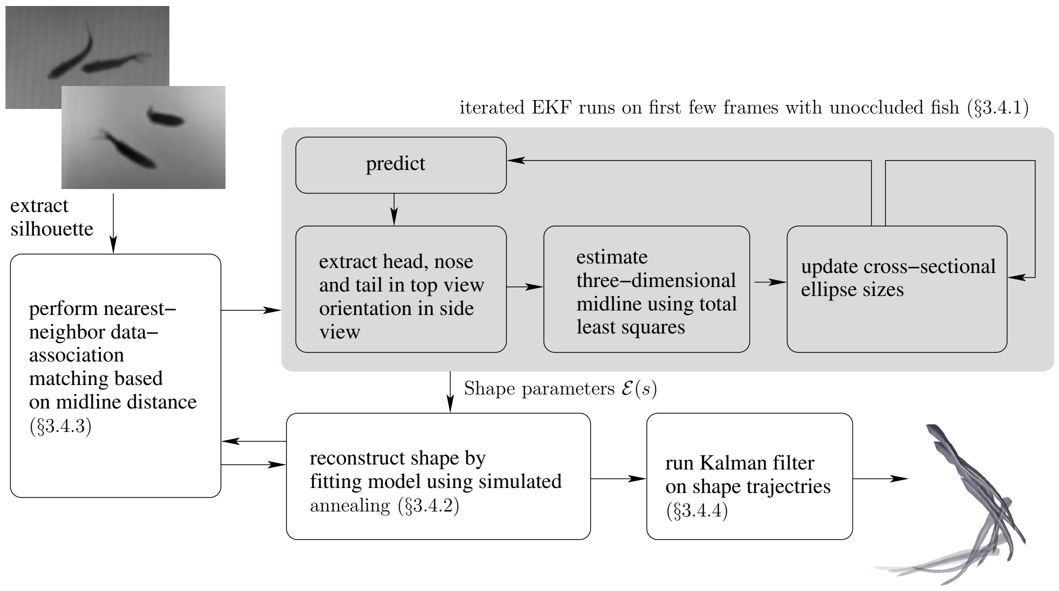
Reconstructing Swimming Kinematics of Schooling Fish

In studying fish schools we are specifically interested in the rapid transmission of information via a nonverbal cue such as a fright response. An example of a fright response is a fast start, which is often the precursor to an escape or attack [98]. Two behaviors associated with fast-start swimming are C-starts and S-starts [99], named for the corresponding body shape during the maneuvers, which take place in less than 100 ms. The propagation of startle responses in a fish school may be indicative of the social transmission of information [11]. By quantifying the shape kinematics of each fish, properties of the school such as the first responder and its position relative to the cue, and the spatio-temporal distribution of successive responders may be extracted. Such information can reveal deeper insights into the decision making process within fish schools.

Therefore, the requirements for the multi-target tracking system described in this chapter were to track deformable targets that exhibit sudden movements and often occlude each other in a given view. The fish tracking system enables automatic shape reconstruction of individual fish in a school using calibrated multi-view video with at least one overhead view. We begin in Section 3.1 with an overview of the tracking framework. In Section 3.2, we describe the fish state comprising position, heading, and locally defined shape. In Section 3.3 we present methods to extract head, nose, and midline from fish silhouette after background subtraction. These features are used in estimating shape geometry. Section 3.4 details parts of the tracking algorithm including shape estimation, shape reconstruction, data association, and filtering. Section 3.5 describes data collection procedure for filming multiple fish in the laboratory. In Section 3.6 we evaluate the performance of tracking algorithm including verification using an independent view. We validate the tracking system on schools of up to eight fish. In Section 3.7, we compare shape kinematics of a fast-start with regular swimming behavior.

3.1 Tracking Framework

The input to the fish tracking system is a sequence of synchronized images from a calibrated multi-view setup in the laboratory. The tracking system consists of the tracking algorithm which uses silhouettes as inputs to reconstruct fish shape and a Kalman filter that smoothes the shape trajectories. The tracking algorithm developed in MATLAB® is run post-filming and begins by locating the frame that has no occlusions in each view called the start frame. (We count the difference between the number of blobs and the expected number of fish in each view. In case no such frame is available we prompt the user to mark the outline of occluded fish manually in the frame with least number of occlusions.) We use the epipolar constraint on blobs in each view to initialize the target positions. The tracker first runs forward, then backwards in time from the start frame; shape estimation, which estimates fish shape geometry, and shape reconstruction, which tracks individual fish shape are run in parallel. For data association, we compute silhouette-target proximity in each view using a midline distance function. Shape estimation runs until an error bound on the estimate is attained. Shape reconstruction continues to use occluding contours of blob silhouettes to project lines into three-dimensional space. Using the combined distance of the fish shape surface from all such lines, the fish position, orientation, and shape is modified until a best fit is obtained. We find the best fit using simulated annealing.



iterated EKF runs on first few frames with unoccluded fish (§3.4.1)
extract silhouette
predict
extract head, nose and tail in top view orientation in side view
estimate three−dimensional midline using total least squares
update cross−sectional ellipse sizes
perform nearest−neighbor data−association matching based on midline distance (§3.4.3)
Shape parameters $\mathcal{E}(s)$
reconstruct shape by fitting model using simulated annealing (§3.4.2)
run Kalman filter on shape trajectries (§3.4.4)

Figure 3.1: Fish Tracking Framework. Generative modeling is used to parameterize a shape model; these parameters are estimated using an iterated EKF. Shape reconstruction is performed by matching measurements from segmented images in multiple cameras to a three-dimensional shape estimate.

3.2 State Space Representation

The fish state consists of three-dimensional position $\boldsymbol{r} \in \mathbb{R}^3$, heading $\boldsymbol{h} \in \mathbb{R}^3$, and shape parameters $\boldsymbol{p} \in \mathbb{R}^p$. In [38], we propose approximating fish shape by a bendable ellipsoid ($p = 1$). We are able to track simple motion using this method, but not C- or S-starts, which motivates the approach described here ($p = 5$).

The shape geometry is modeled about the midline of the fish. There are several ways to generate the midline. In [28], the midline is found by projecting the top-view profile on a plane of orientation. In [79, 29], the midline is generated manually. The midline in our tracking system is generated automatically when the fish is in clear view of all cameras, i.e., when there are no occlusions and both head and tail are visible. The shape geometry is automatically estimated about the midline using an iterated extended Kalman Filter (EKF).

For the purpose of shape estimation and tracking we make the following assumptions about fish motion observed in our experiments:

A1) The fish in our tracking experiments bend laterally [28].

A2) The fish in our tracking experiments turn and pitch, but rarely roll.

A3) The portion of the body from the eyes to the nose (the head) does not bend.

A single fish is characterized by the position of the head, the orientation of the head (the heading vector), and the midline. The midline is a curve that runs from the head to the tail. A surface is generated around the midline to approximate the shape. We define the shape locally using a body-fixed reference frame $\mathcal{B}$. The origin of frame $\mathcal{B}$ is the center of the head with one axis in the direction of the nose. The heading $\boldsymbol{h} \in \mathbb{R}^3$ is a unit vector pointing from the center of the head to the tip of the nose (see Fig. 3.2). Based on assumption (A2), the body-frame axes are completed by performing the cross product of the vertical $\boldsymbol{g} \triangleq \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ with the heading $\boldsymbol{h}$ to get the pitching axis, followed by the cross product between the heading and the pitching axis to get the yaw axis. [1] Given the position of the head $\boldsymbol{r} \in \mathbb{R}^3$, the complete body frame in the world-frame coordinates can be represented by the transformation

$$^{\mathcal{W}}T_{\mathcal{B}} = \begin{bmatrix} \boldsymbol{h} & \boldsymbol{g} \times \boldsymbol{h} & \boldsymbol{h} \times (\boldsymbol{g} \times \boldsymbol{h}) & \boldsymbol{r} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.1}$$

The midline is parameterized in the body frame by $\boldsymbol{f}(s) = \begin{bmatrix} f_1(s) & f_2(s) & f_3(s) \end{bmatrix}^T$, where $s \in [0, 1]$. We assume the functions $f_i(s)$ are differentiable, which permits us to define an orthonormal frame at each point s on the midline. We use this frame to define the body cross section at s. We use the following criteria to determine the type of parameterization for $\boldsymbol{f}(s)$

i. To model an S-start the curve parameterization must permit at least two inflection points, which implies that one of $f_1(s)$ or $f_2(s)$ should be a quartic

[1] Note that since the cross product covers the smaller angle between two vectors, the yaw axis will change sign when the fish nose dives, which rarely happens in our experiments.

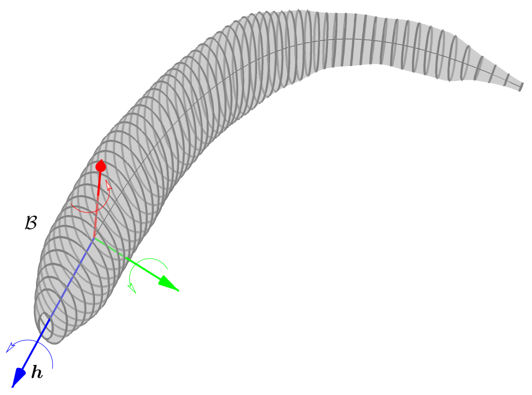


Figure 3.2: Fish body frame. The body frame $\mathcal{B}$ is fixed to the head with the heading vector $\boldsymbol{h}$ pointing towards the tip of the nose. The pitch (green), roll (blue), and yaw (red) axes complete the frame.

polynomial. [2]

ii. To model a C-shape, the curve parameterization must permit two values of $f_2(s)$ for a value of $f_1(s)$. Therefore if $f_2(s)$ is a quartic, $f_1(s)$ should at least be a quadratic in s.

iii. The curve at $s = 0$ must be continuous with the heading. Therefore the slope $\frac{\partial f_2}{\partial f_1}\big|_{s=0} = 0$. We therefore set the coefficient of s in f_2 to zero.

Using the curve selection criteria and assumption A1 we model the fish midline as

$$\begin{aligned} f_1(s) &= p_1 s + p_2 s^2 \\ f_2(s) &= p_3 s^2 + p_4 s^3 + p_5 s^4 \\ f_3(s) &= 0, \end{aligned} \tag{3.2}$$

where $\boldsymbol{p} = \left[p_1, \ldots, p_5\right]^T$ are the polynomial coefficients. The midline generated using polynomial representation (3.2) is extensible. Inextensible curves can be created by parameterizing the curve as a function of the bending angle in two [100, 29] and

[2]An inflection point occurs where the curvature of the curve changes sign. The sign of the curvature is determined by the second derivative $\partial f_2^2(s)/\partial f_1^2(s)$. Therefore $f_2(s)$ should be a fourth degree polynomial, so that the second derivative has two roots. We may choose any of f_1 or f_2 to be fourth degree depending on the framing we choose.

three dimensions [101]. For example, a two dimensional inextensible curve should satisfy $\partial^{\mathcal{W}} f_1(s)/\partial s = \cos(\theta(s)); \partial^{\mathcal{W}} f_2(s)/\partial s = \sin(\theta(s))$, where $\theta(s)$ is the bending angle, and ${}^{\mathcal{W}} f_1(s), {}^{\mathcal{W}} f_2(s)$ are in the world frame. An inextensible curve, however, lends no flexibility to measurement noise such as variation in length due to inconsistent appearance of caudal fins. To ensure inextensibility while permitting minor changes in length we instead place a constraint on total length within the cost function [39]. In this dissertation, we move the constraint to the perturb function that generates a candidate solution thereby making the optimization process faster in time.

3.3 Feature Extraction

The input to the shape-reconstruction algorithm is the silhouette of the blob and the shape estimates. The input to the shape-estimation algorithm is the three-dimensional position of head, nose, and tail of the fish, which we extract from blobs in the top view and side view images.

We extract blobs using the *regionprops* routine in MATLAB, which performs connected-component labeling to extract features such as centroid, area, silhouette, and occluding contour. A second routine called *bwboundaries*, which implements a contour tracing algorithm to arrange the pixels is used to determine curvature and edge pixels. Finally, the distance transform routine *bwdist* calculates the distance of each foreground pixel from the nearest background pixel. To automatically detect head, nose, and tail for generating the midline, we make the following observations (see Fig. 3.3):

(a) the center of the head is the center of the largest circle that fits inside the silhouette. If more than one pixels satisfy this condition, then the one that is farthest from the centroid is selected.

(b) the nose is modeled in the top view by fitting a half ellipse centered at the head. The minor axis of the ellipse is same as the radius of the largest circle

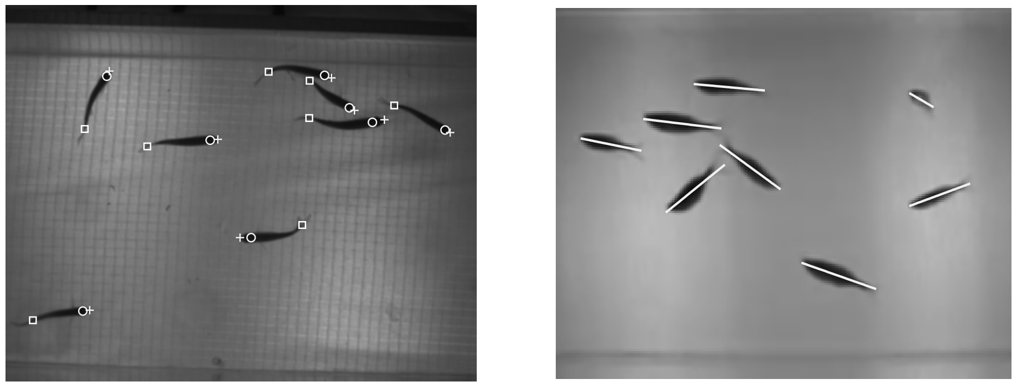


Figure 3.3: Extracting fish head, nose, and tail automatically. The head, nose, and tail of unoccluded fish are extracted to create a midline

.

at the head, and the eccentricity is 0.74, which is determined empirically. The best fit in the sense of orthogonal distance is found by an exhaustive search on orientation angles with increasing resolution from 90°–1°.

(c) the tail marks the point of highest curvature on the occluding contour. Curvature, defined in (3.19), represents the degree of bending.

The location of the nose expressed in pixels in camera 1 is denoted by ${}^1\boldsymbol{u}_n$, the tail by ${}^1\boldsymbol{u}_t$, and the center of the head by ${}^1\boldsymbol{u}_h$. The side views (cameras 2 and 3) give orientation information as well as position information. Let ${}^cl \triangleq ({}^cl_m, {}^cl_r)$ be a line in camera c, where ${}^cl_m$ is the slope and ${}^cl_r$ is the intercept with the vertical axis of the image plane. A least-squares fit on the silhouette in camera c establishes a line along the body. The body frame is oriented so that the heading is aligned with this line in the side view and with the vector from the head to the nose in the top view. Let the camera projection matrix (Appendix B) be denoted by ${}^cP = \begin{bmatrix} {}^cP_1^T & {}^cP_2^T & {}^cP_3^T \end{bmatrix}^T$ with each row ${}^cP_i, i = 1, 2, 3$. For each point detected automatically in the top view we may now use the following equations to find the three-dimensional position (in a least-squares sense) of a point

$$\begin{aligned}{}^{1}P_{1,1-3}\hat{\boldsymbol{r}} &= {}^{1}P_{1,4} - {}^{1}\boldsymbol{u}_1\\ {}^{1}P_{2,1-3}\hat{\boldsymbol{r}} &= {}^{1}P_{2,4} - {}^{1}\boldsymbol{u}_2\\ {}^{2}P_{2,1-3}\hat{\boldsymbol{r}} &= {}^{2}l_m({}^{2}P_{1,1-3}\hat{\boldsymbol{r}} + {}^{2}P_{1,4}) + {}^{2}l_r\end{aligned} \tag{3.3}$$

where $c \in \{2, 3\}$. The above three equations can be solved in either one of the side cameras for the position of the head $\boldsymbol{m}(0) \in \mathbb{R}^3$ and nose.

3.4 Tracking Algorithm

In this section we begin with estimating the midline in the initial frames and then using that estimate to build a parameterized generative model around that midline. The parameters of the generative model are then used in the shape-fitting cost function in subsequent frames.

3.4.1 Estimating Shape Geometry

The midline is represented in world-frame coordinates using transformation ${}^{\mathcal{W}}T_{\mathcal{B}}$, i.e,

$$\begin{bmatrix}\boldsymbol{m}(s)\\ 1\end{bmatrix} = {}^{\mathcal{W}}T_{\mathcal{B}}\begin{bmatrix}\boldsymbol{f}(s)\\ 1\end{bmatrix}. \tag{3.4}$$

We model the fish cross section at point s on the midline by an ellipse $\mathcal{E}(s)$ in the plane that is normal to the midline at s, and compute the ellipse planes at each point using curve framing [77]. The tangent $\boldsymbol{t}(s)$ to the midline at s forms an axis of a local orthogonal frame $\begin{bmatrix}\boldsymbol{x}(s) & \boldsymbol{y}(s) & \boldsymbol{t}(s)\end{bmatrix}$. The local frame at each point on the midline is completed as follows: the normal axis $\boldsymbol{x}(s)$ is $\boldsymbol{x}(s) = \boldsymbol{g} \times \boldsymbol{t}(s)$ and the binormal is $\boldsymbol{y}(s) = \boldsymbol{t}(s) \times \boldsymbol{x}(s)$ (see Fig. 3.4). A point on the cross section $\mathcal{E}(s)$ can be represented in the world frame $\mathcal{W}$ using the transformation matrix

$${}^{\mathcal{W}}T_{\mathcal{E}} = \begin{bmatrix}\boldsymbol{x}(s) & \boldsymbol{y}(s) & \boldsymbol{t}(s) & \boldsymbol{m}(s)\\ 0 & 0 & 0 & 1\end{bmatrix}, \tag{3.5}$$

where

$$\boldsymbol{t}(s) = \begin{bmatrix} \frac{\partial m_1}{\partial s} & \frac{\partial m_2}{\partial s} & \frac{\partial m_3}{\partial s} \end{bmatrix}^T.$$

The distance of a point i on the silhouette ${}^c\boldsymbol{Z}^i \in \mathbb{R}^2$ from any point on the projected curve ${}^c\hat{\boldsymbol{m}}(s) \in \mathbb{R}^2$ is given by $\|{}^c\boldsymbol{Z}^i - {}^1\hat{\boldsymbol{m}}(s)\|$. The midline $\boldsymbol{m}(s)$ is projected onto the image by perspective projection, which uses the camera calibration parameters [102]. The projected midline ${}^c\hat{\boldsymbol{m}}(s)$ on camera c is [85]

$${}^c\hat{\boldsymbol{m}}(s) = \begin{bmatrix} \frac{{}^cw_1}{{}^cw_3} & \frac{{}^cw_2}{{}^cw_3} \end{bmatrix}^T,$$

where ${}^c\boldsymbol{w}(s) = {}^cP\boldsymbol{m}(s)$.

We complete the body frame by applying the no-roll assumption (A2).

The estimated midline parameters $\hat{\boldsymbol{p}}$ are found using a nonlinear cost function that measures the distance of the silhouette to the midline. Let q_i^* be the distance of the point ${}^1\boldsymbol{Z}^i$ in the top-view silhouette to the closest point on the projected midline ${}^1\hat{\boldsymbol{m}}(s)$. The midline parameters $\hat{\boldsymbol{p}}$ are estimated by solving

$$\begin{aligned} &\hat{\boldsymbol{p}} = \underset{\boldsymbol{p}}{\operatorname{argmin}} \sum_i q_i^*, \text{ where} \\ &q_i^* = \min_s \|{}^1\boldsymbol{Z}^i - {}^1\hat{\boldsymbol{m}}(s)\| \text{ subject to} \\ &{}^1\hat{\boldsymbol{m}}(1) = {}^1\boldsymbol{u}_t. \end{aligned} \tag{3.6}$$

We minimize (3.6) it by applying a two-stage optimization process consisting of simulated annealing followed by a gradient based search using the MATLAB function *fmincon* [103]. The output of the simulated annealing algorithm serves as the input to the gradient based optimization search with the constraint described in (3.6). Once a midline is estimated, a surface is generated around it to create a shape model as described next.

To generate the body surface, we need the major axis $a(s)$ and the minor axis $b(s)$ of each elliptical cross section, and an offset $d(s)$ along the normal $\boldsymbol{y}(s)$. Using candidate values for $a(s), b(s)$, $d(s)$, and the transformation matrix above, we scale and transform the cross section $\gamma(v) = \begin{bmatrix} \cos(v) & \sin(v) & 0 \end{bmatrix}^T$, where $v \in [0, 2\pi]$ [76, 35]. The transformation is defined as (see (2.10))

$$\delta(\gamma, s) = M(s)\gamma + \boldsymbol{T}(s), \tag{3.7}$$

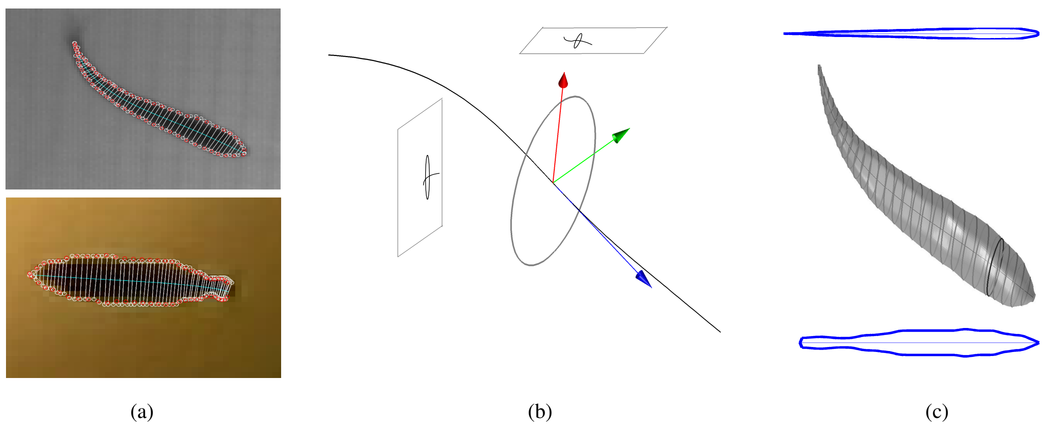


Figure 3.4: Estimating fish shape geometry. To estimate fish shape geometry we (a) fit a midline in the top view and side view. The white circles on the occluding contours are the measurements and estimates are the red circles are the projected estimates of the end points of the ellipse axes. (b) assign estimated major and minor axis values to a cross-sectional ellipse normal to the midline. (c) generate the final surface using the top profile and side profile. (The black ellipse partitions the head and rest of the body.)

where

$$\begin{aligned} M(s) &= \begin{bmatrix} \boldsymbol{x}(s)a(s) & \boldsymbol{y}(s)b(s) & \mathbf{0}_{3\times 1} \end{bmatrix} \\ \boldsymbol{T}(s) &= \boldsymbol{m}(s) + \boldsymbol{y}(s)d(s). \end{aligned} \tag{3.8}$$

The curve $\boldsymbol{m}(s)$ is formed using (3.4). Substituting (3.8) into (3.7), we obtain the surface

$$\begin{aligned} \mathcal{S}(s,v) \triangleq \delta(\gamma(v), s) = \boldsymbol{m}(s) + a(s)\cos(v)\boldsymbol{x}(s) + \\ (b(s)\sin(v) + d(s))\boldsymbol{y}(s), \end{aligned} \tag{3.9}$$

where $s \in [0,1]$ and $v \in [0, 2\pi]$.

