Simulating flight from an insect's perspective: determining whether or not filmed interactions involving *Eristalis tenax* hoverflies are pursuits.

by

Alexander Chan Wei-Ming Mesecke

Supervisors:

Associate Professor Karin Nordström

Doctor Sherry Randhawa

Submitted:

30 October 2019

Resubmitted with amendments:

23 December 2019

Submitted to the College of Science and Engineering in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Biomedical) (Honours), Master of Engineering (Biomedical) at Flinders University – Adelaide Australia

I certify that this work does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

Signed Date

30 OCT. 2019

Alexander Chan Wei-Ming Mesecke

"Try not. Do. Or do not. There is no try."

—— Yoda, Star Wars: Episode V – The Empire Strikes Back

This would not have been possible without all of the support I have received.

I would like to thank in particular my supervisors, Associate Professor Karin Nordström and Doctor Sherry Randhawa.

I would also like to especially acknowledge the undying love, understanding, and forgiveness given to me by my family and friends.

My deepest gratitude goes to all who have helped me along this long and winding road.

Abstract

The aim of this project was to develop a program that would take as input three-dimensional position trajectories filmed as part of a previous experiment. These positions were isolated from recordings of interactions involving *Eristalis tenax* hoverflies, and various other insects and inanimate objects. The program would allow the user to select which of the participants in the interaction was to be the observer, and a simulation of the original flight would be reconstructed from the perspective of that observer.

In order to perform this simulation, additional information not present in the original data had to be estimated. To test the validity of the methods used to create this simulation, the program also determined which interactions were likely to be pursuits, and which were not. This was achieved by using the characteristics of the simulation and the trajectories.

Although the methods were found to be consistent according to statistical tests, the determination of pursuit was inconclusive.

Table of Contents

Abstract iii

Table of Contents iv

List of Figures vi

List of Tables vii

List of Abbreviations viii

1. Introduction 1

    1.1. Background 1

    1.2. Relevant Technologies 5

    1.3. Research Objective 5

2. Literature Review 7

    2.1. Tracking Insect Flight 7

    2.2. Hoverfly Behaviour 9

    2.3. Characterising and Modelling Flight 9

    2.4. Trajectory Reconstruction and Flight Simulation 11

    2.5. Display of Optic Flow 12

    2.6. Target Detection in Visual Clutter 13

    2.7. Heading Estimation and Machine Learning 14

3. Application and Implementation 16

    3.1. Project Management 16

    3.2. Aims and Specifications 16

    3.3. Deliverables and Requirements 16

4. Methods 17

4.1. Data Validation and Pre-Processing 17

4.2. Trajectory Reconstruction and Heading Estimation 18

4.3. Simulation 20

4.4. Determination of Pursuit 22

5. Results 23

6. Discussion 33

6.1. Analysis 33

6.2. Limitations and Improvements 33

6.3. Further Work 34

7. Summary 35

References 36

Appendices a

Appendix A. Code a

Appendix B. Summary of Results g

Appendix C. Results of Statistical Tests ll

Appendix C.1. Kruskal-Wallis ll

Appendix C.2. Conover-Iman nn

Appendix D. Determining Pursuit oo

List of Figures

1.1.	Variation in visibility of insects in the recorded interactions.	2
1.2.	Comparison of honeybee with various hoverflies.	3
1.3.	Illustration of optic flow.	4
1.4.	Effect of heading and direction of motion on optic flow.	4
2.1.	Prototypical movements determined for a hoverfly.	10
4.1.	A typical, valid, dataset.	17
4.2.	Rotating the observer's heading to look directly at the screen.	20
4.3.	Data from the code ready to be input into the 3D Target stimulus.	20
5.1.	Response plot from the regression learner.	28
5.2.	The regression learner's predicted response vs. true response.	31
5.3.	Using the linear regression algorithm on the test set.	32

List of Tables

4.1. Datasets found to be invalid during pre-processing. 18

5.1. Trajectories from which NaN values were removed. 23

5.2. Trajectories that consisted entirely of NaN values. 25

5.3. Trajectories resulting in an error in simulation. 25

5.4. The results of applying the linear regression to the test set. 29

List of Abbreviations

Conspecific	Another animal of the same species.
CSV	Comma-separated value; a spreadsheet file format.
DOM	Direction of motion.
(d)TSDN	(Dipteran) target-selective descending neuron.
Heading	Direction in which the head is facing.
Hoverfly/hoverflies	*Eristalis tenax* hoverfly/hoverflies, unless specified otherwise.
LED	Light-emitting diode.
LOS	Line of sight; the direct line between the two participants.
NaN	Not a number.
(3D) Pose	The combination of (3D) position and orientation.
STMD	Small target motion detector.

1. Introduction

The aim of this project was to develop a piece of code that would integrate with FlyFly, the program currently used in the Motion Vision Lab (Henriksson 2010a). FlyFly was developed in 2010 as an alternative to the open-source visual stimuli program VisionEgg (Straw 2008), and was intended to better suit the specific requirements of the lab and its experiments. FlyFly is still used in the lab to display visual stimuli to *Eristalis tenax* hoverflies, referred to from here on simply as 'hoverflies', when conducting electrophysiological experiments.

This project aimed to allow for the expansion of earlier work done by Thyselius et al. (2018), who recorded interactions that occurred when investigating hoverflies being approached by other insects while feeding in the field. The hoverflies featured in these recordings interacted with either other hoverflies, wasps, bees, or inanimate beads. Hence, not all recorded interactions are between two insects of the same species.

From these interactions, the three-dimensional positions of both of the participants involved were tracked using a high-speed stereo videography setup. The process by which this was achieved will be further elaborated in 1.1. Background and 2.1. Tracking Insect Flight.

The code developed as a result of this project would take datasets such as those obtained by Thyselius et al. (2018) as input. The code would then utilise the existing FlyFly assets to simulate flight from the perspective of one of the participants involved. The user would have the ability to select which participant would provide its perspective as the basis of the flight simulation.

To validate the performance of this code, in light of challenges that will be elaborated in 1.1. Background, 2.3. Characterising and Modelling Flight, 2.4, Trajectory Reconstruction and Flight Simulation, and 2.7. Heading Estimation and Machine Learning, the code aimed to further be able to determine which interactions are and are not pursuits.

1.1. Background

Hoverflies are prime candidates for study in the areas of both neuroscience and aeronautics, due to their relatively simple brain consisting of only one million neurons, as well as their ability to perform fast and agile flight manoeuvres both in constrained environments and in free flight (Geurten et al. 2010; Thyselius et al. 2018). Hoverflies have been studied in flight since the 1970s but this early work focussed on the aerodynamics and mechanics of their flight, rather than the behaviour (Golding, Ennos & Edmunds 2001).

In this research, conventional stimuli presented to hoverflies is usually unnatural, often consisting of rotating stripes or expanding circles. Ideally, naturalistic conditions must be provided to an animal in order to observe truly naturalistic behaviour (Geurten, Kern & Egelhaaf 2012; van Hateren et al. 2005). When using algorithms based on un-naturalistic stimuli to predict responses to naturalistic stimuli, the algorithms may fail (Dyakova & Nordström 2017).

In addition to unnatural stimuli, experimentation is usually performed on hoverflies that are tethered – fixed in position using wax – in open-loop paradigms (Fry et al. 2008). Open-loop paradigms experimentally decouple the sensory stimulus from the motor behaviour, presenting visual stimuli and recording motor responses without altering the stimuli despite the animal's attempts to move (Fry et al. 2008; 2009). As a result, the feedback loop involving the visual system and the motor output normally present in free movement is broken (Fry et al. 2008).

The transferability of results from experiments using these open-loop paradigms to natural free flight is therefore unknown due to the experimental constraints (Fry et al. 2008; 2009). Additionally, the possible stimuli that can be presented to insects in such open-loop paradigms is limited in intricacy (Bagheri et al. 2014b).

This project aims to create a naturalistic stimulus for use in hoverfly research, by simulating flight from the perspective of a hoverfly using the three-dimensional position data taken from real interactions between two insects in the field.

The recordings were taken using cameras with high temporal resolution – a capture frequency of 120 Hz, recording every 8.33 ms – but a relatively low spatial resolution of 640 × 480 pixels (Thyselius et al. 2018). As a result of the low spatial resolution, the participants in the interactions are often difficult to distinguish. This is amplified by the high level of visual clutter, i.e. the background has similar colours and textures to the insects that are to be tracked (Acosta 2010). As a result, tracking the participants often necessitated manual input, as there was insufficient contrast between the participants and the background, even when measures were taken to improve contrast (Thyselius et al. 2018).

Figure 1.1 shows scenes from two separate recordings, illustrating the vast difference in visibility of insects across the recordings.

Figure 1.1: Demonstrating the variation in visibility of insects in the recorded interactions (Thyselius et al. 2018). Left: one participant is clearly visible against a flower in the upper centre of the image. Right: neither participant is clearly visible.

The participants in these interactions always involved at least one hoverfly of the species *Eristalis tenax,* found all over the world, often called drone flies due to their visual similarity to honeybee drones (Golding, Ennos & Edmunds 2001; Thyselius et al. 2018). The term hoverfly will be used in this thesis to refer to *Eristalis tenax*, despite their additional name.

In the literature, the term hoverfly is often used to refer to a wide range of dipteran insects, such as *Syritta pipiens* and *Episyrphus balteatus*. For this reason, care must be taken to differentiate between these different species of flies when referred to as hoverflies. Hence, the use of the term hoverfly in this thesis refers exclusively to *Eristalis tenax*, with other species referred to unambiguously.

Figure 1.2 shows an *Eristalis tenax* hoverfly alongside an *Apis mellifera* honeybee, illustrating how the unarmed hoverfly and the honeybee with its stinger are almost visually identical. For illustration, other insects commonly called hoverflies are also shown.

(b)

Images removed due to copyright restriction.

(a) **(c)**

Figure 1.2: ***(a)*** *Comparison of an* Apis mellifera *honeybee (left) and an* Eristalis tenax *hoverfly (right) (Warren Photographic).* ***(b)*** *A* Syritta pipiens *hoverfly (Storey 2012).* ***(c)*** *An* Episyrphus balteatus *hoverfly (Royal Society for the Protection of Birds).*

The outcome of this project will be a program that will take three-dimensional position data as input, representing flight sequences, and simulate this flight from the perspective of either insect participant as a stimulus for use in further electrophysiological research.

However, due to the limitations of the original experimental setup, this position data does not include any information on the direction of flight of the insects involved. This is problematic as the accurate reconstruction of a flight from the perspective of the original pilot relies on the way it was facing throughout the flight, as this has a direct effect on the information seen by the flying animal over the course of the flight.

The information seen by an animal as it moves through space is termed optic flow. Optic flow is rigorously defined as “the change of structured patterns of light on the retina that lend to an impression of movement of the visual imagery projected onto the retina,” and gives animals the ability to visually estimate their own motion (Raudies 2013). When an animal moves, its optic flow is composed of the perceived relative motion of the animal’s static surroundings generated by the animal’s own motion, in addition to the real motion of other moving objects (Boeddeker et al. 2005).

Figure 1.3 illustrates this concept by recolouring different components of the field of vision according to how they would be perceived by an animal in motion. Each of the coloured components in Figure 1.3 would appear to be moving, due to the viewer's self-motion.

Image removed due to copyright restriction.

Figure 1.3: An illustration of optic flow (Gonzalez-Bellido, Fabian & Nordström 2016).

In Figure 1.3, the viewer is moving straight forwards, hence the optic flow radiates from the centre of the image. The backboard and net are static figures, but would appear to move as a result of the viewer's own motion. In this case, the viewer would perceive the backboard and net to be moving nearer to them and also vertically higher, as indicated by the lines in Figure 1.3. The other players – dynamic figures – would also be perceived by the viewer to move, according to how they are indeed moving in the real world.

The optic flow illustrated in Figure 1.3 is conceptually simple, as the direction of motion is aligned with the direction in which the viewer is looking. However, the optic flow observed by an animal in motion depends not only on the direction it is moving, but also on the direction it is looking, from here on termed 'heading'. Figure 1.4 illustrates the effect a change in heading has on optic flow.

Image removed due to copyright restriction.

Figure 1.4: Different heading with same direction of motion produces different optic flow. Left: going forwards, and Right: going backwards along the same track (Ellis 2014).

In Figure 1.4, the rollercoaster traverses the same track in both instances, but facing in different directions. The arrows indicate the perceived motion of those elements of the visual field. As a result, although both images feature the same direction of motion, indeed along the same path, the difference in heading produces different optic flow.

Similarly, a heading far from the true heading for the simulated hoverfly flight would produce totally different optic flow to that which would have been experienced by the hoverflies in the recordings. In order for the simulated flight to be realistic, the optic flow presented to the viewer must realistically represent that which would have been experienced by the original flying insect.

This is one major challenge of this project, as the data supplied as the basis of this project lacks information on the orientation of the body and head of the insects. In order to correctly represent the optic flow, the heading of the insects at each frame must therefore be estimated. This estimation will form the bulk of the project, and the accuracy of the simulation will depend on the factors used in this process of estimation.

As the orientation of the body is not considered in this project, the (unrealistic) assumption that the head and body are aligned (Land 1992; Olberg, Worthington & Venator 2000) does not need to be made.

1.2. Relevant Technologies

Similar technologies to FlyFly already exist, with different solutions being found to the problem of creating naturalistic stimuli. FlyFly itself was created in an effort to improve upon the VisionEgg software (Henriksson 2010a). However, using trajectories from recorded flights in the wild is a different question. As will be outlined in 2.4. Trajectory Reconstruction and Flight Simulation, other studies have attempted to display reconstructed flights to insects, however, these had other limitations.

One technology of note is GapFlyt, an autonomous flying vehicle which makes use of optical flow and parallax to detect holes. By taking pictures around a hole, and analysing which features appear to have moved more or less than others between pictures, the edges of a hole can be approximated (Ackerman 2018). In this way, visual information from the cameras provides closed-loop feedback to the motors, altering flight in real time. However, GapFlyt relies on textured features, and would not perform as well on walls painted white (Ackerman 2018).

1.3. Research Objective

As laid out in 1.1. Background, this project aimed to create a program that would simulate hoverfly flights given 3D data tracked from interactions recorded in the field. This simulation would run using the FlyFly visual stimulus program currently used in the Motion Vision Lab.

The performance and hence validity of the created program would be evaluated by using the simulations to determine which interactions were and were not pursuits, based on the characteristics of the simulation.

By reconstructing interactions which may have been pursuits, target detection can be evaluated alongside optic flow. Depending on the direction of motion, optic flow may help or hinder target detection (Nicholas et al. 2018), and so this project will have a direct future application in research, with its resultant visual stimuli being planned to be used in electrophysiology.

As a whole, small target detection, particularly in visual clutter, has applications in fields such as automation and defence, particularly in bio-inspired robotics.

2. Literature Review

A literature review was conducted using the Google Scholar, ScienceDirect, PubMed, and IEEE Xplore databases to search for articles investigating tracking insect flight in three dimensions, modelling insect flight for the purposes of biomimicry, and otherwise characterising insect flight behaviours. The review also searched for articles regarding simulating flight, reconstructing trajectories from discrete points, estimation of heading, displaying optic flow, insects' ability to detect targets in background clutter, and general hoverfly behaviour.

As a result, this literature review has been divided into seven sections, according to the order in which the project considered these areas. First, the tracking of flight in three dimensions was investigated, in particular comparing different experimental setups. Next, hoverfly behaviour was explored, in an effort to find elements that could be used in determining a strategy for flight control. Third, this was expanded upon in flight characterisation and modelling. In the fourth section, reconstruction of trajectories was investigated, in order to consider alternative methods to flight simulation. Fifth, display of optic flow was investigated, considering different methods of doing so. The sixth section looks at insects detecting targets in cluttered backgrounds, again to look for elements to include in a flight control strategy. Finally, the main focus of this project, heading estimation, was considered. Machine learning was also investigated.

2.1. Tracking Insect Flight

To obtain the data to be used as input in this project, two cameras were set up orthogonally, allowing the centre of mass of each insect to be mapped in three dimensions (Thyselius et al. 2018). For each two-insect interaction, the positions of the insects were recorded as a set of XYZ-coordinates, isolated at each frame. This experimental setup was very similar to that used in other studies, and in particular followed previous work done by Wardill et al. (2017).

However, unlike other studies, the experimental setup from whence the data was obtained could not enable the orientation of the insects' bodies nor their heads to be determined (Geurten et al. 2010; Geurten, Kern & Egelhaaf 2012; Golding, Ennos & Edmunds 2001; van Hateren et al. 2005). This was due to the low resolution of the footage obtained and the highly textured surroundings, meaning at times the hoverflies were unable to be identified by the image processing software (Thyselius et al. 2018).

Olberg, Venator, and Worthington (2000) simplified measuring body orientation by characterising interactions in a plane perpendicular to the axis of the camera lens, allowing angles to be used to describe the orientation of the insects' bodies. However, this could only be done when the insects flew roughly at right angles to the camera.

Other studies overcame this problem in different ways. In order to be able to identify body orientation during flight, Geurten et al. (2010) utilised an experimental setup in which one camera filmed hoverflies from above, in addition to another camera filming from the side. Similarly, Boeddeker, Kern & Egelhaaf (2002) used one camera recording from the side and

another from the bottom, orthogonally oriented, to capture body orientation. Although not in insects, Eckmeier et al. (2008) used a one camera above, one camera from the side setup to record the movements of finches in three dimensions. In each of these experiments, aligning the two cameras orthogonally, as opposed to a simple angular offset, was necessary to identify the orientation of the animals' bodies in 3D space, not simply position.

In those studies, however, this was made possible by the animals being observed within cages of various sizes, constraining the area in which they were able to fly, unlike the data provided which was recorded outdoors. Alternatively, some setups made use of a uniform background, in order to improve contrast. Additionally, the method used in the experiment done by Thyselius et al. (2018) was deliberately chosen to be easy to set up and calibrate, in order to be able to be used in a wide range of settings. As this data was gathered in the wild, as opposed to within a constrained area, the interactions should therefore more closely resemble truly naturalistic behaviour, and thus when used as stimuli should be more naturalistic, in turn evoking a more naturalistic response.

As outlined in 1.1. Background, the orientation of an animal's head is involved in controlling its gaze, and as a result affects its perceived optic flow (Boeddeker et al. 2005; Eckmeier et al. 2008; Geurten, Kern & Egelhaaf 2012; van Hateren et al. 2005; Lindemann et al. 2003; Tammero & Dickinson 2002). If the orientation of the head cannot be tracked in flight, the body orientation can be used to approximate the orientation of the head instead (Boeddeker et al. 2005; Tammero & Dickinson 2002). To overcome the limitation of having no information on head orientation, it was assumed that the insects faced straight ahead in the direction of flight, with the concession that this assumption would likely not hold outdoors, and that the reconstructed optic flow would not be completely true to life (Boeddeker et al. 2005; Tammero & Dickinson 2002). This assumption was based on the fact that both *Calliphora vicina* blowflies and *Drosophila melanogaster* fruit flies face in the direction of flight for the majority of cruising flight.

In the other aforementioned experiments, the resolution of the images obtained from the orthogonally oriented cameras was high enough to determine the orientation of the head in addition to the body (Geurten et al. 2010; Geurten, Kern & Egelhaaf 2012; Golding, Ennos & Edmunds 2001). Instead of using a camera to track the head position of blowflies, van Hateren et al. (2005) used a slightly different approach, measuring voltages induced by magnetic coils embedded in the blowflies' heads to track their positions relative to the cage in which they were filmed flying. As a result, trajectories could be established for not just the body but also the head, subsequently describing exactly the path taken by the animal, and its heading along that path. Goulard et al. (2015) also achieved this, using a high-resolution camera and a uniform background to track natural markers on the head of the insect and thereby determine its orientation.

In order to demonstrate that naturalistic stimuli could be created from trajectories lacking information on head orientation, van Hateren et al. (2005) reconstructed a set of trajectories with head orientations and one without. The article referenced Collett and Land's landmark 1978 paper on hoverflies, which did not record in three dimensions,

determining only body position and rotation in the pitch axis. Similarly, although Golding, Ennos, and Edmunds (2001) used a camera from above, body positions were recorded in only two dimensions, allowing for the construction of flight paths but with no information on altitude or body orientation. Van Hateren et al. (2005) noted that naturalistic stimuli could be created from trajectories like these, but that any flight that included sequences of sideward translation would lead to the generation of sideward optic flow. Hence, operating on the assumption that the direction of gaze does not differ from the direction of body orientation would lead to an unrealistic flight trajectory, and in turn an inaccurate flight simulation.

Although it is fallacious to assume that the head position of an insect can be used to infer its body position (Olberg, Worthington & Venator 2000), with some smaller insects, head orientation is tightly coupled to body orientation (Land 1992). However, in this project the head position will be estimated independently of the body position, thus making no assumptions about the coupling of body orientation to head position.