To estimate the values $\mathcal{E}(s) = \begin{bmatrix} a(s) & b(s) & d(s) \end{bmatrix}^T$ using the top-view and side-view observations, we measure the length of the line segment contained in the occluding contour and normal to the midline (see Fig. 3.4). For the top view, we substitute ${}^1v = \{0, \pi\}$ in equation (3.9) to produce the end points of the major axis, $a(s)$; ${}^2v = \{\pi/2, 3\pi/2\}$ for an orthogonal side-view produces the values for the minor axis, $b(s)$, and $d(s)$. A perspective projection of a surface point $\mathcal{S}(s,v)$ on camera c is denoted by ${}^cS(s, {}^cv)$.[3] The measurements ${}^c\boldsymbol{E}(s) = \begin{bmatrix} e_1(s) & e_2(s) \end{bmatrix}^T$ are related to

[3]For a side view camera oriented at an angle ϕ with the vertical the value of $v = \{\phi, \phi + \pi\}$

$a(s), b(s), d(s)$ in the respective camera views by the nonlinear measurement model [4]

$$\begin{aligned} {}^c\boldsymbol{E}_{\mathcal{E}}(s) &= {}^cH_{\mathcal{E}}(\mathcal{E}(s)) + {}^c\boldsymbol{n}_e \\ &= \begin{pmatrix} \frac{\|{}^c\mathcal{S}(s,{}^cv_1) - {}^c\mathcal{S}(s,{}^cv_2)\|}{2} \\ \left\| \frac{{}^c\mathcal{S}(s,{}^cv_1) + {}^c\mathcal{S}(s,{}^cv_2)}{2} - {}^c\hat{\boldsymbol{u}}(s) \right\| \end{pmatrix} + {}^c\boldsymbol{n}_e \end{aligned} \tag{3.10}$$

where $\|\cdot\|$ is the 2-norm and $\boldsymbol{n}_e \in \mathbb{R}^2$ is the measurement noise. The side-view camera had greater noise due to lower resolution and straight line assumption A1 of the midline, which although compensated by the use of the offset $d(s)$, causes measurement errors near the nose and tail. The nonlinear measurement model (3.10) is used on multiple views to estimate the size of elliptical cross sections using a gradient based optimization method. Since the measurements depend on fish position and orientation, we run the shape estimation algorithm for multiple frames in an iterated EKF to obtain an accurate estimate of the shape geometry. (See Appendix A for linearization of the measurement model.)

As a fish moves, the change in thickness of a its cross section is in sub-millimeters [104]. We approximate the change in the thickness of a fish cross section as a constant value Gaussian disturbance model

$$d\mathcal{E}(s) = d\boldsymbol{w}_{\mathcal{E}} \tag{3.11}$$

where $\dot{\boldsymbol{w}}_{\mathcal{E}} \in \mathbb{R}^3$ is Gaussian white noise process.

The EKF is initialized on the start frame with unoccluded fish. Across-view data association for the start frame is solved using the Hungarian method [86] on the value of epipolar constraint (2.12) for blob centroid pairs. The EKF runs in parallel with the shape reconstruction algorithm until the error norm of the state covariance matrix $P_{\mathcal{E}}$ is less than a threshold $t_{\mathcal{E}}$. The EKF update is run at each step by linearizing the measurement model about the current estimate. (A single update

[4]Note that the above measurement model assumes that the occluding contour of a fish is a projection of the extreme ends of the elliptical cross sections. Since the camera distance (1 m) is much larger than the fish cross section (2.5 cm), this assumption introduces sub-pixel measurement error.

of the EKF is equivalent to a single step of a Gauss-Newton optimization method [105].) The EKF given in Table 3.1 iterates at each frame until the improvement in successive iterations is less than a threshold $t_{\mathcal{E}_k}$.

Table 3.1: **EKF Algorithm for Shape Estimation**

Input: Motion model (3.11), measurement model $H_{\mathcal{E}}$ (3.10), covariance matrices for measurement noise R_E and disturbance $Q_{\mathcal{E}}$

Initialize: State estimate $\mathcal{E}[0]^-$, and covariance matrix $P_{\mathcal{E}}[0]^-$, prior to the first measurement (we drop the s dependence for clarity)

For each time step $k = 1, 2, \dots$

```
i. Iterate until $\|P_{\mathcal{E}} - P_{\mathcal{E}}^-\| >= t_{\mathcal{E}_k}$
    1: Compute gain matrix: $W = P_{\mathcal{E}}^-[k]H^T S^{-1}$, where $S = HP_{\mathcal{E}}[k]^- H + R_E$ is the measurement prediction covariance, and $H = \frac{\partial H_{\mathcal{E}}}{\partial \mathcal{E}}(\hat{\mathcal{E}}[k]^-)$
    2: Update state estimate: $\hat{\mathcal{E}}[k] = \hat{\mathcal{E}}[k]^- + W(\boldsymbol{E}[k] - H(\hat{\mathcal{E}}[k]^-, \boldsymbol{n}))$
    3: Update state covariance: $P_{\mathcal{E}}[k] = (1 - WH)P_{\mathcal{E}}[k]^-$
    4: Reassign state and covariance: $\hat{\mathcal{E}}[k]^- = \hat{\mathcal{E}}[k]$, $P_{\mathcal{E}}[k]^- = P_{\mathcal{E}}[k]$.
ii. Predict state prior to next measurement: $\hat{\mathcal{E}}[k+1]^- = F(\mathcal{E}\hat{[k]}, \boldsymbol{w})$
iii. Compute covariance: $P_{\mathcal{E}}[k+1]^- = FP_{\mathcal{E}}[k]F^T + Q_{\mathcal{E}}$, where $F = \frac{\partial F}{\partial \mathcal{E}}(\hat{\mathcal{E}}[k])$
```

3.4.2 Fitting Shape to Measurements

Once a model is generated we produce a three-dimensional line from each point on the occluding contour O. The distance of each line to the model surface $\mathcal{S}$ is used to optimize the state estimate [106]. We represent a line L in three dimensions by Plücker coordinates [106]. The advantage of this representation is that it defines a line uniquely and its distance to a point is a straightforward operation. Let $L \triangleq \begin{bmatrix} \boldsymbol{l}_v^T & \boldsymbol{l}_m^T \end{bmatrix}^T$, where $\boldsymbol{l}_v \in \mathbb{R}^3$ is the unit vector representing the direction of the line and $\boldsymbol{l}_m = \boldsymbol{l}_r \times \boldsymbol{l}_v$ is the moment of any point $\boldsymbol{l}_r \in \mathbb{R}^3$ on the line. The distance

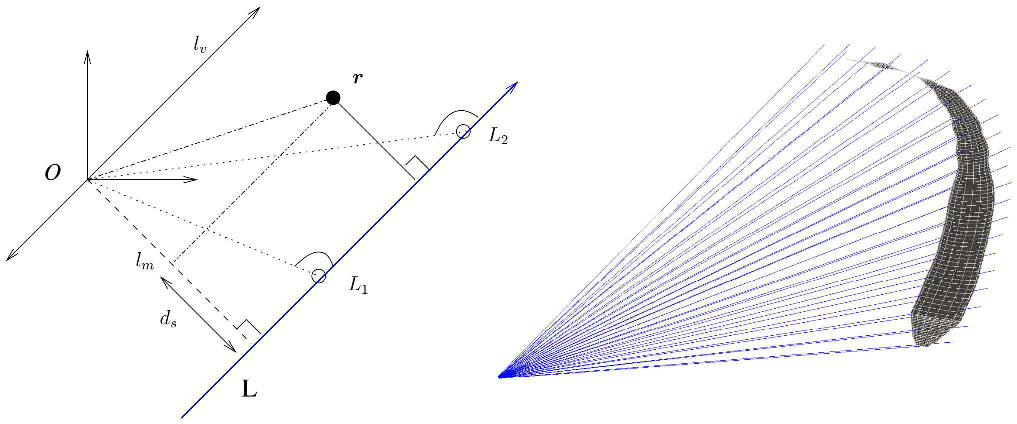


Figure 3.5: Shape fitting cost function. Any point on the line L (a), $L_1, L_2, ...$, represented in a frame O will have $d_s = 0$; here l_m is the projection of the position vector (w.r.t O) of any point on the line. The distance of a point s from the line L is simply the projection of $\boldsymbol{r}$ minus l_m. The shape fitting cost function (b) is the sum of the distances of each point on the surface from the lines projected out from the each view.

of point $\boldsymbol{r}$ from the line L is given by $\|\boldsymbol{r} \times \boldsymbol{l}_v - \boldsymbol{l}_m\|$. The cost function is a measure of the total distance of a surface from an occluding contour. We denote a point on the surface $\mathcal{S}$ by $\mathcal{S}_i$. The state estimate $\hat{\boldsymbol{X}}_t$ is obtained by solving

$$\begin{aligned} \hat{\boldsymbol{X}}_t &= \underset{\boldsymbol{X}_t}{\operatorname{argmin}} \sum_{c \in (1,2)} \sum_{o \in {}^cO} {}^cD_o^* \text{ where} \\ {}^cD_o^* &= \min_{i \in \mathcal{S}} \|\mathcal{S}_i(\boldsymbol{X}_t) \times {}^c\boldsymbol{l}_{v,o} - {}^c\boldsymbol{l}_{m,o}\|. \end{aligned} \tag{3.12}$$

In [39] we added a non-decreasing function of Δl, the difference in length between the midline as computed from shape-estimation and from the candidate state $\boldsymbol{X}_t$, $g(\Delta l)$, to the cost function (3.12). The value $g(\Delta l) = K_g \|\Delta l\|^2$, where $K_g > 0$, served as a penalty on change in length. Here we an upper bound of 2 mm on Δl by modifying the perturb function.

To resolve occlusions using (3.12) we minimize the cost for the joint state $\boldsymbol{X}_t = \{\boldsymbol{X}_o\}, o = 1, \ldots, N_o$, where N_o is the number of targets in an occlusion. The surface $\mathcal{S}$ is the combined surface generated from all targets in the joint state. Similarly, the Plücker lines are generated from the occluding contours of the associated blobs of all targets in the occlusion.

To implement simulated annealing for solving (3.12), we note that the cost function, which measures the total distance between three-dimensional lines from a camera and a surface, is a bounded distance function. Convergence is shown for a bounded cost function in a logarithmic cooling schedule [61]. In [39] we ran a geometrically decreasing cooling schedule with an inner loop criterion to simulate a homogeneous Markov chain. Much of the time was lost in evaluating candidate solutions that did not satisfy the length constraint or were generally far from the initial solution. In this dissertation we sample candidate solutions more efficiently from an alternative perturb function described next.

Perturb function

In a simulated annealing algorithm, the perturb function $\boldsymbol{r} : \mathbb{R}^n \to \mathbb{R}^n$ selects a possible candidate solution in each iteration. In [39] we used a simple additive Gaussian distribution function with unit variance to perturb the current solution. This generated a large number of candidate solutions that were unrealistic, for example, rolling motion (which violates assumption A2), large pitch and yaw angles, and an increase in length of the midline. Since the shape is defined within the body frame, a realistic perturbation can be described as a sum of a rigid-body motion about the head and a local shape change.

We use the group of rigid body motions called the special Euclidean group, $SE(3)$, to define a random disturbance in orientation and position of the fish and tune the variance in each of the six degrees of freedom to represent a realistic fish motion. We also verify length of the midline before proposing a candidate solution. To sample shape parameters $\boldsymbol{p} = \begin{bmatrix} p_1, \ldots, p_5 \end{bmatrix}^T$ we note that in a straight midline, p_1 represents the length of the fish and $p_2, \ldots, p_5$ are all zero. A bent midline corresponds to non-zero values in $p_2, \ldots, p_5$. We model the fish midline as having constant length with the tendency to straighten out. We therefore model changes in p_1 using Gaussian white noise $\dot{w}_{p,i}$, and model $p_2, \ldots, p_5$ as exponentially decaying variables with rate $\lambda > 0$. Let the fish body frame (3.1) be represented by $g = {}^{\mathcal{W}}T_{\mathcal{B}} \in SE(3)$. The stochastic perturb function (with matrix multiplication as the composition rule for SE(3)) can be represented as [107, 47]

$$dg = gdW_g \tag{3.13a}$$

$$dp_1 = dw_{p,1} \tag{3.13b}$$

$$dp_i = -\lambda p_i dt + dw_{p,i},\ i = 2, \ldots, 5, \tag{3.13c}$$

where dW_g is a standard Wiener process on $se(3)$ [108] that denotes a disturbance input on each degree of freedom of a target. Let $E_i, i = \{1, 2...6\}$, be the basis elements of $se(3)$, the Lie algebra of SE(3) [109] and $\varepsilon_i = \mathbb{N}(0, \sigma^2_{\varepsilon,i})$ be a zero-mean Gaussian random variable representing the corresponding variance. The basis elements represent motion along each of the six degrees of freedom in the body frame and are given as [109]:

$$\text{roll} = E_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \text{pitch} = E_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix};$$

$$\text{yaw} = E_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \text{surge} = E_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix};$$

$$\text{sway} = E_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \qquad \text{heave} = E_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{3.14}$$

The variance $\sigma^2_{\varepsilon,i}$ of each degree of freedom is set so that the fish yaws more than it pitches and does not roll. The first-order Euler discretized version of (3.13) for a small time-step Δ is [110, 108]

$$\tilde{g}^j = g^j \exp(\sum_{i=1}^{6} E_i \sqrt{\Delta} \varepsilon_i) \tag{3.15a}$$

$$\tilde{p}_1^j = p_1^j + w_{p,1}\sqrt{\Delta} \tag{3.15b}$$

$$\tilde{p}_i^j = p_i^j \exp(-\lambda\Delta) + w_{p,i}\sqrt{\Delta},\ i = 2, \ldots, 5. \tag{3.15c}$$

Note that (3.15a) assumes that the motion along each degree of freedom is independent. Equations (3.15b and 3.15c) are run in a loop until the length of the midline is within 2 mm of the actual length measured during shape estimation. This form also ensures that the orthonormality of the body-fixed frame is preserved at every time step. We compare the performance of simulated annealing algorithm using the modified perturb function in Section 3.6.

3.4.3 Data Association

To associate targets to measurements across views and through time we use nearest-neighbor matching, which associates each target to a measurement based on Euclidean distance. However, the Euclidean distance between centroid positions may not provide an accurate association when the fish are close to one another, so we establish another metric described here.

The measurements in our case are silhouettes on a camera frame. Let the set of measurements on a camera frame be indexed by j. $\boldsymbol{Z}_j$ denotes a silhouette on the camera frame. The points in a silhouette are indexed by i, i.e, $\boldsymbol{Z}_j^i \in \boldsymbol{Z}_j$. Note that $\boldsymbol{Z}_j^i \in \mathbb{R}^2$ is measured in pixels. To match a silhouette with a target, we project the midline from each target onto the camera image plane. We then assign a silhouette to the target if it is the "closest" silhouette to the midline. The generalized distance metric computes the sum of the minimum distance of each point on the midline to a silhouette. Let ${}^c\hat{\boldsymbol{m}}_t(s)$ denote the predicted midline of target t. The measurement index j_t of the measurement that is assigned to target t in frame c is computed by solving

$$\begin{aligned} j_t &= \operatorname*{argmin}_j \sum_s q_s^* \text{ where} \\ q_s^* &= \min_i \| {}^c\boldsymbol{Z}_j^i - {}^c\hat{\boldsymbol{m}}_t(s) \|. \end{aligned} \tag{3.16}$$

Note that in (3.16) the minimum distance from the midline is computed. This is because we are not attempting to fit the midline to a silhouette, but rather to find how far it is from a given silhouette.

In the case of an occlusion, two or more targets are assigned the same silhouette. Since different targets may occlude in each view, an occlusion consists of all targets that share a silhouette in any view. The occluded blob is used as a combined measurement across views to fit shapes of all fish involved in an occlusion.

3.4.4 Smoothing

The optimization output is rarely smooth because errors in the measurements are absorbed into the estimates. We smooth the estimates by passing the output state through a Kalman filter. Fish movement comprises change in position, orientation, and shape. We model velocity and heading vector as being subject to Gaussian disturbance

$$d\dot{\boldsymbol{r}} = d\boldsymbol{w}_r,\ d\boldsymbol{h} = d\boldsymbol{w}_h, \tag{3.17}$$

where $\dot{\boldsymbol{w}}_r, \dot{\boldsymbol{w}}_h \in \mathbb{R}^3$ indicate white noise processes. Based on the reasoning for perturb function we model the change in shape according to (3.13c). Since the state equations for $\boldsymbol{r}, \boldsymbol{h}$ and $\boldsymbol{p}$ are decoupled the Kalman filter can be run separately for each.

3.5 Data Collection

In order to test our tracking framework, we filmed trials of one, two, four, and eight giant danio (*Danio aequipinnatus*) in a 0.61 m × 0.30 m × 0.40 m (24"×12"×16"), 20 gallon tank. Each trial lasted for 1–3 seconds. Three cameras

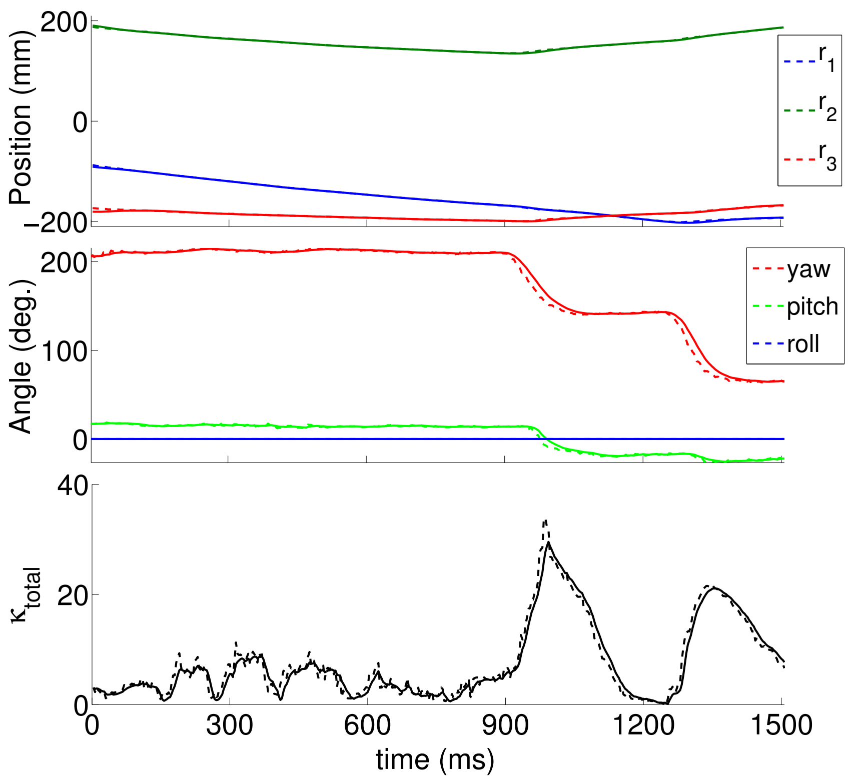


Figure 3.6: Smoothing optimized data. Time series plots of position, orientation, and total curvature for a single fish. The plots are shown before filtering (dashed lines) and after filtering (solid lines). The two peaks in the total curvature correspond to turns. κ_{total} is defined in Section 3.7

were used to film the fish (see Fig. 3.7). Two cameras were used for tracking and the third camera was used for validation. A DRS Lightning RDT high-resolution camera was placed above the tank to capture the top view at 250 frames per second (fps) and 1280 $\times$ 1024 pixel resolution. Two Casio EX-F1 Pro cameras were placed orthogonal to each other facing the tank sides. These cameras captured images at 300 fps and 512 $\times$ 384 pixel resolution. To ensure an adequately lit background, the remaining three sides of the tank were back-lit by a 150W fluorescent light source diffused by 1/4 stop with a diffuser fabric. Videos from the three cameras were synced by marking a frame in each video with a distinct common event. Simultaneous events during a trial were generated in the field of view of all three cameras by a string of flashing LEDs. The full videos were then synced and verified using a custom Linux shell script. (Every fifth frame in the 250 fps video was repeated.)

At the beginning of each experiment, a short video of the tank was recorded

Table 3.2: Parameter Values for Tracking Fish

Parameter	Value	Description
α	0.05	Background update coefficient (initial)
α	0.0001	Background update coefficient (final)
λ	5	Coefficient of decay for midline parameters $\boldsymbol{p}$
$\Sigma_{\mathcal{E}}$	diag{1,1,1}	Disturbance variance in elliptical cross section (mm)
$t_{\mathcal{E}}$	3	Threshold on shape covariance matrix norm for EKF
$t_{\mathcal{E}_k}$	0.02	Threshold on shape covariance improvement for iEKF
R_E	$\text{diag}\{1,1,2,2\}$	Noise covariance for shape estimation (pixels^2)

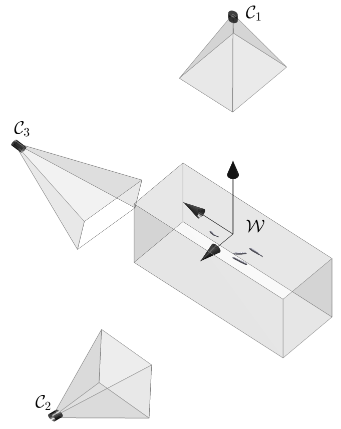


Figure 3.7: Setup and framing. Camera views $\mathcal{C}_1, \mathcal{C}_2$, and $\mathcal{C}_3$, and world frame $\mathcal{W}$. Cameras $\mathcal{C}_1$ and $\mathcal{C}_2$ are used for tracking; Camera $\mathcal{C}_3$ is used for validation purposes.

without any fish, so that we could model the background for background subtraction. Each tracking sequence starts with a set of background images, wherein the background is modeled as a running average with a tuning parameter ${}^{c}\alpha$ [111]:

$$ {}^{c}B_{k+1} = {}^{c}B_{k}(1 - {}^{c}\alpha) + {}^{c}\alpha {}^{c}I_{k+1}, \tag{3.18} $$

where ${}^{c}B_0$ is the first background image and ${}^{c}I_k$ is the current image of camera c. The value of ${}^{c}\alpha$ was kept high initially to characterize lighting fluctuations and was lowered when there were fish present. (See Table 3.2 for parameter values used for tracking.)

Camera calibration was performed using the MATLAB calibration toolbox [112]. A planar checkerboard was filmed underwater at different orientations inside the tank. Extrinsic calibration was performed by moving the checkerboard between the cameras and propagating the extrinsic parameters between overlapping camera views until all camera positions and orientations were known with respect to the world frame. The reprojection error during calibration for each camera was in subpixels. In three dimensions, the error was computed by comparing the known distance between checkerboard points (ranging between 30 mm and 210 mm apart) with the distance between estimated position. The average error over 50 such observations was 0.7 $\pm$ 0.37 mm. The world frame was chosen to be directly below the top camera such that the vertical axis pointed up (see Fig. 3.7). The top-view camera and the tank were aligned using a bubble level.

Once the calibration was performed, fish were introduced into the tank from a separate tank in sets of 1, 2, 4 and 8. Three trials were conducted for each set. Filming was started approximately ten minutes after the fish were introduced. The input to the tracking system was a set of synced frames from each camera (top and side) and the calibration parameters for each camera. The output is a time series of the state vector $\boldsymbol{X}$ for each fish. The number of fish was constant during each trial.

3.6 Performance and Validation

We analyze convergence and finite time behavior of the simulated annealing algorithm on artificial data. An estimate from experimental data is used as the initial value for the SA algorithm. A goal state is created by changing the position, orientation and shape parameters of the initial shape by random values that are recorded. The SA algorithm is run for four cases representing two choices each for perturb function and cooling schedule. For the perturb function we chose random Gaussian distribution with unit variance [39], denoted by G1 and (3.15) denoted by G2. We compute the values of $\sigma_{\varepsilon,i} = \{0^\circ, 10^\circ, 24^\circ, 56\text{mm}, 27\text{mm}, 14\text{mm}\}$ corresponding to roll, pitch, yaw, and forward, sideways, and downward motion from existing trajectory data. For cooling schedule we chose a geometric cooling schedule $\tau^{j+1} = 0.9\tau^j$ [39] denoted by C1 and logarithmic cooling schedule $\tau^{j+1} = \tau^0 / \log(j+1)$ denoted by C2. For C1 we also include an inner loop criteria to run perturb and update the solution (steps 1 and 2 in 2.2) 100 times before the current temperature is updated. (Not doing so quickly brings the temperature down significantly without a significant change in cost.) $\tau^0 = 10$ in each case, and the freezing temperature is 0.01 for C1 and 1.25 for C2.