2.2. Hoverfly Behaviour

Pursuing prey is not the only motivation for insect pursuit flights; both dragonflies and hoverflies pursue members of their own species, termed 'conspecifics' (Land 1992). Pursuit of conspecifics usually entails males chasing females in order to mate. When pursuing females, male hoverflies use an interception strategy, aiming ahead of the female (Olberg, Worthington & Venator 2000). In interception, the image of the prey on the pursuer's retina is held at a constant angle offset from straight ahead, which has the effect of minimising the drift of the retinal image of the prey (Olberg, Worthington & Venator 2000).

When in pursuit, male hoverflies and dragonflies pursue targets from below and behind, fixating their targets in the dorso-frontal visual field (Nordström & O'Carroll 2009). By doing so, the target can generally be visualised against the open sky, improving contrast. This behaviour correlates with the fact that male hoverflies have a region in their field of view wherein light capture is increased, termed the "acute zone" or the "bright zone" (Nordström & O'Carroll 2009; Straw, Warrant & O'Carroll 2006). This adaption possibly improves contrast of targets in this region, and male hoverflies keep females fixated in this zone when in pursuit (Straw, Warrant & O'Carroll 2006).

Hence, trajectories that result from male hoverflies chasing females would be expected to show the target being kept in the upper-frontal part of the display.

2.3. Characterising and Modelling Flight

In order to characterise and model the complex flight behaviour of hoverflies, numerous studies have attempted to break it down into simpler pieces. The same approach has been taken in describing other types of complex motion, such as defining the movements of mice in terms of simple sequences that occur often, recognising movements that make up sign language, as well as in building locomotion from small movements (Braun, Geurten & Egelhaaf 2010; Geurten et al. 2010).

The flight of many species of flies, as well as that of birds and bees, has been shown to consist of short fast turns, called saccades, and segments of non-turning flight in between (Boeddeker et al. 2005; Eckmeier et al. 2008; Geurten et al. 2010; Geurten, Kern & Egelhaaf 2012; van Hateren et al. 2005; Tammero & Dickinson 2002). Geurten et al. (2010) further classified the flight of hoverflies into nine 'prototypical movements,' which form basic building blocks to describe flight sequences. Braun, Geurten, and Egelhaaf (2010) also were able to classify blowfly flight into nine prototypical movements.

As outlined previously, the heading of the insects will have to be estimated in order to provide an accurate simulation of the original flight. With the paths of motion provided in the three-dimensional position data, the prototypical movements and the transition probabilities between these prototypical movements can be used to better inform the estimated heading throughout the flights. Figure 2.1 shows the prototypical movements determined for hoverflies.

Images removed due to copyright restriction.

Figure 2.1: Prototypical movements determined for a hoverfly (Geurten et al. 2010). Left: a diagrammatic representation of the colour-coded prototypical movements. Right: a schematic explanation of the terms of reference for the colour coding.

Geurten et al. (2010) found their hoverfly prototypical movements applied largely equivalently in two different sizes of flight arenas. Boeddeker et al. (2005) also found that blowflies exhibited the same flight styles whether confined in a cage or observed in the wild. Geurten, Kern, and Egelhaaf (2012) compared the data from Geurten et al. (2010) and Braun, Geurten, and Egelhaaf (2010), concluding that hoverflies and blowflies both separated flight into saccades and straight flight, and that the optic flow during saccades was similar for both species. Despite this, the optic flow in straight flight was very different between the two species. As a result, blowfly movement sequences cannot be used to characterise the flight of hoverflies, but those of hoverflies can.

Although these prototypical movements were initially promising for this project, they rely on knowledge of the heading of the insect. Instead of assigning a prototypical movement

to a section of flight based on its heading, it is perhaps possible to instead use a distribution of prototypical movements over the entire flight. The heading could then be determined to match that section of flight to the prototypical movement, but this will have to be explored in further work.

2.4. Trajectory Reconstruction and Flight Simulation

The stimuli displayed to insects under study classically has consisted of either moving stripes or moving dots. Prior to the advent of these stimuli being generated electronically, these stimuli were the only stimuli able to be generated mechanically. The aim of this project is to produce a stimulus that will be more naturalistic than these simple illusions of motion.

Displaying a reconstruction of flight to hoverflies as a stimulus has been attempted before (Boeddeker et al. 2005; Geurten, Kern & Egelhaaf 2012; van Hateren et al. 2005; Lindemann et al. 2003; 2008; Tammero & Dickinson 2002). However, with the exception of Boeddeker et al. (2005), the insects from which these simulations were reconstructed were filmed flying in cages. As a result, the reconstructed flights could only show the interiors of the cages in which they were filmed. Boeddeker et al. (2005) were able to film blowflies outside, but due to the physical limitations of the camera setup, the video could only be replayed at 1 frame per second. This is well under the temporal resolution of fly vision, and had the effect of over-exaggerating the motion of the background surroundings. Thyselius et al. (2018) recorded hoverflies outside at 120 frames per second, and so this problem would not be present in the data provided. However, it must be noted that for different experimental setups, different display frequencies may be used for stimuli, and so the final product must be able to be resampled to match the display.

As has been outlined previously, in order to simulate flight, the original trajectory and heading must be established from the data provided, which consists in this case only of points in three-dimensional space. This is certainly possible, as shown by O'Neill (2000), who used only the position and acceleration data from a flight recorder to accurately drive a helicopter flight simulator.

In order to create a realistic simulation, a realistic path must be constructed between the positional points. Al-Jarrah and Hasan (2009) investigated a method for an unmanned aerial vehicle to plan a flight path between 3D waypoints in real time, making reference to Dijkstra's algorithm. This algorithm is used to find the shortest path between points, and may be useful in establishing a baseline path that can be adjusted according to the movements realistically available to the hoverflies to create a path through the points.

For direct applicability to hoverfly flight, the prototypical movements mentioned in 2.3. Characterising and Modelling Flight, found by Geurten et al. (2010), would be the main basis for adjusting the path flown by the hoverflies. Geurten et al. (2010) established a Markov model for the hoverfly movement sequences, which is a set of discrete states and the probabilities of transition between those states (Wallisch et al. 2014). By creating a Markov model, it is then possible to apply the Viterbi algorithm, used to determine the

most probable path through the Markov states taken to generate a set of observed outputs (Wallisch et al. 2014).

Both Markov models and the Viterbi algorithm are straightforward to implement in MATLAB, but simple Markov models may not describe the system fully. Hidden Markov models are extensions of simple Markov models that describe the probability of each state not only transitioning between states, but also of outputting multiple possible outputs, and are able to model systems more comprehensively (Wallisch et al. 2014). These hidden Markov models can then be refined using the Baum-Welch algorithm to determine each state's probability of output. In this way, the Markov model provides a method to not only describe the reconstructed flight sequences but also measure their accuracy.

For this project, the observed outputs would be the three-dimensional positions recorded by Thyselius et al. (2018). A Markov model could be constructed using the states and the transition probabilities from Geurten et al. (2010). The Viterbi algorithm and the Baum-Welch algorithm would then be applied to estimate the most probable path taken by the hoverfly to pass through those points, also giving an estimation of the heading between the points. This information could then be used to construct a simulation.

Geurten et al. (2010) made a comparison between the prototypical movements and transition probabilities to words and the grammar connecting those words in natural language. In this project, the data provided gives only the words, and the grammar from established literature will have to be applied in order to make meaningful sense of the story.

2.5. Display of Optic Flow

If the simulations are to be displayed in real-time, as captured by Thyselius et al. (2018), they will appear very fast to the human eye. For comparison, dragonfly pursuit flights were found to have an average duration of 184 ms, calculated from the point where pursuit was commenced to capture (Bagheri et al. 2015). The simulations from this project can be expected to last only a few seconds, and display motion at a very fast rate.

Reproducing naturalistic visual stimuli in an experimental setting is complicated, as existing visual displays are limited in terms of the naturalness of the stimuli they can be used to display (Dyakova & Nordström 2017). In order to evaluate insect vision in the natural world, the natural world must first be able to be displayed. LED arenas, commonly used to display optic flow, have sufficient temporal resolution, however, their spatial resolution is generally too low for the optical resolution of hoverflies (Dyakova & Nordström 2017). The spatial resolution of LED arenas depends on the size of the LEDs used in construction, which generally ranges between 2 and 4 degrees in diameter (Henriksson 2010a). Hoverfly eyes have a spatial resolution of 1 to 2 degrees, which can be approximated as the width of a thumb at arm's length (Henriksson 2010a; Nordström, Bolzon & O'Carroll 2011).

The temporal resolution of hoverfly vision is more than double that of a human's, at about 125 Hz compared to around 50 (Henriksson 2010a). Temporal resolution can become an issue when the motion between two successive images in an animation is too large,

thereby appearing jerky (Henriksson 2010a). One way to combat this is to use motion blur between the two positions, effectively smoothing the motion by reducing the change between frames (Henriksson 2010a). This feature is not implemented in FlyFly at present.

Bagheri et al. (2014b) used a cylindrical arena to display simulated motion of a target against a static background. The insect was able to pursue the target within the arena, with the target being manipulated to not collide with the walls of the arena. The target was also manipulated such that it would not be lost from view each time it turned (Bagheri et al. 2015). This would artificially force the target to remain in the field of view at all times. Not only was the target's position maintained within the viewport of the camera, rotations were also performed in order to keep the target fixated no more than 5 degrees from the centre of the frontal field of view (Bagheri et al. 2015). As a result, the target was always near the centre of vision.

2.6. Target Detection in Visual Clutter

Dipteran target-selective descending neurons (dTSDNs) are unresponsive to targets moving in visual clutter, unless the background meets certain requirements (Nicholas et al. 2018). Diptera is the order to which hoverflies, blowflies, and fruit flies – among other flies – belong. Naturalistic backgrounds suppress the responses of dTSDNs, and hence dTSDNs are responsive to backgrounds with un-naturalistic spatial characteristics (Nicholas et al. 2018). Additionally, if the background is perceived to be moving slowly or in the opposite direction relative to the target motion, dTSDNs will respond.

It is suggested that the unresponsiveness of dTSDNs to target motion in the same direction as that perceived to be of the background is due to the rarity of this occurring in actual target pursuits (Nicholas et al. 2018). For target motion and perceived background motion to be in the same direction, the observer must be turning away from the target, causing the background to appear to move in the opposite direction.

By hovering, the perceived background motion due to self-motion is minimised, maximising the dTSDN response, and so hoverflies often detect their targets whilst in a hovering flight pattern (Nicholas et al. 2018; Nordström & O'Carroll 2009).

In addition to TSDNs, small target motion detector (STMD) neurons have been described in hoverflies, which exhibit strong responses to the motion of small targets against backgrounds with visual clutter (Nordström, Bolzon & O'Carroll 2011). These STMDs exhibit an ability to track targets moving in non-continuous paths until the interruption is higher than a threshold of approximately 7 degrees (Nordström, Bolzon & O'Carroll 2011).

Additionally, neuronal response was found to build up as targets moved along continuous trajectories (Bagheri et al. 2015). In contrast to TSDNs, some STMDs respond to targets even when they move without a velocity relative to the background (Nordström & O'Carroll 2009). Pursuits that failed to end in capture were found to be generally due to inadequate target detection rather than unsuccessful pursuit (Bagheri et al. 2014a).

If a hoverfly were able to perfectly follow the motion of its target, keeping the image of the target perfectly in the centre of view, the target would not be perceived to move at all

(Nicholas et al. 2018). Instead, the background would continue to be perceived to move. In order to investigate this hypothesis, reconstructed target pursuits can be replayed.

2.7. Heading Estimation and Machine Learning

In order to determine the heading of the hoverflies, their 3D orientation (in addition to their position) can be estimated. This combination of 3D orientation and position is referred to as pose, and pose estimation is a widespread challenge for computer vision. Pose estimation is used in automated industrial picking, human feature detection, and autonomous vehicle egomotion estimation, visually informing the vehicle of its own motion.

For ground vehicles, egomotion estimation techniques make use of the limited degrees of freedom to specify three-dimensional movement as a combination of forward translation and side-to-side yaw (Angelopoulou & Bouganis 2014). This is insufficient for aerial motion, where translations and rotations alike can be performed in three dimensions, with six degrees of freedom. When using optic flow to determine self-motion, avoiding rotational motion can simplify the estimation of linear velocity from translational motion (Raudies 2013). Hence, heading becomes the direction of linear velocity, disregarding rotational motion (Raudies 2013).

Estimating 3D pose from a single image is inherently an under-constrained problem, as many 3D poses can project to the same image (Arnab, Doersch & Zisserman 2019). Similarly, the same object projects different silhouettes according to its pose (Liu et al. 2012). With the data provided, the ground truth of the pose of the insects was not available, even given the video footage. Liu et al. (2012) circumvented the absence of ground truth by comparing the pose estimated from all eight cameras used with poses estimated from differing numbers of cameras, hence estimating the ground truth itself.

With a 3D model of a textured object, feature points can be matched between the model and images to estimate the pose of the object in the image (Mitash, Boularias & Bekris 2019). However, this requires good lighting conditions and a 3D model of the object. When pose estimation is performed for industrial parts, 3D CAD models are usually readily available, and when not already existing, the number of identical parts processed lends to easy and accurate model acquisition (Liu et al. 2012). In this case, 3D models are neither already existing nor readily available for *Eristalis tenax* hoverflies, let alone the other species of insects that were participants in the filmed interactions.

In the field of human pose estimation, most existing datasets were collected in constrained lab environments, with uniform lighting, background, and viewpoints (Yang et al. 2018). In the context of this project, none of these characteristics are exhibited by the data. The data is instead typical of “in the wild” footage, with features such as occlusion, poor lighting, and motion blur (Arnab, Doersch & Zisserman 2019). Most pose estimation algorithms, particularly those trained on uniform datasets, fail to estimate pose in the wild (Arnab, Doersch & Zisserman 2019).

An alternative method to quantitative 3D pose estimation is ordinal depth assignment. By specifying which of two key points on an object is in front of the other, the relative depth ordering of each of the key points selected for an object can be assigned (Arnab, Doersch & Zisserman 2019). In this way, a 3D pose for the object can be estimated.

Access to a large training dataset helps to both speed up an algorithm's learning, as well as improve its applicability to testing data (Liu et al. 2012; Ukita & Uematsu 2019). When a large dataset is used, however, this data must be labelled, which requires a significant amount of manual effort (Kawana et al. 2018; Mitash, Boularias & Bekris 2019). An alternative is to use synthetic data, however, there is an inherent difference between the theoretical examples synthesised as training examples and real-world testing data (Mitash, Boularias & Bekris 2019).

3. Application and Implementation

3.1. Project Management

Originally, this project was planned for completion in May 2019. However, due to the limitations of the project, in particular those of the data, there were many challenges.

Initially, all of the elements described in the literature review were to be included, such as using prototypical movements and transition probabilities to validate heading estimations by measuring distributions of prototypical movements. However, the prototypical movements could not be determined without assigning a local coordinate axis to the fly, which in turn required an estimate of the heading.

Machine learning was also considered, in particular the use of computer vision. This was in fact attempted, with considerable time spent trying to label at least the orientation of visible hoverflies in the video footage. Ultimately this did not produce fruitful results.

Path planning, using flight dynamics in combination with behavioural studies, was also originally desired. Ideally, a flight strategy was to be developed, determining the heading most likely used to traverse between two successive data points.

These additional requirements led to scope bloat, pushing the endpoint of the project out to October 2019. Determination of pursuit was added at this point in order to simplify the project, removing these additional complicating features.

When reviewing the literature, there were even further methods that were found that could prove useful, but these must be explored in future work.

3.2. Aims and Specifications

The specifications for the end product of this project were quite straightforward. First, the program must accept the datasets provided by Thyselius et al. (2018) as input. The user must be able to select which of the two participants they wished to be the observer, and the other would thus be the target.

The output of the program would be a simulation of the recorded flight as would have been experienced by the insect itself. It was also specified that the program should be as straightforward to use as the existing software, and provide similar functionality, such as allowing for experimental variation.

One key specification was that the program must allow the user to adjust the output of the program, as they may have additional information or knowledge.

3.3. Deliverables and Requirements

The main deliverable of the project is the program itself, with the specified input, output and functionality as described above. It was established as a further requirement that data produced by the program should be saved for future use, and that the program should integrate with the FlyFly environment.

4. Methods

Initially, computer vision was investigated as an option for estimating the heading of the hoverflies. As the data consisted not only of the 3D trajectories but also the video recordings of the interactions, it was proposed that the videos could be used as a ground truth to be used to evaluate the accuracy of the heading estimation from the trajectories.

However, as illustrated in Figure 1.1, and outlined in 2.1. Tracking Flight and 2.7. Heading Estimation and Machine Learning, the resolution of the videos did not allow for the tracking of multiple points on the bodies of the insects. Indeed, even tracking the centre of mass of the insect was not a straightforward task, for both the computer and the user.

Johnson and Everingham (2010) were able to crop and scale labelled body parts in their experiments so that they were all roughly 180 pixels in length. In some parts of the videos for this project, however, the insects are indistinguishable from the background, and are only a few pixels in size. As a result, there was no way a point cloud could be registered to represent a model of the insect's body and thus determine its orientation.

The use of a visually uniform background facilitates object detection and tracking. The more different the colour of the background to the objects that are to be tracked, the less difficulty there will be differentiating between object and background (Acosta 2010). However, it is considered too restrictive for a general system to use a plain background, and algorithms trained on plain backgrounds will not be able to perform in natural situations (Acosta 2010). As the data used in this project was collected in the wild, the background present is naturally visually cluttered.

4.1. Data Validation and Pre-Processing

Firstly, the data had to be pre-processed to ensure it was valid. The original 16 datasets provided were uniformly in `.mat` format, with the 3D positions through the duration of the interaction given as a two-matrix cell. These matrices were of the same length, each having three rows. These rows specify the X-, Y-, and Z-coordinates, respectively, at each frame for each participant. Figure 4.1 shows part of a typical dataset.

all_3Dcordinates{1, 1}

	1	2	3	4	5	6	7	8	9	10	11
1	57.3566	58.3969	57.3164	57.6137	57.6209	58.4229	59.5509	63.1064	63.9993	63.4618	63
2	858.0259	856.4084	858.3907	857.7071	858.1820	857.0737	855.9323	850.6198	850.4487	850.7287	851
3	-123.3357	-123.8782	-123.5232	-123.9036	-124.0322	-124.0293	-124.5582	-124.7647	-124.8753	-125.3181	-125
4											

all_3Dcordinates{1, 2}

	1	2	3	4	5	6	7	8	9	10	11
1	48.1613	48.6439	49.8283	51.2133	51.8422	52.4808	53.3276	54.6777	57.2353	57.8441	59.
2	900.2174	897.3857	896.1029	893.4990	892.2531	891.5368	890.1174	888.4522	885.2879	884.5531	881.
3	-57.5036	-59.2066	-58.1201	-58.4376	-58.7024	-59.0444	-59.9025	-60.4363	-62.2038	-65.0056	-64.
4											

Figure 4.1: The first few elements of a typical, valid, dataset. Above: matrix 1 of the two-matrix cell. Below: matrix 2 of the two-matrix cell.

At this stage, the dataset must be valid: it must be a two-matrix cell, with each matrix being the same size and having three rows. This is to ensure that there is XYZ data for both of the two participants over the entire interaction. For the purposes of this project, data that was deemed to be invalid was discarded. However, in future, the user should have the ability to edit invalid datasets. This could entail the program extrapolating from the present data in order to fill the gaps. As in 3.2. Aims and Specifications, the user must always retain control over how the data is adjusted.

A further 90 datasets were provided for analysis over the course of the project. When preparing the 106 datasets, 16 were found to be invalid. Datasets were given an ID according to the order in which they were taken from each individual folder of recordings. Each dataset is from a separate instance, however, some datasets come from different points in the same recording. Table 4.1 lists which datasets were invalid.

Table 4.1: Datasets found to be invalid during pre-processing. These invalid datasets are invalid as they do not consist of two matrices.

ID of invalid dataset
001
002
003
007
037
038
039
040
041
046
074
075
076
077
078
079

These datasets were unable to be used for the purposes of this experiment. Again, however, in future these datasets should be able to be edited and used in simulation. The validation code additionally relies on the imported dataset being of a specific format. For future use, this will have to be rectified in case datasets of different origin are used. If the datasets are deemed invalid, the program does not proceed.

4.2. Trajectory Reconstruction and Heading Estimation

After choosing the dataset to be processed, the user is able to specify which participant is to be used as the basis of the reconstruction. There are competing systems for how to term

the two participants – e.g. target and pursuer (Olberg, Worthington & Venator 2000; Wardill et al. 2017), occupant and challenger (Thyselius et al. 2018), fly A and fly B. In the context of this program, observer and target are used. In this sense, the observer is the fly to whom the stimulus will be displayed. The target is the fly whose motion will be displayed relative to the observer's own position. The user selects this by specifying which matrix (1 or 2) represents the observer, with the other matrix representing the target.