Fig. 3.8 shows a single run of the SA algorithm for each of the four combinations. We note that the perturb function (3.15) significantly improves finite time performance for both types of cooling schedule. All combinations eventually attain the same cost as the ground truth data but the combination G1C2 took about 20,000 evaluations compared to average 1648 evaluations by G2C2 combination. We now describe the results of validation of the tracking system on real data.

Results for the tracking system are reported here for five out of the twelve trials. In every trial, we were able to track multiple fish shapes even during occlusions. The maximum density of the fish schools that we tracked was one fish per 2.5 gallons. (The actual density was higher because the fish schooled in only a fraction of the tank volume.) We used two methods to determine the accuracy of our tracking algorithm. First, the estimated shape and track reconstruction were verified

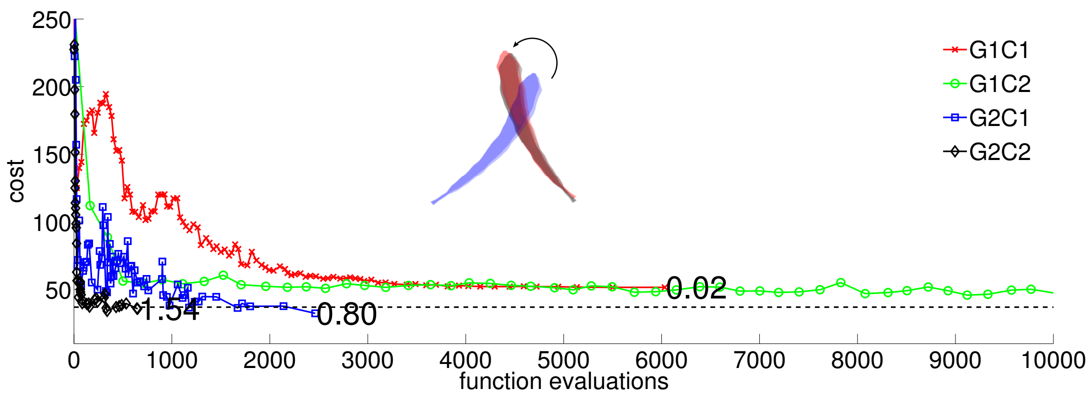


Figure 3.8: Finite time behavior of simulated annealing for shape reconstruction. We compare finite time behavior of simulated annealing for different choice of cooling schedules and perturb functions. An artificial shape fitting problem consists of the initial shape (blue) and measurements generated from the goal shape (gray). The final fit (red) is shown. Simulated annealing runs are shown for four different cases. The final stopping temperature for each run is shown at the end. (Combination G1C2 which stopped after 20,000 evaluations is cut short for clarity.) The cost function value (black dash line) for ground truth is also shown.

using an independent camera. Fig. 3.9 illustrates the accuracy of the tracker using the projected estimate on the third camera. Second, we randomly selected a set of frames across multiple videos and manually marked ten control points along the midline in the top view. The midline was then manually generated by interpolating a curve between the ten marked points. The orthogonal distance between each point on the estimated midline and the manually generated midline was computed at each point. Fig. 3.10 depicts the average, maximum, and minimum error on the midline. Comparing the manually generated midline and tracked midline in the top view for a single fish shows a maximum average error of five pixels at the tip of the tail. The tail error is primarily due to the inconsistent appearance of the semi-transparent caudal fin in the silhouette measurements.

The maximum total variation in length observed in eight fish that undergo a series of startle responses is less than 1 mm while the maximum change in distance between two elliptical cross section is 0.5 mm. Occlusions of two and three fish were tracked reliably as evidenced by Figures 3.11 and 3.12.

Since the tracking process depends on the silhouettes in each camera frame to estimate the fish position, orientation, and shape, the tracking accuracy is affected

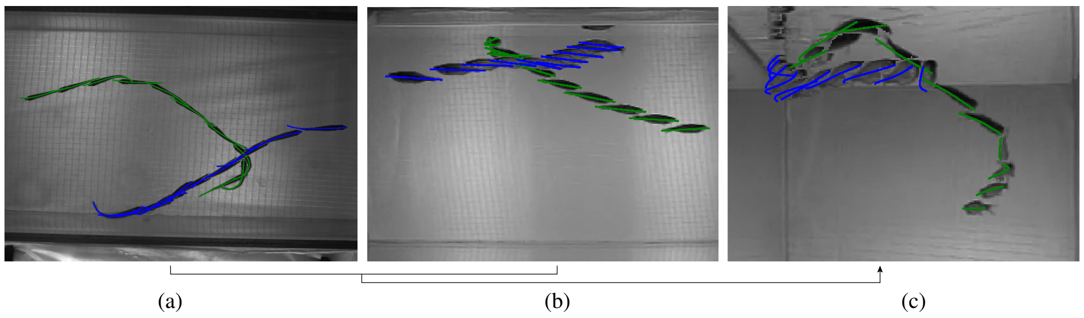


Figure 3.9: Validation using independent camera. The shape estimated from the (a) top and (b) side camera is projected onto a (c) multi-exposure image from the independent camera.

by the number of fish in an occlusion. In our setup, with the low-resolution side cameras, we found loss of accuracy in occlusions with four or more fish (See Fig. 3.13). There were no data association errors, although these are expected for dense occlusions. We intend to address this problem by increasing the camera resolution and number so that the views with the fewest occlusions can be used to estimate shape. The inaccuracies in the tracker result primarily from (a) the modeling assumption that the fish midline lies on an inclined plane; (b) dense occlusions, during which the limited resolution of the cameras make it difficult to resolve the silhouettes into individual shapes; and (c) the curve parameterization which may be insufficient to represent complex curves. The accuracy of the tracker can be further improved by segregating the head and orientation tracking from shape tracking when there are no occlusions. A particle filter may be run to track the head and orientation while simulated annealing can be used to estimate shape.

Inaccuracies may also result due to refraction between air and water. In the case of our setup where the camera image plane was parallel to the water surface and centered with respect to the face of the tank, errors due to refraction were low (Section 3.5; also see Appendix B), however, mounting the cameras at an angle to the water surface would require compensation for refraction effects.

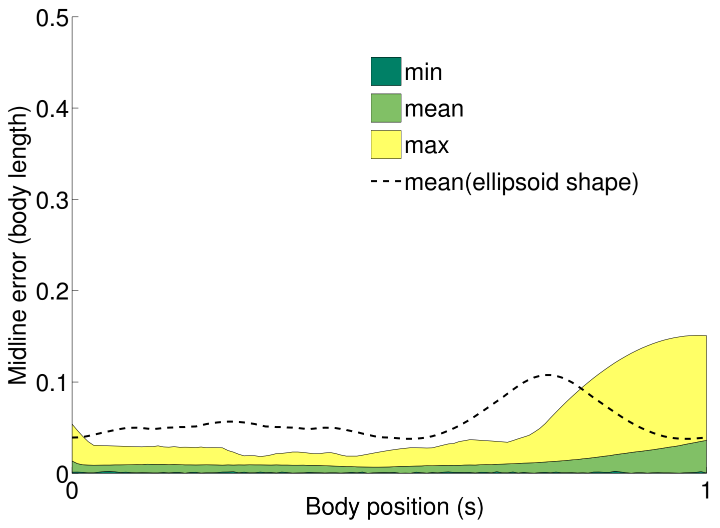


Figure 3.10: Error in midline fit. The midline was manually selected on a random set of 100 top-view frames. The distance between the projected estimate and the manually generated midline as measured in the top camera frame. For comparison with previous work, we also computed the mean error (dashed line) for a fish shape modeled as a bent ellipsoid [38].

3.7 Kinematic Data of Fast Start Response

The shape-tracking system described in this paper yields a new opportunity to study fish behavior. The full-body reconstruction at every step allows one to automatically detect and quantify fast-start behavior, which we are doing in ongoing work outside the scope of this paper. Fig. 3.14 compares the curvature profile for a coasting motion with the profile for a fright response. We compute curvature and total curvature from the midline $\boldsymbol{f}(s)$ as [113]

$$\kappa = \frac{|f_1' f_2'' - f_2' f_1''|}{(f_1'^2 + f_2'^2)^{3/2}} \text{ and } \kappa_{\text{total}} = \int_0^1 \kappa(s) ds. \tag{3.19}$$

In the first case, the fish was filmed without any disturbance. The second case is a midline reconstruction of a single fish from a multi-fish trial during which the fish was startled by a visual stimulus.

When no fright stimulus was presented, the curvature is high towards the tail. A coasting turn takes more than one hundred milliseconds and the curvature profile is flat. In the case of a fright response (an S-start), high curvature appears along the midline. The turn occurs in approximately 40 ms and appears as a dark band at 450

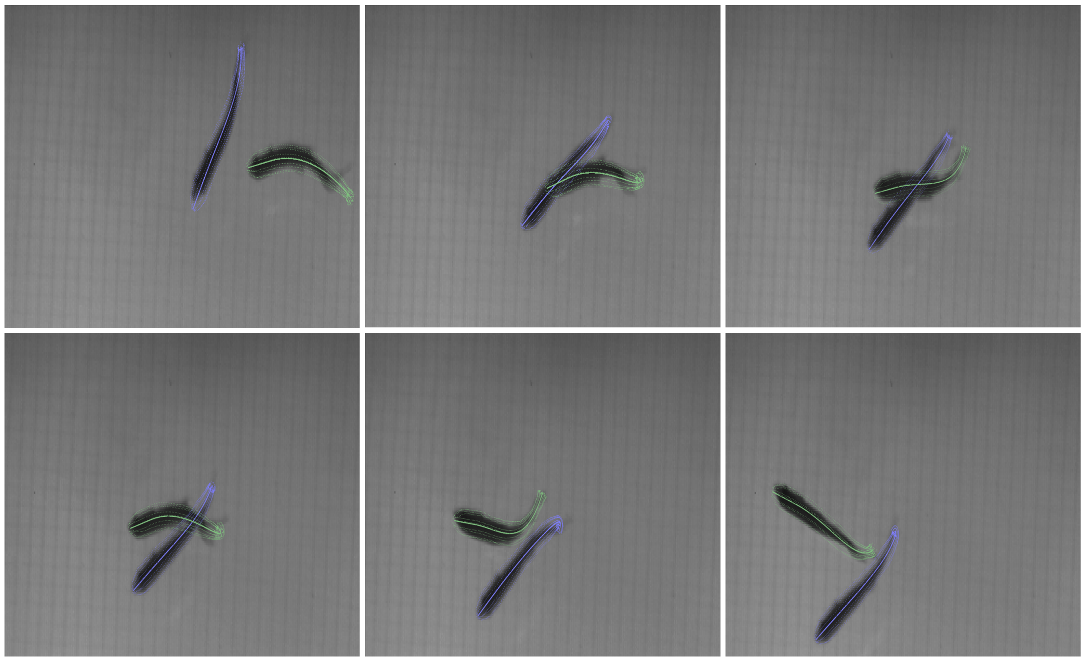


Figure 3.11: Tracking two fish. Sequence of frames showing shape tracking during an occlusion.

ms. A thin dark region near body length 0.9 appears in the curvature plots due to the combined effect of tail beat movement and inaccuracy in the tail reconstruction due to inconsistent appearance of the caudal fin. The three-dimensional reconstruction of each of these turns shows the distance travelled by each fish during the turn. The coasting fish travels 54 mm in 500 ms where as the startled fish travelled 160 mm in the same time.

The kinematic parameters that can be extracted from a dataset largely depend on the type of input video. We show that with a relatively-close range high-speed video it is possible to reconstruct shape of individual fish in a school. The same school of fish filmed from a larger distance with low resolution cameras [38] provides data for a different type of analysis that require longer time-scales and larger individual distances travelled. With the increase in the number of targets and larger distances covered within a single frame, the challenges now shift to frequent occlusions and highly unpredictable motion. To extract long trajectories it makes sense to use motion coherence to resolve occlusions. This can be done probabilistically

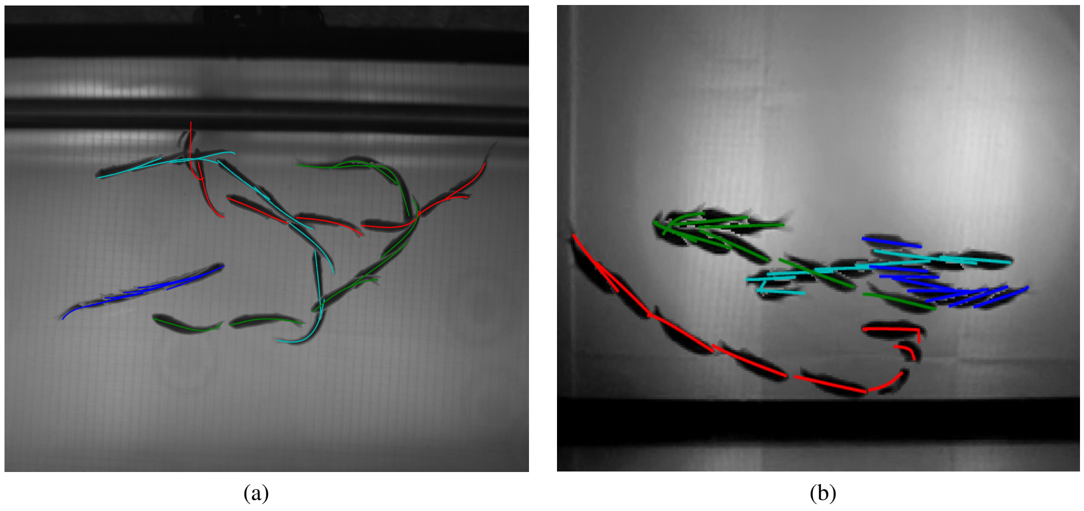


Figure 3.12: Tracking four fish. Top and side views of (a) four and (b) eight fish tracked through 500 and 250 frames. Due to low resolution, the tracking accuracy is reduced (blue midline) in the side-view camera for eight fish during dense occlusions.

as the targets are tracked or offline using an optimization method that joins track segments over multiple frames. In contrast from fish schools, videos of mosquito swarms present a different challenge where motion-blurred point-mass targets move in a random fashion. In that sense, techniques developed for tracking mosquito swarms described in the next chapter may be readily adapted to track a large school of fish from a distance.

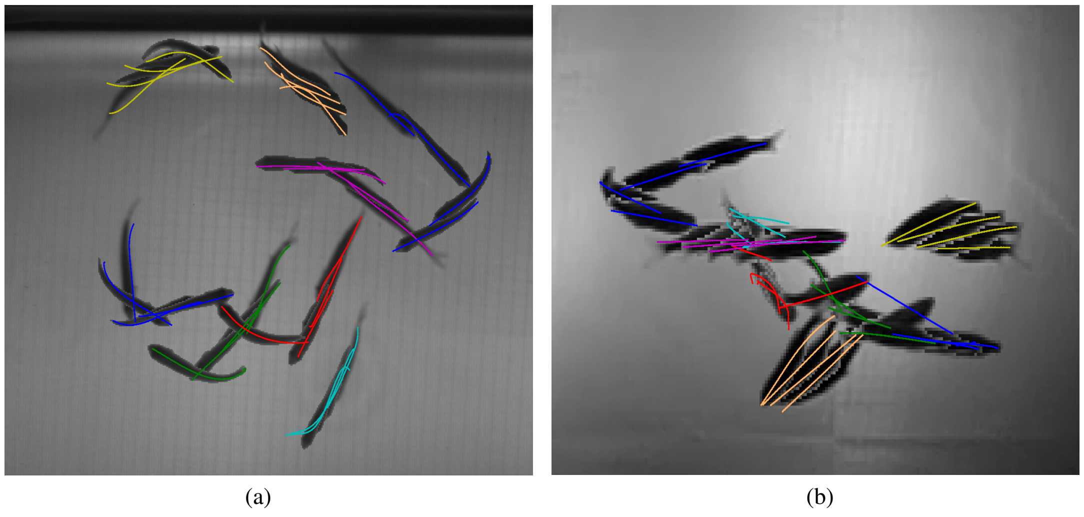


Figure 3.13: Tracking eight fish. Top and side views of (a) four and (b) eight fish tracked through 500 and 250 frames. Due to low resolution, the tracking accuracy is reduced (blue midline) in the side-view camera for eight fish during dense occlusions.

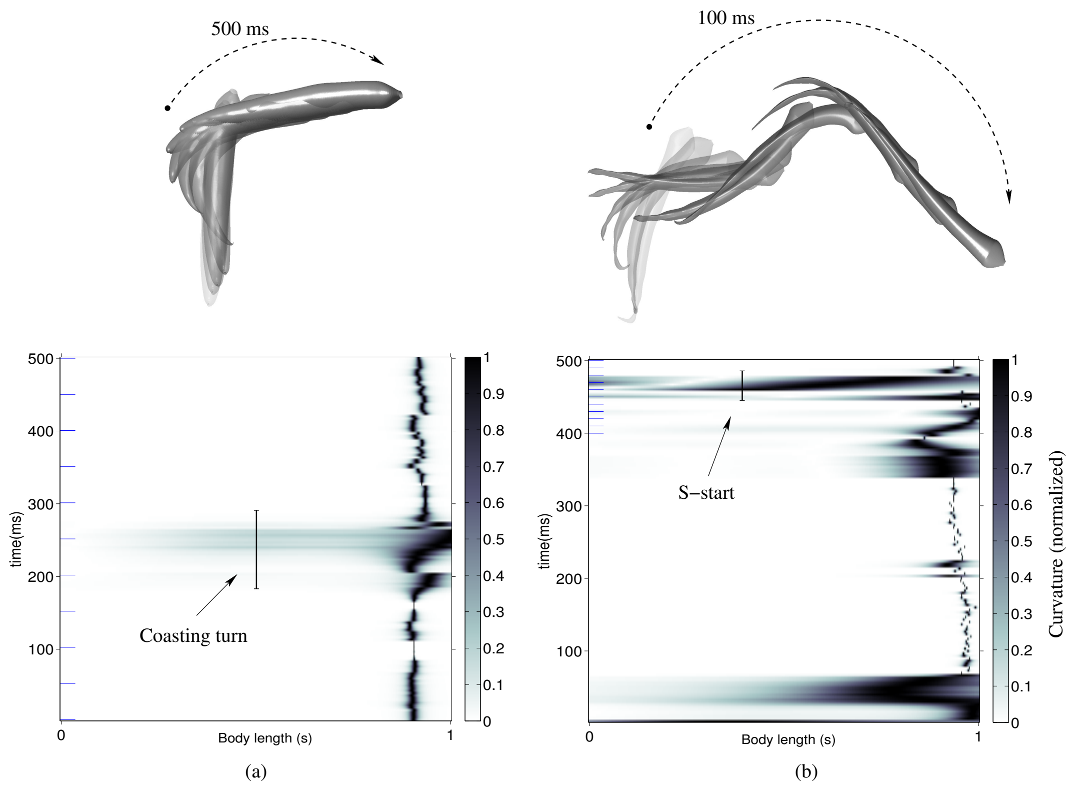


Figure 3.14: Curvature profile comparison of fast-start and coasting. Curvature profile using the three-dimensional reconstruction at fixed intervals (blue ticks). Curvature profile of fish midline during (a) a coasting turn with no fright stimulus and (b) a fast-start turn. The curvature is normalized through the body length at each time step. The dotted arc marks the beginning and end of the turn. The time in ms denotes the duration of the turn.

Chapter 4

Reconstructing Flight Kinematics of Swarming Mosquitoes

Quantitative observations of the flight patterns of wild mosquitoes are critical to expanding our understanding of swarming and mating behavior [21, 45, 46, 22, 20, 114]. Female *Anopheles gambiae* find male swarms in order to mate [115, 20]. A single mating event results in all of the fertilized eggs that a female mosquito lays in her lifetime [23, 116]. Although the basis of mate selection has generated much interest [115, 117, 118, 23, 114], generation of three-dimensional trajectory data of mosquitoes in wild swarms has not been previously accomplished.

This chapter describes an automated multi-target tracking system that reconstructs the three-dimensional flight kinematics of individual mosquitoes in wild swarms. We collect data using two cameras operating synchronously at 25 frames per second. (The frame rate is limited by the ambient light.) The cameras and a laptop are powered by an uninterrupted power supply (UPS) for up to thirty minutes. The mosquitoes appear as dark streaks or dots on a light background. At high speeds, the mosquito streaks fade, making them hard to detect, and even harder to track. Because the swarms are dense, occlusions are frequent, and often appear in both camera frames. We tested the system by filming swarms and mating events of *An. gambiae* in a rural village in Mali in August 2010. Fig. 4.1 shows a pair of magnified and enhanced sample frames from this field experiment.

The tracking system is implemented in MATLAB® and consists of two parts: an automated component that outputs track segments called tracklets and a human-supervised component that is used to verify and combine the tracklets into full-length

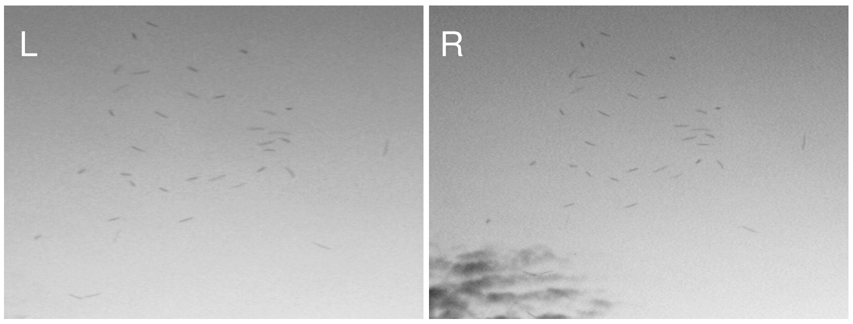


Figure 4.1: Stereo images of a mosquito swarm. The pair of images above are magnified and enhanced versions of raw footage obtained from the author's field work in Mali.

tracks. Tracklets produced by the automated component typically range between 15–25 frames (0.6–1 s) long and can be used to extract position and velocity data for 80% of the swarm. The human-supervised component uses a particle filter to combine tracklets into individual mosquito tracks. It takes up to 20 minutes to generate a 10-second track (250 frames). When validated using data filmed in Mali in August 2010, the tracking system produced 30–40 second trajectories of individual mosquitoes in swarms of 6–25 mosquitoes. We have reconstructed six swarms and six mating events from these data. We evaluate the performance of the automated component of the tracking system using an established metric based on position error and target cardinality; tracking accuracy was also evaluated using two independent rigs to track the same swarm.

Section 4.1 provides an overview of the tracking framework and lists the tracking algorithm. Section 4.2 assigns the state and measurements for each mosquito. Section 4.3 presents foreground segmentation method including adaptive seeking of missing measurements. Section 4.4 describes the tracking algorithm including the likelihood function and the multi-hypothesis data association. Section 4.5 describes the human supervised track linking and verification tool. Section 4.6 presents the data collection methods including a description of stereo camera setup and the design parameters used in filming and tracking. Section 4.7 evaluates the performance of the tracking algorithm. Section 4.8 presents representative kinematic data of swarming and mating events. The data presented in this chapter was collected with assistance from Dr. Nicholas Manoukis from U.S. Department of Agriculture, Moussa Diallo from Malaria Research and Training center, Bamako, Mali, and with support from residents of the village of Donéguébogou, Mali.