It is also important to highlight the difference between motion and position. The datasets provide information on only the *positions* of the observer and target at discrete points in time. The direction of *motion* between those positions can only be inferred from the data points given. For this project, direction of motion was taken to be the straight line between position data points.

In a similar vein, the heading through the flight must also be inferred. The frequency at which the data points were captured is important to determining the direction of motion and heading, as well as to the accuracy of the simulation. The user is given the ability to specify this frequency, although Thyselius et al. (2018) claim the sampling frequency of the 3D position data to be 120 s^{-1}. However, it should be noted that 120 Hz is below the upper temporal resolution limit of flies (Fry et al. 2008; Henriksson 2010a), and so ideally a higher capture frequency would be preferable.

The intention of this code is to provide an informed estimate of the heading of the observer, in order to accurately reconstruct the optic flow and the target motion. From there, the flight from the perspective of the observer can be simulated.

FlyFly uses gluPerspective in its 3D projection algorithm, which makes the assumption that the user is positioned directly in front of the screen, facing perpendicular to it, and looking directly at the centre of the screen (Kooima 2008). In more general terms, perspective projection requires the viewing point to lie on a coordinate axis intersecting with the view plane (Power 1996).

In its current state, the code aligns the observer's heading with either the observer's direction of motion ('DOM') between position coordinates, or the straight line (line of sight, 'LOS') between the observer and target at each position coordinate. In future, an algorithm will be added to the program that will use the trajectory to estimate the heading.

An ID was assigned to the resultant trajectory, depending on the method used to generate it. If matrix 1 was selected as the observer, `obs0`, representing a value of 0 for the `observerFlag` variable, used throughout the code, was appended to the ID. Similarly, if matrix 2 was selected as the observer, `obs1` was appended to the ID. Then, if the direction of motion method was used to rotate the heading, `dom` was further appended to the ID. Otherwise, `los` for line of sight was appended. Hence, as an example, if matrix 1 in dataset number 043 were selected as the observer, and the line of sight method was used to rotate the heading, the resultant ID would be `043obs0los`.

By rotating the observer's estimated heading to the Z-axis, the coordinate axes for the simulation are also rotated such that the aforementioned perspective projection

requirements are met. 3D motion is then displayed as if a camera were attached to the observer through its flight, with forward motion displayed along the (negative) Z-axis. Figure 4.2 illustrates the process of rotating the coordinate axes in order to display flight from the perspective of the observer.

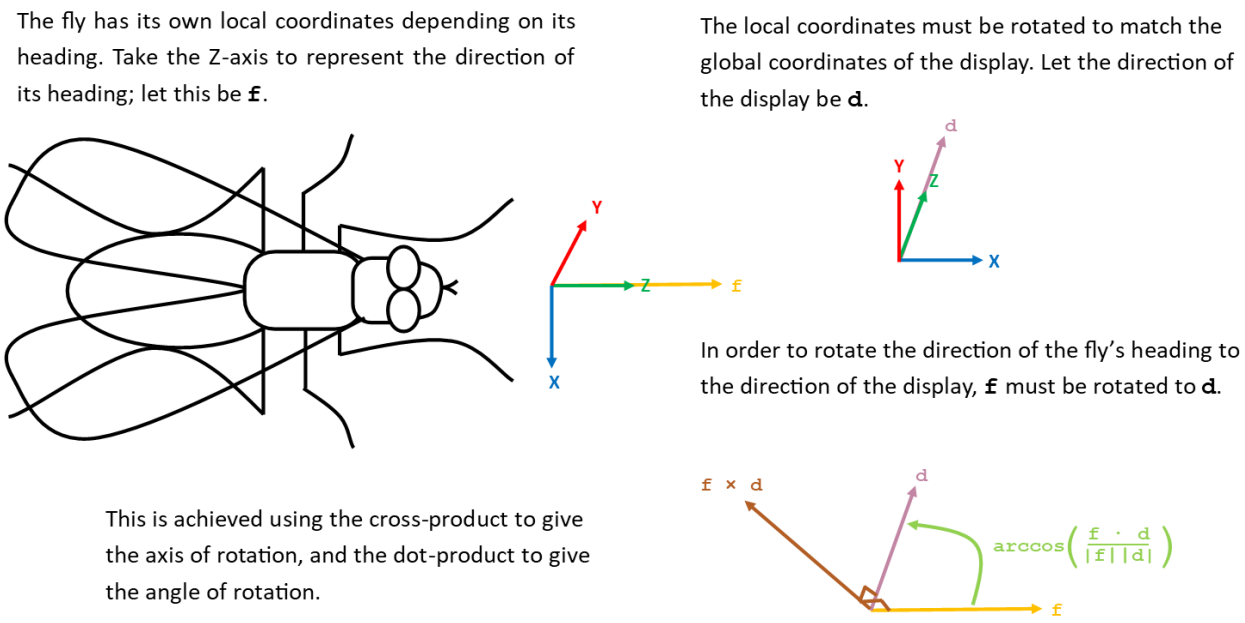

Figure 4.2: Rotating the observer's heading to look directly at the screen.

As illustrated in Figure 4.2, the same rotation applied to the observer's heading is also applied to the target's relative positions. For the code used in 4.1. Data Validation and Pre-Processing and 4.2. Trajectory Reconstruction and Heading Estimation, see Appendix A. Code.

4.3. Simulation

As of present, the code prints the relevant data to CSV to be imported into the 3D Target and Starfield FlyFly stimuli. For illustration, Figure 4.3 shows how the data is ready from the code, but has to be input into the 3D Target stimulus.

Image removed due to copyright restriction.

Figure 4.3: Data from the code ready to be input into the 3D Target stimulus, which has to first be initialised (Henriksson 2010b).

However, this involves changing the working directory if the stimulus code is located in a different folder to the FlyFly directory, opening up the layer manager, and inputting the appropriate number of trials. The user must then copy the data from each of the spreadsheets into FlyFly.

A far more convenient solution would be to have a stimulus accessible from the FlyFly main menu, that imports the data from the CSVs directly without needing extra setup. This must also retain the option of including background optic flow, which can presently be done by using the 3D Target stimulus alone *sans* Starfield. Including the code as a FlyFly stimulus would also then allow the capture frequency to be set as a stimulus setting, without requiring prompting every time the code is run.

It is also important to note that in display of the background optic flow, simultaneous rotations are prohibited. This is due to the mathematical implementation of rotation within the code, and changing this method would allow for simultaneous rotations.

It was noted that any rotation can be decomposed into a sequence of rotations about the three orthogonal axes (Liu et al. 2012), however, for the purposes of display, rotation was cut to only about one axis. It was decided that yaw would be used for changes in X, alongside lift and thrust for changes in Y and Z respectively. This is in line with hoverfly flight exhibiting yaw saccades (Geurten & Egelhaaf 2010; Land 1992). As a result, the reconstructions feature mainly translational optic flow, with some rotational components.

Another point to note is that the rotations used to align the heading with the z-axis are not physical rotations, and so do not produce rotational optic flow. These are mathematical rotations.

As the drawing uses the 3D Target and Starfield stimuli, the culling and clipping criteria apply. Off-screen targets are not drawn, nor are targets too close to the screen. This is a result of the glFrustum algorithm used in 3D projection. Objects that are too far away are clipped as they would otherwise be indistinguishable (Power 1996). This limit is set in the code, at 200 cm. Additionally, objects behind or too near to the eye are clipped (Power 1996). Again, this limit is set in the code, but also depends on the resultant dot size. This is in turn a factor of the graphics card used, and the limit set in the code is 6 cm. This also takes into account the standard experimental distance of the fly from the screen at 7 cm, although this too can be adjusted in the code.

For the purposes of this experiment, a variable was recorded to keep track of when the target would have been visible to the observer, ignoring the culling and clipping algorithms. This does not take into account the actual visual capabilities of the insect, considering hoverfly eyes can only resolve between 1 to 2 degrees (Henriksson 2010a). A dot of radius 1 cm, the default minimum dot size, at 200 cm, the default far clipping plane, would subtend an angular size of approximately 0.6 degrees, and so would be indistinguishable. Hence this method is only an approximation for foveation.

4.4. Determination of Pursuit

Pursuit is not clearly defined. Gibson (1957) defines pursuit as being the process of maximising the optical size of a target in the observer's field of view. On this basis, the target visibility variable recorded in 4.3. Simulation was used as a factor in determining whether or not an interaction was a pursuit. When chasing a target moving at speed, animals can either follow at a higher speed in order to catch up, or attempt to intercept the target's path (Fabian et al. 2018).

Male hoverflies use a deviated pursuit strategy when chasing females, maintaining the perceived target position at a constant non-zero angle from straight ahead (Fabian et al. 2018). Hence, simulations based on data taken from male hoverflies pursuing females should show the observer attempting to keep the target at a constant non-zero angle from straight ahead.

Pursuit was determined according to the footage of the interactions, giving a simple 1 or 0 for likely to be a pursuit or not. The MATLAB Regression Learner toolbox was then used, with input as the proportion of the interaction that the target was visible, along with the number of frames between the target being visible. The target speed was also used as a factor, as this was calculated from the relative distance between the observer and target.

The average change in target speed was used to represent the change in distance between observer and target from frame to frame, and so a more negative number would suggest an overall closing of distance. 44 of 344 datasets were chosen at random as test sets, with the remaining 300 datasets left as training sets, giving a ratio of test sets to training sets of 0.147.

5. Results

As outlined in 4. Methods, 16 of the 106 datasets were invalid and so could not be further processed for testing. Similarly, after processing datasets to prepare input for the simulation, 52 of the 360 processed trajectories contained not-a-number ('NaN') values. These NaN values cause the simulation to crash as they are unsupported in cells in MATLAB, and so they were edited out of the trajectories, skipping frames with NaN values.

The simulations were still run with these trajectories, and the data recorded as if the trajectories were unedited. Table 5.1 lists the trajectories that had NaN values edited.

Table 5.1: Trajectories from which NaN values were removed in order to allow the simulation to not crash.

ID of trajectory containing NaN value
008obs0dom
009obs0dom
010obs1dom
013obs0dom
014obs0dom
014obs1dom
017obs1dom
019obs0dom
020obs1dom
021obs1dom
022obs1dom
023obs0dom
024obs1dom
025obs0dom
026obs0dom
028obs0dom
030obs0dom
031obs0dom
036obs1dom
045obs0dom
049obs1dom
051obs0dom
053obs0dom
054obs0dom
057obs0dom
058obs0dom
061obs0dom

ID of trajectory containing NaN value
062obs0dom
064obs1dom
067obs0dom
068obs0dom
085obs1dom
088obs0dom
090obs0dom
091obs0dom
092obs0dom
093obs0dom
094obs1dom
095obs0dom
096obs0dom
097obs0dom
099obs0dom
100obs0dom
102obs0dom
102obs0los
102obs1dom
103obs0dom
103obs1dom
104obs1dom
105obs0dom
105obs1dom
106obs0dom

In addition to these 52 trajectories that could be edited, there were 12 more trajectories that consisted entirely of NaN values. The simulation could not be run at all using these trajectories, hence they were deemed invalid. These, too, were discarded. It is unclear what caused these trajectories to contain NaN values, and even more unclear what caused the 12 trajectories to be entirely invalid. Table 5.2 lists the trajectories that consisted entirely of NaN values.

Table 5.2: Trajectories that were deemed invalid as they consisted entirely of NaN values.

ID of completely invalid trajectory
044obs0dom
044obs0los
044obs1dom
044obs1los
084obs0dom
084obs0los
084obs1dom
084obs1los
087obs0dom
087obs0los
087obs1dom
087obs1los

As a result, only 348 trajectories could be used in simulations. Of these, 296 were unedited. However, although the datasets were valid for simulation, 70 of 348 trajectories resulted in the simulation encountering an error. These errors occurred at the end of the simulation, and seemed to be only an error in counting the number of frames dropped in simulation, although the cause is again unclear. Hence, these trajectories were also used in analysis, but tagged as having failed. Table 5.3 lists the trajectories which led to a failed simulation.

Table 5.3: Trajectories resulting in an error in simulation.

ID of trajectory resulting in failed simulation
004obs1dom
005obs0los
006obs0dom
006obs0los
010obs1los
012obs1dom
012obs1los
014obs0dom
015obs0los
016obs0los
019obs0los
020obs0los
020obs1los
023obs0dom
024obs0dom
025obs1los

ID of trajectory resulting in failed simulation
027obs1los
029obs1los
031obs0los
032obs0los
033obs0los
035obs1dom
035obs1los
036obs0dom
036obs0los
042obs0los
042obs1dom
042obs1los
043obs0los
045obs1los
047obs0dom
050obs0dom
050obs1dom
053obs1dom
054obs1dom
055obs1los
056obs1dom
066obs1los
069obs0dom
070obs0dom
070obs0los
071obs1los
072obs0dom
081obs1los
083obs0los
083obs1dom
092obs0dom
092obs0los
093obs0los
093obs1dom
094obs1dom
094obs1los
096obs1dom
096obs1los

ID of trajectory resulting in failed simulation
098obs1dom
099obs0dom
099obs1dom
100obs0dom
101obs0dom
101obs0los
102obs0dom
102obs1dom
102obs1los
105obs0dom
105obs0los
105obs1dom
106obs0dom
106obs0los
106obs1dom
106obs1los

At this point, all data had been collected from simulation. See Appendix B. Summary of Results for a full table of results.

Kruskal-Wallis one-way ANOVA tests were then conducted on the datasets, to determine the homogeneity of the population. The first test considered the 348 datasets as a whole, and found it significantly unlikely that the datasets came from the same population ($p < 0.001$). All subsequent Kruskal-Wallis tests found each of the considered datasets to be unlikely to come from the same population. These findings were all at $p < 0.001$ except for those testing datasets using the line of sight method for the observer being matrix 1 (`obs0los`; *vide* 4.2. Trajectory Reconstruction and Heading Estimation). This finding was at $p < 0.05$. See Appendix C.1. Kruskal-Wallis for a fuller description of the results of the Kruskal-Wallis tests.

However, Kruskal-Wallis tests for equality of medians, with the assumption made that the distributions were the same. With the null hypothesis being rejected, the medians were found to be unequal, hence the conclusion that at least one if not several of the many datasets considered did not come from the same population.

For a more powerful analysis, Conover-Iman pairwise tests with Bonferroni correction were then used. When the entire dataset was considered as a whole, 627 of 58996 comparisons were found to be significant, i.e. approximately 1.1% of datasets were found to significantly be unlikely to come from the same population.

Comparisons within methods gave differing results. The direction of motion method found 489 of 14706 (approximately 3.3%) comparisons to be significant, whereas the line of sight method found only 51 of 14706 (0.3%) comparisons to be significant.

In line with the finding from the Kruskal-Wallis tests that `obs0los` had the least significant rejection value of all comparisons, the Conover-Iman test found all of the data from this group to support the null hypothesis. Hence, this dataset can be said to be homogeneous. Because of this homogeneity in results from inputs that should be part of the same set, the method can be suggested to be consistent.

In contrast, the direction of motion method applied to the same set of trajectories returned 98 of 3655 (2.7%) of comparisons as significant. For comparison, the same two methods applied to the observer as matrix 2 found 40 of 3655 (1.1%) and 185 of 3655 (5.1%) respectively to be significant. When considering the group of trajectories where the observer was set as matrix 1 as a whole, 114 of 14706 (0.8%) were found to be significant, compared to the `obs1` population, finding 324 of 14706 (2.2%) of comparisons to be significant. See Appendix C.2. Conover-Iman for the full results of the Conover-Iman tests.

Overall, setting the observer as matrix 1 gave more consistent results regardless of method of heading estimation. That being said, all datasets are fairly homogeneous, with all except the `obs1dom` population having at least 95% supporting the null hypothesis.

As outlined in 4.4. Determination of Pursuit, the MATLAB Regression Learner toolbox was used to model an algorithm for determining pursuit, based on the factors listed previously. See Appendix D. Determining Pursuit for a full table of the data supplied to the algorithm. The algorithm did not perform well, potentially due to the binary value of the output it was trying to predict. This will be discussed further in 6.1. Analysis. Of the four learning methods selected by the toolbox, linear regression had the lowest RMS error, of 0.49827. However, this is still very high. Figure 5.1 shows the response plot from the regression learner, with the supplied binary values of pursuit in blue, and the predicted values in yellow.

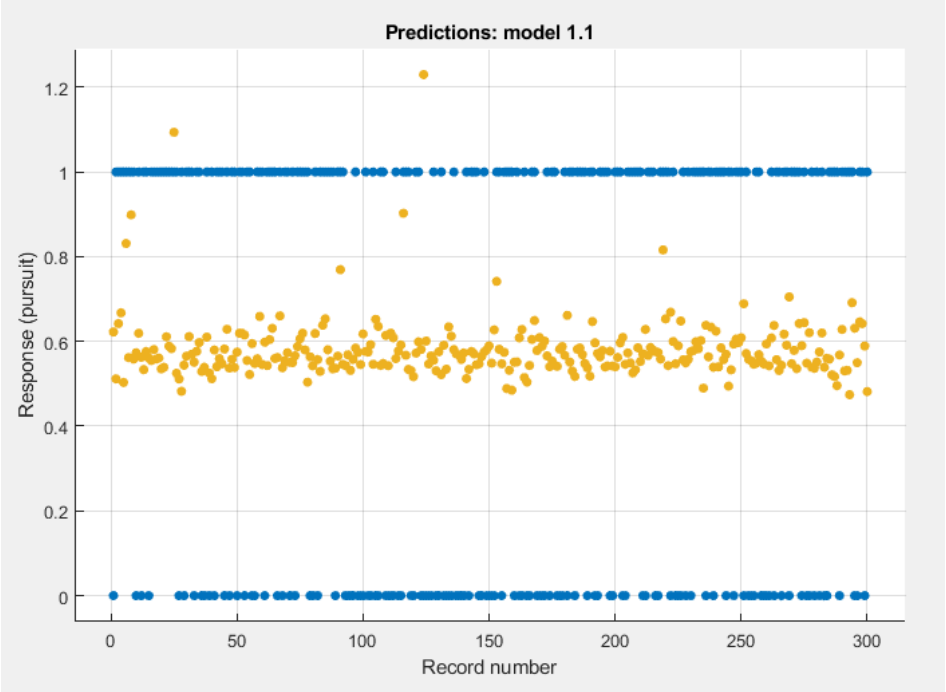

Figure 5.1: Response plot from the regression learner.

As can be seen in the response plot in Figure 5.1, the regression learner was fairly conservative, with most of its responses sitting around the median of 0.56. In fact, when the algorithm was applied to the 44-member test set, none of the values were below 0.5, as shown in Table 5.4. This meant that rounding could not be used to differentiate between what it had decided was a pursuit and what was not.

Table 5.4: The results of applying the linear regression to the test set.

Trajectory ID	Pursuit	Value determined by linear regression
027obs1los	0	0.579231739349939
048obs0dom	1	0.572569172039877
058obs1los	1	0.598728969640561
101obs1los	1	0.539723543989143
047obs1dom	1	0.514440608570384
105obs0los	0	0.544088691732749
030obs1dom	0	0.535310967203228
072obs0dom	1	0.561817484175091
068obs1los	1	0.584341461152492
058obs0dom	0	0.538635604803950
071obs1los	0	0.564804292741779
069obs0dom	1	0.550261569351687
020obs0dom	0	0.532805342145664
015obs1los	0	0.602190350089980
106obs1dom	1	0.528816311728632
019obs0los	0	0.593647887513588
006obs1dom	1	0.920370687481880
060obs0dom	1	0.567273211416535
096obs1dom	0	0.538414408627345

Trajectory ID	Pursuit	Value determined by linear regression
069obs0los	1	0.554058202006299
021obs0los	1	0.615332900748642
036obs0los	1	0.606149257074336
051obs0los	1	0.613869311325594
105obs0dom	0	0.562771136236102
018obs0dom	1	0.541648630431018
094obs0los	1	0.548770230957606
067obs0dom	1	0.576443526629890
042obs0dom	0	0.527491857267609
021obs0dom	1	0.528802373887515
048obs1dom	0	0.579622773521143
053obs0dom	0	0.670535474267514
015obs0dom	0	0.524725338994316
062obs0los	1	0.609717888124430
024obs0dom	1	0.578022847449142
042obs1los	1	0.577494326787606
061obs1los	0	0.546471654006704
026obs0los	1	0.584060632790005
029obs1dom	1	0.586047987171685
064obs1los	1	0.595610767921597
027obs1dom	0	0.603542056288610
091obs1dom	1	0.597314733300175
105obs1dom	0	0.556265434946909

Trajectory ID	Pursuit	Value determined by linear regression
080obs1dom	1	0.540426708263338
034obs0los	0	0.564925293786210

If the value given by the regression learner is to be interpreted as a probability of whether or not a pursuit, the regression learner can be said to mainly determine pursuits as having an essentially equal possibility of being or not being a pursuit. The conservativeness of the regression learner can be seen more clearly in Figure 5.2, where the data clearly bunches near 0.5, particularly for data that was tagged as not being a pursuit.