4.1 Tracking Framework

Our aim in designing the mosquito tracking system is to combine nearly indistinguishable measurements available from stereo images recorded at discrete times into trajectories that represent real mosquitoes (targets). The mosquito tracking system takes a sequence of stereo image pairs as input and produces three-dimensional tracks as output. Fig. 4.2 depicts a block diagram of the tracking system. The automated component models blobs as straight lines and extracts the midpoint and endpoint of each blob as measurements. We find missing blobs using a gating volume generated around predicted measurements. Measurement pairs, i.e., one from each camera, that satisfy the epipolar constraint (2.12) are selected for data association. We again use gating volumes to group targets and measurements into independent sets called clusters. Instead of generating definite tracks, hypotheses connecting measurements to targets are propagated to the next step using a particle filter. Based on the probability of each hypothesis at the current time step, the number of hypotheses at a previous time step are reduced to a single assignment. A single target particle filter verifies and combines tracklets under human supervision; the combined tracks are passed through a Kalman smoother [119]. The tracking algorithm is summarized in Table 4.1.

4.2 State Space Representation

We represent target t at time step k by the state vector $\boldsymbol{X}_t[k] \in \mathbb{R}^6$, which contains the target's instantaneous three-dimensional position and velocity. The measurement $\boldsymbol{Z}_m[k] \in \mathbb{R}^6$ in our case consists of the two-dimensional positions of the midpoint and two endpoints of an elongated blob in an image that corresponds

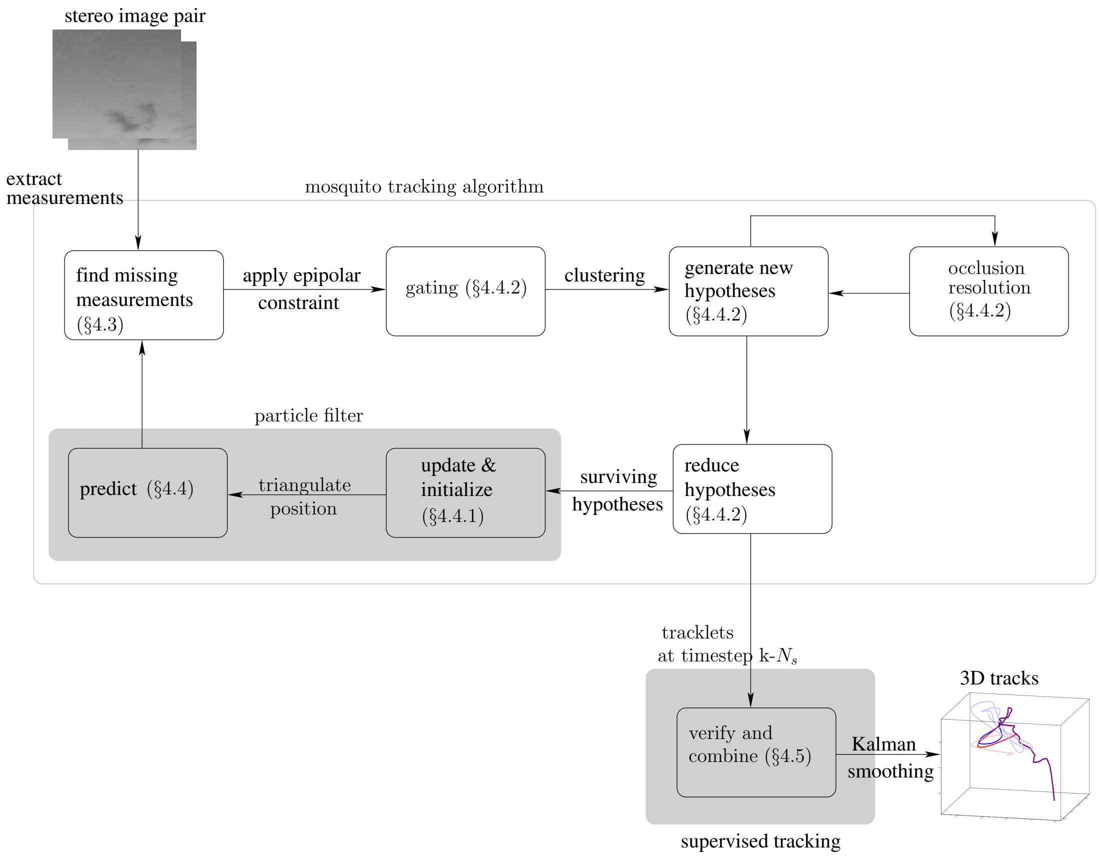

stereo image pair
extract measurements
mosquito tracking algorithm
find missing measurements (§4.3)
apply epipolar constraint
gating (§4.4.2)
clustering
generate new hypotheses (§4.4.2)
occlusion resolution (§4.4.2)
particle filter
predict (§4.4)
triangulate position
update & initialize (§4.4.1)
surviving hypotheses
reduce hypotheses (§4.4.2)
tracklets at timestep k-N_s
3D tracks
verify and combine (§4.5)
Kalman smoothing
supervised tracking

Figure 4.2: Block-diagram of the mosquito tracking system.

to the motion-blurred silhouette of a flying mosquito in each of the two images.

4.3 Feature Extraction

During observation of mosquito swarms, which may appear silhouetted in front of trees under a cloudy sky, it may not be possible to use a static (mosquito-free) background to segment the mosquitoes out of the image stream. Instead we create a dynamic background by choosing the highest intensity point within a sliding window [120]. Let $B_{u,v}$ be the background image value at the pixel position (u, v) and $t_{win} = 2d + 1$ be the width of the sliding window centered at time step k. The background value at time k is

$$B_{u,v}[k] = \max_{i \in [k-d, k+d]} B_{u,v}[i]. \tag{4.1}$$

The binarized foreground F is obtained by subtracting the background B from the current image I and applying an intensity threshold t_{int}, i.e., $F_{u,v}[k] = \max(I_{u,v}[k] - B_{u,v}[k], t_{int}) - t_{int}$. We automatically select the value of t_{int} by running the background subtraction algorithm recursively on different segments of the image sequence until the number of blobs detected are within an acceptable range of the expected number of mosquitoes. We extract blobs using the *regionprops* routine in MATLAB®, which performs connected-component labeling to extract features such as centroid, area, and bounding ellipse. We remove large insects and birds from the foreground by applying a threshold on the blob area. (See Table 4.2 for the values of the threshold parameters.)

Due to the duration of the camera exposure ($\delta t_e = 25$ ms)[1], fast mosquitoes (1–4 m/s) appear as elongated image blobs or streaks. Depending on the mosquito speed, the streaks may fail to appear in the foreground for a given value of t_{int}. Existing strategies for low signal-to-noise environments include the track-before-detect approach [97], which permits raw sensor data as the input. The success for track-before-detect relies on the low target density and relatively straight movement of targets in the measurement space [89]. However, using raw sensor data is not a viable option for mosquito tracking, because it generates more false targets than observed in a single noisy image. Instead, we search for the missing streak in a new foreground generated using a threshold of $0.75t_{int}$. The search is performed within the gating volume of the predicted measurement. If a missing measurement is found, it is added to the list of existing measurements.

A measurement $\boldsymbol{Z}^c = [\boldsymbol{e}^c_-, \boldsymbol{u}^c, \boldsymbol{e}^c_+]^T$ from camera c contains the image locations of a streak's start $\boldsymbol{e}^c_-$, midpoint $\boldsymbol{u}^c$, and end $\boldsymbol{e}^c_+$. These values are extracted by performing a least-squares fit on the pixel positions of the blob by modeling it as a straight line. The streak therefore represents a perspective projection of the mosquito trajectory for the duration of exposure δt_e. To gate three-dimensional points arising from measurement pairs, we apply the epipolar constraint (2.12) on

[1]The duration of exposure (25 ms) is less than the time between frames at (40 ms). The remaining time (15 ms) is for image processing.

midpoints, one from each camera. Only measurement pairs from a true target should satisfy the above constraint; clutter or mismatched measurement pairs should not. We use the midpoint and endpoint locations to define a likelihood functions for position and velocity.

4.4 Tracking Algorithm

The tracking algorithm is a multi-hypothesis tracker that associates every measurement to a hypothesized target. This section describes the likelihood function and data association methods.

4.4.1 Likelihood Function

A constant-velocity model suffices to describe the mosquito motion during the exposure ($\delta t_e = 25$ ms). Let $\boldsymbol{r} \in \mathbb{R}^3$ be the three-dimensional location of the midpoint of a streak. The start and end of the streak are located at $\boldsymbol{r}_- = \boldsymbol{r} - \dot{\boldsymbol{r}}\frac{\delta t_e}{2}$ and $\boldsymbol{r}_+ = \boldsymbol{r} + \dot{\boldsymbol{r}}\frac{\delta t_e}{2}$, respectively. The corresponding point on the image plane is given by the perspective projection model [85],

$$f^c(\boldsymbol{r}) = \left(\frac{w_1}{w_3}, \frac{w_2}{w_3}\right), \tag{4.2}$$

where $\boldsymbol{w} = P\boldsymbol{r} \in \mathbb{R}^3$, and P is the camera projection matrix. Let $\mathbb{N}(\boldsymbol{u}; f(\boldsymbol{r}), \Sigma)$ denote a normal density function evaluated at $\boldsymbol{u}$ with mean $f(\boldsymbol{r})$ and covariance matrix $\Sigma \in \mathbb{R}^{2\times 2}$. Assuming that the measurement is normally distributed about the true value, the likelihood of midpoint $\boldsymbol{u}^c$ given $\boldsymbol{r}$ is

$$P^c_{mp}(\boldsymbol{u}^c \big| \boldsymbol{r}) = \mathbb{N}(\boldsymbol{u}^c; f^c(\boldsymbol{r}), \Sigma_{mp}), \tag{4.3}$$

We set the diagonal entries of Σ_{mp} equal to the length of the major and minor axes of the streak's bounding ellipse in the streak frame; the off-diagonal entries are zero.

As with the midpoint likelihood function, we assume the endpoint likelihood function is based on a normal density function. However, due to uncertainty in the labelling of the start and end of the streak, the endpoint likelihood function is bimodal. The directional ambiguity is described by a sum of conditional probabilities on the order of endpoints. Let Σ_{ep} be the covariance of the endpoint position in pixels (computed empirically). The endpoint likelihood function is

$$\begin{aligned} P_{ep}^{c}(\boldsymbol{e}_{-}^{c}, \boldsymbol{e}_{+}^{c} | \boldsymbol{r}, \dot{\boldsymbol{r}}) = 1 - & \\ & (1 - \mathbb{N}(\boldsymbol{e}_{-}^{c}; f^{c}(\boldsymbol{r}_{-}), \Sigma_{ep}) \mathbb{N}(\boldsymbol{e}_{+}^{c}; f^{c}(\boldsymbol{r}_{+}), \Sigma_{ep})) \\ & (1 - \mathbb{N}(\boldsymbol{e}_{-}^{c}; f^{c}(\boldsymbol{r}_{+}), \Sigma_{ep}) \mathbb{N}(\boldsymbol{e}_{+}^{c}; f^{c}(\boldsymbol{r}_{-}), \Sigma_{ep})), \end{aligned} \tag{4.4}$$

where $\boldsymbol{r}_{\pm} = \boldsymbol{r} \pm \dot{\boldsymbol{r}}(\delta t_e/2)$. The combined effect of using a pair of points in the endpoint likelihood function (4.4) is to reduce the set of velocity values along the camera axis, which is otherwise unobservable.

The combined position and velocity likelihood function is

$$\begin{aligned} P(\boldsymbol{Z}|\boldsymbol{X}) &= \prod_{c=1,2} P(\boldsymbol{e}_{-}^{c}, \boldsymbol{u}^{c}, \boldsymbol{e}_{+}^{c} | \boldsymbol{r}, \dot{\boldsymbol{r}}) \\ &= \prod_{c=1,2} P_{mp}^{c}(\boldsymbol{u}^{c} | \boldsymbol{r}) P_{ep}^{c}(\boldsymbol{e}_{-}^{c}, \boldsymbol{e}_{+}^{c} | \boldsymbol{r}, \dot{\boldsymbol{r}}). \end{aligned} \tag{4.5}$$

Fig. 4.3 shows the combined position and velocity likelihood function. The likelihood function (4.5) is used to assign weights to particles prior to resampling in the particle filter. We update the position estimates using triangulation [121], thereby effectively marginalizing out the position from the combined position and velocity filtering pdf.

A velocity likelihood function improves the reliability of data association by placing predicted measurements closer to actual measurements. We compared the absolute velocity estimation error between a stand-alone position likelihood function and the combined position and velocity likelihood function (4.5). To create ground-truth data we isolate a single mosquito track in both camera frame for 8 seconds. We then interpolate the position values to every 1/800th of a second. These values were then used to create an artificial mosquito streak during the time of exposure δt_e with from a 1 cm sphere. We then project the streak on a synthetic left and right camera white image with 1392×1024 pixel resolution that matches our experimental setup. To achieve the same faded streak effect, when the mosquito flies across the image, we reduce the intensity value of a pixel by 30 every time it is visited on the screen

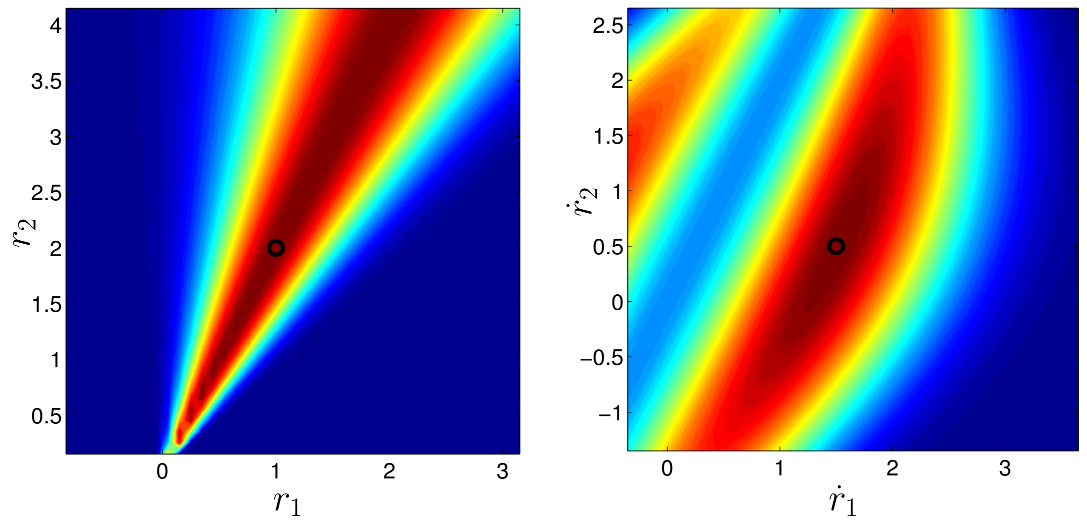
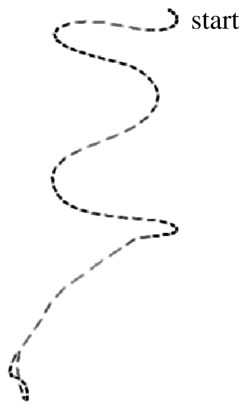


Figure 4.3: Velocity likelihood function. Top-down view of the (a) position and (b) velocity likelihood functions on a plane orthogonal to the image and parallel to the camera axis. The camera is located at $(r_1, r_2) = (0, 0)$.

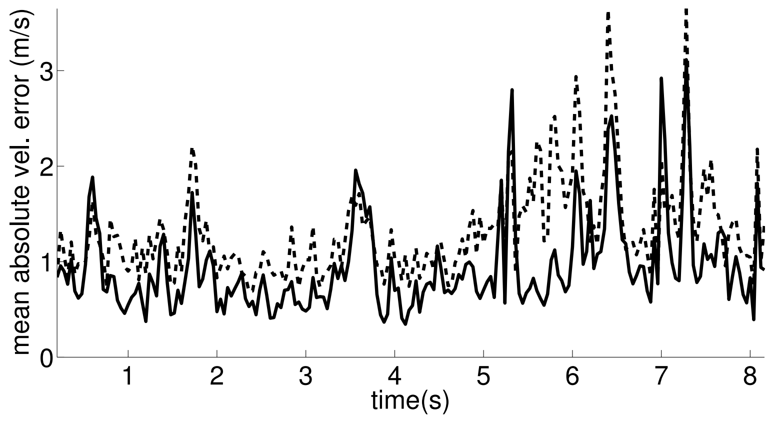
during exposure. We track this dataset using multiple Monte-Carlo runs of a particle filter. The combined position and velocity likelihood function performed better than the stand-alone position likelihood function, with an average improvement of 27% (see Fig. 4.4). We update the position in both cases using triangulation. Average position error was 5.78 ± 4.26 mm.

4.4.2 Data Association

Prior to weighting a target distribution with a likelihood function, we must first address the data-association problem. The mosquito data-association problem is challenging due to the variable number of targets. To address the uncertainty in association (for example, did the paths of two mosquitoes cross each other, or was it a close encounter?), we use a deferred-logic method called the multiple hypothesis tracker (MHT) [82]. Each assignment of measurements to targets is set aside as a hypothesis and acted upon in a future time-step when we are more certain. The certainty is computed using the probability of a hypothesis that depends on the innovation $\nu^c = \boldsymbol{u}^c - f^c(\boldsymbol{r})$ of each measurement-target assignment in the hypothesis,



(a)



(b)

Figure 4.4: Mean absolute velocity estimate error. (a) We simulated an artificial mosquito by projecting a 1 cm sphere on synthetic images for the time of exposure. Note that the streak fades with increasing velocity in the parallel to the image. (b) Mean of absolute velocity estimate error for this data over multiple Monte-Carlo runs using a stand-alone position likelihood function (dash-dot) compared to a combined position and velocity likelihood function (solid).

probability of detection, and the covariance of the predicted measurement S

$$P_{pos}(\boldsymbol{u}_m^{1,2}|\boldsymbol{r}_{(m)}) = \prod_{c=1,2} \mathbb{N}(\boldsymbol{u}_m^c; f^c(\boldsymbol{r}_{(m)}), S), \tag{4.6}$$

where the covariance $S = \text{cov}(f^c(\boldsymbol{r}_{(m)}))$ computed over all samples in the particle filter distribution.[2]

We reduce the number of hypotheses by clustering and prune them by selecting a few best hypotheses based on their probability at each step. Clustering is performed by dividing the measurement and hypothesized targets at each step into independent sets. At each time step, measurements are associated to each cluster based on the combined gating volume of all targets within the cluster. Measurements that do not belong to any cluster form their own clusters. Two clusters that consist of the same measurement are combined to form a single cluster. Similarly, we split clusters that consist of targets only assigned to a single measurement. Hypotheses are computed for each cluster independently. Hypotheses within a large cluster (more than 10 measurements) are limited to a single localized global nearest neighbor assignment [86]. (Such an assignment can generate a few more hypotheses

[2]Note that this is an approximation since the particle distribution may not be truly Gaussian.

by using the Murty's algorithm [88]. Using a single scanback [82] at each step, we choose the hypothesis with the highest probability to reduce to one the number of hypotheses at the previous step. Child hypotheses resulting from a pruned parent hypotheses are also removed.

New targets are automatically initialized from unassigned measurements and automatically confirmed as a threshold is applied on hypotheses probability. New target distributions are sampled from a normal distribution with a low standard deviation (5 mm) in position about the triangulated point, and a large standard deviation (500 mm/s) in velocity about zero. The combined likelihood function resamples the distribution to equally favor particles getting projected on either side of the streak in the next timestep.

Occlusions are not directly addressed as part of any data-association strategy, because existing strategies assume that each measurement can at most arise from a single target and that motion coherence will automatically associate the right tracks in a future timestep. In our case, occlusions undermine the velocity estimate, making future associations less reliable. An occlusion is detected if (a) two measurement pairs within a hypothesis consist of the same measurement from a single camera, or (b) multiple hypotheses assign the same measurement to two or more targets. We interpret an occlusion as a combination of individual streaks, which are then used to extract velocity information as described in Section 4.4.

In order to cluster the pixels in an occlusion blob we use the information about the number of mosquitoes hypothesized in the occlusion as well as their position and velocity estimates to model the blob as a mixture of Gaussians. We use MATLAB® file exchange function *emgm* [122] to run the expectation maximization algorithm [123] with position estimates for initial means and velocity estimates for diagonalized covariance matrix to hard-cluster the pixels into individual streaks. This set of individual streaks are used as an initial guess to soft-cluster[3] the pixels into more accurate overlapping streaks. Using the shortest distance of a pixel from the line

[3]Soft clustering allows a single pixel to be assigned to more than one cluster, whereas hard-clustering assigns each pixel to exactly one cluster.

that passes through the split streak, we allow multiple assignments of each pixel to individual streaks. Fig. 4.5 shows four instances of splitting an occluded blob into individual mosquito streaks.

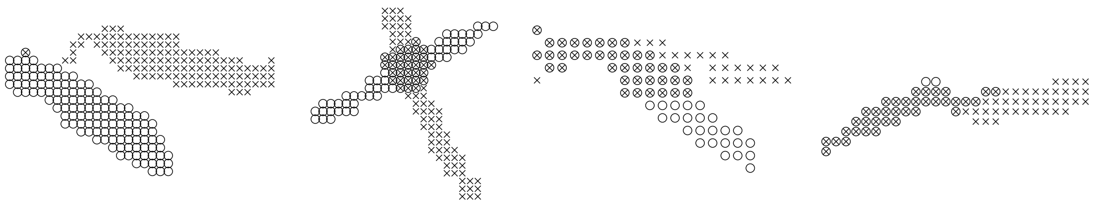
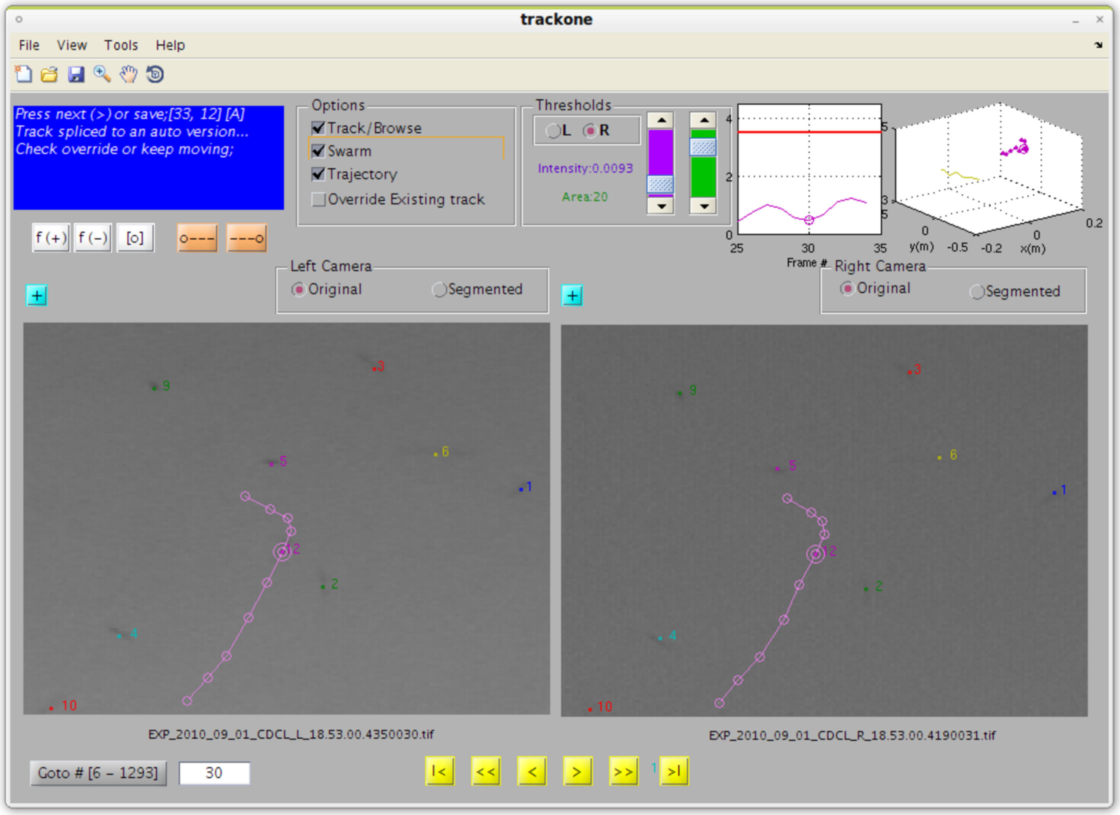


Figure 4.5: Resolving occlusions by clustering blobs. Four examples of occlusion resolution by clustering pixels of each blob into individual mosquito streaks. Each streak is denoted by a different marker type.