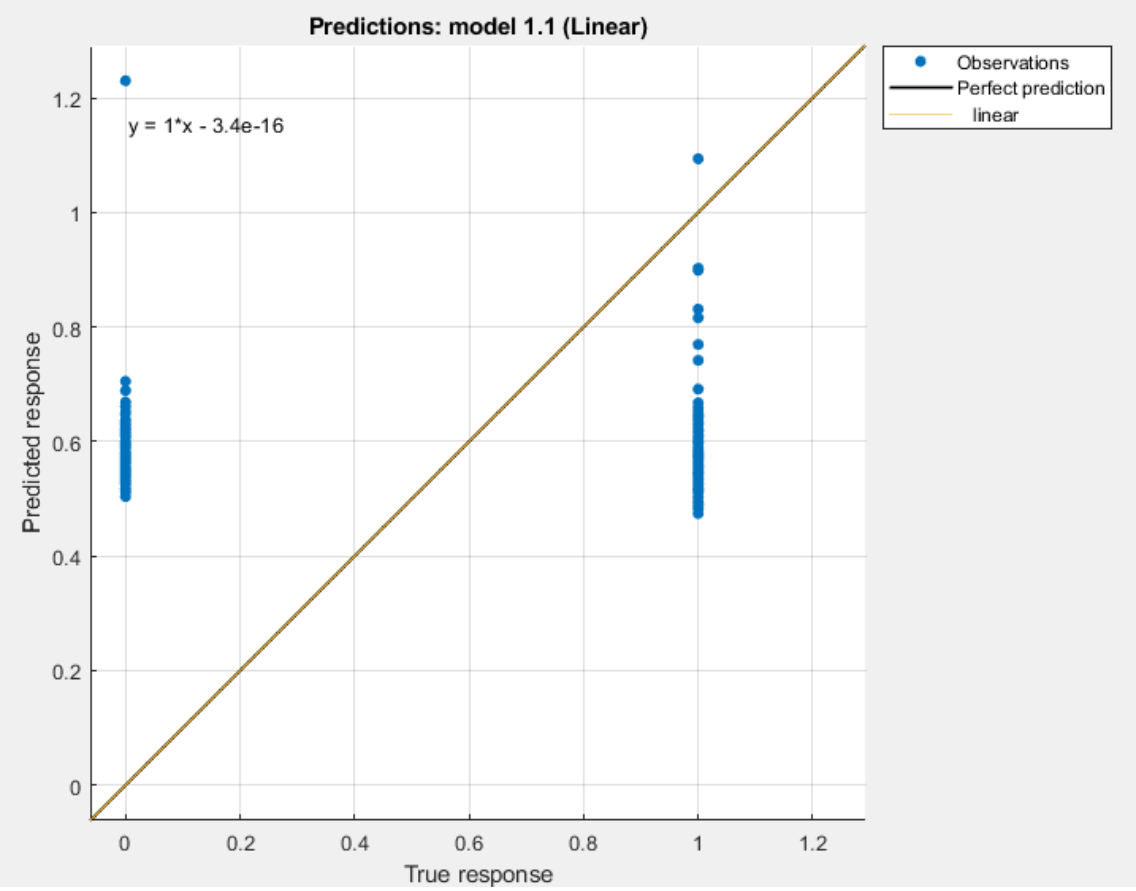

Figure 5.2: Predicted response vs. true response, as determined by the regression learner.

The equation of the line of best fit, shown in Figure 5.2, suggests that only the pursuit variable has an effect on whether or not the interaction is a pursuit. With such a small intercept, and a slope of 1, the line of best fit will predict a 1 when the pursuit variable is 1, or 0 if not.

When the algorithm was applied to the test set, similar results were produced. Figure 5.3 shows these results.

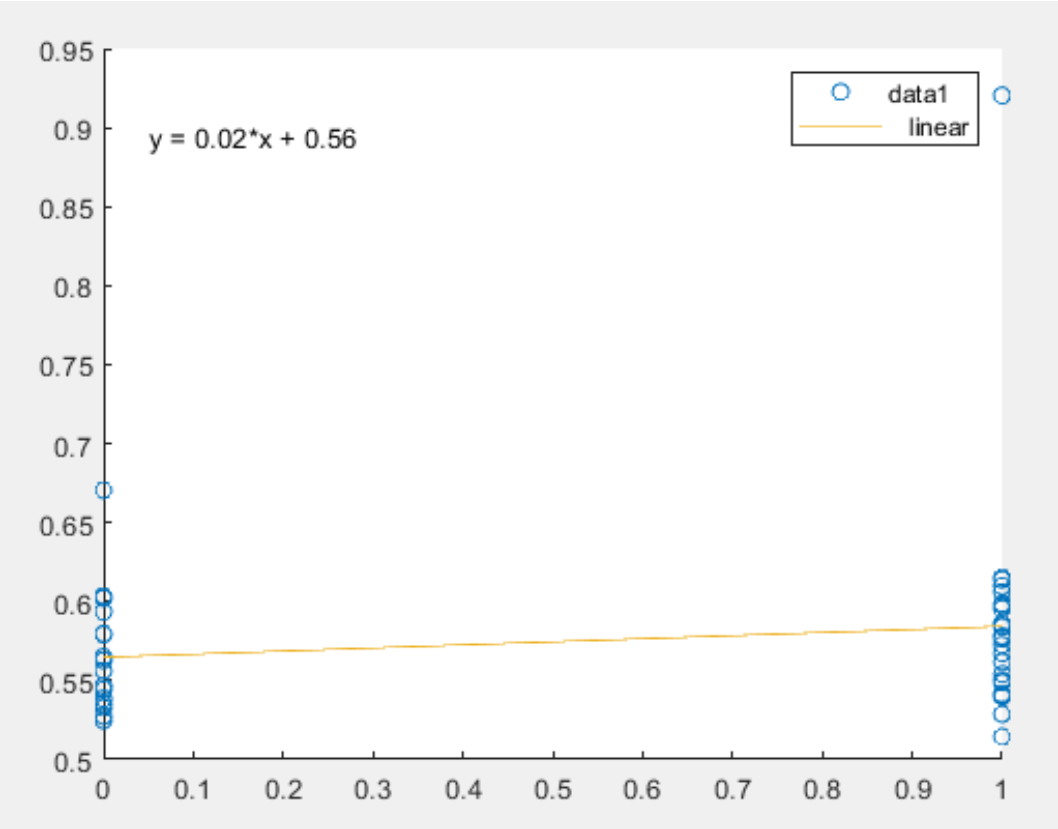

Figure 5.3: Using the linear regression algorithm on the test set.

Again, the data bunches around the median, indicating an equal probability of pursuit.

6. Discussion

6.1. Analysis

As outlined in 5. Results, the homogeneity of all but one of the datasets is remarkable. This suggests that the methods used are at least somewhat robust. The fact that the consistency was markedly higher for setting the observer as matrix 1 suggests that this data itself was more consistent. As the films always involved hoverflies interacting with other insects, be they other hoverflies or insects of other species, it could be reasonable to assume that the hoverfly was always tagged as matrix 1. As a result, data from matrix 1 across the interactions would exhibit a great degree of self-similarity, especially compared to that of matrix 2, where the species of insect may differ between interactions. Indeed, in some interactions the other participant was not an insect at all, but a bead on a string.

However, although the results from the statistical tests were impressive, the results from the linear regression were not. Whether this was a function of the data itself, or of the methods used to manipulate the data, is unknown. However, although some of the interactions were tagged as likely to be pursuits according to the footage, Thyselius et al. (2018) found it unlikely that any of the interactions were deliberate pursuits, according to the characteristics of the flights. It would be interesting to investigate the effect of changing the pursuit variable from being a binary yes or no to a probability or likelihood.

That being said, despite the intended use of the determination of pursuit as a metric for validation of the methods, the inconclusive results of the regression make it hard to discount the comparatively stronger results of the statistical tests.

In particular, the finding that the group of trajectories arising from matrix 2 data treated with the line of sight method were all significantly likely to come from the same population lends support to the validity of applying this method to this group of data.

From a deliverables point of view, the end product meets the original specifications. The program takes input in the form of 3D position datasets – although at this point they must be of a specific format – and estimates the heading along the trajectory in order to create a simulation of the flight as would have been experienced by the original fly. The user can select which insect will be at the centre of motion, and by using the FlyFly assets, experimental variation can be implemented.

6.2. Limitations and Improvements

The data used in this project had its limitations, as has been outlined in 1.1 Background, 2.1. Tracking Insect Flight, 2.4. Trajectory Reconstruction and Flight Simulation, and 2.5. Display of Optic Flow. In textured surroundings such as those filmed by Thyselius et al. (2018), increasing the resolution of the image generally increases the amount of information available, in contrast to images of “urban” scenes, with more visually uniform objects (Brinkworth 2009). In this sense, increased image resolution for caged flights would have little effect on the information provided, whereas higher resolution for the interactions filmed would give more information.

The data collection built upon the experimental setup used by Wardill et al. (2017), although in those experiments, cameras were placed at right angles to the anterior-posterior body axis of the fly, allowing for the pitch and yaw angles of the body to be measured. Additionally, a piece of white fabric was used as a backdrop in all experiments to improve contrast, even in the wild. Despite the similarities between the experimental setups, these differences that are also present have a great impact on the information contained in the data collected. If new data were to be collected for use with this program, ideally it should also contain information on the heading of the insects involved in the interactions, without needing to make an estimate.

There were further limitations in the way that the simulation was performed as part of this experiment. As outlined in 4.3. Simulation, rotational movements were vastly simplified. However, this does not necessarily accurately reflect free insect flight, as flight control depends on control of roll and pitch in addition to yaw (Goulard, Vercher & Viollet 2016). These two components of rotation were removed for simplification, but in future should be implemented if the rotation algorithm can be adjusted to make this possible.

An additional limitation was also found in the simulation. The default dot size used to draw targets on the screen is a 1 cm radius, being the lowest that this size could be set to. This size is used as the true size of the object, and is projected onto the screen accordingly. However, this is very likely to not be representative of the true size of the targets involved in the actual pursuits. That being said, this size was chosen as sizes any larger were not supported by the graphics card. As a result, simulations would crash irretrievably.

6.3. Future Work

Originally, a control strategy was to be developed to determine the heading based on the characteristics of the flight. Much research has been done to attempt to find algorithms to adequately describe the flight behaviour of insects. Fry et al. (2009) mentioned a PID control scheme underlying the control of flight altitude in honeybees. Franceschini, Ruffier, and Serres (2007) described an extremely simple method by which insects control their flight speed and altitude. It was proposed that insects use a simple ratio between ground height and ground speed during cruising flight, in order to maintain a constant optic flow experienced (Lawson & Srinivasan 2017). This also has the effect of controlling their air speed, effected by changing the pitch of their bodies.

Fabian et al. (2017) found proportional navigation sufficient to explain both timing and magnitude of steering responses exhibited by *Holcocephala* robber flies and *Coenosia* killer flies in pursuit. The gain constants and delays describing the proportional navigation controllers differed between species. It was suggested that other insects may also use proportional navigation with species-specific parameters. Future work on this project could investigate tuning a proportional controller to hoverflies, based on their trajectories, as well as ground height and ground speed, in order to determine their heading and thus optic flow.

Physical constraints about the actions animals take can also be used as information to constrain possible poses (Arnab, Doersch & Zisserman 2019).

7. Summary

There is further work still to be done on this project, with improvements that can be made in order to meet the original goals. In particular, investigating a control strategy may ease the process of estimating the heading of the hoverflies. This could be used to improve the performance of the algorithm developed with regression learning.

If new data at higher resolutions could be sourced, computer vision could be investigated. A ground truth could then be provided for the heading, or alternatively, a ground truth could be estimated, as mentioned in 2.7. Heading Estimation and Machine Learning.

Altogether, however, despite the challenges faced throughout this project, in particular the limitations placed upon the project due to the data provided and the tools to be used, a program was developed that meets the original specifications.

References

Ackerman, E 2018, 'Insect-inspired vision system helps drones pass through small gaps', *IEEE Spectrum*.

Acosta, VB 2010, '3D model based pose estimation in cluttered scenes' [Masters thesis], *University College London*.

Al-Jarrah, MA & Hasan, MM 2009, 'HILS setup of dynamic flight path planning in 3D environment with flexible mission planning using Ground Station', *Journal of the Franklin Institute*, vol. 348, pp. 45-65.

Angelopoulou, ME & Bouganis, CS 2014, 'Vision-based egomotion estimation on FPGA for unmanned aerial vehicle navigation', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 6, pp. 1070-83.

Arnab, A, Doersch, C & Zisserman, A 2019, 'Exploiting temporal context for 3D human pose estimation in the wild', *Cornell University archive: Computer Vision and Pattern Recognition.*

Bagheri, ZM, Wiederman, SD, Cazzolato, BS, Grainger, S & O'Carroll, DC 2014, 'A biologically inspired facilitation mechanism enhances the detection and pursuit of targets of varying contrast', *2014 International Conference on Digital Image Computing: Techniques and Applications (DICTA).*

Bagheri, ZM, Wiederman, SD, Cazzolato, BS, Grainger, S & O'Carroll, DC 2014, 'Performance assessment of an insect-inspired target tracking model in background clutter', *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV).*

Bagheri, ZM, Wiederman, SD, Cazzolato, BS, Grainger, S & O'Carroll, DC 2015, 'Properties of neuronal facilitation that improve target tracking in natural pursuit simulations', *Journal of the Royal Society Interface*, vol. 12, no. 108.

Boeddeker, N, Kern, R & Egelhaaf, M 2002, 'Chasing a dummy target: smooth pursuit and velocity control in male blowflies', *Proceedings of the Royal Society of London B*, vol. 270, pp. 393-399.

Boeddeker, N, Lindemann, JP, Egelhaaf, M & Zeil, J 2005, 'Responses of blowfly motion-sensitive neurons to reconstructed optic flow along outdoor flight paths', *Journal of Comparative Physiology*, vol. 191, no. 12, pp. 1143-55.

Braun, E, Geurten, BRH & Egelhaaf, M 2010, 'Identifying prototypical components in behaviour using clustering algorithms', *PLoS ONE*, vol. 5, no. 2, pp. 1-15.

Brinkworth, R 2009, 'Robust models for optic flow coding in natural scenes inspired by insect biology', *PLoS Computational Biology*, vol. 5, no. 11.

Collett, TS & Land, MF 1978, 'How hoverflies compute interception courses', *Journal of Comparative Physiology*, vol. 125, pp. 191-204.

Conover, WJ & Iman, RL 1979, 'On multiple-comparisons procedures', *University of California: Los Alamos Scientific Laboratory* [informal report].

Dyakova, O & Nordström, K 2017, 'Image statistics and their processing in insect vision', *Current Opinion in Insect Science*, vol. 24, pp. 7-14.

Eckmeier, D, Geurten, BRH, Kress, D, Mertes, M, Kern, R, Egelhaaf, M & Bischof, HJ 2008, 'Gaze strategy in the free flying zebra finch (Taeniopygia guttata)', *PLoS ONE*, vol. 5, no. 12, pp. 1-9.

Ellis, D 2014, 'Racer - Kings Island (forwards and backwards POV)' [YouTube video], viewed 24 September 2019, <https://www.youtube.com/watch?v=e4aGiW6AIDk>.

Fabian, ST, Sumner, ME, Wardill, TJ, Rossoni, S & Gonzalez-Bellido, PT 2018, 'Interception by two predatory fly species is explained by a proportional navigation feedback controller', *Journal of the Royal Society Interface*, vol. 15, no. 147.

Franceschini, N, Ruffier, F & Serres, J 2007, 'A bio-inspired flying robot sheds light on insect piloting abilities', *Current Biology*, vol. 17, no. 4, pp. 329-35.

Fry, SN, Rohrseitz, N, Straw, AD & Dickinson, MH 2008, 'TrackFly: virtual reality for a behavioural system analysis in free-flying fruit flies', *Journal of Neuroscience Methods*, vol. 171, no. 1, pp. 110-7.

Fry, SN, Rohrseitz, N, Straw, AD & Dickinson, MH 2009, 'Visual control of flight speed in Drosophila melanogaster', *The Journal of Experimental Biology*, vol. 212, pp. 1120-30.

Geurten, BRH, Kern, R, Braun, E & Egelhaaf, M 2010, 'A syntax of hoverfly flight prototypes', *The Journal of Experimental Biology*, vol. 213, no. 14, pp. 2461-75.

Geurten, BRH, Kern, R & Egelhaaf, M 2012, 'Species-specific flight styles of flies are reflected in the response dynamics of a homolog motion-sensitive neuron', *Frontiers in Integrative Neuroscience*, vol. 6, no. 11, pp. 1-15.

Gibson, JJ 1957, 'Visually controlled locomotion and visual orientation in animals', *British Journal of Psychology*, vol. 49, no. 3, pp. 182-94.

Golding, Y, Ennos, A & Edmunds, M 2001, 'Similarity in flight behaviour between the honeybee Apis mellifera (Hymenoptera: apidae) and its presumed mimic, the dronefly Eristalis tenax (Diptera: syrphidae)', *The Journal of Experimental Biology*, vol. 204, pp. 139-45.

Gonzalez-Bellido, PT, Fabian, ST & Nordström, K 2016, 'Target detection in insects: optical, neural and behavioral optimizations', *Current Opinion in Neurobiology*, vol. 41, pp. 122-8.

Goulard, R, Julien-Laferriere, A, Fleuriet, J, Vercher, JL & Viollet, S 2015, 'Behavioural evidence for a visual and proprioceptive control of head roll in hoverflies (Episyrphus balteatus)', *Journal of Experimental Biology*, vol. 218, pp. 3777-87.

Goulard, R, Vercher, JL & Viollet, S 2016, 'To crash or not to crash: how do hoverflies cope with free-fall situations and weightlessness?', *Journal of Experimental Biology*, vol. 219, pp. 2497-503.

van Hateren, JH, Kern, R, Schwerdtfeger, G & Egelheaaf, M 2005, 'Function and coding in the blowfly H1 neuron during naturalistic optic flow', *The Journal of Neuroscience*, vol. 25, no. 17, pp. 4343-52.

Henriksson, J 2010, 'FlyFly – a user friendly interface for MatLab and the Psychophysics toolbox' [Masters thesis], *Uppsala University*.

Henriksson, J 2010, 'FlyFly user manual' [reference document], viewed 21 October 2019, <https://hoverflyvision.weebly.com/uploads/4/8/1/0/48109195/user_manual_v2_2.docx>.

Johnson, S & Everingham, M 2010, 'Clustered pose and nonlinear appearance models for human pose estimation', *Proceedings of the British Machine Vision Conference*.

Kawana, Y, Ukita, N, Huang, JB & Yang, MH 2018, 'Ensemble convolutional neural networks for pose estimation', *Computer Vision and Image Understanding*, vol. 169, pp. 62-74.

Kooima, R 2008, 'Generalized perspective projection', *Louisiana State University*.

Land, MF 1992, 'Visual tracking and pursuit: humans and arthropods compared', *Journal of Insect Physiology*, vol. 38, no. 12, pp. 939-51.

Lawson, KK & Srinivasan, MV 2017, 'Flight control of fruit flies: dynamic response to optic flow and headwind', *Journal of Experimental Biology*, vol. 220, pp. 2005-16.

Lin, CM, Tsai, CY, Lai, YC, Li, SA & Wong, CC 2018, 'Visual object recognition and pose estimation based on a deep semantic segmentation network', *IEEE Sensors Journal*, vol. 18, no. 22, pp. 9370-81.

Lindemann, JP, Kern, R, Michaelis, C, Meyer, P, van Hateren, JH & Egelhaaf, M 2003, 'FliMax, a novel stimulus device for panoramic and highspeed presentation of behaviourally generated optic flow', *Vision Research*, vol. 43, no. 7, pp. 779-91.

Lindemann, JP, Weiss, H, Möller, R & Egelhaaf, M 2008, 'Saccadic flight strategy facilitates collision avoidance: closed-loop performance of a cyberfly', *Biological Cybernetics*, vol. 98, pp. 213-27.

Liu, MY, Tuzel, O, Veeraraghavan, A, Taguchi, Y, Marks, TK & Chellappa, R 2012, 'Fast object localization and pose estimation in heavy clutter for robotic bin picking', *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 951-73.

Liu, Z, Liu, G, Li, J & Ye, D 2016, 'Pose estimation of rigid object in point cloud', *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*.

Mitash, C, Boularias A & Bekris, K 2019, 'Physics-based scene-level reasoning for object pose estimation in clutter', *The International Journal of Robotics Research.*

Nicholas, S, Supple, J, Leibbrandt, R, Gonzalez-Bellido, PT & Nordström, K 2018, 'Integration of small- and wide-field visual features in target-selective descending neurons of both predatory and nonpredatory dipterans', *The Journal of Neuroscience*, vol. 38, no. 50, pp. 10725-33.

Nordström, K & O'Carroll, DC 2009, 'Feature detection and the hypercomplex property in insects', *Trends in Neurosciences*, vol. 32, no. 7, pp. 389-91.

Nordström, K, Bolzon, DM & O'Carroll, DC 2011, 'Spatial facilitation by a high-performance dragonfly target-detecting neuron', *Biology Letters*, vol. 7, no. 4, pp. 588-92.

O'Neill, JK 2000, 'Backdriving a full motion simulator using flight recorder data', Master's thesis, Institute for Aerospace Studies, University of Toronto, Toronto, Canada.

Olberg, RM, Worthington, AH & Venator, KR 2000, 'Prey pursuit and interception in dragonflies', *Journal of Comparative Physiology A,* vol. 186, pp. 155-62.

Power, K 1996, 'Projections and viewing transformations', *Glasnost: Institute of Technology, Carlow, Ireland.*

Raudies, F 2013, 'Optic flow', *Scholarpedia*, vol. 8, no. 7.

Royal Society for the Protection of Birds n.d., 'Marmalade hoverfly: Episyrphus balteatus', *Royal Society for the Protection of Birds*, viewed 26 October 2019, <https://www.rspb.org.uk/birds-and-wildlife/wildlife-guides/other-garden-wildlife/insects-and-other-invertebrates/flies/marmalade-hoverfly/>.

Schilstra, C & van Hateren, JH 1999, 'Blowfly flight and optic flow: I. thorax kinematics and flight dynamics', *The Journal of Experimental Biology*, vol. 202, pp. 1481-90.

Storey, M 2012, 'Syritta pipiens', *Discover Life*, viewed 26 October 2019, <https://www.discoverlife.org/mp/20p?see=I_MWS107048&res=640&mobile=1>

Straw, AD, Warrant, EJ & O'Carroll, DC 2006, 'A 'bright zone' in male hoverfly (Eristalis tenax) eyes and associated faster motion detection and increased contrast sensitivity', *Journal of Experimental Biology*, vol. 209, pp. 4339-54.

Tammero, LF & Dickinson, MH 2002, 'The influence of visual landscape on the free flight behavior of the fruit fly Drosophila melanogaster', *The Journal of Experimental Biology*, vol. 205, pp. 327-43.

Thyselius, M, Gonzalez-Bellido, PT, Wardill, TJ & Nordström, K 2018, 'Visual approach computation in feeding hoverflies', *The Journal of Experimental Biology*, vol. 221, no. 10, pp. 1-9.

Ukita, N & Uematsu, Y 2019, 'Semi- and weakly-supervised human pose estimation', *Cornell University archive: Computer Vision and Pattern Recognition.*

Wallisch, P, Lusignan, ME, Benayoun, MD, Baker, TI, Dickey, AS & Hatsopoulos, NG 2014, *MATLAB for Neuroscientists: an introduction to scientific computing in MATLAB*, 2nd edn, Elsevier Academic, Amsterdam.

Wardill, TJ, Fabian, ST, Pettigrew, AC, Stavenga, DG, Nordström, K & Gonzalez-Bellido, PT 2017, 'A novel interception strategy in a miniature robber fly with extreme visual acuity', *Current Biology*, vol. 27, pp. 854-9.

Warren Photographic: Image Library of Animals in Action n.d., *Honey bee (Apis mellifera) worker beside hoverfly (Eristalis tenax) to show mimicry in form and size*, viewed 26 February 2019, <https://www.warrenphotographic.co.uk/38654-drone-fly-mimicking-honey-bee>

Yang, W, Ouyang, W, Wang, X, Ren, J, Li, H & Wang, X 2018, '3D human pose estimation in the wild by adversarial learning', *Cornell University archive: Computer Vision and Pattern Recognition.*

Appendix A. Code

This appendix contains the code used in 4.1. Data Validation and Pre-Processing and 4.2. Trajectory Reconstruction and Heading Estimation. This code is the main deliverable of the project, and takes the datasets provided by Thyselius et al. (2018) as input. At present, the validation step relies on the datasets being in the same, uniform, format.

```
%% 0. Clear all
clear all

%% 1. Navigate to desired dataset
% Opens user interface to select the dataset to be worked on
[trajectory, filepath] = uigetfile('*.mat');
% Loads the selected dataset into the MATLAB Workspace
load([filepath,trajectory])

%% 2. Check dataset is valid
% First, check that the cell has two matrices
if size(all_3Dcordinates,2) == 2
    % Next, check that the matrices are the same size
    if size(all_3Dcordinates{1}) ==
size(all_3Dcordinates{2})
        % Finally, check both matrices have 3 rows
        if size(all_3Dcordinates{1},1) == 3
            % Continue if all the above are satisfied
        else
            msgbox('Selected dataset is invalid: matrices do
not have three rows.', 'Invalid Value', 'error')
            return
        end
    else
        msgbox('Selected dataset is invalid: matrices are
not of the same size.', 'Invalid Value', 'error')
        return
    end
else
    msgbox('Selected dataset is invalid: cell does not
contain two matrices.', 'Invalid Value', 'error')
    return
end

% Here is where you could also include an interactive option
for the user
% to adjust the datasets so that they are valid, as well as
a way to check
% if the selected dataset has already been processed

%% 3. Select leader and pursuer
% Prompt user to choose which matrix is which
```

```
warning('off','MATLAB:questdlg:StringMismatch') % Suppress
warning about default button
% Format: questdlg(quest,dlgtitle,btn1,btn2,(btn3),defbtn)
prompt = questdlg(['Which matrix represents the pursuing
fly? The perspective ' ...
    'of this fly will be used as the basis of the
reconstruction and simulation.'], ...
    'Select leader and pursuer','1','2','defbtn'); %
Pressing Enter will not close dialog box
observerFlag = str2num(prompt)-1;

%% 4. Convert positions from absolute to relative
% Both observer and target's original motion will be saved
for later use
observerPositions = all_3Dcordinates{observerFlag+1};
targetPositions = all_3Dcordinates{2-observerFlag};
% Reverse direction of z-axis to match FlyFly's definition
observerPositions(3,:) = -observerPositions(3,:);
targetPositions(3,:) = -targetPositions(3,:);
% This line of code will create a matrix with relative
positions depending
% on which matrix was selected as the observer. The vectors
also represent
% the direct line of sight between the observer and the
target at each frame
targetRelativePositions = targetPositions -
observerPositions;

%% 5. Get direction of motion between frames
% Vector at each frame representing the direction of motion
to next position
for m=1:length(targetPositions)-1
    for n=1:3
        % Change in position between trials
        targetMotion(n,m) = targetPositions(n,m+1) -
targetPositions(n,m);
        observerMotion(n,m) = observerPositions(n,m+1) -
observerPositions(n,m);
    end
end

%% 6. Choose how to rotate positions
% Heading of fly must be aligned with the z-axis in order to
display with
% FlyFly -> the heading must first be chosen
alignmentMethod = questdlg(['What would you like to align
the pursuer''s heading ' ...
    'with?'],'Select heading alignment','Direction of
Motion','Line of Sight', ...
    'Choose for me','defbtn');
```

```
% Once user has selected how they want to align the
pursuer's heading,
% set this so that the motion can be rotated accordingly
switch alignmentMethod
    case 'Direction of Motion'
        % The pursuer will face along the direction of its
own motion
        observerHeading = observerMotion;
        alignmentMethod = 'dom';
    case 'Line of Sight'
        % The pursuer will face along the direct line of
sight between it
        % and the leader
        observerHeading = targetRelativePositions;
        alignmentMethod = 'los';
    case 'Choose for me'
        observerHeading = [];
        alignmentMethod = 'alg';
    otherwise
        msgbox('Alignment will be chosen for you.')
        observerHeading = [];
end

%% 7. Algorithm for estimating most suitable heading
if isempty(observerHeading)
    % algorithm goes here
end

%% 8. Rotate heading to align with z-axis
% This code uses the angle-axis method to create a rotation
matrix, aligning
% the selected heading with the z-axis. This rotation is
then applied to the
% relative positions of the target.
zAxis = [0 0 1];
% Does not process entire matrix so matrix dimensions do not
exceed the loop index
for n = 1:length(observerHeading)-1
    % Use cross-product to get axis of rotation
    xprod{n} = cross(zAxis,observerHeading(:,n));
    % Use dot-product to get angle of rotation
    dprod(n) = dot(zAxis,observerHeading(:,n));
    theta(n) = acos(dprod(n)/norm(observerHeading(:,n)));
    % MATLAB uses Rodrigues' formula here
    rotationVector{n} = theta(n)*xprod{n};
    rotationMatrix{n} =
rotationVectorToMatrix(rotationVector{n});
    % Apply rotation to observer's motion for display of
optic flow
    rotatedObserverMotion{n} =
rotationMatrix{n}*observerMotion(:,n);
```

```
    % And to target's relative positions
    rotatedTargetPos{n} =
rotationMatrix{n}*targetRelativePositions(:,n);
end
% Convert positions into a usable format
rotatedTargetPos = cell2mat(rotatedTargetPos);
rotatedObserverMotion = cell2mat(rotatedObserverMotion);

%% 9. Calculate velocity
% Target moves in straight line therefore velocity is
calculated from Euclidean
% distance moved between each position coordinate and the
time between capture
% of each position coordinate
for k = 1:length(rotatedTargetPos)-1
    targetDistance(k) = sqrt(rotatedTargetPos(1,k)^2 +
rotatedTargetPos(2,k)^2 + rotatedTargetPos(3,k)^2);
    observerDistance(k) = sqrt(rotatedObserverMotion(1,k)^2
+ rotatedObserverMotion(2,k)^2 +
rotatedObserverMotion(3,k)^2);
end
% % Check with user what the coordinate capture frequency is
% trialRate = inputdlg({'At what frequency were the position
coordinates captured (times per second)?'}, ...
%     'Confirm position coordinate capture
frequency',1,{'30'});
% trialRate = str2num(trialRate{1});
trialRate = 30;
% Time between each coordinate capture is reciprocal of
capture frequency
interTrialTime = 1/trialRate;
% Calculate target velocity based on this information
targetVelocity = targetDistance/interTrialTime;
observerVelocity = observerDistance/interTrialTime;

%% 10. Save values for 3D Target to CSV
% 3D Target takes positions in the form of azimuth,
elevation, and distance.
% The coordinate system in the FlyFly guide was used in this
conversion method,
% converting the XYZ positions to az/el/r. Angles are
specified in degrees.
for n=1:length(rotatedTargetPos)
    sphericalRelativePos(1,n)=90-
atan2d(rotatedTargetPos(3,n),rotatedTargetPos(1,n));

sphericalRelativePos(2,n)=atan2d(rotatedTargetPos(2,n),sqrt(
rotatedTargetPos(1,n).^2 + rotatedTargetPos(3,n).^2));
    sphericalRelativePos(3,n)=rotatedTargetPos(3,n);
    % Also get observer yaw
```

```
    observerYaw(1,n) = 90-
atan2d(rotatedObserverMotion(3,n),rotatedObserverMotion(1,n)
);
end
% Prepare values to be imported directly into 3D Target
threeDTargetData = [ones(1,length(sphericalRelativePos)-
1); ...   % target size
    sphericalRelativePos(1,1:end-
1); ...                          % start azimuth
    sphericalRelativePos(2,1:end-
1); ...                          % start elevation
    sphericalRelativePos(3,1:end-
1); ...                          % start distance

sphericalRelativePos(1,2:end); ...
% end azimuth

sphericalRelativePos(2,2:end); ...
% end elevation

sphericalRelativePos(3,2:end); ...
% end distance

targetVelocity];
% velocity
% Write to CSV
csvwrite(['3DTargetData_',trajectory(1,1:3),'_obs',num2str(o
bserverFlag),'_',alignmentMethod,'.csv'],threeDTargetData)

%% 11. Save values for Starfield to CSV
% Prepare values to be imported directly into Starfield
starfieldData = [ones(1,length(rotatedObserverMotion)-
1); ...     % dot size
    ones(1,length(rotatedObserverMotion)-
1); ...                  % dot density
    zeros(1,length(rotatedObserverMotion)-
1); ...                  % sideslip
    rotatedObserverMotion(2,1:end-
1)./interTrialTime; ...          % lift
    rotatedObserverMotion(3,1:end-
1)./interTrialTime; ...          % thrust
    zeros(1,length(rotatedObserverMotion)-
1); ...                  % pitch
    observerYaw(1:end-
1)./interTrialTime; ...                        % yaw
    zeros(1,length(rotatedObserverMotion)-
1); ...                  % roll
    zeros(1,length(rotatedObserverMotion)-
1); ...                  % background noise
    ones(1,length(rotatedObserverMotion)-
1); ...                  % retain into next trial
```

```
60.*observerDistance./observerVelocity];
% time
% Write to CSV
csvwrite('StarfieldData.csv',starfieldData)

%% To be included later:
% 12. Set up FlyFly session with appropriate number of
layers and trials
% 13. Import values directly into FlyFly
```

Appendix B. Summary of Results

This appendix contains a summary of the results of 4.3. Simulation, noting which datasets and trajectories were invalid or otherwise of note. As can be seen in the table, some datasets had NaN values that were edited and resulted in a reconstruction that failed, whereas others did not lead to an error. Likewise, some failed reconstructions resulted from trajectories that did not previously have NaN values.

Table B.1: A summary of the results of 4.3. Simulation.

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
001obs0dom	-	-	-	-	Dataset invalid: one matrix
001obs0los	-	-	-	-	Dataset invalid: one matrix
001obs1dom	-	-	-	-	Dataset invalid: one matrix
001obs1los	-	-	-	-	Dataset invalid: one matrix
002obs0dom	-	-	-	-	Dataset invalid: one matrix
002obs0los	-	-	-	-	Dataset invalid: one matrix
002obs1dom	-	-	-	-	Dataset invalid: one matrix
002obs1los	-	-	-	-	Dataset invalid: one matrix
003obs0dom	-	-	-	-	Dataset invalid: one matrix
003obs0los	-	-	-	-	Dataset invalid: one matrix
003obs1dom	-	-	-	-	Dataset invalid: one matrix

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
003obs1los	-	-	-	-	Dataset invalid: one matrix
004obs0dom	19	61	31.1%	3.1	
004obs0los	17	71	23.9%	4	
004obs1dom	16	67	23.9%	4	Reconstruction failed
004obs1los	18	59	30.5%	3.157894737	
005obs0dom	3	20	15.0%	5.25	
005obs0los	10	39	25.6%	3.636363636	Reconstruction failed
005obs1dom	12	51	23.5%	4	
005obs1los	372	414	89.9%	1.112600536	
006obs0dom	373	421	88.6%	1.128342246	Reconstruction failed
006obs0los	27	84	32.1%	3.035714286	Reconstruction failed
006obs1dom	370	400	92.5%	1.080862534	
006obs1los	31	91	34.1%	2.875	
007obs0dom	-	-	-	-	Dataset invalid: one matrix

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
007obs0los	-	-	-	-	Dataset invalid: one matrix
007obs1dom	-	-	-	-	Dataset invalid: one matrix
007obs1los	-	-	-	-	Dataset invalid: one matrix
008obs0dom	11	44	25.0%	3.75	NaN edited
008obs0los	27	90	30.0%	3.25	
008obs1dom	367	393	93.4%	1.070652174	
008obs1los	18	96	18.8%	5.105263158	
009obs0dom	19	74	25.7%	3.75	NaN edited
009obs0los	18	85	21.2%	4.526315789	
009obs1dom	380	423	89.8%	1.112860892	
009obs1los	28	69	40.6%	2.413793103	
010obs0dom	376	435	86.4%	1.156498674	
010obs0los	19	79	24.1%	4	
010obs1dom	7	44	15.9%	5.625	NaN edited

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
010obs1los	29	68	42.6%	2.3	Reconstruction failed
011obs0dom	35	176	19.9%	4.916666667	
011obs0los	47	181	26.0%	3.791666667	
011obs1dom	23	158	14.6%	6.625	
011obs1los	41	190	21.6%	4.547619048	
012obs0dom	0	7	0.0%	8	
012obs0los	5	9	55.6%	1.666666667	
012obs1dom	5	9	55.6%	1.666666667	Reconstruction failed
012obs1los	2	9	22.2%	3.333333333	Reconstruction failed
013obs0dom	29	124	23.4%	4.166666667	NaN edited
013obs0los	46	159	28.9%	3.404255319	
013obs1dom	38	155	24.5%	4	
013obs1los	53	169	31.4%	3.148148148	
014obs0dom	10	98	10.2%	9	NaN; Reconstruction failed

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
014obs0los	24	83	28.9%	3.36	
014obs1dom	20	65	30.8%	3.142857143	NaN edited
014obs1los	20	90	22.2%	4.333333333	
015obs0dom	6	53	11.3%	7.714285714	
015obs0los	28	96	29.2%	3.344827586	Reconstruction failed
015obs1dom	16	74	21.6%	4.411764706	
015obs1los	33	93	35.5%	2.764705882	
016obs0dom	3	56	5.4%	14.25	
016obs0los	14	57	24.6%	3.866666667	Reconstruction failed
016obs1dom	11	53	20.8%	4.5	
016obs1los	24	97	24.7%	3.92	
017obs0dom	57	368	15.5%	6.362068966	
017obs0los	76	427	17.8%	5.558441558	
017obs1dom	21	113	18.6%	5.181818182	NaN edited

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
017obs1los	86	428	20.1%	4.931034483	
018obs0dom	17	74	23.0%	4.166666667	
018obs0los	24	102	23.5%	4.12	
018obs1dom	29	83	34.9%	2.8	
018obs1los	32	95	33.7%	2.909090909	
019obs0dom	29	162	17.9%	5.433333333	NaN edited
019obs0los	58	208	27.9%	3.542372881	Reconstruction failed
019obs1dom	34	191	17.8%	5.485714286	
019obs1los	46	182	25.3%	3.893617021	
020obs0dom	17	89	19.1%	5	
020obs0los	31	138	22.5%	4.34375	Reconstruction failed
020obs1dom	13	93	14.0%	6.714285714	NaN edited
020obs1los	38	156	24.4%	4.025641026	Reconstruction failed
021obs0dom	16	83	19.3%	4.941176471	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
021obs0los	32	81	39.5%	2.484848485	
021obs1dom	9	63	14.3%	6.4	NaN edited
021obs1los	30	77	39.0%	2.516129032	
022obs0dom	12	52	23.1%	4.076923077	
022obs0los	20	65	30.8%	3.142857143	
022obs1dom	14	54	25.9%	3.666666667	NaN edited
022obs1los	18	67	26.9%	3.578947368	
023obs0dom	17	145	11.7%	8.111111111	NaN; Reconstruction failed
023obs0los	62	233	26.6%	3.714285714	
023obs1dom	45	195	23.1%	4.260869565	
023obs1los	54	191	28.3%	3.490909091	
024obs0dom	24	75	32.0%	3.04	Reconstruction failed
024obs0los	27	93	29.0%	3.357142857	
024obs1dom	7	56	12.5%	7.125	NaN edited

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
024obs1los	23	89	25.8%	3.75	
025obs0dom	17	64	26.6%	3.611111111	NaN edited
025obs0los	21	80	26.3%	3.681818182	
025obs1dom	7	64	10.9%	8.125	
025obs1los	26	77	33.8%	2.888888889	Reconstruction failed
026obs0dom	5	76	6.6%	12.83333333	NaN edited
026obs0los	70	277	25.3%	3.915492958	
026obs1dom	48	252	19.0%	5.163265306	
026obs1los	54	277	19.5%	5.054545455	
027obs0dom	29	172	16.9%	5.766666667	
027obs0los	44	188	23.4%	4.2	
027obs1dom	50	173	28.9%	3.411764706	
027obs1los	49	184	26.6%	3.7	Reconstruction failed
028obs0dom	40	196	20.4%	4.804878049	NaN edited