4.5 Track Linking and Verification Tool: trackone

We experienced track splits due to missed or dropped measurements (see discussion in Section 4.7). The resulting track segments called tracklets are combined under human supervision one mosquito at a time in a GUI script that called *trackone.*

Trackone comprises of a particle filter that runs in the background and a GUI to display the output (see Fig. 4.6). The particle filter operates on the three-dimensional tracklet data as a restricted state space to track a single mosquito and uses a constant velocity motion model and nearest-neighbor matching to find the closest measurements at the end of a tracklet. (The nearest-neighbor match is less restrictive that MHT and allows for quicker visual verification of the match by the user.) The GUI displays left and right camera frames with the option to switch between raw and segmented images. Intensity and area thresholds can be varied for each camera. Once a mosquito is selected in a camera frame, an epipolar line centered on the current estimate is drawn in the other camera frame to highlight possible matches. The closest tracklet at the current time step within 25 mm is automatically combined to the current estimate. The GUI permits overriding the tracklet suggestion, or, verifying the same by fast-forwarding to the end of the tracklet. Forward and backward tracking is allowed. At the end of a tracklet the user may press next to associate with the nearest measurement pair or mark the mosquito manually in any one frame. To aid verification, speed and three-dimensional position of currently tracked mosquito is displayed online with the option of viewing a segment of the trajectory in each frame. All mosquitoes tracked can be viewed to check for possible track switches. Tracks can be switched, deleted, and created by specifying the unique mosquito id assigned at the first instance the mosquito is selected. The amount of time spent in generating a 10-second track ranges between 5-20 minutes based on the track segmentation.


Press next (>) or save;[33, 12] [A]
Track spliced to an auto version...
Check override or keep moving;

Options
- [✓] Track/Browse
- [✓] Swarm
- [✓] Trajectory
- [ ] Override Existing track

Thresholds: L [✓ R]
Intensity:0.0093
Area:20

Left Camera: [✓ Original] Segmented
Right Camera: [✓ Original] Segmented

EXP_2010_09_01_CDCL_L_18.53.00.4350030.tif
EXP_2010_09_01_CDCL_R_18.53.00.4190031.tif

Goto # [6 – 1293] 30

Figure 4.6: GUI tool for verification and linking of track segments. We develop a MATLAB® graphical user interface called **trackone** to combine tracklets under human supervision.

4.6 Data Collection

To validate the mosquito-tracking system in the field, we used a pair of phase-locked Hitachi KP-F120CL cameras in a stereo configuration. Fig. 4.7 shows a schematic of the data collection system. The video streams were recorded using a 2.8 GHz quad core laptop, an Imperx FrameLink Express frame grabber (Imperx Inc, Boca Raton, FL USA), and Streampix 5 software (Norpix Inc, Quebec, Canada). Each camera was calibrated onsite using a checkerboard and the MATLAB® Calibration Toolbox [112]. Reprojection error, which is a measure of calibration accuracy, was in sub-pixels for each camera. Relative camera orientation and position was determined by extrinsic calibration by taking multiple pictures of a stationary checkerboard with both cameras. During filming, the camera height, azimuth and elevation were recorded to create a ground-fixed reference frame. We used a Kestrel 4500 portable weather station (Nielsen-Kellerman, Boothwyn, PA USA) to sample other environmental factors such as wind velocity and humidity at 0.1 Hz.

Filming was done in the village of Donéguébogou, Mali in Western Africa. Donéguébogou is 29 km north of Bamako and has been the site of previous research on *An. gambiae* mosquitoes [20, 117]. Swarms formed approximately 20 minutes after subset, initially with only one or two males then increasing in numbers, and lasted for 20 minutes. Most couples were seen 5-10 minutes after the swarm was first observed. Couples formed only for a few minutes during this period, then were no longer observed, though the males continued to swarm for many minutes after the last couple had formed. We filmed swarms of *An. gambiae* that formed over bare ground or markers.

We filmed twenty-one swarms and thirteen mating events between August 17, 2010 and September 3, 2010. Out of the twenty-one swarms, eighteen formed over bare ground and three formed over natural markers. (A natural marker is an area of high contrast with the rest of the ground such as a patch of grass.) *An. gambiae* can be divided into two incipient species namely the M and S molecular forms. In [117] a strong association between the swarming marker type and molecular form

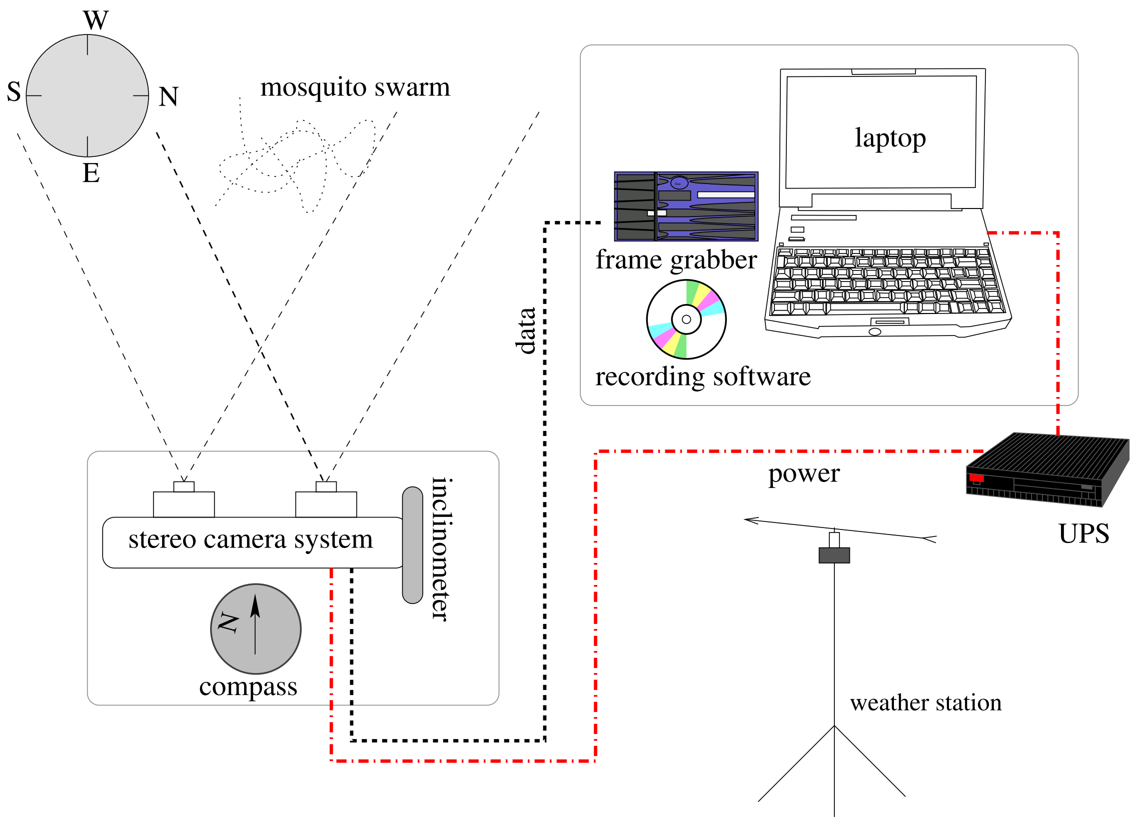


Figure 4.7: Mosquito tracking data collection system. The data collection system consists of the stereo camera rig and a laptop powered by an uninterrupted power supply (UPS). A capture signal is sent from the frame grabber to record frames in sync. A Kestrel weather station records environmental factors such as temperature and wind velocity at 0.1 Hz.

has been found. The M form was found to swarm over natural markers, whereas the S form swarmed over bare ground [117]. We collected a few mosquitoes from each swarm and performed a polymerase chain reaction (PCR) test to determine the molecular form. All sequences presented here were of type S. Each day two teams of 3–5 people with identical camera rigs selected separate swarming sites for filming. The swarming sites were usually within a few hundred meters of each other. Swarming sites were surveyed the day before to record average swarm size and location. Filming locations spread throughout the village (see Fig. 4.8) were chosen based on swarm size (less than about 100 mosquitoes for tractability in tracking) and the presence of few trees or houses in the background (i.e., in the direction of the setting sun). Once filming began, 60–90s stereo video sequences were recorded as 10-bit synchronized tiff images on separate solid state drives. (The drives were backed up daily on to two separate disks.) A filming session typically produced 5–8 video sequences before it became too dark to film.

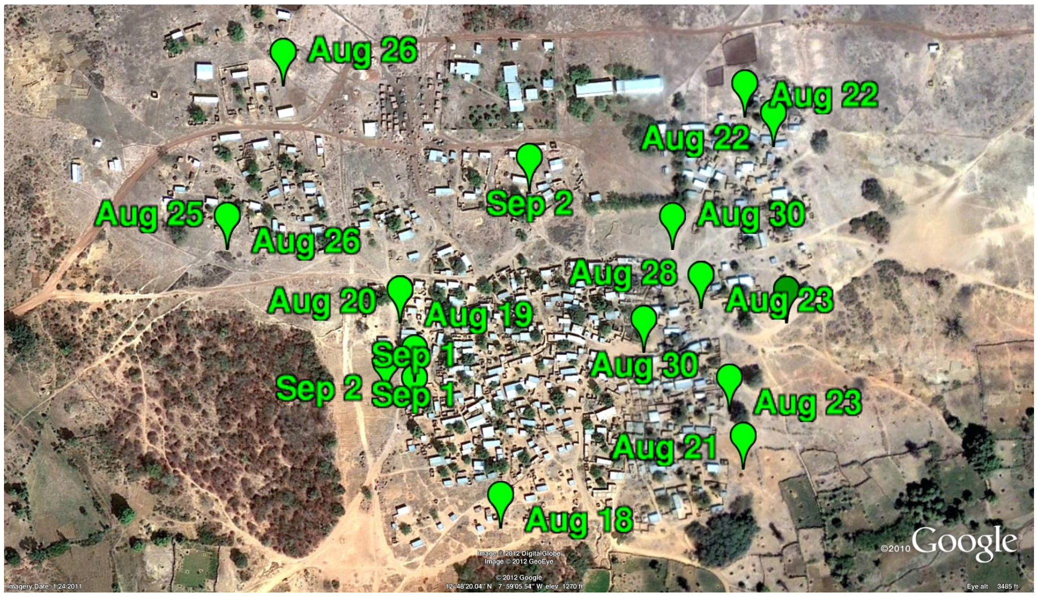


Figure 4.8: Filming locations in Mali, Africa. GPS measurements of filming locations in Mali, Africa ©2012 Google, ©2012 DigitalGlobe.

Female mosquitoes are difficult to detect and track because they fly faster than the average male (see tracking results), and appear as a faint streak much of the time. A mosquito couple is distinguishable to the human eye due to its distinct flying pattern and darker appearance against the sky. Upon spotting a couple we noted the frame number displayed on the laptop screen. The couples were located after filming for the evening was complete by manually reviewing the video footage at the designated frame. Out of the two mating mosquitoes, the female mosquito was identified as the mosquito that entered the swarm last by playing the sequence backwards and tracking the pair, first as a couple and then individually. Parameter values used for data collection and tracking are described below and in Table 4.2. The evaluation of tracking performance and validation follow.

4.7 Performance and Validation

The camera baseline b, i.e., the distance between cameras, affects the disparity Δu in pixel positions of an object in a stereo camera setup [85]. A large disparity reduces uncertainty along the camera axis, which in turn improves accuracy as well as the ability to resolve occlusions. For a stereo-camera configuration with focal length f and no vertical offset between centers, the baseline and disparity are related according to $\Delta u = (bf)/z$ [85], where z is the distance along the camera axis of the target from the stereo setup. The overlap between camera views is $(I_w - \Delta u)/I_w$, where I_w is the image width resolution in pixels. A large overlap is desirable for maximum coverage. Since the majority of swarms were filmed with $1.5 \leq z \leq 2.5$ m, we selected a baseline of 20 cm to achieve 80–90% overlap and 3–5 pixel difference between two mosquitoes that are 3 cm apart (approx. 2 body lengths) along the camera axis.

In addition to the intensity threshold t_{int} described in Section 4.3, foreground segmentation requires setting the sliding window t_{win} and a threshold on area of the blobs t_{area}. We selected $t_{win} = 7$ frames centered on the current frame, although swarms filmed at short ranges required a sliding window in the range of 3–5. We computed the area-threshold limits $20 \leq t_{area} \leq 150$ from several different swarms to achieve the best rejection of noise as well as large insects.

The covariance $\Sigma_{ep} = \text{diag}\{4, 4\}$ pixels^2 for endpoints was computed by manually selecting the endpoints of streaks in a random sampling of frames and comparing with the calculated value. The disturbance $\boldsymbol{w}$ for the constant-velocity model was sampled from $\mathbb{N}(0, 100 \text{ m}^2/\text{s}^4)$, whose covariance was found by fitting a normal distribution to the acceleration values of manually generated tracks.

We tested the position accuracy of our tracking system using a calibration checkerboard with squares of known dimensions by manually clicking pairs of points whose separation distance was in the range 3–40 cm. This method yielded an error of 5 ± 5 mm for 50 pairs. We also reconstructed tracks from a single swarming event on Aug. 29 using two independent stereo camera rigs. We created a common reference frame by measuring the height, azimuth and elevation of the cameras. The videos were time-synced using a laser pointer flashed at the end of the sequence. The mean distance between independent tracks of the same mosquito (200 data points) was 4.4 $\pm$ 1.3 cm, although up to 3 cm error can be attributed to the inter-frame time difference between the camera systems (caused due to delay within a single frame that was used to match the laser flash). A mosquito flying at an average speed of 1.5 m/s will cover 3 cm in 1/50th of a second.

Fig. 4.9 shows the results of using the OSPA metric (see Appendix, equation (C.1)) to compare tracks from the multi-target tracking system to the manually generated ground-truth. We tested two swarms with 10 and 20 mosquitoes, respectively. The order parameter and the cut-off parameter for computing OSPA values were set 2 and 50 mm respectively. Decomposing OSPA into position and cardinality errors shows that the average root mean square (RMS) position errors in the 10- and 20-mosquito swarms were 2.17 $\pm$ 0.58 cm and 2.3 $\pm$ 0.46 cm, respectively. Correspondingly, average absolute position errors for the 10- and 20-mosquito swarms were 1.74 $\pm$ 0.56 cm and 2.03 $\pm$ 0.47 cm, respectively. A low cardinality error was often accompanied by relatively high position error during periods when the swarm was dense, because of occlusions and false tracks. As would be expected for a stereo setup, position error was highest (44%) in the range measurement (along the camera optical axis) as compared to either of the other two dimensions. OSPA was larger for the 20-mosquito swarm, mainly due to cardinality errors. The position error is likely a consequence of image noise, which resulted in partially segmented streaks. (We mitigate this problem by filtering trajectory data using a Kalman smoother.) Average reprojection error on the images was less than 2 pixels.

The labeling error, which captures track continuity and identity swaps, was computed separately. An identity swap results in a labeling error of 2 before or after the swap in the sequence. Track fragmentation results in a labeling error of 1 after the disconnect occurs. We randomly selected 100 instances of 25 continuous frames in a swarm of 10 mosquitoes. The average labeling error (most of which was due to track fragmentation) was 2.1 $\pm$ 1.4 tracks. A simple average of track lengths across six swarms ranged between 15–25 frames corresponding to 0.6–1 s. Track fragmentation occurs due to early terminations, which can be caused by the following (see Fig. 4.10):

- Partially segmented streaks due to noise, cloudy background, and clutter. Partially segmented streaks in one frame often violate the epipolar constraint. Decreasing the intensity threshold to get full streaks adds noise to the measurements. (A possible solution that we are exploring in ongoing work is to reconstruct the streak using velocity estimates.)
- Occlusion between a tracked and untracked target. Occlusions between a tracked (known) targets and an as yet uninitialized target are not detected. The success rate of surviving such an occlusion depends on the motion of the tracked target after the occlusion. A maneuver or successive occlusion may terminate the track.

4.8 Kinematic Data of Mating Behavior

This section presents a subset of the three-dimensional trajectory data generated using the mosquito tracking system.

To create representative trajectory dataset, we selected six video sequences that contain a mating event. We call these the mating sequences. We refer to the mating mosquitoes as the female and the focal male. We also selected six other video sequences with no female present, called the male-only sequences, to produce full-length trajectories of swarming behavior. Trajectory data presented here are from swarms filmed on August 20, 21, 25, 26, 28, and 29 and September 1. Male-only sequences last between 20–35 seconds, whereas mating sequences start a few seconds prior to the detection of female within the field of view and end when the couple flies out of field of view (0–5 s).

Fig. 4.11 shows the position and velocity of a randomly selected male *An. gambiae* in the Aug. 29 male-only sequence, which was a swarm that formed over bare ground (S form). The swarm consisted of 20 mosquitoes at the beginning of the sequence and dropped to 19 after 10 seconds. The mosquito movement is characterized by quasi-periodic motion in each of the three spatial dimensions. The instantaneous mean position of the mosquitoes in the swarm, i.e., the swarm centroid, is also shown. The inertial frame (0,0,0) is located at ground level under the camera rig; the inertial frame is oriented along east-west, north-south, and vertical directions. The 3σ bounds for position and velocity of all of mosquitoes in this swarm are shown in gray. Swarm size (twice the 3σ bounds) averaged 1.17 m in the horizontal plane and 0.56 m in the vertical. The average swarm size across all planes ranged between 0.52–1.86 m. The average height of the swarm was 1.89 m. The average velocity along each dimension is close to zero with a highest standard deviation in the east-west direction (0.514 m/s) followed by the north-south (0.332 m/s) and the vertical (0.281 m/s).

Fig. 4.12 shows the ratio between horizontal and vertical speed for each swarm. The Aug. 28 sequence was filmed on a day with relatively high wind (approx. 0.8 m/s) as compared to other sequences. The mosquito movement for that swarm was characterized by a rolling motion in the direction of the wind and relatively higher vertical velocities. In five out of the six swarms that we used to generate male-only sequences, we witnessed mating events at a later time. The horizontal and vertical speeds of female mosquito that formed couples are also plotted in Fig. 4.12. Non-parametric Kruskal-Wallis tests on each dataset show that the average male and female speeds in the same sequence are significantly different for each sequence. The maximum p-value among all mating sequences was 0.0003. (In contrast the maximum p-value for male speeds during the same mating sequence was 0.051.)

Fig. 4.13 shows the position and velocity of a female mosquito that formed a successful couple in the Aug. 29 sequence. The mating sequence was filmed about a minute after the male-only sequence on the same date. The female appeared in the field of view 5 seconds prior to couping. The movement of the female crosses the 3σ boundaries of the swarm in the north-south dimension. The average speed of the female was higher than the male mosquito until just before the couple forms, when the focal male speeds up. The vertical movement shows that the female stayed in the lower half of the swarm. A three-dimensional reconstruction of the mating mosquitoes in six mating sequences is shown in Fig. 4.14. Across all mating sequences, the female mosquito covered an average 59% more distance than the focal male during the same time interval.

Fig. 4.15 shows the separation distance and speeds from six mating sequences. The amount of time we observed the females in the swarm before forming a couple was up to 5 seconds. In each mating sequence that lasted longer than 0.5 seconds, the number of close encounters (moments when the separation distance between the mating mosquitoes dropped below 3 body lengths, or 4 cm) with the successful male mosquito was in the range 3–6.

Table 4.1: **Mosquito Tracking Algorithm**

Input:	Sequence of synced images from a stereo-camera setup, camera calibration matrices, parameters in Table 4.2
Output:	Estimated three-dimensional mosquito trajectories

For each time step k:

1: *Extract measurements:* Model each blob as a straight line and find the midpoint and endpoints.

2: *Find missing measurements, if any:* Ensure that each hypothesized target has at least one measurement within the gating volume; if not, lower the intensity threshold. If a measurement is found append it to the existing set of measurements.

3: *Validate:* Use the epipolar constraint (2.12) to generate valid measurement pairs, one from each camera view.

4: *Cluster:* Use gating volume of each target within a cluster to add measurements to that cluster. A cluster is the smallest set of measurements and targets that exist independently; combine/divide existing clusters as needed.

5: *Compute hypotheses:* Generate hypotheses for each cluster, and compute probabilities.

$\hookrightarrow$ *Resolve occlusions:* If an occlusion is detected split the image blob into individual streaks as described in §III-C and recompute the hypotheses.

6: *Hypothesis reduction:* Based on the most probable hypothesis at k and scanback range N_s, reduce the number of hypotheses at $k - N_s$ to a single assignment.

7: *Initialize & update:* Initialize tentative targets from unassociated measurement pairs; resample target states based on hypotheses using the three-dimensional estimate and velocity likelihood function (4.5). Each new target forms a new cluster.

8: *Predict:* Use the constant velocity motion model with random (Gaussian) disturbance to propagate hypotheses to timestep $k + 1$.