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
028obs0los	64	268	23.9%	4.138461538	
028obs1dom	33	241	13.7%	7.117647059	
028obs1los	66	284	23.2%	4.253731343	
029obs0dom	14	51	27.5%	3.466666667	
029obs0los	22	69	31.9%	3.043478261	
029obs1dom	8	43	18.6%	4.888888889	
029obs1los	19	62	30.6%	3.15	Reconstruction failed
030obs0dom	17	95	17.9%	5.333333333	NaN edited
030obs0los	53	195	27.2%	3.62962963	
030obs1dom	18	141	12.8%	7.473684211	
030obs1los	38	172	22.1%	4.435897436	
031obs0dom	23	245	9.4%	10.25	NaN edited
031obs0los	104	357	29.1%	3.40952381	Reconstruction failed
031obs1dom	62	289	21.5%	4.603174603	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
031obs1los	73	275	26.5%	3.72972973	
032obs0dom	16	76	21.1%	4.529411765	
032obs0los	39	140	27.9%	3.525	Reconstruction failed
032obs1dom	19	93	20.4%	4.7	
032obs1los	42	152	27.6%	3.558139535	
033obs0dom	25	130	19.2%	5.038461538	
033obs0los	63	224	28.1%	3.515625	Reconstruction failed
033obs1dom	53	181	29.3%	3.37037037	
033obs1los	69	217	31.8%	3.114285714	
034obs0dom	14	52	26.9%	3.533333333	
034obs0los	20	77	26.0%	3.714285714	
034obs1dom	8	37	21.6%	4.222222222	
034obs1los	12	72	16.7%	5.615384615	
035obs0dom	25	70	35.7%	2.730769231	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
035obs0los	25	90	27.8%	3.5	
035obs1dom	15	58	25.9%	3.6875	Reconstruction failed
035obs1los	22	84	26.2%	3.695652174	Reconstruction failed
036obs0dom	14	55	25.5%	3.733333333	Reconstruction failed
036obs0los	31	84	36.9%	2.65625	Reconstruction failed
036obs1dom	4	45	8.9%	9.2	NaN edited
036obs1los	19	59	32.2%	3	
037obs0dom	-	-	-	-	Dataset invalid: one matrix
037obs0los	-	-	-	-	Dataset invalid: one matrix
037obs1dom	-	-	-	-	Dataset invalid: one matrix
037obs1los	-	-	-	-	Dataset invalid: one matrix
038obs0dom	-	-	-	-	Dataset invalid: one matrix
038obs0los	-	-	-	-	Dataset invalid: one matrix
038obs1dom	-	-	-	-	Dataset invalid: one matrix

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
038obs1los	-	-	-	-	Dataset invalid: one matrix
039obs0dom	-	-	-	-	Dataset invalid: one matrix
039obs0los	-	-	-	-	Dataset invalid: one matrix
039obs1dom	-	-	-	-	Dataset invalid: one matrix
039obs1los	-	-	-	-	Dataset invalid: one matrix
040obs0dom	-	-	-	-	Dataset invalid: one matrix
040obs0los	-	-	-	-	Dataset invalid: one matrix
040obs1dom	-	-	-	-	Dataset invalid: one matrix
040obs1los	-	-	-	-	Dataset invalid: one matrix
041obs0dom	-	-	-	-	Dataset invalid: one matrix
041obs0los	-	-	-	-	Dataset invalid: one matrix
041obs1dom	-	-	-	-	Dataset invalid: one matrix
041obs1los	-	-	-	-	Dataset invalid: one matrix
042obs0dom	14	74	18.9%	5	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
042obs0los	29	90	32.2%	3.033333333	Reconstruction failed
042obs1dom	9	73	12.3%	7.4	Reconstruction failed
042obs1los	19	60	31.7%	3.05	Reconstruction failed
043obs0dom	11	73	15.1%	6.166666667	
043obs0los	23	84	27.4%	3.541666667	Reconstruction failed
043obs1dom	15	85	17.6%	5.375	
043obs1los	17	77	22.1%	4.333333333	
044obs0dom	-	-	-	-	Entire trajectory NaN
044obs0los	-	-	-	-	Entire trajectory NaN
044obs1dom	-	-	-	-	Entire trajectory NaN
044obs1los	-	-	-	-	Entire trajectory NaN
045obs0dom	7	63	11.1%	8	NaN edited
045obs0los	25	98	25.5%	3.807692308	
045obs1dom	10	80	12.5%	7.363636364	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
045obs1los	23	91	25.3%	3.833333333	Reconstruction failed
046obs0dom	-	-	-	-	Dataset invalid: one matrix
046obs0los	-	-	-	-	Dataset invalid: one matrix
046obs1dom	-	-	-	-	Dataset invalid: one matrix
046obs1los	-	-	-	-	Dataset invalid: one matrix
047obs0dom	1	13	7.7%	7	Reconstruction failed
047obs0los	13	64	20.3%	4.642857143	
047obs1dom	4	26	15.4%	5.4	
047obs1los	11	46	23.9%	3.916666667	
048obs0dom	4	26	15.4%	5.4	
048obs0los	13	70	18.6%	5.071428571	
048obs1dom	7	42	16.7%	5.375	
048obs1los	9	59	15.3%	6	
049obs0dom	1	39	2.6%	20	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
049obs0los	24	83	28.9%	3.36	
049obs1dom	4	38	10.5%	7.8	NaN edited
049obs1los	25	76	32.9%	2.961538462	
050obs0dom	1	17	5.9%	9	Reconstruction failed
050obs0los	7	45	15.6%	5.75	
050obs1dom	0	30	0.0%	31	Reconstruction failed
050obs1los	16	42	38.1%	2.529411765	
051obs0dom	4	49	8.2%	10	NaN edited
051obs0los	35	94	37.2%	2.638888889	
051obs1dom	21	78	26.9%	3.590909091	
051obs1los	28	90	31.1%	3.137931034	
052obs0dom	3	31	9.7%	8	
052obs0los	17	37	45.9%	2.111111111	
052obs1dom	1	11	9.1%	6	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
052obs1los	19	60	31.7%	3.05	
053obs0dom	0	23	0.0%	24	NaN edited
053obs0los	19	74	25.7%	3.75	
053obs1dom	14	50	28.0%	3.4	Reconstruction failed
053obs1los	13	51	25.5%	3.714285714	
054obs0dom	10	72	13.9%	6.636363636	NaN edited
054obs0los	20	77	26.0%	3.714285714	
054obs1dom	30	90	33.3%	2.935483871	Reconstruction failed
054obs1los	23	104	22.1%	4.375	
055obs0dom	26	76	34.2%	2.851851852	
055obs0los	29	102	28.4%	3.433333333	
055obs1dom	9	85	10.6%	8.6	
055obs1los	25	83	30.1%	3.230769231	Reconstruction failed
056obs0dom	4	29	13.8%	6	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
056obs0los	10	47	21.3%	4.363636364	
056obs1dom	13	37	35.1%	2.714285714	Reconstruction failed
056obs1los	22	58	37.9%	2.565217391	
057obs0dom	12	46	26.1%	3.615384615	NaN edited
057obs0los	31	112	27.7%	3.53125	
057obs1dom	35	88	39.8%	2.472222222	
057obs1los	26	96	27.1%	3.592592593	
058obs0dom	12	75	16.0%	5.846153846	NaN edited
058obs0los	22	99	22.2%	4.347826087	
058obs1dom	13	58	22.4%	4.214285714	
058obs1los	30	98	30.6%	3.193548387	
059obs0dom	1	34	2.9%	17.5	
059obs0los	17	50	34.0%	2.833333333	
059obs1dom	14	35	40.0%	2.4	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
059obs1los	5	49	10.2%	8.333333333	
060obs0dom	24	82	29.3%	3.32	
060obs0los	12	61	19.7%	4.769230769	
060obs1dom	17	79	21.5%	4.444444444	
060obs1los	22	76	28.9%	3.347826087	
061obs0dom	11	46	23.9%	3.916666667	NaN edited
061obs0los	21	89	23.6%	4.090909091	
061obs1dom	17	84	20.2%	4.722222222	
061obs1los	23	93	24.7%	3.916666667	
062obs0dom	12	61	19.7%	4.769230769	NaN edited
062obs0los	32	88	36.4%	2.696969697	
062obs1dom	12	59	20.3%	4.615384615	
062obs1los	25	87	28.7%	3.384615385	
063obs0dom	5	34	14.7%	5.833333333	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
063obs0los	38	104	36.5%	2.692307692	
063obs1dom	26	78	33.3%	2.925925926	
063obs1los	30	88	34.1%	2.870967742	
064obs0dom	12	77	15.6%	6	
064obs0los	25	87	28.7%	3.384615385	
064obs1dom	17	70	24.3%	3.944444444	NaN edited
064obs1los	21	91	23.1%	4.181818182	
065obs0dom	12	66	18.2%	5.153846154	
065obs0los	19	80	23.8%	4.05	
065obs1dom	14	85	16.5%	5.733333333	
065obs1los	20	100	20.0%	4.80952381	
066obs0dom	14	72	19.4%	4.866666667	
066obs0los	27	99	27.3%	3.571428571	
066obs1dom	16	79	20.3%	4.705882353	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
066obs1los	32	87	36.8%	2.666666667	Reconstruction failed
067obs0dom	14	67	20.9%	4.533333333	NaN edited
067obs0los	31	90	34.4%	2.84375	
067obs1dom	10	75	13.3%	6.909090909	
067obs1los	36	97	37.1%	2.648648649	
068obs0dom	11	64	17.2%	5.416666667	NaN edited
068obs0los	22	80	27.5%	3.52173913	
068obs1dom	17	65	26.2%	3.666666667	
068obs1los	23	84	27.4%	3.541666667	
069obs0dom	19	86	22.1%	4.35	Reconstruction failed
069obs0los	24	105	22.9%	4.24	
069obs1dom	14	67	20.9%	4.533333333	
069obs1los	28	84	33.3%	2.931034483	
070obs0dom	19	78	24.4%	3.95	Reconstruction failed

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
070obs0los	27	78	34.6%	2.821428571	Reconstruction failed
070obs1dom	20	70	28.6%	3.380952381	
070obs1los	22	81	27.2%	3.565217391	
071obs0dom	17	86	19.8%	4.833333333	
071obs0los	36	93	38.7%	2.540540541	
071obs1dom	17	67	25.4%	3.777777778	
071obs1los	21	82	25.6%	3.772727273	Reconstruction failed
072obs0dom	17	67	25.4%	3.777777778	Reconstruction failed
072obs0los	29	95	30.5%	3.2	
072obs1dom	27	97	27.8%	3.5	
072obs1los	34	93	36.6%	2.685714286	
073obs0dom	22	89	24.7%	3.913043478	
073obs0los	25	85	29.4%	3.307692308	
073obs1dom	22	91	24.2%	4	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
073obs1los	27	77	35.1%	2.785714286	
074obs0dom	-	-	-	-	Dataset invalid: one matrix
074obs0los	-	-	-	-	Dataset invalid: one matrix
074obs1dom	-	-	-	-	Dataset invalid: one matrix
074obs1los	-	-	-	-	Dataset invalid: one matrix
075obs0dom	-	-	-	-	Dataset invalid: one matrix
075obs0los	-	-	-	-	Dataset invalid: one matrix
075obs1dom	-	-	-	-	Dataset invalid: one matrix
075obs1los	-	-	-	-	Dataset invalid: one matrix
076obs0dom	-	-	-	-	Dataset invalid: one matrix
076obs0los	-	-	-	-	Dataset invalid: one matrix
076obs1dom	-	-	-	-	Dataset invalid: one matrix
076obs1los	-	-	-	-	Dataset invalid: one matrix
077obs0dom	-	-	-	-	Dataset invalid: one matrix

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
077obs0los	-	-	-	-	Dataset invalid: one matrix
077obs1dom	-	-	-	-	Dataset invalid: one matrix
077obs1los	-	-	-	-	Dataset invalid: one matrix
078obs0dom	-	-	-	-	Dataset invalid: one matrix
078obs0los	-	-	-	-	Dataset invalid: one matrix
078obs1dom	-	-	-	-	Dataset invalid: one matrix
078obs1los	-	-	-	-	Dataset invalid: one matrix
079obs0dom	-	-	-	-	Dataset invalid: one matrix
079obs0los	-	-	-	-	Dataset invalid: one matrix
079obs1dom	-	-	-	-	Dataset invalid: one matrix
079obs1los	-	-	-	-	Dataset invalid: one matrix
080obs0dom	26	71	36.6%	2.666666667	
080obs0los	22	92	23.9%	4.043478261	
080obs1dom	15	70	21.4%	4.4375	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
080obs1los	24	97	24.7%	3.92	
081obs0dom	7	42	16.7%	5.375	
081obs0los	23	76	30.3%	3.208333333	
081obs1dom	17	59	28.8%	3.333333333	
081obs1los	15	68	22.1%	4.3125	Reconstruction failed
082obs0dom	8	47	17.0%	5.333333333	
082obs0los	21	61	34.4%	2.818181818	
082obs1dom	11	42	26.2%	3.583333333	
082obs1los	9	48	18.8%	4.9	
083obs0dom	14	86	16.3%	5.8	
083obs0los	23	78	29.5%	3.291666667	Reconstruction failed
083obs1dom	13	65	20.0%	4.714285714	Reconstruction failed
083obs1los	24	80	30.0%	3.24	
084obs0dom	-	-	-	-	Entire trajectory NaN

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
084obs0los	-	-	-	-	Entire trajectory NaN
084obs1dom	-	-	-	-	Entire trajectory NaN
084obs1los	-	-	-	-	Entire trajectory NaN
085obs0dom	0	0	0.0%	0	
085obs0los	0	0	0.0%	0	
085obs1dom	0	0	0.0%	0	NaN edited
085obs1los	0	0	0.0%	0	
086obs0dom	5	44	11.4%	7.5	
086obs0los	10	43	23.3%	4	
086obs1dom	6	40	15.0%	5.857142857	
086obs1los	12	47	25.5%	3.692307692	
087obs0dom	-	-	-	-	Entire trajectory NaN
087obs0los	-	-	-	-	Entire trajectory NaN
087obs1dom	-	-	-	-	Entire trajectory NaN

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
087obs1los	-	-	-	-	Entire trajectory NaN
088obs0dom	16	66	24.2%	3.941176471	NaN edited
088obs0los	28	93	30.1%	3.24137931	
088obs1dom	14	86	16.3%	5.8	
088obs1los	27	95	28.4%	3.428571429	
089obs0dom	9	42	21.4%	4.3	
089obs0los	30	104	28.8%	3.387096774	
089obs1dom	10	71	14.1%	6.545454545	
089obs1los	40	101	39.6%	2.487804878	
090obs0dom	23	91	25.3%	3.833333333	NaN edited
090obs0los	20	85	23.5%	4.095238095	
090obs1dom	20	93	21.5%	4.476190476	
090obs1los	22	89	24.7%	3.913043478	
091obs0dom	14	63	22.2%	4.266666667	NaN edited

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
091obs0los	19	89	21.3%	4.5	
091obs1dom	17	81	21.0%	4.555555556	
091obs1los	34	110	30.9%	3.171428571	
092obs0dom	8	34	23.5%	3.888888889	NaN; Reconstruction failed
092obs0los	25	92	27.2%	3.576923077	Reconstruction failed
092obs1dom	20	69	29.0%	3.333333333	
092obs1los	31	104	29.8%	3.28125	
093obs0dom	4	59	6.8%	12	NaN edited
093obs0los	13	94	13.8%	6.785714286	Reconstruction failed
093obs1dom	11	69	15.9%	5.833333333	Reconstruction failed
093obs1los	19	98	19.4%	4.95	
094obs0dom	12	59	20.3%	4.615384615	
094obs0los	26	110	23.6%	4.111111111	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
094obs1dom	11	51	21.6%	4.333333333	NaN; Reconstruction failed
094obs1los	21	81	25.9%	3.727272727	Reconstruction failed
095obs0dom	10	56	17.9%	5.181818182	NaN edited
095obs0los	24	106	22.6%	4.28	
095obs1dom	22	93	23.7%	4.086956522	
095obs1los	34	98	34.7%	2.828571429	
096obs0dom	3	27	11.1%	7	NaN edited
096obs0los	23	66	34.8%	2.791666667	
096obs1dom	6	46	13.0%	6.714285714	Reconstruction failed
096obs1los	17	78	21.8%	4.388888889	Reconstruction failed
097obs0dom	10	71	14.1%	6.545454545	NaN edited
097obs0los	9	69	13.0%	7	
097obs1dom	20	74	27.0%	3.571428571	
097obs1los	24	82	29.3%	3.32	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
098obs0dom	11	72	15.3%	6.083333333	
098obs0los	24	97	24.7%	3.92	
098obs1dom	21	78	26.9%	3.590909091	Reconstruction failed
098obs1los	24	99	24.2%	4	
099obs0dom	9	55	16.4%	5.6	NaN; Reconstruction failed
099obs0los	33	97	34.0%	2.882352941	
099obs1dom	12	58	20.7%	4.538461538	Reconstruction failed
099obs1los	28	100	28.0%	3.482758621	
100obs0dom	17	79	21.5%	4.444444444	NaN; Reconstruction failed
100obs0los	25	83	30.1%	3.230769231	
100obs1dom	16	98	16.3%	5.823529412	
100obs1los	29	104	27.9%	3.5	
101obs0dom	12	56	21.4%	4.384615385	Reconstruction failed

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
101obs0los	34	92	37.0%	2.657142857	Reconstruction failed
101obs1dom	10	69	14.5%	6.363636364	
101obs1los	25	98	25.5%	3.807692308	
102obs0dom	155	993	15.6%	6.371794872	NaN; Reconstruction failed
102obs0los	397	2977	13.3%	7.48241206	NaN edited
102obs1dom	180	1815	9.9%	10.03314917	NaN; Reconstruction failed
102obs1los	232	1348	17.2%	5.789699571	Reconstruction failed
103obs0dom	76	654	11.6%	8.506493506	NaN edited
103obs0los	211	1041	20.3%	4.91509434	
103obs1dom	98	763	12.8%	7.717171717	NaN edited
103obs1los	201	1001	20.1%	4.96039604	
104obs0dom	37	272	13.6%	7.184210526	
104obs0los	74	362	20.4%	4.84	

Trajectory ID	Number of frames visible	Number of frames total	Percentage visible	Average frames to visible	Notes
104obs1dom	38	242	15.7%	6.230769231	
104obs1los	68	367	18.5%	5.333333333	
105obs0dom	37	439	8.4%	11.57894737	NaN; Reconstruction failed
105obs0los	115	737	15.6%	6.362068966	Reconstruction failed
105obs1dom	51	549	9.3%	10.57692308	NaN; Reconstruction failed
105obs1los	98	758	12.9%	7.666666667	
106obs0dom	40	577	6.9%	14.09756098	NaN; Reconstruction failed
106obs0los	133	781	17.0%	5.835820896	Reconstruction failed
106obs1dom	98	680	14.4%	6.878787879	Reconstruction failed
106obs1los	112	787	14.2%	6.973451327	Reconstruction failed

Appendix C. Results of Statistical Tests

This appendix contains the results of the statistical tests applied to the results to determine the consistency of the methods described in the main body.

Appendix C.1. Kruskal-Wallis

This section contains the results of the Kruskal-Wallis tests. First, the entire dataset was compared as a whole, then the datasets within the two methods (direction of motion, line of sight), were compared. Next, datasets according to different selection of observer were tested, and finally, data within the four individual groups was compared.

Table C.1.1: Results from the Kruskal-Wallis test applied to the dataset as a whole.

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	2.10933e+10	343	6.14966e+07	2180.69	2.96135e-264
Error	9.54058e+10	11701	8.15365e+06		
Total	1.16499e+11	12044			

Table C.1.2: Results from the Kruskal-Wallis test applied to the DOM dataset.

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	2.90836e+09	171	1.70079e+07	1655.64	4.08333e-241
Error	6.4194e+09	5139	1.24915e+06		
Total	9.32776e+09	5310			

Table C.1.3: Results from the Kruskal-Wallis test applied to the LOS dataset.

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	1.60167e+09	171	9.36647e+06	507.01	5.39091e-35
Error	1.96682e+10	6562	2.99729e+06		
Total	2.12699e+10	6733			

Table C.1.4: Results from the Kruskal-Wallis test applied to the `obs0` *dataset.*

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	1.77221e+09	171	1.03638e+07	821.75	1.07718e-85
Error	1.03006e+10	5427	1.89802e+06		
Total	1.20728e+10	5598			

Table C.1.5: Results from the Kruskal-Wallis test applied to the `obs1` *dataset.*

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	3.6222e+09	171	2.11824e+07	1347.9	7.94924e-182
Error	1.36974e+10	6274	2.1832e+06		
Total	1.73196e+10	6445			

Table C.1.6: Results from the Kruskal-Wallis test applied to the `obs0dom` *dataset.*

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	2.33105e+08	85	2742416.6	696.85	7.92246e-97
Error	5.29586e+08	2195	241269.1		
Total	7.62691e+08	2280			

Table C.1.7: Results from the Kruskal-Wallis test applied to the `obs0los` *dataset.*

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	8.54953e+07	85	1005826.5	108.71	0.0425
Error	2.52315e+09	3232	780676.4		
Total	2.60864e+09	3317			

Table C.1.8: Results from the Kruskal-Wallis test applied to the `obs1dom` *dataset.*

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	5.31452e+08	85	6252376.4	953.81	5.53352e-147
Error	1.15628e+09	2944	392757.3		
Total	1.68773e+09	3029			

Table C.1.9: Results from the Kruskal-Wallis test applied to the `obs1los` *dataset.*

Source	SS	df	MS	Chi-sq	Prob>Chi-sq
Groups	3.09345e+08	85	3639352.3	391.01	8.80675e-41
Error	2.39243e+09	3330	718446		
Total	2.70177e+09	3415			

Appendix C.2. Conover-Iman

The Conover-Iman results were far too large to attach, with one table having almost 60000 rows, and so are available online here instead:

https://drive.google.com/drive/folders/1oVZnaqdoAOR2U3DOMD18mzhW6yPe3U_6

In order, the files contain the results of the Conover-Iman tests applied to the following populations:

1. collatedTest: All 344 valid trajectories collated as a whole (58996 comparisons)
2. domTest: All trajectories treated with the DOM method (14706 comparisons)
3. losTest: All trajectories treated with the LOS method (14706 comparisons)
4. obs0domTest: All trajectories from the `obs0dom` population (3655 comparisons)
5. obs0losTest: All trajectories from the `obs0los` population (3655 comparisons)
6. obs0Test: All trajectories with observer set as matrix 1 (14706 comparisons)
7. obs1domTest: All trajectories from the `obs1dom` population (3655 comparisons)
8. obs1losTest: All trajectories from the `obs1los` population (3655 comparisons)
9. obs1Test: All trajectories with observer set as matrix 2 (14706 comparisons)

Appendix D. Determining Pursuit

This appendix contains the data used to supply the regression learner. Trajectories were tagged as pursuit or not, according to the footage. 300 of the 344 datasets were selected at random as training data, and the remaining 44 were left as test sets. The training data was then input to the regression learner to build an algorithm.

Table D.1: Data supplied to the regression learner.

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
004obs0dom	31.1%	3.1	-16.97421053	0
004obs0los	23.9%	4	-17.43297297	0
004obs1dom	23.9%	4	-16.97421053	0
004obs1los	30.5%	3.157894737	-17.43297297	0
005obs0dom	15.0%	5.25	-149.784	1
005obs0los	25.6%	3.636363636	-144.0230769	1
005obs1dom	23.5%	4	-149.784	1
005obs1los	89.9%	1.112600536	-106.9885714	1
006obs0dom	88.6%	1.128342246	-70.67021277	1
006obs0los	32.1%	3.035714286	-72.20652174	1
006obs1dom	92.5%	1.080862534	-70.67021277	1
006obs1los	34.1%	2.875	-83.0375	1
008obs0dom	25.0%	3.75	-72.64255319	0
008obs0los	30.0%	3.25	-74.22173913	0
008obs1dom	93.4%	1.070652174	-72.64255319	1
008obs1los	18.8%	5.105263158	-83.27317073	1
009obs0dom	25.7%	3.75	-76.40851064	0
009obs0los	21.2%	4.526315789	-78.06956522	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
009obs1dom	89.8%	1.112860892	-76.40851064	1
009obs1los	40.6%	2.413793103	-78.06956522	1
010obs0dom	86.4%	1.156498674	-114.6	1
010obs0los	24.1%	4	-128.2428571	1
010obs1dom	15.9%	5.625	-114.6	0
010obs1los	42.6%	2.3	-57.3	0
011obs0dom	19.9%	4.916666667	-111.5473684	0
011obs0los	26.0%	3.791666667	-112.7340426	0
011obs1dom	14.6%	6.625	-111.5473684	0
011obs1los	21.6%	4.547619048	-286.4054054	0
012obs0dom	0.0%	8	-234.9	1
012obs0los	55.6%	1.666666667	-195.75	1
012obs1dom	55.6%	1.666666667	-234.9	1
012obs1los	22.2%	3.333333333	-24.9893617	1
013obs0dom	23.4%	4.166666667	-56.70777778	1
013obs0los	28.9%	3.404255319	-57.34494382	1
013obs1dom	24.5%	4	-56.70777778	1
013obs1los	31.4%	3.148148148	-65.43205128	1
014obs0dom	10.2%	9	-2.263333333	1
014obs0los	28.9%	3.36	-2.645454545	1
014obs1dom	30.8%	3.142857143	-2.263333333	0
014obs1los	22.2%	4.333333333	-4.968292683	0
015obs0dom	11.3%	7.714285714	-43.32857143	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
015obs0los	29.2%	3.344827586	-44.38536585	0
015obs1dom	21.6%	4.411764706	-43.32857143	0
015obs1los	35.5%	2.764705882	-39.56086957	0
016obs0dom	5.4%	14.25	-14.10404255	1
016obs0los	24.6%	3.866666667	-14.41065217	1
016obs1dom	20.8%	4.5	-14.10404255	1
016obs1los	24.7%	3.92	-2.857284483	1
017obs0dom	15.5%	6.362068966	-57.11587983	1
017obs0los	17.8%	5.558441558	-172.8311688	1
017obs1dom	18.6%	5.181818182	-57.11587983	1
017obs1los	20.1%	4.931034483	-266.16	1
018obs0dom	23.0%	4.166666667	-27.34117647	1
018obs0los	23.5%	4.12	-27.888	1
018obs1dom	34.9%	2.8	-27.34117647	1
018obs1los	33.7%	2.909090909	-17.87692308	1
019obs0dom	17.9%	5.433333333	-88.93398058	0
019obs0los	27.9%	3.542372881	-89.80588235	0
019obs1dom	17.8%	5.485714286	-88.93398058	0
019obs1los	25.3%	3.893617021	-110.3638554	0
020obs0dom	19.1%	5	-34.33809524	0
020obs0los	22.5%	4.34375	-35.60987654	0
020obs1dom	14.0%	6.714285714	-34.33809524	0
020obs1los	24.4%	4.025641026	-62.70434783	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
021obs0dom	19.3%	4.941176471	-25.52553191	1
021obs0los	39.5%	2.484848485	-27.26590909	1
021obs1dom	14.3%	6.4	-23.21276596	1
021obs1los	39.0%	2.516129032	-29.26097561	1
022obs0dom	23.1%	4.076923077	-19.36785714	0
022obs0los	30.8%	3.142857143	-20.85769231	0
022obs1dom	25.9%	3.666666667	-18.07785714	1
022obs1los	26.9%	3.578947368	-7.395	1
023obs0dom	11.7%	8.111111111	-65.86725664	0
023obs0los	26.6%	3.714285714	-66.45535714	0
023obs1dom	23.1%	4.260869565	-65.86725664	1
023obs1los	28.3%	3.490909091	-161.8043478	1
024obs0dom	32.0%	3.04	-24.54255319	1
024obs0los	29.0%	3.357142857	-26.21590909	1
024obs1dom	12.5%	7.125	-24.54255319	1
024obs1los	25.8%	3.75	-26.21590909	1
025obs0dom	26.6%	3.611111111	-22.07234043	1
025obs0los	26.3%	3.681818182	-22.55217391	1
025obs1dom	10.9%	8.125	-22.07234043	0
025obs1los	33.8%	2.888888889	-15.03478261	0
026obs0dom	6.6%	12.83333333	-90.70629371	1
026obs0los	25.3%	3.915492958	-91.34507042	1
026obs1dom	19.0%	5.163265306	-90.70629371	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
026obs1los	19.5%	5.054545455	-128.4257426	0
027obs0dom	16.9%	5.766666667	-101.2058824	1
027obs0los	23.4%	4.2	-102.2079208	1
027obs1dom	28.9%	3.411764706	-101.2058824	0
027obs1los	26.6%	3.7	-72.18881119	0
028obs0dom	20.4%	4.804878049	-45.24189189	0
028obs0los	23.9%	4.138461538	-45.54965986	0
028obs1dom	13.7%	7.117647059	-45.24189189	0
028obs1los	23.2%	4.253731343	-180.9675676	0
029obs0dom	27.5%	3.466666667	-147.883871	1
029obs0los	31.9%	3.043478261	-143.2625	1
029obs1dom	18.6%	4.888888889	-147.883871	1
029obs1los	30.6%	3.15	-55.90731707	1
030obs0dom	17.9%	5.333333333	-53.02826087	0
030obs0los	27.2%	3.62962963	-53.61098901	0
030obs1dom	12.8%	7.473684211	-53.02826087	0
030obs1los	22.1%	4.435897436	-25.81269841	0
031obs0dom	9.4%	10.25	-21.11510417	1
031obs0los	29.1%	3.40952381	-21.22565445	1
031obs1dom	21.5%	4.603174603	-21.11510417	1
031obs1los	26.5%	3.72972973	-63.3453125	1
032obs0dom	21.1%	4.529411765	-40.70307692	0
032obs0los	27.9%	3.525	-41.3390625	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
032obs1dom	20.4%	4.7	-40.70307692	1
032obs1los	27.6%	3.558139535	-20.35153846	1
033obs0dom	19.2%	5.038461538	-27.98320611	0
033obs0los	28.1%	3.515625	-28.19846154	0
033obs1dom	29.3%	3.37037037	-27.98320611	0
033obs1los	31.8%	3.114285714	-79.69130435	0
034obs0dom	26.9%	3.533333333	-49.1787234	0
034obs0los	26.0%	3.714285714	-50.24782609	0
034obs1dom	21.6%	4.222222222	-49.1787234	1
034obs1los	16.7%	5.615384615	-50.24782609	1
035obs0dom	35.7%	2.730769231	-19.04851064	1
035obs0los	27.8%	3.5	-19.4626087	1
035obs1dom	25.9%	3.6875	-19.04851064	0
035obs1los	26.2%	3.695652174	-19.4626087	0
036obs0dom	25.5%	3.733333333	-32.04468085	1
036obs0los	36.9%	2.65625	-34.22954545	1
036obs1dom	8.9%	9.2	-32.04468085	1
036obs1los	32.2%	3	-32.74130435	1
042obs0dom	18.9%	5	-26.05531915	0
042obs0los	32.2%	3.033333333	-26.62173913	0
042obs1dom	12.3%	7.4	-26.05531915	1
042obs1los	31.7%	3.05	-26.62173913	1
043obs0dom	15.1%	6.166666667	-65.72340426	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
043obs0los	27.4%	3.541666667	-67.15217391	1
043obs1dom	17.6%	5.375	-65.72340426	1
043obs1los	22.1%	4.333333333	-70.20454545	1
045obs0dom	11.1%	8	-77.8787234	1
045obs0los	25.5%	3.807692308	-79.57173913	1
045obs1dom	12.5%	7.363636364	-77.8787234	0
045obs1los	25.3%	3.833333333	-98.92702703	0
047obs0dom	7.7%	7	-30.1848	1
047obs0los	20.3%	4.642857143	-29.02384615	1
047obs1dom	15.4%	5.4	-30.1848	1
047obs1los	23.9%	3.916666667	-29.02384615	1
048obs0dom	15.4%	5.4	-145.2642857	1
048obs0los	18.6%	5.071428571	-140.2551724	1
048obs1dom	16.7%	5.375	-145.2642857	0
048obs1los	15.3%	6	-107.0368421	0
049obs0dom	2.6%	20	-14.35680851	1
049obs0los	28.9%	3.36	-15.33568182	1
049obs1dom	10.5%	7.8	-14.35680851	1
049obs1los	32.9%	2.961538462	-18.23702703	1
050obs0dom	5.9%	9	-94.0125	1
050obs0los	15.6%	5.75	-90.252	1
050obs1dom	0.0%	31	-94.0125	0
050obs1los	38.1%	2.529411765	-70.509375	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
051obs0dom	8.2%	10	-45.56595745	1
051obs0los	37.2%	2.638888889	-46.55652174	1
051obs1dom	26.9%	3.590909091	-45.56595745	1
051obs1los	31.1%	3.137931034	-57.88108108	1
052obs0dom	9.7%	8	-104.1821429	1
052obs0los	45.9%	2.111111111	-100.5896552	1
052obs1dom	9.1%	6	-104.1821429	1
052obs1los	31.7%	3.05	-100.5896552	1
053obs0dom	0.0%	24	-84.92857143	0
053obs0los	25.7%	3.75	-76.70967742	0
053obs1dom	28.0%	3.4	-79.26666667	1
053obs1los	25.5%	3.714285714	-69.94117647	1
054obs0dom	13.9%	6.636363636	-28.04893617	0
054obs0los	26.0%	3.714285714	-31.9173913	0
054obs1dom	33.3%	2.935483871	-31.23829787	1
054obs1los	22.1%	4.375	-31.9173913	1
055obs0dom	34.2%	2.851851852	-7.354680851	1
055obs0los	28.4%	3.433333333	-7.514565217	1
055obs1dom	10.6%	8.6	-7.354680851	0
055obs1los	30.1%	3.230769231	-9.342432432	0
056obs0dom	13.8%	6	-100.1333333	1
056obs0los	21.3%	4.363636364	-96.128	1
056obs1dom	35.1%	2.714285714	-100.1333333	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
056obs1los	37.9%	2.565217391	-75.1	1
057obs0dom	26.1%	3.615384615	-51.54680851	1
057obs0los	27.7%	3.53125	-52.6673913	1
057obs1dom	39.8%	2.472222222	-51.54680851	1
057obs1los	27.1%	3.592592593	-57.68333333	1
058obs0dom	16.0%	5.846153846	-61.14468085	0
058obs0los	22.2%	4.347826087	-62.47391304	0
058obs1dom	22.4%	4.214285714	-61.14468085	1
058obs1los	30.6%	3.193548387	-77.67027027	1
059obs0dom	2.9%	17.5	-88.55517241	0
059obs0los	34.0%	2.833333333	-85.60333333	0
059obs1dom	40.0%	2.4	-88.55517241	1
059obs1los	10.2%	8.333333333	-65.84871795	1
060obs0dom	29.3%	3.32	-27.03404255	1
060obs0los	19.7%	4.769230769	-27.62173913	1
060obs1dom	21.5%	4.444444444	-27.03404255	0
060obs1los	28.9%	3.347826087	-30.9902439	0
061obs0dom	23.9%	3.916666667	-22.10425532	0
061obs0los	23.6%	4.090909091	-22.58478261	0
061obs1dom	20.2%	4.722222222	-22.10425532	0
061obs1los	24.7%	3.916666667	-23.61136364	0
062obs0dom	19.7%	4.769230769	-44.97446809	1
062obs0los	36.4%	2.696969697	-45.95217391	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
062obs1dom	20.3%	4.615384615	-44.97446809	0
062obs1los	28.7%	3.384615385	-45.95217391	0
063obs0dom	14.7%	5.833333333	-68.44255319	0
063obs0los	36.5%	2.692307692	-69.93043478	0
063obs1dom	33.3%	2.925925926	-68.44255319	1
063obs1los	34.1%	2.870967742	-69.93043478	1
064obs0dom	15.6%	6	-129.8829787	1
064obs0los	28.7%	3.384615385	-138.7386364	1
064obs1dom	24.3%	3.944444444	-129.8829787	1
064obs1los	23.1%	4.181818182	-132.7065217	1
065obs0dom	18.2%	5.153846154	-20.30957447	1
065obs0los	23.8%	4.05	-20.75108696	1
065obs1dom	16.5%	5.733333333	-20.30957447	0
065obs1los	20.0%	4.80952381	-20.75108696	0
066obs0dom	19.4%	4.866666667	-73.56595745	1
066obs0los	27.3%	3.571428571	-75.16521739	1
066obs1dom	20.3%	4.705882353	-73.56595745	1
066obs1los	36.8%	2.666666667	-78.58181818	1
067obs0dom	20.9%	4.533333333	-111.3021277	1
067obs0los	34.4%	2.84375	-116.7521739	1
067obs1dom	13.3%	6.909090909	-114.2680851	1
067obs1los	37.1%	2.648648649	-134.265	1
068obs0dom	17.2%	5.416666667	-75.35106383	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
068obs0los	27.5%	3.52173913	-76.98913043	0
068obs1dom	26.2%	3.666666667	-75.35106383	1
068obs1los	27.4%	3.541666667	-76.98913043	1
069obs0dom	22.1%	4.35	-50.25319149	1
069obs0los	22.9%	4.24	-51.34565217	1
069obs1dom	20.9%	4.533333333	-50.25319149	1
069obs1los	33.3%	2.931034483	-51.34565217	1
070obs0dom	24.4%	3.95	-30.95957447	0
070obs0los	34.6%	2.821428571	-31.6326087	0
070obs1dom	28.6%	3.380952381	-30.95957447	1
070obs1los	27.2%	3.565217391	-31.6326087	1
071obs0dom	19.8%	4.833333333	-52.0106383	1
071obs0los	38.7%	2.540540541	-53.14130435	1
071obs1dom	25.4%	3.777777778	-52.0106383	0
071obs1los	25.6%	3.772727273	-53.14130435	0
072obs0dom	25.4%	3.777777778	-49.34893617	1
072obs0los	30.5%	3.2	-50.42173913	1
072obs1dom	27.8%	3.5	-49.34893617	0
072obs1los	36.6%	2.685714286	-50.42173913	0
073obs0dom	24.7%	3.913043478	-49.68085106	0
073obs0los	29.4%	3.307692308	-50.76086957	0
073obs1dom	24.2%	4	-49.68085106	1
073obs1los	35.1%	2.785714286	-50.76086957	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
080obs0dom	36.6%	2.666666667	-36.60425532	0
080obs0los	23.9%	4.043478261	-37.4	0
080obs1dom	21.4%	4.4375	-36.60425532	1
080obs1los	24.7%	3.92	-46.4972973	1
081obs0dom	16.7%	5.375	-195.0473684	0
081obs0los	30.3%	3.208333333	-200.3189189	0
081obs1dom	28.8%	3.333333333	-195.0473684	1
081obs1los	22.1%	4.3125	-200.3189189	1
082obs0dom	17.0%	5.333333333	-199.0677419	0
082obs0los	34.4%	2.818181818	-192.846875	0
082obs1dom	26.2%	3.583333333	-199.0677419	0
082obs1los	18.8%	4.9	-150.5146341	0
083obs0dom	16.3%	5.8	-151.3170213	0
083obs0los	29.5%	3.291666667	-154.6065217	0
083obs1dom	20.0%	4.714285714	-151.3170213	1
083obs1los	30.0%	3.24	-192.2135135	1
086obs0dom	11.4%	7.5	-44.24814815	1
086obs0los	23.3%	4	-42.66785714	1
086obs1dom	15.0%	5.857142857	-44.24814815	0
086obs1los	25.5%	3.692307692	-34.13428571	0
088obs0dom	24.2%	3.941176471	-90.21568627	0
088obs0los	30.1%	3.24137931	-92.02	0
088obs1dom	16.3%	5.8	-90.21568627	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
088obs1los	28.4%	3.428571429	-92.02	1
089obs0dom	21.4%	4.3	-66.75098039	1
089obs0los	28.8%	3.387096774	-68.086	1
089obs1dom	14.1%	6.545454545	-66.75098039	0
089obs1los	39.6%	2.487804878	-70.92291667	0
090obs0dom	25.3%	3.833333333	-176.4333333	0
090obs0los	23.5%	4.095238095	-179.962	0
090obs1dom	21.5%	4.476190476	-176.4333333	1
090obs1los	24.7%	3.913043478	-199.9577778	1
091obs0dom	22.2%	4.266666667	-150.9938776	1
091obs0los	21.3%	4.5	-154.1395833	1
091obs1dom	21.0%	4.555555556	-150.9938776	1
091obs1los	30.9%	3.171428571	-176.1595238	1
092obs0dom	23.5%	3.888888889	-88.56326531	1
092obs0los	27.2%	3.576923077	-90.40833333	1
092obs1dom	29.0%	3.333333333	-88.56326531	0
092obs1los	29.8%	3.28125	-103.3238095	0
093obs0dom	6.8%	12	-104.0795918	1
093obs0los	13.8%	6.785714286	-106.2479167	1
093obs1dom	15.9%	5.833333333	-104.0795918	1
093obs1los	19.4%	4.95	-106.2479167	1
094obs0dom	20.3%	4.615384615	-33.8	1
094obs0los	23.6%	4.111111111	-36.00434783	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
094obs1dom	21.6%	4.333333333	-33.8	0
094obs1los	25.9%	3.727272727	-37.64090909	0
095obs0dom	17.9%	5.181818182	-123.5918367	0
095obs0los	22.6%	4.28	-126.1666667	0
095obs1dom	23.7%	4.086956522	-123.5918367	1
095obs1los	34.7%	2.828571429	-163.6756757	1
096obs0dom	11.1%	7	-76.30526316	1
096obs0los	34.8%	2.791666667	-76.30526316	1
096obs1dom	13.0%	6.714285714	-74.34871795	0
096obs1los	21.8%	4.388888889	-78.36756757	0
097obs0dom	14.1%	6.545454545	-68.61842105	1
097obs0los	13.0%	7	-68.61842105	1
097obs1dom	27.0%	3.571428571	-66.85897436	1
097obs1los	29.3%	3.32	-54.32291667	1
098obs0dom	15.3%	6.083333333	-34.79795918	0
098obs0los	24.7%	3.92	-35.52291667	0
098obs1dom	26.9%	3.590909091	-34.79795918	0
098obs1los	24.2%	4	-43.72051282	0
099obs0dom	16.4%	5.6	-108.7510638	0
099obs0los	34.0%	2.882352941	-111.1152174	0
099obs1dom	20.7%	4.538461538	-108.7510638	1
099obs1los	28.0%	3.482758621	-138.1432432	1
100obs0dom	21.5%	4.444444444	-135.1265306	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
100obs0los	30.1%	3.230769231	-137.9416667	0
100obs1dom	16.3%	5.823529412	-135.1265306	0
100obs1los	27.9%	3.5	-137.9416667	0
101obs0dom	21.4%	4.384615385	-99.45918367	1
101obs0los	37.0%	2.657142857	-101.53125	1
101obs1dom	14.5%	6.363636364	-99.45918367	1
101obs1los	25.5%	3.807692308	-3.804449649	1
102obs0dom	15.6%	6.371794872	-6.072531646	0
102obs0los	13.3%	7.48241206	-7.51339076	0
102obs1dom	9.9%	10.03314917	-6.068690702	0
102obs1los	17.2%	5.789699571	-21.75646259	0
103obs0dom	11.6%	8.506493506	-42.43678161	1
103obs0los	20.3%	4.91509434	-44.84210526	1
103obs1dom	12.8%	7.717171717	-44.35440613	0
103obs1los	20.1%	4.96039604	-127.3103448	0
104obs0dom	13.6%	7.184210526	-72.93203883	0
104obs0los	20.4%	4.84	-76.48469388	0
104obs1dom	15.7%	6.230769231	-72.77184466	1
104obs1los	18.5%	5.333333333	-51.3390411	1
105obs0dom	8.4%	11.57894737	-62.46268657	0
105obs0los	15.6%	6.362068966	-64.58354756	0
105obs1dom	9.3%	10.57692308	-62.49502488	0
105obs1los	12.9%	7.666666667	-71.98567335	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
106obs0dom	6.9%	14.09756098	-35.91911765	1
106obs0los	17.0%	5.835820896	-36.00737101	1
106obs1dom	14.4%	6.878787879	-35.91911765	1
106obs1los	14.2%	6.973451327	-396.0810811	1

Table D.2: The training data supplied to the regression learner.