Table 4.2: Parameter Values Tracking Mosquitoes

Parameter	Value	Description
b	20 cm	Stereo camera configuration baseline
t_{win}	7 frames	Sliding window for segmentation
Σ_{ep}	diag{4,4} pixels^2	Covariance of endpoint error
σ_w	100 m^2/s^4	Covariance of disturbance
δt_e	25 ms	Duration of camera exposure
t_{gate}	16	Threshold for gating volume
t_e	.5	Threshold on epipolar constraint
t_{area}	(20, 150)	Minimum and maximum blob areas
N_s	1 frame	Scanback for MHT
N_p	200	Number of samples in particle filter

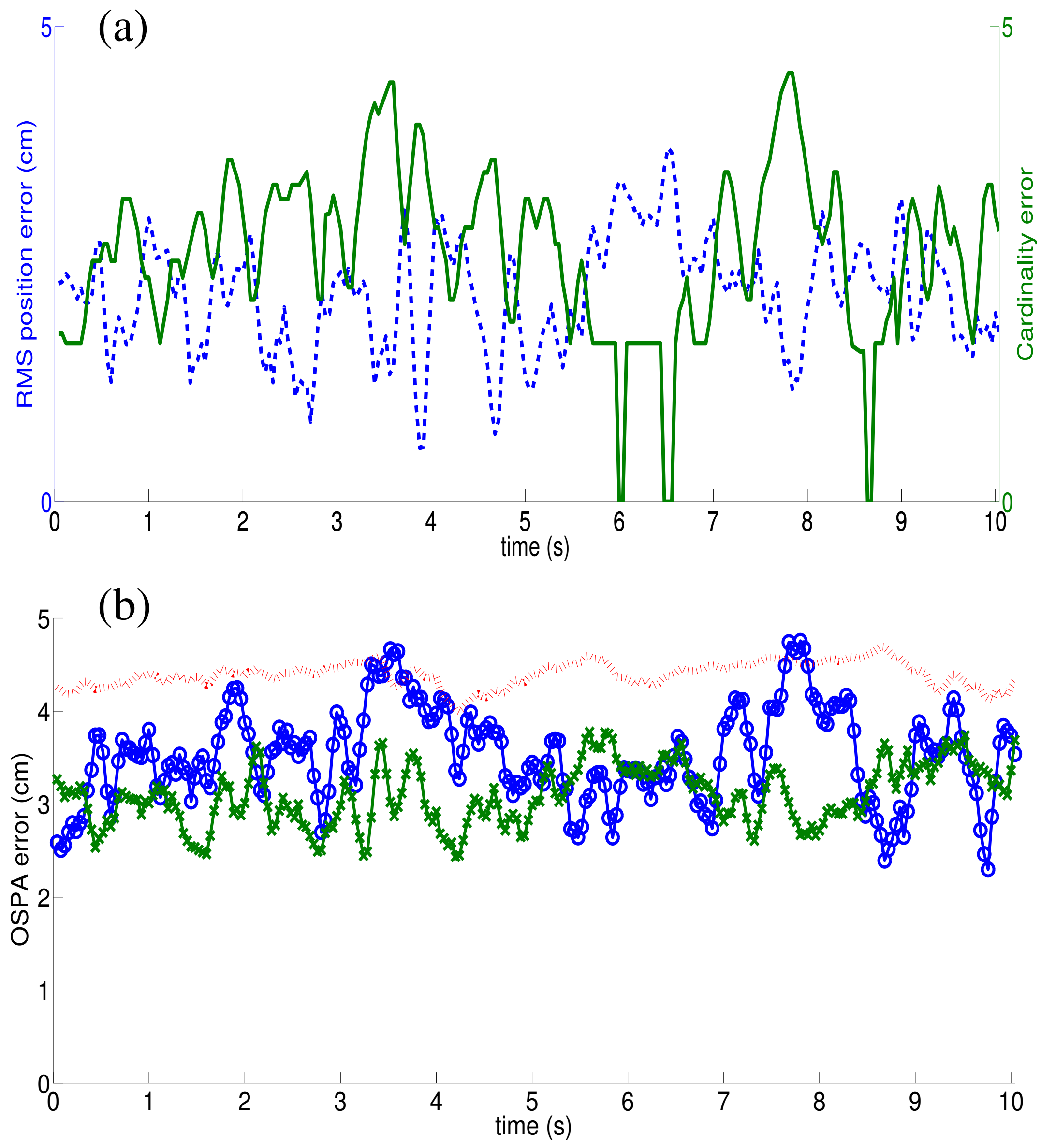


Figure 4.9: Tracking performance. (a) Position (dashed blue) and cardinality error (solid green) for a swarm of 10 mosquitoes. (b) OSPA error for different methods and swarm sizes: nearest neighbor [36] (dotted) for a swarm of 20 mosquitoes and single-scan MHT for two swarms of sizes 10 (crosses) and 20 (circles), respectively.

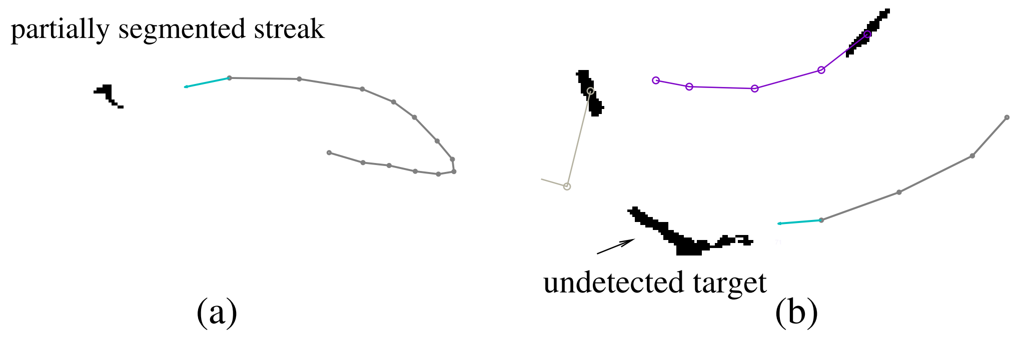


Figure 4.10: Track terminations. Track terminations due to (a) partially segmented streak and (b) occlusion with a yet undetected target. The predicted velocity of the terminated target is shown in cyan.

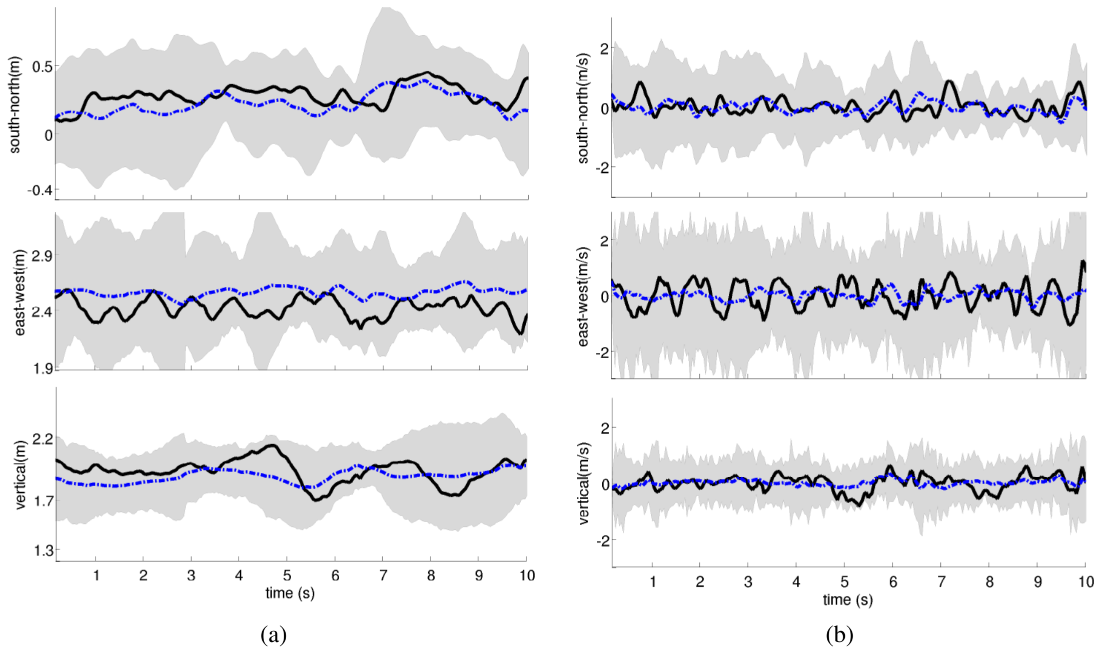


Figure 4.11: 3D position and velocity of a single male mosquito. Instantaneous three-dimensional (a) position and (b) velocity tracks of a male mosquito (black solid) in the Aug. 29 sequence (S molecular form). Mean (dotted blue) and 3 σ bounds (gray) for all mosquitoes also shown.

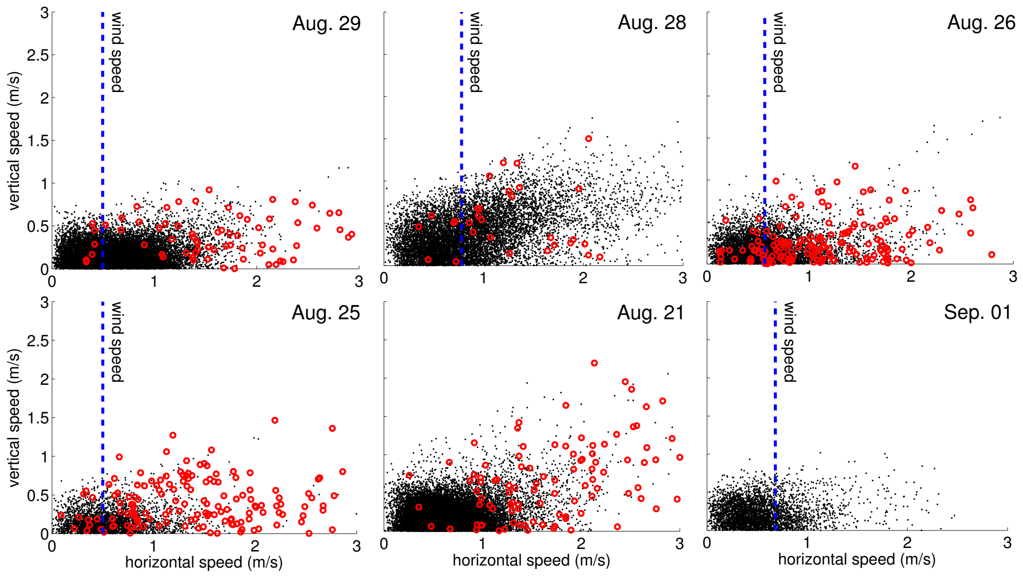


Figure 4.12: Horizontal versus vertical speeds of all mosquitoes. Instantaneous horizontal versus vertical speed of male mosquitoes in six sequences (black dots). Female speeds (red circles) and average wind speed (blue dashed lines) are shown if available.

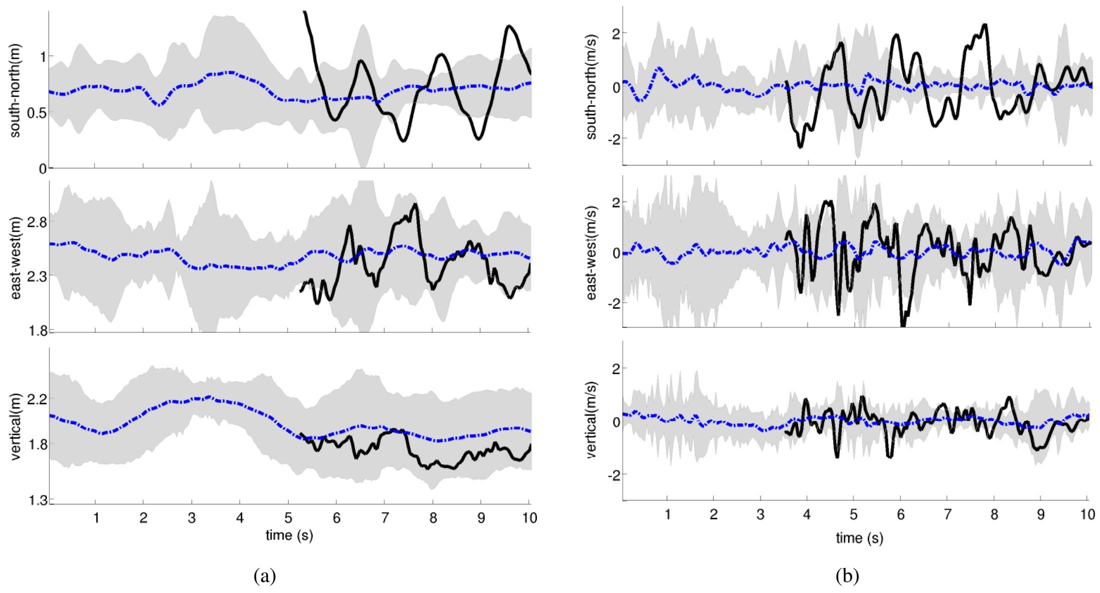


Figure 4.13: 3D position and velocity of a single female mosquito. Instantaneous three-dimensional position (a) and velocity (b) of a female mosquito that coupled in the Aug. 29 sequence filmed 70 seconds later than the male position and velocity shown in Fig. 4.11. Mean (dotted blue) and 3σ bounds (gray) of all males are also shown.

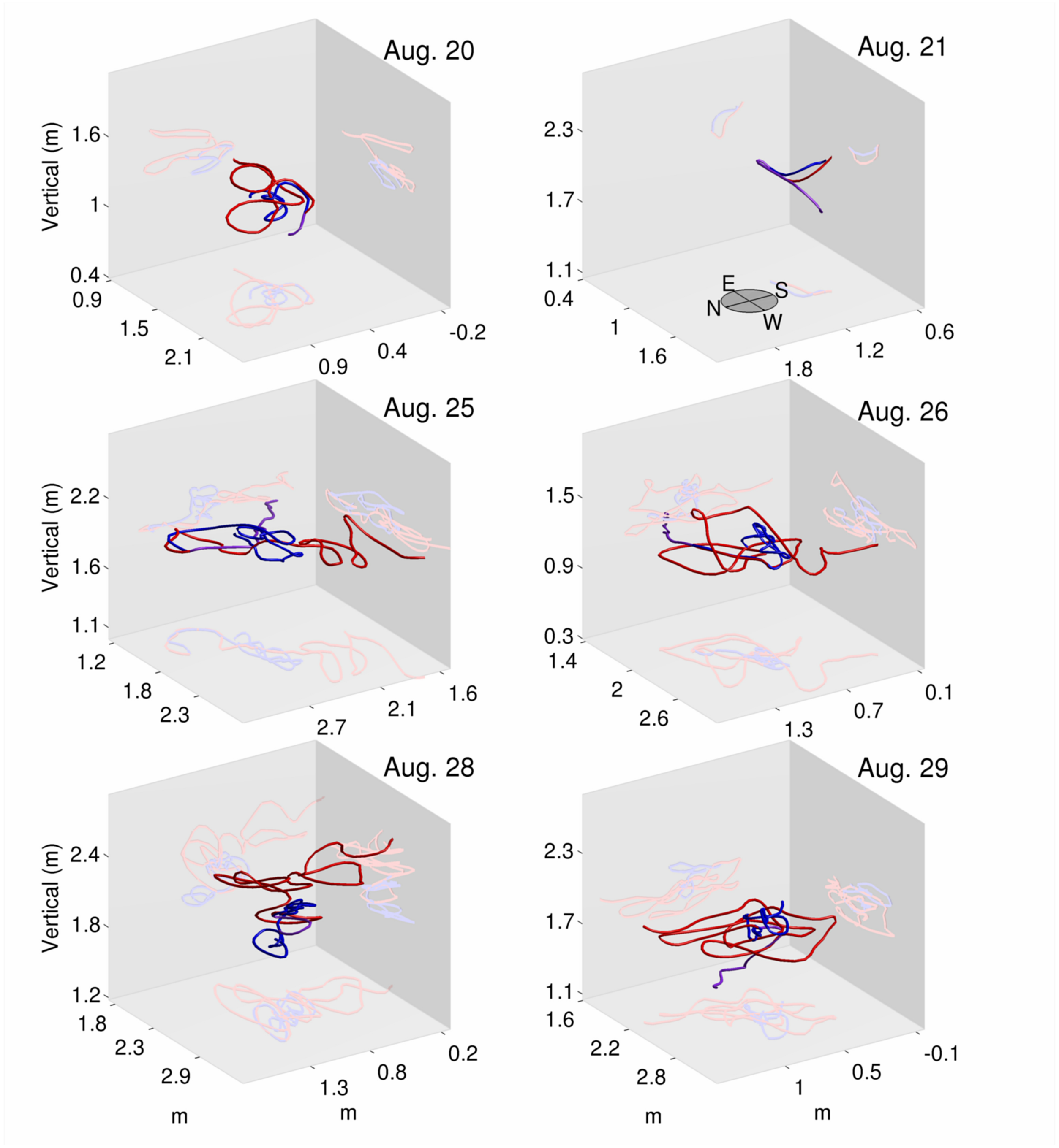


Figure 4.14: 3D reconstruction of mosquito mating events. Three-dimensional reconstruction of the *An. gambiae* mating events in the wild. The female mosquito track (red) and male mosquito track (blue) are shown. The couple is shown in purple. Pre-coupling tracks are projected on to two-dimensional planes on each side. The magnetic direction for all swarms is the same (shown in Aug. 21 mating sequence).

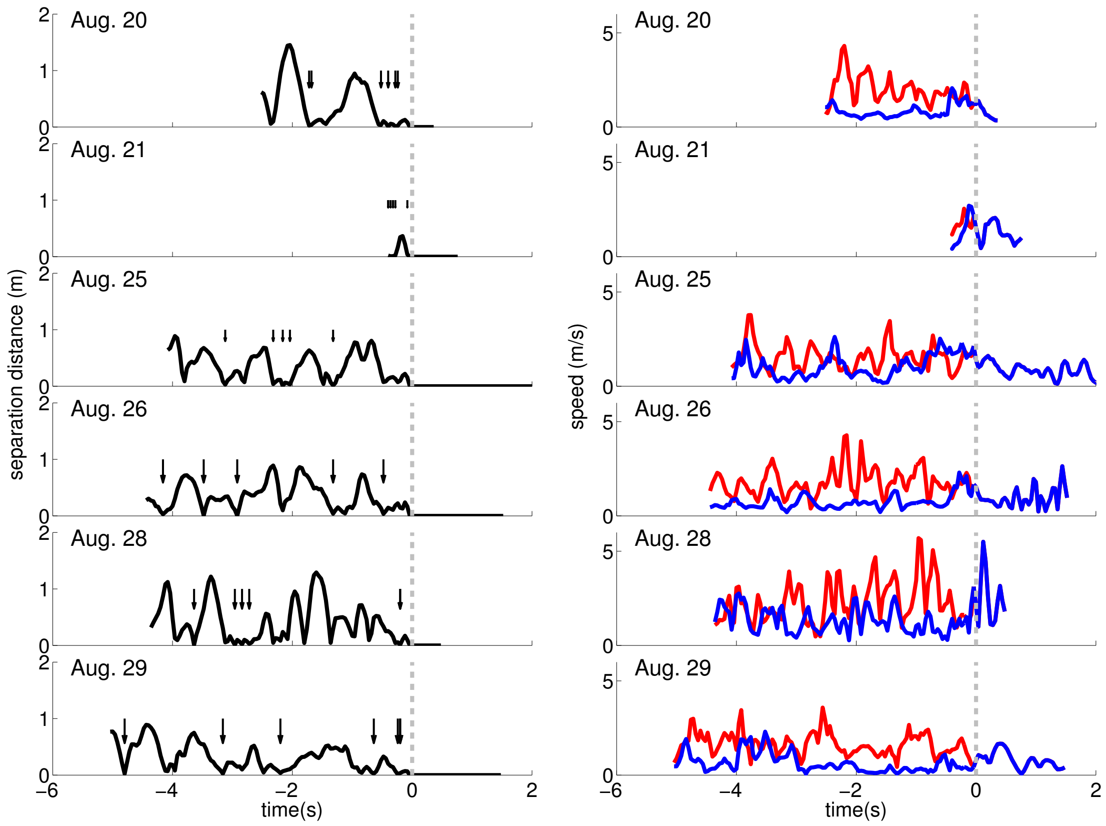


Figure 4.15: Distance and speeds of mating mosquitoes. Relative distance (a) and speeds (b) of mating male and female *An. gambiae* mosquitoes (a) in six mating sequences. Time 0 s occurs when the separation distance first drops below 2 cm. Arrows depict close encounters (separation distance less than 4 cm).

Chapter 5

Conclusion

5.1 Summary

In this dissertation we describe multi-target tracking systems for studying fish schools and mosquito swarms. Together these systems span variability in target density, extent on image, and movement. The tracking systems are constructed on a Bayesian framework and are adapted to address occlusions and missed detections. In each case, the research has yielded an unprecedented volume of trajectory data for subsequent analysis.

The input to the fish tracking system is a sequence of synchronized high-speed video images from a top-view and a side-view camera. We automatically initialize the tracker so that high-volume datasets may be generated without the need to manually set a prior. In a multi-layered framework, we use simulated annealing (SA) to reconstruct fish shape through occlusions and smooth the shape trajectories using a Kalman filter. We provide a perturb function to inform the local search and improve finite time behavior of the SA algorithm. Performance is evaluated using an independent camera view, as well as manually generated ground truth. The tracking system is validated on schools of up to eight fish in a 37.8 liters (10 gallon) tank and is currently in use to study information transmission in fish schools.

The mosquito tracking system is designed to address noisy, low frame-rate (25 frames per second) video streams from a stereo camera system. We provide an adaptive algorithm to search for faded mosquito streaks across the image plane and a likelihood function that utilizes streak endpoints to extract velocity information. A modified multi-hypothesis tracker probabilistically addresses occlusions and a particle filter estimates the trajectories. The output of the tracking algorithm is a set of track segments with an average length of 0.6–1 seconds. The segments are verified and combined under human supervision to create individual tracks up to the duration of the video (90s). We evaluate tracking performance using an established metric for multi-target tracking and validate the accuracy using independent stereo measurements of a single swarm. Three-dimensional reconstructions of *An. gambiae* swarming and mating events are presented. In ongoing work, we are investigating these trajectories to characterize swarming and mating behavior.

5.2 Suggestions for Future Research

The ability to track multiple animals in a group can be extended in at least two different directions: (a) interactive experiments with animal groups, and (b) higher level interaction models that assist and inform the tracking algorithm

5.2.1 Interactive Experiments with Animal Groups

A desired goal would be to interact with animal groups as they are filmed, possibly by subjecting a single individual or a subgroup to external stimuli. For biologists interactive experiments where they are able to modify the animal's environment and study its response will allow quicker analysis of different decision making strategies. For engineers interactive systems close the loop for studying response of an animal group to specific inputs for system identification. The stimulus may be presented in the form of an artificial robot that interacts with the group [87, 124] or on a computer screen in a virtual reality setup [48, 34]. To be able to interact with an animal we must track its position and orientation in real-time.

The primary design challenge in performing interactive experiments is to be able to close the loop on target tracking by using the estimates to drive the manipulation of the group. The tracking systems described in this dissertation are post-filming tools, because the computational time for processing each image and tracking each target is more than the frame rate. At the same time, components of the tracking system such as automatic head and nose detection and subsequent three-dimensional localization that we use to estimate the midline can be used for tracking fish pose in real time [48].

In ongoing work [48], we design a virtual-reality framework for investigating startle-response behavior in a single fish. Using real-time three-dimensional tracking at 15 frames per second, we generate looming stimuli at a specific location on a computer screen, such that the shape and size of the looming stimuli change according to the fish's perspective and location in the tank. To attain real-time speeds we use a Microsoft^{TM} Cinema Lifecam at low resolution of 640 $\times$ 480 pixels mounted above a fish tank looking straight below. Computational load is further reduced by obtaining the second perspective through a mirror mounted at an angle to the side of the tank. Fish pose and velocity were validated at each time step to ensure that (1) the fish heading and direction of motion was aligned (2) the fish was moving at a speed less than 150 cm/s (3) the fish was more than one body length away from the tank walls. We demonstrate the effectiveness of the setup through experiments on Giant danio and compute the success rate in eliciting a startle response by presenting the stimulus to 39 individual fish from different orientations. Fig. 5.1 shows a schematic of the test environment.

Extending this setup to multiple fish will again be driven by the requirements. An experiment where the stimulus is presented based on the fish school properties may not require maintaining long trajectories of each fish. In the case where longer trajectories are needed we must resort to occlusion resolution. Although shape information helps in resolving occlusions, detailed shape required for analysis may not be needed until after the experiment. (In our virtual reality setup, for example, we record with a high-speed camera in parallel for later analysis.) To resolve occlusions we may track shape selectively with a lower resolution model such as Gaussian mixtures [87]. Fig. 5.2 shows two such attempts on existing datasets. We see that although accurate shape information is difficult to extract, it is possible to determine position and orientation of relatively straight fish.