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
082obs0dom	17	5.333333	-199.068	0
022obs1los	26.9	3.578947	-7.395	1
021obs1los	39	2.516129	-29.261	1
091obs1los	30.9	3.171429	-176.16	1
060obs0los	19.7	4.769231	-27.6217	1
005obs1los	89.9	1.112601	-106.989	1
093obs1los	19.4	4.95	-106.248	1
010obs0dom	86.4	1.156499	-114.6	1
025obs0dom	26.6	3.611111	-22.0723	1
096obs1los	21.8	4.388889	-78.3676	0
056obs1los	37.9	2.565217	-75.1	1
094obs1los	25.9	3.727273	-37.6409	0
103obs0los	20.3	4.915094	-44.8421	1
073obs1dom	24.2	4	-49.6809	1
104obs0los	20.4	4.84	-76.4847	0
050obs0dom	5.9	9	-94.0125	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
018obs1los	33.7	2.909091	-17.8769	1
010obs0los	24.1	4	-128.243	1
101obs1dom	14.5	6.363636	-99.4592	1
089obs0dom	21.4	4.3	-66.751	1
071obs0dom	19.8	4.833333	-52.0106	1
013obs1los	31.4	3.148148	-65.4321	1
029obs1los	30.6	3.15	-55.9073	1
053obs1dom	28	3.4	-79.2667	1
008obs1dom	93.4	1.070652	-72.6426	1
012obs1los	22.2	3.333333	-24.9894	1
102obs0dom	15.6	6.371795	-6.07253	0
049obs1dom	10.5	7.8	-14.3568	1
004obs0los	23.9	4	-17.433	0
049obs1los	32.9	2.961538	-18.237	1
071obs0los	38.7	2.540541	-53.1413	1
043obs1los	22.1	4.333333	-70.2045	1
033obs0los	28.1	3.515625	-28.1985	0
092obs0dom	23.5	3.888889	-88.5633	1
095obs1dom	23.7	4.086957	-123.592	1
098obs1los	24.2	4	-43.7205	0
071obs1dom	25.4	3.777778	-52.0106	0
052obs1los	31.7	3.05	-100.59	1
102obs0los	13.3	7.482412	-7.51339	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
096obs0dom	11.1	7	-76.3053	1
073obs0los	29.4	3.307692	-50.7609	0
056obs0dom	13.8	6	-100.133	1
026obs0dom	6.6	12.83333	-90.7063	1
014obs0los	28.9	3.36	-2.64545	1
068obs0los	27.5	3.521739	-76.9891	0
006obs1los	34.1	2.875	-83.0375	1
102obs1dom	9.9	10.03315	-6.06869	0
066obs1dom	20.3	4.705882	-73.566	1
025obs0los	26.3	3.681818	-22.5522	1
008obs0dom	25	3.75	-72.6426	0
029obs0dom	27.5	3.466667	-147.884	1
096obs0los	34.8	2.791667	-76.3053	1
072obs1los	36.6	2.685714	-50.4217	0
027obs0dom	16.9	5.766667	-101.206	1
052obs1dom	9.1	6	-104.182	1
095obs0los	22.6	4.28	-126.167	0
061obs0los	23.6	4.090909	-22.5848	0
068obs1dom	26.2	3.666667	-75.3511	1
095obs1los	34.7	2.828571	-163.676	1
101obs0dom	21.4	4.384615	-99.4592	1
100obs0dom	21.5	4.444444	-135.127	0
032obs1dom	20.4	4.7	-40.7031	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
093obs0dom	6.8	12	-104.08	1
064obs0los	28.7	3.384615	-138.739	1
049obs0los	28.9	3.36	-15.3357	1
019obs1dom	17.8	5.485714	-88.934	0
083obs1los	30	3.24	-192.214	1
033obs0dom	19.2	5.038462	-27.9832	0
017obs0los	17.8	5.558442	-172.831	1
057obs0los	27.7	3.53125	-52.6674	1
028obs0dom	20.4	4.804878	-45.2419	0
088obs1dom	16.3	5.8	-90.2157	1
019obs0dom	17.9	5.433333	-88.934	0
036obs1dom	8.9	9.2	-32.0447	1
005obs0los	25.6	3.636364	-144.023	1
056obs1dom	35.1	2.714286	-100.133	1
043obs0los	27.4	3.541667	-67.1522	1
014obs0dom	10.2	9	-2.26333	1
028obs1dom	13.7	7.117647	-45.2419	0
061obs0dom	23.9	3.916667	-22.1043	0
099obs1los	28	3.482759	-138.143	1
014obs1dom	30.8	3.142857	-2.26333	0
097obs0los	13	7	-68.6184	1
029obs0los	31.9	3.043478	-143.263	1
017obs1los	20.1	4.931034	-266.16	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
006obs0los	32.1	3.035714	-72.2065	1
043obs1dom	17.6	5.375	-65.7234	1
070obs1dom	28.6	3.380952	-30.9596	1
010obs1dom	15.9	5.625	-114.6	0
024obs0los	29	3.357143	-26.2159	1
012obs1dom	55.6	1.666667	-234.9	1
065obs0los	23.8	4.05	-20.7511	1
060obs1dom	21.5	4.444444	-27.034	0
026obs1dom	19	5.163265	-90.7063	0
020obs1dom	14	6.714286	-34.3381	0
055obs1los	30.1	3.230769	-9.34243	0
035obs0dom	35.7	2.730769	-19.0485	1
058obs0los	22.2	4.347826	-62.4739	0
094obs1dom	21.6	4.333333	-33.8	0
050obs1los	38.1	2.529412	-70.5094	0
064obs0dom	15.6	6	-129.883	1
022obs0los	30.8	3.142857	-20.8577	0
062obs1los	28.7	3.384615	-45.9522	0
054obs1los	22.1	4.375	-31.9174	1
059obs0dom	2.9	17.5	-88.5552	0
063obs0los	36.5	2.692308	-69.9304	0
050obs0los	15.6	5.75	-90.252	1
069obs1dom	20.9	4.533333	-50.2532	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
011obs0los	26	3.791667	-112.734	0
035obs1dom	25.9	3.6875	-19.0485	0
100obs1los	27.9	3.5	-137.942	0
008obs0los	30	3.25	-74.2217	0
013obs1dom	24.5	4	-56.7078	1
060obs1los	28.9	3.347826	-30.9902	0
023obs0los	26.6	3.714286	-66.4554	0
006obs0dom	88.6	1.128342	-70.6702	1
031obs0dom	9.4	10.25	-21.1151	1
062obs0dom	19.7	4.769231	-44.9745	1
065obs1los	20	4.809524	-20.7511	0
014obs1los	22.2	4.333333	-4.96829	0
059obs1los	10.2	8.333333	-65.8487	1
057obs1dom	39.8	2.472222	-51.5468	1
098obs1dom	26.9	3.590909	-34.798	0
050obs1dom	0	31	-94.0125	0
033obs1los	31.8	3.114286	-79.6913	0
015obs1dom	21.6	4.411765	-43.3286	0
034obs0dom	26.9	3.533333	-49.1787	0
023obs1dom	23.1	4.26087	-65.8673	1
023obs0dom	11.7	8.111111	-65.8673	0
009obs0dom	25.7	3.75	-76.4085	0
024obs1los	25.8	3.75	-26.2159	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
025obs1los	33.8	2.888889	-15.0348	0
045obs1dom	12.5	7.363636	-77.8787	0
083obs0los	29.5	3.291667	-154.607	0
092obs1dom	29	3.333333	-88.5633	0
054obs1dom	33.3	2.935484	-31.2383	1
095obs0dom	17.9	5.181818	-123.592	0
099obs0dom	16.4	5.6	-108.751	0
073obs0dom	24.7	3.913043	-49.6809	0
072obs1dom	27.8	3.5	-49.3489	0
086obs0dom	11.4	7.5	-44.2481	1
098obs0dom	15.3	6.083333	-34.798	0
106obs1los	14.2	6.973451	-396.081	1
067obs1dom	13.3	6.909091	-114.268	1
016obs0los	24.6	3.866667	-14.4107	1
055obs1dom	10.6	8.6	-7.35468	0
048obs1los	15.3	6	-107.037	0
053obs1los	25.5	3.714286	-69.9412	1
015obs0los	29.2	3.344828	-44.3854	0
083obs0dom	16.3	5.8	-151.317	0
054obs0dom	13.9	6.636364	-28.0489	0
010obs1los	42.6	2.3	-57.3	0
012obs0los	55.6	1.666667	-195.75	1
070obs1los	27.2	3.565217	-31.6326	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
098obs0los	24.7	3.92	-35.5229	0
097obs1dom	27	3.571429	-66.859	1
042obs1dom	12.3	7.4	-26.0553	1
047obs1los	23.9	3.916667	-29.0238	1
047obs0dom	7.7	7	-30.1848	1
030obs1los	22.1	4.435897	-25.8127	0
093obs0los	13.8	6.785714	-106.248	1
080obs0dom	36.6	2.666667	-36.6043	0
100obs0los	30.1	3.230769	-137.942	0
103obs0dom	11.6	8.506494	-42.4368	1
086obs1dom	15	5.857143	-44.2481	0
030obs0dom	17.9	5.333333	-53.0283	0
091obs0los	21.3	4.5	-154.14	1
090obs1los	24.7	3.913043	-199.958	1
042obs0los	32.2	3.033333	-26.6217	0
081obs0dom	16.7	5.375	-195.047	0
004obs0dom	31.1	3.1	-16.9742	0
088obs0los	30.1	3.241379	-92.02	0
056obs0los	21.3	4.363636	-96.128	1
068obs0dom	17.2	5.416667	-75.3511	0
005obs1dom	23.5	4	-149.784	1
008obs1los	18.8	5.105263	-83.2732	1
022obs0dom	23.1	4.076923	-19.3679	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
033obs1dom	29.3	3.37037	-27.9832	0
026obs1los	19.5	5.054545	-128.426	0
013obs0dom	23.4	4.166667	-56.7078	1
082obs1dom	26.2	3.583333	-199.068	0
036obs1los	32.2	3	-32.7413	1
094obs0dom	20.3	4.615385	-33.8	1
102obs1los	17.2	5.7897	-21.7565	0
089obs0los	28.8	3.387097	-68.086	1
051obs1dom	26.9	3.590909	-45.566	1
093obs1dom	15.9	5.833333	-104.08	1
097obs0dom	14.1	6.545455	-68.6184	1
032obs0dom	21.1	4.529412	-40.7031	0
016obs1los	24.7	3.92	-2.85728	1
009obs1los	40.6	2.413793	-78.0696	1
045obs1los	25.3	3.833333	-98.927	0
100obs1dom	16.3	5.823529	-135.127	0
031obs0los	29.1	3.409524	-21.2257	1
045obs0los	25.5	3.807692	-79.5717	1
047obs0los	20.3	4.642857	-29.0238	1
031obs1dom	21.5	4.603175	-21.1151	1
011obs0dom	19.9	4.916667	-111.547	0
089obs1dom	14.1	6.545455	-66.751	0
034obs1dom	21.6	4.222222	-49.1787	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
035obs0los	27.8	3.5	-19.4626	1
088obs1los	28.4	3.428571	-92.02	1
019obs1los	25.3	3.893617	-110.364	0
065obs1dom	16.5	5.733333	-20.3096	0
055obs0dom	34.2	2.851852	-7.35468	1
018obs0los	23.5	4.12	-27.888	1
021obs1dom	14.3	6.4	-23.2128	1
017obs1dom	18.6	5.181818	-57.1159	1
072obs0los	30.5	3.2	-50.4217	1
031obs1los	26.5	3.72973	-63.3453	1
030obs0los	27.2	3.62963	-53.611	0
059obs0los	34	2.833333	-85.6033	0
057obs1los	27.1	3.592593	-57.6833	1
018obs1dom	34.9	2.8	-27.3412	1
097obs1los	29.3	3.32	-54.3229	1
009obs0los	21.2	4.526316	-78.0696	0
053obs0los	25.7	3.75	-76.7097	0
080obs1los	24.7	3.92	-46.4973	1
009obs1dom	89.8	1.112861	-76.4085	1
059obs1dom	40	2.4	-88.5552	1
032obs1los	27.6	3.55814	-20.3515	1
011obs1los	21.6	4.547619	-286.405	0
048obs0los	18.6	5.071429	-140.255	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
025obs1dom	10.9	8.125	-22.0723	0
103obs1los	20.1	4.960396	-127.31	0
089obs1los	39.6	2.487805	-70.9229	0
058obs1dom	22.4	4.214286	-61.1447	1
104obs0dom	13.6	7.184211	-72.932	0
027obs0los	23.4	4.2	-102.208	1
088obs0dom	24.2	3.941176	-90.2157	0
055obs0los	28.4	3.433333	-7.51457	1
069obs1los	33.3	2.931034	-51.3457	1
057obs0dom	26.1	3.615385	-51.5468	1
063obs1los	34.1	2.870968	-69.9304	1
051obs0dom	8.2	10	-45.566	1
028obs1los	23.2	4.253731	-180.968	0
066obs0los	27.3	3.571429	-75.1652	1
090obs1dom	21.5	4.47619	-176.433	1
063obs0dom	14.7	5.833333	-68.4426	0
023obs1los	28.3	3.490909	-161.804	1
034obs1los	16.7	5.615385	-50.2478	1
051obs1los	31.1	3.137931	-57.8811	1
013obs0los	28.9	3.404255	-57.3449	1
020obs1los	24.4	4.025641	-62.7043	0
024obs1dom	12.5	7.125	-24.5426	1
106obs0los	17	5.835821	-36.0074	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
070obs0los	34.6	2.821429	-31.6326	0
064obs1dom	24.3	3.944444	-129.883	1
083obs1dom	20	4.714286	-151.317	1
063obs1dom	33.3	2.925926	-68.4426	1
081obs0los	30.3	3.208333	-200.319	0
005obs0dom	15	5.25	-149.784	1
011obs1dom	14.6	6.625	-111.547	0
061obs1dom	20.2	4.722222	-22.1043	0
070obs0dom	24.4	3.95	-30.9596	0
036obs0dom	25.5	3.733333	-32.0447	1
066obs0dom	19.4	4.866667	-73.566	1
035obs1los	26.2	3.695652	-19.4626	0
020obs0los	22.5	4.34375	-35.6099	0
092obs1los	29.8	3.28125	-103.324	0
086obs1los	25.5	3.692308	-34.1343	0
073obs1los	35.1	2.785714	-50.7609	1
099obs0los	34	2.882353	-111.115	0
106obs0dom	6.9	14.09756	-35.9191	1
104obs1dom	15.7	6.230769	-72.7718	1
054obs0los	26	3.714286	-31.9174	0
066obs1los	36.8	2.666667	-78.5818	1
092obs0los	27.2	3.576923	-90.4083	1
082obs0los	34.4	2.818182	-192.847	0

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
043obs0dom	15.1	6.166667	-65.7234	1
099obs1dom	20.7	4.538462	-108.751	1
016obs0dom	5.4	14.25	-14.104	1
081obs1dom	28.8	3.333333	-195.047	1
004obs1los	30.5	3.157895	-17.433	0
067obs1los	37.1	2.648649	-134.265	1
062obs1dom	20.3	4.615385	-44.9745	0
090obs0dom	25.3	3.833333	-176.433	0
086obs0los	23.3	4	-42.6679	1
004obs1dom	23.9	4	-16.9742	0
022obs1dom	25.9	3.666667	-18.0779	1
028obs0los	23.9	4.138462	-45.5497	0
091obs0dom	22.2	4.266667	-150.994	1
080obs0los	23.9	4.043478	-37.4	0
103obs1dom	12.8	7.717172	-44.3544	0
052obs0dom	9.7	8	-104.182	1
104obs1los	18.5	5.333333	-51.339	1
045obs0dom	11.1	8	-77.8787	1
016obs1dom	20.8	4.5	-14.104	1
032obs0los	27.9	3.525	-41.3391	0
101obs0los	37	2.657143	-101.531	1
065obs0dom	18.2	5.153846	-20.3096	1
017obs0dom	15.5	6.362069	-57.1159	1

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
012obs0dom	0	8	-234.9	1
052obs0los	45.9	2.111111	-100.59	1
090obs0los	23.5	4.095238	-179.962	0
105obs1los	12.9	7.666667	-71.9857	0
081obs1los	22.1	4.3125	-200.319	1
067obs0los	34.4	2.84375	-116.752	1
082obs1los	18.8	4.9	-150.515	0
049obs0dom	2.6	20	-14.3568	1

Table D.3: The data used as test sets for the regression learner.

Trajectory ID	Percentage visible	Average # frames between visibility	Average change in speed	Pursuit
027obs1los	26.6	3.7	-72.1888	0
048obs0dom	15.4	5.4	-145.264	1
058obs1los	30.6	3.193548	-77.6703	1
101obs1los	25.5	3.807692	-3.80445	1
047obs1dom	15.4	5.4	-30.1848	1
105obs0los	15.6	6.362069	-64.5835	0
030obs1dom	12.8	7.473684	-53.0283	0
072obs0dom	25.4	3.777778	-49.3489	1
068obs1los	27.4	3.541667	-76.9891	1
058obs0dom	16	5.846154	-61.1447	0
071obs1los	25.6	3.772727	-53.1413	0
069obs0dom	22.1	4.35	-50.2532	1

020obs0dom	19.1	5	-34.3381	0
015obs1los	35.5	2.764706	-39.5609	0
106obs1dom	14.4	6.878788	-35.9191	1
019obs0los	27.9	3.542373	-89.8059	0
006obs1dom	92.5	1.080863	-70.6702	1
060obs0dom	29.3	3.32	-27.034	1
096obs1dom	13	6.714286	-74.3487	0
069obs0los	22.9	4.24	-51.3457	1
021obs0los	39.5	2.484848	-27.2659	1
036obs0los	36.9	2.65625	-34.2295	1
051obs0los	37.2	2.638889	-46.5565	1
105obs0dom	8.4	11.57895	-62.4627	0
018obs0dom	23	4.166667	-27.3412	1
094obs0los	23.6	4.111111	-36.0043	1
067obs0dom	20.9	4.533333	-111.302	1
042obs0dom	18.9	5	-26.0553	0
021obs0dom	19.3	4.941176	-25.5255	1
048obs1dom	16.7	5.375	-145.264	0
053obs0dom	0	24	-84.9286	0
015obs0dom	11.3	7.714286	-43.3286	0
062obs0los	36.4	2.69697	-45.9522	1
024obs0dom	32	3.04	-24.5426	1
042obs1los	31.7	3.05	-26.6217	1
061obs1los	24.7	3.916667	-23.6114	0

026obs0los	25.3	3.915493	-91.3451	1
029obs1dom	18.6	4.888889	-147.884	1
064obs1los	23.1	4.181818	-132.707	1
027obs1dom	28.9	3.411765	-101.206	0
091obs1dom	21	4.555556	-150.994	1
105obs1dom	9.3	10.57692	-62.495	0
080obs1dom	21.4	4.4375	-36.6043	1
034obs0los	26	3.714286	-50.2478	0