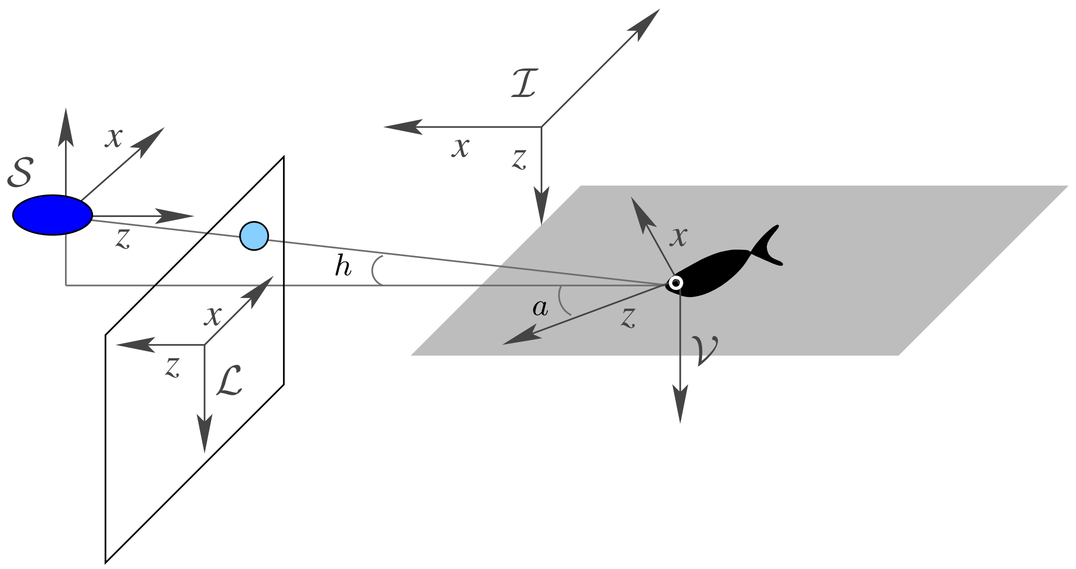


Figure 5.1: Schematic of the virtual reality setup. Reference frames $\mathcal{V}$ (viewer), $\mathcal{S}$ (stimulus), $\mathcal{L}$ (screen) and $\mathcal{I}$ (inertial). The azimuth (a) and elevation (h) of the stimulus with respect to the fish are also shown.

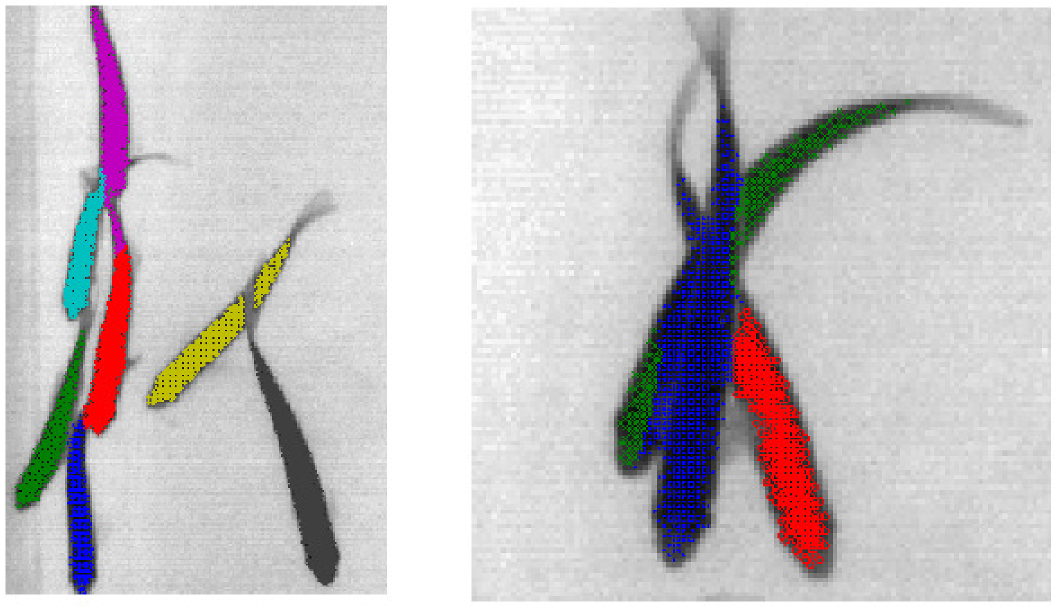


Figure 5.2: Resolving fish occlusions modeled as Gaussian mixtures. We cluster fish occlusions modeled as Gaussian mixtures using an expectation minimization algorithm. It took 0.4 seconds to cluster seven fish (left) and 0.1 seconds to cluster three fish (right).

5.2.2 Interacting Motion Models for Robust Tracking

In Chapter 1 we motivate the need to quantify trajectory data in order to validate models of collective behavior. In turn, a dynamic model of collective behavior can be used to improve tracking performance by predicting individual motion in response to other members of the group. The general idea of using higher-order models for an improved tracking performance is not new: In [125] a Markov Random Field (MRF) motion prior is used to model interactions between multiple ants. A particle filter with MCMC sampling tracks the joint space of of multiple targets. They identify the failure modes when a unique behavior—overlapping targets due to carrying behavior in ants—occurs and suggest incorporating the same into the tracker. In [96], bees are tracked using a multi-scale Markov model to track waggle dance in honeybees. Simple behaviors such as straight motion, turns, waggle, and staying still are learnt and then incorporated into the tracker to obtain improved performance. Each behavior is independent and the authors suggest incorporating transition between behaviors into the model.

The first step to incorporate such models into tracking would be to characterize and detect behaviors. This falls into the general category of activity recognition in humans [126, 127] and flies [31, 128]. In fish, for example, we characterize fast-start behavior in Section 3.7 by a sudden change in body shape, position and orientation. Using this information and relative position of the fish in the school, the shape kinematics of the first responder can be used to predict the motion of others. Such behaviors can also be used to selectively assign tracking resources to different parts of the image. For example, the tracking system may switch between a lower-resolution coarse shape description and high-resolution shape description based on activity cues that are of interest to the researcher.

Existing models of collective behavior can also be used to assist and improve tracking accuracy. For example, it has been shown previously in swarming that each insect interacts with the rest of the swarm in the form of an attractive force towards the centroid and a drag force due to its speed. Kinematic reconstructions of mosquitoes in Section 4.8 show that every mosquito tends to move in and out of the swarm in a periodic pattern. Such motion is characteristic of grouping behavior which can be characterized in terms of velocity autocorrelation [129, 80]. Further analysis of trajectory data in mosquitoes can motivate simple models that adaptively increase acceleration noise based on position relative to the centroid. Going further, once an interaction topology or a time-dependent model thereof is established, a model that would include the positions of other mosquitoes within an interacting distance may be used in the prediction step of the tracking system.

Appendix A

Linearizing Shape Estimation Measurement Model

We linearize the shape-estimation measurement model for the purpose of using in an extended Kalman filter. The measurement model defined in (3.10) is

$$ {}^cH_{\mathcal{E}}(\mathcal{E}(s)) = \begin{pmatrix} {}^ce_1(s) \\ {}^ce_2(s) \end{pmatrix} = \begin{pmatrix} \frac{\|{}^c\mathcal{S}(s,{}^cv_1) - {}^c\mathcal{S}(s,{}^cv_2)\|}{2} \\ \left\| \frac{{}^c\mathcal{S}(s,{}^cv_1) + {}^c\mathcal{S}(s,{}^cv_2)}{2} - {}^c\hat{\boldsymbol{u}}(s) \right\| \end{pmatrix} \tag{A.1} $$

where ${}^c\mathcal{S}(s,v)$ is a point $\mathcal{S}(s,v)$ on elliptical cross section as defined in (3.9) at the midline s and angle v projected on camera 1. The first order approximation of ∂H_s is the Jacobian of H_s evaluated at the current estimate $\hat{\mathcal{E}}(s)$ and is given by (we drop the s dependency for clarity)

$$ \frac{\partial H_{\mathcal{E}}}{\partial \mathcal{E}} = \begin{pmatrix} \frac{\partial e_1}{\partial a} & \frac{\partial e_1}{\partial b} & \frac{\partial e_1}{\partial d} \\ \frac{\partial e_2}{\partial a} & \frac{\partial e_2}{\partial b} & \frac{\partial e_2}{\partial d} \end{pmatrix}. \tag{A.2} $$

where, each element is given by

$$ \begin{aligned} \frac{\partial H_s}{\partial \mathcal{E}}(1,1) = \frac{\partial e_1}{\partial a} &= \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial a} + \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial a} \\ &= \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial \mathcal{S}_{v_1}} \frac{\partial S_{v_1}}{\partial a} + \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial \mathcal{S}_{v_2}} \frac{\partial S_{v_2}}{\partial a} \\ \frac{\partial H_s}{\partial \mathcal{E}}(1,2) = \frac{\partial e_1}{\partial b} &= \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial \mathcal{S}_{v_1}} \frac{\partial S_{v_1}}{\partial b} + \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial \mathcal{S}_{v_2}} \frac{\partial S_{v_2}}{\partial b} \\ \frac{\partial H_s}{\partial \mathcal{E}}(1,3) = \frac{\partial e_1}{\partial d} &= \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial \mathcal{S}_{v_1}} \frac{\partial S_{v_1}}{\partial d} + \frac{\partial e_1}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial \mathcal{S}_{v_2}} \frac{\partial S_{v_2}}{\partial d}. \\ \frac{\partial H_s}{\partial \mathcal{E}}(2,1) = \frac{\partial e_2}{\partial a} &= \frac{\partial e_2}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial \mathcal{S}_{v_1}} \frac{\partial S_{v_1}}{\partial a} + \frac{\partial e_2}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial \mathcal{S}_{v_2}} \frac{\partial S_{v_2}}{\partial a} \\ \frac{\partial H_s}{\partial \mathcal{E}}(2,2) = \frac{\partial e_2}{\partial b} &= \frac{\partial e_2}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial \mathcal{S}_{v_1}} \frac{\partial S_{v_1}}{\partial b} + \frac{\partial e_2}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial \mathcal{S}_{v_2}} \frac{\partial S_{v_2}}{\partial b} \\ \frac{\partial H_s}{\partial \mathcal{E}}(2,3) = \frac{\partial e_2}{\partial d} &= \frac{\partial e_2}{\partial^1 \mathcal{S}_{v_1}} \frac{\partial^1 \mathcal{S}_{v_1}}{\partial \mathcal{S}_{v_1}} \frac{\partial S_{v_1}}{\partial d} + \frac{\partial e_2}{\partial^1 \mathcal{S}_{v_2}} \frac{\partial^1 \mathcal{S}_{v_2}}{\partial \mathcal{S}_{v_2}} \frac{\partial S_{v_2}}{\partial d} \end{aligned} \tag{A.3} $$

Using the above relations the Jacobian can be written as

$$
\begin{aligned}
\frac{\partial H_{\mathcal{E}}}{\partial \mathcal{E}} &= \frac{\partial \boldsymbol{e}}{\partial {}^1\mathcal{S}_v}\frac{\partial {}^1\mathcal{S}_v}{\partial \mathcal{S}_v}\frac{\partial S_v}{\partial \mathcal{E}} \\
&= \begin{pmatrix} \frac{\partial e_1}{\partial {}^1\mathcal{S}_{v_1}} & \frac{\partial e_1}{\partial {}^1\mathcal{S}_{v_2}} \\ \frac{\partial e_2}{\partial {}^1\mathcal{S}_{v_1}} & \frac{\partial e_2}{\partial {}^1\mathcal{S}_{v_2}} \end{pmatrix} \begin{pmatrix} \frac{\partial {}^1\mathcal{S}_{v1}}{\partial \mathcal{S}_{v1}} & \mathbf{0}_{2\times 3} \\ \frac{\partial {}^1\mathcal{S}_{v2}}{\partial \mathcal{S}_{v2}} & \mathbf{0}_{2\times 3} \end{pmatrix} \begin{pmatrix} \frac{\partial S_{v1}}{\partial \mathcal{E}} \\ \frac{\partial S_{v2}}{\partial \mathcal{E}} \end{pmatrix},
\end{aligned}
\tag{A.4}
$$

where $\mathbf{0}_{2\times3}$ is a 2×3 zero matrix. The partial derivatives are computed by using the chain rule. These are $\frac{\partial \boldsymbol{e}}{\partial {}^1\mathcal{S}_v}$ which is a 2×4 matrix relating the measurements $\boldsymbol{e} \in \mathbb{R}^2$ to the projection of two surface end points given by

$$
\begin{aligned}
\frac{\partial e_1}{\partial {}^1\mathcal{S}_{v1}} &= \begin{pmatrix} \frac{\partial e_1}{\partial {}^1\mathcal{S}_{v1,1}} & \frac{\partial e_1}{\partial {}^1\mathcal{S}_{v1,2}} \end{pmatrix} \\
&= \begin{pmatrix} \frac{({}^1\mathcal{S}_{v1,1} - {}^1\mathcal{S}_{v2,1})}{4e_1} & \frac{({}^1\mathcal{S}_{v1,2} - {}^1\mathcal{S}_{v2,2})}{4e_1} \end{pmatrix}; \\
\frac{\partial e_2}{\partial {}^1\mathcal{S}_{v1}} &= \begin{pmatrix} \frac{\partial e_2}{\partial {}^1\mathcal{S}_{v1,1}} & \frac{\partial e_2}{\partial {}^1\mathcal{S}_{v1,2}} \end{pmatrix} \\
&= \begin{pmatrix} \frac{({}^1\mathcal{S}_{v1,1} + {}^1\mathcal{S}_{v2,1})}{4e_1} & \frac{({}^1\mathcal{S}_{v1,2} + {}^1\mathcal{S}_{v2,2})}{4e_1} \end{pmatrix};
\end{aligned}
\tag{A.5}
$$

the perspective projection model $\frac{\partial {}^1\mathcal{S}_v}{\partial \mathcal{S}_v}$, which is a 4×6 matrix relating the three dimensional surface endpoints to their projections on the image. The top 2×3 matrix is

$$
\frac{\partial {}^1\mathcal{S}_{v1}}{\partial \mathcal{S}_{v1}} = \begin{pmatrix} \frac{\partial {}^1\mathcal{S}_{v,1}}{\partial \mathcal{S}_{v,1}} & \frac{\partial {}^1\mathcal{S}_{v,1}}{\partial \mathcal{S}_{v,2}} & \frac{\partial {}^1\mathcal{S}_{v,1}}{\partial \mathcal{S}_{v,3}} \\ \frac{\partial {}^1\mathcal{S}_{v,2}}{\partial \mathcal{S}_{v,1}} & \frac{\partial {}^1\mathcal{S}_{v,2}}{\partial \mathcal{S}_{v,2}} & \frac{\partial {}^1\mathcal{S}_{v,2}}{\partial \mathcal{S}_{v,3}} \end{pmatrix},
\tag{A.6}
$$

where each element in the partial derivative $\frac{\partial {}^1\mathcal{S}_{v1}}{\partial \mathcal{S}_{v1}}$ is computed using the distortion-free perspective projection model (B.1). Using $\boldsymbol{u} \in \mathbb{R}^2$ to represent the camera measurement of the three-dimensional point $\boldsymbol{r} \in \mathbb{R}^3$ ($\boldsymbol{u}, \boldsymbol{r}$ correspond to ${}^1\mathcal{S}_{v1}, \mathcal{S}_{v1}$ respectively).

$$
\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{r}} = K_{2\times2} \begin{pmatrix} \frac{1}{{}^1r_3} & 0 & -\frac{{}^1r_1}{{}^1r_3^2} \\ 0 & \frac{1}{{}^1r_3} & -\frac{{}^1r_2}{{}^1r_3^2} \end{pmatrix} R,
\tag{A.7}
$$

where $K_{2\times2}$ is first two rows and columns of the camera matrix $K \in \mathbb{R}^{3\times3}$, $R \in \mathbb{R}^{3\times3}$ is the rotation matrix, and ${}^1r_i$ is the transformed position coordinate in camera frame 1; and the generative model relation $\frac{\partial S_v}{\partial \mathcal{E}}$, which is a 6×3 matrix connecting the ellipse parameters to the the end points. The top 3×3 matrix is

$$\begin{aligned}\frac{\partial S_{v1}}{\partial \mathcal{E}} &= \begin{pmatrix} \frac{\partial S_{v1}}{\partial a} & \frac{\partial S_{v1}}{\partial b} & \frac{\partial S_{v1}}{\partial d} \end{pmatrix} \\ &= \begin{pmatrix} x_1 \cos(v1) & y_1 \sin(v1) & y_1 \\ x_2 \cos(v1) & y_2 \sin(v1) & y_2 \\ x_3 \cos(v1) & y_3 \sin(v1) & y_3 \end{pmatrix}.\end{aligned} \tag{A.8}$$

The first order series expansion of the nonlinear measurement model described in this section is decoupled at every s, which allows running a separate filter for each point on the midline.

Appendix B

Multiple-view Camera Geometry

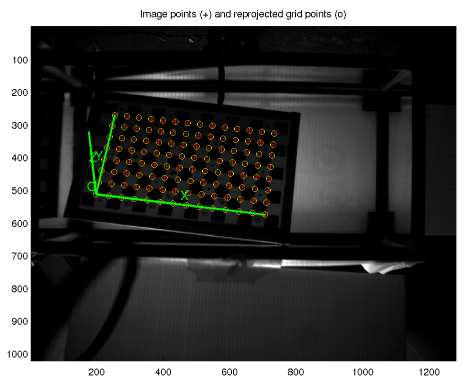
Camera Calibration

We use perspective projection to model a camera. Let $\boldsymbol{r} \in \mathbb{R}^3$ be a three-dimensional point mapped to a two-dimensional pixel position $\boldsymbol{u} \in \mathbb{R}^2$. Using the camera's intrinsic parameters called focal length f, principal point $[u_0, v_0]^T$, and distortion coefficients $\{k_1, k_2\}$, and the extrinsic parameters position $\boldsymbol{t} \in \mathbb{R}^3$ and orientation $R \in SO3$ the three-dimensional position and it's distortion-free pixel image are related by a nonlinear mapping [102]

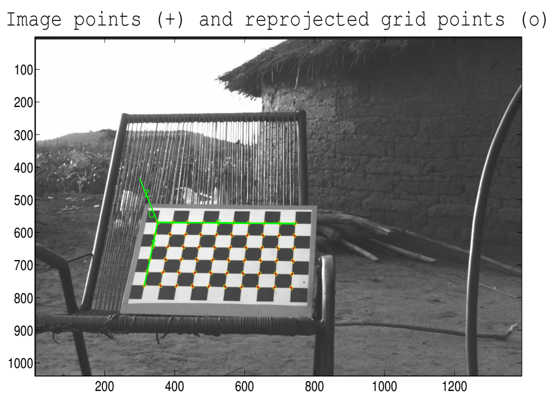
$$s\tilde{\boldsymbol{u}} = K \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix} \tilde{\boldsymbol{r}}, \tag{B.1}$$

where K is the 3×3 camera matrix, s is a scaling factor and $\tilde{\boldsymbol{u}} = [\boldsymbol{u}^T, 1]^T$ denotes the homogeneous representation. The 3×4 matrix $P = K \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix}$ is called the projection matrix [85]. Solving the camera calibration problem requires estimating the values of $R, \boldsymbol{t}$ and $\boldsymbol{r}$ given a set of points $\boldsymbol{u}_i$. This is a under-determined nonlinear optimization problem that requires additional constraints in order to be solved. In addition, for a multi-view setup the values of $R, \boldsymbol{t}$ must be transformed to a common inertial frame.

In [130] these values are found by performing an optimization over a large number of points imposing constraints in the form of known positions of each point. This method has been utilized in calibrating cameras that cover a large volume [47, 131]. The multi-camera self-calibration method in [132] uses image sequence of a single light source in a dark environment to automatically calibrate each camera in a multi-view setup. Constraints are implemented in the form of nearness of points in successive images in a sequence. At least three cameras are required for



(a)



(b)

Figure B.1: Camera calibration. The checkerboard frame represents the position and orientation of the checkerboard in each camera. We use the extrinsic calibration from two views to define a new inertial frame.

this technique to work. The MATLAB calibration toolbox [112] which is based on the technique described in [102] uses multiple images of a planar checkerboard to accurately calibrate the camera parameters. Homography constraints based on planar set of points are used to solve the optimization problem.

For all of our setups we used the MATLAB calibration toolbox. The toolbox comes with features that output the calibration error as well as distortion coefficients giving an estimate of the cost of ignoring distortion in the measurement model. The toolbox can be easily extended for a multi-view setup by placing the checkerboard for extrinsic calibration in full view of all cameras [39, 36]. Fig. B.1 shows the extrinsic calibration frame projected onto the checkerboard for both setups. When this is not possible, the checkerboard may be waved such that it is in at least two camera views at all times. Extrinsic calibration is then performed by propagating the extrinsic parameters between overlapping cameras until all camera positions and orientations are known with respect to a common inertial frame [38]. A world frame may be extracted by locating the checkerboard corner with respect to ground and magnetic direction [36].

The refraction error in our experiments is small because of the relatively small working volume [133].

Appendix C

Optimal Subpattern Assignment Metric

Consider the joint target estimate $(\boldsymbol{X}^1[k], \boldsymbol{X}^2[k])$ at timestep k from datasets 1 and 2 in which the number of targets is m and n, respectively. We denote the distance between estimates $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ by $d^c(\boldsymbol{X}_i, \boldsymbol{X}_j) \triangleq \min(c, d(\boldsymbol{X}_i^1, \boldsymbol{X}_j^2))$, where $c > 0$ is the cut-off distance that determines the penalty on cardinality errors and $d(\boldsymbol{X}_i, \boldsymbol{X}_j)$ is the Euclidean distance between $\boldsymbol{X}_i$, and $\boldsymbol{X}_j$. Let Π_n denotes the set of permutations of $\{1, \ldots, n\}$ and $\pi(i)$ denotes the element of π assigned to i that achieves the global minimum mean-square error, such as obtained using the Hungarian method [86]. The OSPA metric $d_p^c(\boldsymbol{X}^1[k], \boldsymbol{X}^2[k])$ at timestep k is [134]

$$\begin{aligned} &d_p^c(\boldsymbol{X}^1[k], \boldsymbol{X}^2[k]) = \\ &\left(\frac{1}{n} \Big(\min_{\pi \in \Pi_n} \sum_{i=1}^{m} d^c(\boldsymbol{X}_i^1[k], \boldsymbol{X}_{\pi(i)}^2[k])^p + c^p(n-m) \Big) \right)^{1/p}. \end{aligned} \tag{C.1}$$

The OSPA metric consists of two parameters that weight the cardinality and position accuracy in a multi-target tracking system. These are the cut-off parameter c, which determines the penalty on cardinality mismatch, and the order parameter p, which penalizes outliers. A high value of c results in mis-assignments construed as large position errors and should only be used if such errors are expected. A modification to the OSPA metric [135] includes a measure of labeling error to capture track-fragmentation and identity swaps. To compute the modified metric let $d^c(\boldsymbol{X}_i, \boldsymbol{X}_j) \triangleq d(\boldsymbol{X}_i, \boldsymbol{X}_j) + l\bar{\delta}(i,j)$, where $\bar{\delta}(i,j)$ is the Kronecker delta function and $l \in [0, c]$ is the penalty on labeling error. Track labels are assigned in a global optimum sense using the Hungarian method [86] to estimated tracks that are closest to the ground truth.

Bibliography

[1] J. K. Parrish and W. M. Hammer, *Animal groups in three dimensions.* Cambridge University Press, 1997.

[2] N. I. Hristov, T. L. Hedrick, T. H. Kunz, and M. Betke, “Tracking a large number of objects from multiple views,” in *IEEE 12th International Conference on Computer Vision.* IEEE, Sep. 2009, pp. 1546–1553.

[3] A. Poore, “Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking,” *Computational Optimization and Applications*, vol. 3, no. 1, pp. 27–57, 1994.

[4] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, “Novel Type of Phase Transition in a System of Self-Driven Particles,” *Physical Review Letters*, vol. 75, pp. 1226–1229, Aug. 1995.

[5] S. V. Viscido, J. K. Parrish, and D. Grünbaum, “Individual behavior and emergent properties of fish schools: A comparison of observation and theory,” *Marine Ecology Progress Series*, vol. 10.3354/me, pp. 239–249, 2004.

[6] A. Jadbabaie and A. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules,” in *IEEE Transactions on Automatic Control*, vol. 48, no. 6, Jun. 2003, pp. 988–1001.

[7] W. Ren and R. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, May 2005.

[8] R. Olfati-Saber, J. Fax, and R. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[9] T. Balch and F. Dellaert, “How multirobot systems research will accelerate our understanding of social animal behavior,” *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1445–1463, 2006.

[10] H. Min and Z. Wang, “Design and analysis of Group Escape Behavior for distributed autonomous mobile robots,” in *Int. Conf. on Robotics and Automation.* IEEE, 2011, pp. 6128–6135.

[11] D. V. Radakov, *Schooling in the ecology of fish.* J. Wiley, 1973.

[12] D. J. T. Sumpter, “The principles of collective animal behaviour,” *Phil. Trans. of the Royal Society B: Biological Sciences*, vol. 361, no. 1465, pp. 5–22, 2006.

[13] A. Ward, J. Herbert-Read, D. Sumpter, and J. Krause, "Fast and accurate decisions through collective vigilance in fish shoals," *Proc. of the National Academy of Sciences*, vol. 108, no. 6, p. 2312, 2011.

[14] M. Ballerini, N. Cabibbo, R. Candelier, A. Cavagna, E. Cisbani, I. Giardina, V. Lecomte, A. Orlandi, G. Parisi, A. Procaccini, M. Viale, and V. Zdravkovic, "Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study," *Proc. Nat. Acad. Sciences*, vol. 105, no. 4, pp. 1232–1237, 2008.

[15] Y. Katz and K. Tunstrø m, "Inferring the structure and dynamics of interactions in schooling fish," *Proceedings of the National Academy of Sciences*, 2011.

[16] D. A. Paley, N. E. Leonard, R. Sepulchre, D. Grunbaum, and J. K. Parrish, "Oscillator models and collective motion," *IEEE Control Systems Magazine*, vol. 27, no. 4, pp. 89–105, 2007.

[17] R. Lukeman, Y.-X. Li, and L. Edelstein-Keshet, "Inferring individual rules from collective behavior." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 28, pp. 12 576–80, Jul. 2010.

[18] N. Abaid and M. Porfiri, "Fish in a ring: spatio-temporal pattern formation in one-dimensional animal groups," *J R Soc Interface*, 2010.

[19] S. Berman, Q. Lindsey, M. S. Sakar, V. Kumar, and S. C. Pratt, "Experimental Study and Modeling of Group Retrieval in Ants as an Approach to Collective Transport in Swarm Robotic Systems," *Proceedings of the IEEE*, vol. 99, no. 9, pp. 1470–1481, 2011.

[20] N. C. Manoukis, A. Diabate, A. Abdoulaye, M. Diallo, A. Dao, A. S. Yaro, J. M. C. Ribeiro, and T. Lehmann, "Structure and Dynamics of Male Swarms of Anopheles gambiae," *J. Medical Entomology*, vol. 46, no. 2, pp. 227–235, 2009.

[21] J. Charlwood and M. Jones, "Mating in the mosquito, Anopheles gambiae sl," *Physiological Entomology*, vol. 5, no. 4, pp. 315–320, 1980.

[22] J. Charlwood, J. Pinto, C. Sousa, H. Madsen, C. Ferreira, and V. do Rosario, "The swarming and mating behaviour of Anopheles gambiae ss (Diptera: Culicidae) from Sao Tome Island," *J. of Vector Ecology*, vol. 27, no. 2, pp. 178–183, 2002.

[23] J. Thailayil, K. Magnusson, H. Godfray, A. Crisanti, and F. Catteruccia, "Spermless males elicit large-scale female responses to mating in the malaria mosquito Anopheles gambiae," *Proceedings of the National Academy of Sciences*, vol. 108, no. 33, pp. 13 677–13 681, 2011.

[24] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2-3, pp. 90–126, 2006.

[25] S. M. Ali S., "Floor Fields for Tracking in High Density Crowd Scenes," *European Conference on Computer Vision*, 2008.

[26] N. O. Handegard, R. Patel, and V. Hjellvik, "Tracking individual fish from a moving platform using a split-beam transducer," *The Journal of the Acoustical Society of America*, vol. 118, pp. 2210–2223, 2005.

[27] C. Schell, S. P. Linder, and J. R. Zeidler, "Tracking highly maneuverable targets with unknown behavior," *Proc. of the IEEE 92(3)*, pp. 558–574, 2004.

[28] N. F. Hughes and L. H. Kelly, "New techniques for 3-D video tracking of fish swimming movements in still or flowing water," *Canadian J. of Fisheries and Aquatic Sciences*, vol. 53, no. 11, pp. 2473–2483, Nov. 1996.

[29] E. Fontaine, D. Lentink, S. Kranenbarg, U. K. Muller, J. L. Van Leeuwen, A. H. Barr, and J. W. Burdick, "Automated visual tracking for studying the ontogeny of zebrafish swimming," *Journal of Experimental Biology*, vol. 211, no. 8, pp. 1305–1316, 2008.

[30] Z. Khan, T. Balch, and F. Dellaert, "An MCMC-based particle filter for tracking multiple interacting targets," *Proc. European Conf. on Computer Vision (ECCV)*, vol. 3024, pp. 279–290, 2004.

[31] K. Branson, A. A. Robie, J. Bender, P. Perona, and M. H. Dickinson, "High-throughput ethomics in large groups of Drosophila," *Nature Methods*, vol. 6, pp. 451–457, 2009.

[32] M. Betke, D. Hirsh, A. Bagchi, N. Hristov, N. Makris, and T. Kunz, "Tracking Large Variable Numbers of Objects in Clutter," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE. Minneapolis, USA: IEEE, Jun. 2007, pp. 1–8.

[33] A. Veeraraghavan, M. Srinivasan, R. Chellappa, E. Baird, and R. Lamont, "Motion based correspondence for 3D tracking of multiple dim objects," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2006.

[34] A. D. Straw, K. Branson, T. R. Neumann, and M. H. Dickinson, "Multi-camera real-time three-dimensional tracking of multiple flying animals," *J. Royal Society Interface*, vol. 8, no. 56, pp. 395–409, 2010.

[35] E. I. Fontaine, F. Zabala, M. H. Dickinson, and J. W. Burdick, "Wing and body motion during flight initiation in Drosophila revealed by automated visual tracking," *Journal of Experimental Biology*, vol. 212, no. 9, p. 1307, 2009.

[36] S. Butail, N. Manoukis, M. Diallo, A. S. Yaro, A. Dao, S. F. Traoré, J. M. Ribeiro, T. Lehmann, and D. A. Paley, "3D tracking of mating events in wild swarms of the malaria mosquito Anopheles gambiae." *Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2011, no. 2, pp. 720–3, Jan. 2011.

[37] S. Butail, N. Manoukis, M. Diallo, J. M. C. Ribeiro, T. Lehmann, and D. A. Paley, "Reconstructing the flight kinematics of swarming and mating in wild mosquitoes," *J. Royal Soc. Interface*, 2012.

[38] S. Butail and D. A. Paley, "3D reconstruction of fish schooling kinematics from underwater video," in *IEEE Int. Conf. on Robotics and Automation*, Anchorage, AK, USA, May 2010, pp. 2438–2443.

[39] ——, "Three-dimensional reconstruction of the fast-start swimming kinematics of densely schooling fish," *J. Royal Soc. Interface*, vol. 9, no. 66, pp. 77–88, 2011.

[40] D. H. Theriault, Z. Wu, N. I. Hristov, S. M. Swartz, K. S. Breuer, T. H. Kunz, and M. Betke, "Reconstruction and Analysis of 3D Trajectories of Brazilian Free-tailed Bats in Flight," in *International Conference on Pattern Recognition (ICPR)*, 2010.

[41] S. Fry, M. Bichsel, P. Müller, and D. Robert, "Tracking of flying insects using pan-tilt cameras," *J. of Neuroscience Methods*, vol. 101, no. 1, pp. 59–67, 2000.

[42] D. Zou, Q. Zhao, H. S. Wu, and Y. Q. Chen, "Reconstructing 3D motion trajectories of particle swarms by global correspondence selection," in *Int. Conf. Computer Vision*, 2009, pp. 1578–1585.

[43] J. Hardie and S. Young, "Aphid flight-track analysis in three dimensions using video techniques," *Physiological Entomology*, vol. 22, no. 2, pp. 116–122, 1997.

[44] D. Grover, J. Tower, and S. Tavaré, "O fly, where art thou?" *J. of the Royal Society Interface*, vol. 5, no. 27, p. 1181, 2008.

[45] G. Gibson, "Swarming behaviour of the mosquito Culex pipiens quinquefasciatus: a quantitative analysis," *Physiological Entomology*, vol. 10, no. 3, pp. 283–296, Sep. 1985.

[46] T. Ikawa, H. Okabe, T. Mori, K. Urabe, and T. Ikeshoji, "A method for reconstructing three-dimensional positions of swarming mosquitoes," *J. of Insect Behavior*, vol. 7, no. 2, pp. 237–248, 1994.

[47] S. Butail and D. Paley, "Vision-based estimation of three-dimensional position and pose of multiple underwater vehicles," in *IEEE Int. Conf. on Intelligent Robots and Systems*, 2009, pp. 2477–2482.

[48] S. Butail, A. Chicoli, and D. A. Paley, "Putting the fish in the fish tank: Immersive VR for animal behavior experiments," in *IEEE Int. Conf. on Robotics and Automation*, 2012, pp. 5018–5023.

[49] V. Granville, M. Krivanek, and J.-P. Rasson, "Simulated annealing: a proof of convergence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 652–656, Jun. 1994.

[50] C. Andrieu, N. De Freitas, A. Doucet, and M. Jordan, "An introduction to MCMC for machine learning," *Machine learning*, vol. 50, no. 1, pp. 5–43, 2003.

[51] Y. Bar-Shalom, *Tracking and data association.* San Diego, CA, USA: Academic Press Professional, Inc., 1987.

[52] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Proc.*, vol. 50, no. 2, pp. 174–188, 2002.

[53] E. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proc. of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium.* IEEE, pp. 153–158.

[54] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEEE Proc. on Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[55] N. D. Freitas and N. Gordon, *Sequential Monte Carlo Methods in Practice.* Birkhäuser, 2001.

[56] Z. Khan, T. Balch, and F. Dellaert, "A Rao-Blackwellized particle filter for eigentracking," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 980–986, 2004.

[57] R. van der Merwe, A. Doucet, N. D. Freitas, and E. Wan, "The Unscented Particle Filter," 2000.

[58] S. Chib and E. Greenberg, "Understanding the metropolis-hastings algorithm," *American Statistician*, vol. 49, no. 4, pp. 327–335, 1995.

[59] S. Kirkpatrick, "Optimization by simulated annealing: Quantitative studies," *Journal of Statistical Physics*, vol. 34, no. 5, pp. 975–986, 1984.

[60] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and finite-time behavior of simulated annealing," *Advances in Applied Probability*, pp. 747–771, 1986.

[61] D. Henderson, S. Jacobson, and A. Johnson, "The theory and practice of simulated annealing," *Handbook of metaheuristics*, pp. 287–319, 2003.

[62] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[63] G. Casella and E. George, "Explaining the Gibbs sampler," *American Statistician*, pp. 167–174, 1992.

[64] J. Gall, B. Rosenhahn, T. Brox, and H. P. Seidel, "Optimization and filtering for human motion capture," *International J. of Computer Vision*, vol. 87, no. 1, pp. 75–92, 2010.

[65] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys (CSUR)*, vol. 38, no. 4, pp. 1–45, 2006.

[66] B. Stenger, P. R. S. Mendonca, and R. Cipolla, "Model-based 3D tracking of an articulated hand," *IEEE Computer Vision and Pattern Recognition*, vol. 2, pp. 310–315, Dec. 2001.

[67] H. Sidenbladh, M. Black, and D. Fleet, "Stochastic tracking of 3D human figures using 2D image motion," *European Conference on Computer Vision*, pp. 702–718, 2000.

[68] D. Terzopoulos and D. Metaxas, "Dynamic 3D models with local and global deformations: deformable superquadrics," *IEEE trans. on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 703–714, 1991.

[69] L. Ristroph, G. J. Berman, A. J. Bergou, Z. J. Wang, and I. Cohen, "Automated hull reconstruction motion tracking (HRMT) applied to sideways maneuvers of free-flying insects," *Journal of Experimental Biology*, vol. 212, no. 9, p. 1324, 2009.

[70] D. Koller, J. Weber, and J. Malik, "Robust Multiple Car Tracking with Occlusion Reasoning," *California Partners for Advanced Transit and Highways (PATH)*, 1994.

[71] K. Branson and S. Belongie, "Tracking multiple mouse contours (without too many samples)," in *IEEE Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 1039–1046.

[72] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. on Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.

[73] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *International J. of computer vision*, vol. 22, no. 1, pp. 61–79, 1997.

[74] A. Tekalp and P. V. Beek, "Two-dimensional mesh-based visual-object representation for interactive synthetic/natural digital video," *Proceedings of the IEEE*, vol. 86, no. 6, pp. 1029–1051, 1998.

[75] S. Dambreville, Y. Rathi, and A. Tannenbaum, "Tracking deformable objects with unscented Kalman filtering and geometric active contours," in *American Control Conference.* Minneapolis, USA: IEEE, Jun. 2006, pp. 2856—-2861.

[76] J. M. Snyder and J. T. Kajiya, "Generative modeling: A symbolic system for geometric modeling," in *Proc. of the 19th conf. on Computer graphics and interactive techniques.* New York, NY, USA: ACM, Jul. 1992, pp. 369–378.

[77] A. J. Hanson, "Quaternion Frenet frames: Making optimal tubes and ribbons from curves," *Technical Report*, no. 1.111, pp. 1–7, 1993.

[78] Q. Zhu, M. Wolfgang, D. Yue, and M. Triantafyllou, "Three-dimensional flow structures and vorticity control in fish-like swimming," *Journal of Fluid Mechanics*, vol. 468, no. 1-28, pp. 2–2, 2002.

[79] E. D. Tytell and G. V. Lauder, "The C-start escape response of Polypterus senegalus: bilateral muscle activity and variation during stage 1 and 2," *Journal of Experimental Biology*, vol. 205, no. 17, pp. 2591–2603, 2002.

[80] A. Okubo, "Dynamical aspects of animal grouping: swarms, schools, flocks, and herds," *Advances in Biophysics*, vol. 22, no. 0, pp. 1–94, 1986.

[81] X. Rong Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I: Dynamic models," *IEEE Trans. Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, 2003.

[82] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[83] Q. Zheng and R. Chellappa, "Automatic feature point extraction and tracking in image sequences for arbitrary camera motion," *International Journal of Computer Vision*, vol. 15, no. 1-2, pp. 31–76, Jun. 1995.

[84] Y. Yao and R. Chellappa, "Tracking a Dynamic Set of Feature Points," *IEEE Transactions on Image Processing*, vol. 4, no. 10, pp. 1382–1395, Oct. 1995.

[85] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004.

[86] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, pp. 83–97, 1955.

[87] D. Swain, I. Couzin, and N. E. Leonard, "Real-time feedback-controlled robotic rish for behavioral experiments with fish schools," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 150–163, 2011.

[88] I. J. Cox and S. L. Hingorani, "An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 138–150, 1996.

[89] S. Blackman, “Multiple hypothesis tracking for multiple target tracking,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[90] P. Gabriel, J. Verly, J. Piater, and A. Genon, “The state of the art in multiple object tracking under occlusion in video sequences,” in *Advanced Concepts for Intelligent Vision Systems*, 2003, pp. 166–173.

[91] Z. Khan, “MCMC data association and sparse factorization updating for real time multitarget tracking with merged and multiple measurements,” *IEEE Trans. on Pattern Anal. and Machine Intel.*, vol. 28, no. 12, pp. 1960–1972, 2006.

[92] Z. Wu, T. Kunz, and M. Betke, “Efficient track linking methods for track graphs using network-flow and set-cover techniques,” in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1185–1192.

[93] A. Perera, C. Srinivas, A. Hoogs, and G. Brooksby, “Multi-Object Tracking Through Simultaneous Long Occlusions and Split-Merge Conditions,” in *Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2006, pp. 666–673.

[94] O. Tuzel, R. Subbarao, and P. Meer, “Simultaneous multiple 3D motion estimation via mode finding on Lie groups,” in *Proc. IEEE Int. Conf. on Computer Vision ICCV*, vol. 1, Oct. 2005, pp. 18–25.

[95] Z. Wu, N. Hristov, T. Kunz, and M. Betke, “Tracking-reconstruction or reconstruction-tracking? Comparison of two multiple hypothesis tracking approaches to interpret 3D object motion from several camera views,” in *Workshop on Motion and Video Computing.* IEEE, 2010, pp. 1–8.

[96] A. Veeraraghavan, R. Chellappa, and M. Srinivasan, “Shape-and-behavior encoded tracking of bee dances,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, p. 463, 2008.

[97] D. J. Salmond and H. Birch, “A particle filter for track-before-detect,” *In Proc. of American Control Conference*, vol. 5, pp. 3755–3760, Jun. 2001.

[98] J. J. Videler, *Fish swimming.* Springer, 1993.

[99] P. Domenici and R. W. Blake, “The kinematics and performance of fish fast-start swimming,” *Journal of Experimental Biology*, vol. 200, no. 8, pp. 1165–1178, 1997.

[100] D. Yong, “Strings, Chains, and Ropes,” *SIAM Review*, vol. 48, no. 4, pp. 771–781, 2006.

[101] S. Gueron and N. Liron, “Simulations of Three-dimensional Ciliary Beats and Cilia Interactions,” *Biophysical Journal*, vol. 65, no. 1, pp. 499–507, 1993.

[102] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Int. Conf. on Computer Vision*, vol. 1. Kerkyra, Greece: IEEE Computer Society, Sep. 1999, pp. 666–673.

[103] J. Nocedal and S. J. Wright, *Numerical optimization.* Springer Verlag, 1999.

[104] R. Shadwick, J. Steffensen, S. Katz, and T. Knower, "Muscle dynamics in fish during steady swimming," *American zoologist*, vol. 38, no. 4, pp. 755–770, 1998.

[105] B. M. Bell and F. W. Cathey, "The iterated Kalman filter update as a Gauss-Newton method," *IEEE Trans. on Automatic Control*, vol. 38, no. 2, pp. 294–297, 1993.

[106] B. Rosenhahn, T. Brox, and J. Weickert, "Three-dimensional shape knowledge for joint image segmentation and pose tracking," *International Journal of Computer Vision*, vol. 73, no. 3, pp. 243–262, 2007.

[107] L. Scardovi, N. E. Leonard, and R. Sepulchre, "Stabilization of collective motion in three dimensions: A consensus approach," *IEEE Conf. on Decision and Control*, pp. 2931–2936, 2007.

[108] J. Kwon, M. Choi, F. C. Park, and C. Chun, "Particle filtering on the Euclidean group: framework and applications," *Robotica*, vol. 25, no. 6, pp. 725–737, 2007.

[109] Y. Wang and G. S. Chirikjian, "Error propagation on the Euclidean group with applications to manipulator kinematics," *IEEE Trans. on Robotics*, vol. 22, no. 4, pp. 591–602, 2006.

[110] D. J. Higham, "An Algorithmic Introduction to Numerical Simulation of Stochastic Differential Equations," *SIAM Review*, vol. 43, no. 3, pp. 525–546, 2001.

[111] M. Piccardi, "Background subtraction techniques: a review," *Int. Conf. on Systems, Man and Cybernetics*, vol. 4, pp. 3099–3104, Oct. 2004.

[112] J.-Y. Bouguet. Camera Calibration Toolbox for Matlab. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

[113] F. Mokhtarian and A. K. Mackworth, "A theory of multiscale, curvature-based shape representation for planar curves," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 8, pp. 789–805, 1992.

[114] A. Diabate, A. S. Yaro, A. Dao, M. Diallo, D. L. Huestis, and T. Lehmann, "Spatial distribution and male mating success of Anopheles gambiae swarms," *BMC Evolutionary Biology*, vol. 11, no. 1, p. 184, Jun. 2011.

[115] J. Charlwood and J. Pinto, "Male size does not affect mating success (of Anopheles gambiae in Sao Tome)," *Medical and Veterinary Entomomology*, vol. 16, no. 1, pp. 109–111, 2002.

[116] J. Bryan, "Results of consecutive matings of female Anopheles gambiae species B with fertile and sterile males," *Nature*, vol. 218, no. 5140, pp. 489–489, 1968.

[117] A. Diabaté, A. Dao, A. S. Yaro, A. Adamou, R. Gonzalez, N. C. Manoukis, S. F. Traore, R. W. Gwadz, and T. Lehmann, "Spatial swarm segregation and reproductive isolation between the molecular forms of Anopheles gambiae," *Proceedings of the Royal Society B: Biological Sciences*, pp. 4215–42 222, Sep. 2009.

[118] C. Pennetier, B. Warren, K. R. Dabiré, I. J. Russell, and G. Gibson, ""Singing on the wing" as a mechanism for species recognition in the malarial mosquito Anopheles gambiae." *Curr Biol*, vol. 20, no. 2, pp. 131–136, Jan. 2010.

[119] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[120] A. Cavagna, I. Giardina, A. Orlandi, G. Parisi, A. Procaccini, M. Viale, and V. Zdravkovic, "The STARFLAG handbook on collective animal behaviour: 1. Empirical methods," *Animal Behaviour*, vol. 76, no. 1, pp. 217–236, 2008.

[121] R. Hartley and P. Sturm, "Triangulation," *Computer vision and image understanding*, vol. 68, no. 2, pp. 146–157, 1997.

[122] M. Chen, "EM algorithm for Gaussian mixture model (MATLAB File Exchange, file id: 26184)," 2012.

[123] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[124] S. Marras and M. Porfiri, "Fish and robots swimming together: attraction towards the robot demands biomimetic locomotion." *Journal of the Royal Society, Interface*, Feb. 2012.

[125] Z. Khan, T. Balch, and F. Dellaert, "MCMC-based particle filtering for tracking a variable number of interacting targets," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1805–1918, 2005.

[126] P. Turaga and R. Chellappa, "Machine Recognition of Human Activities: A survey," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473–1488, 2008.

[127] R. Poppe, "A survey on vision-based human action recognition," *Image and Vision Computing*, 2010.

[128] H. Dankert, L. Wang, E. D. Hoopfer, D. J. Anderson, and P. Perona, "Automated monitoring and analysis of social behavior in Drosophila." *Nature methods*, vol. 6, no. 4, pp. 297–303, Apr. 2009.

[129] A. Okubo and H. C. Chiang, "An analysis of the kinematics of swarming of Anarete Pritchardi Kim (Diptera: Cecidomyiidae)," *Researches on Population Ecology*, vol. 16, no. 1, pp. 1–42, 1974.

[130] R. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal on Robotics and Automation*, vol. 3, no. 4, pp. 323–344, Aug. 1987.

[131] J. R. Smithanik, E. M. Atkins, and R. M. Sanner, "Visual positioning system for an underwater space simulation environment," *Journal of Guidance, Control, and Dynamics*, vol. 29, pp. 858–869, 2006.

[132] T. Svoboda, D. Martinec, and T. Pajdla, "A Convenient Multicamera Self-Calibration for Virtual Environments," *Presence: Teleoperators and Virtual Environments*, vol. 14, no. 4, pp. 407–422, Aug. 2005.

[133] V. Gourgoulis and N. Aggeloussis, "Reconstruction accuracy in underwater three-dimensional kinematic analysis," *J. of Science and Medecine in Sport*, vol. 11, no. 2, pp. 90–95, 2008.

[134] D. Schuhmacher, B. Vo, and B. Vo, "A consistent metric for performance evaluation of multi-object filters," *Signal Processing, IEEE Transactions on*, vol. 56, no. 8, pp. 3447–3457, 2008.

[135] B. Ristic, B. Vo, D. Clark, and B. Vo, "A metric for performance evaluation of multi-target tracking algorithms," *IEEE Trans. on Signal Processing*, vol. 59, no. 7, pp. 3452–3457, 2011.