**Video Tracking and Data Analysis Software for Monitoring Group Behaviour in 2D videos**

Final Report for CS39440 Major Project

*Author*: Oliwia Marek (okm@aber.ac.uk)

*Supervisor*: Dr. Otar Akanyeti (ota1@aber.ac.uk)

23rd March 2018

Version 0.1 (Draft)

This report is submitted as partial fulfilment of a BSc degree in  
Artificial Intelligence and Robotics (GH7P)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

**Declaration of originality**

I confirm that:

* This submission is my own work, except where clearly indicated.
* I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
* I have read the regulations on Unacceptable Academic Practice from the University’s Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
* In submitting this work I understand and agree to abide by the University’s regulations governing these issues.

Name …………Oliwia Marek……………

Date ………………23/03/18………………

**Consent to share this work**

By including my name below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name …………Oliwia Marek……………

Date ………………23/03/18………………

**Acknowledgements**

I am grateful to…

I’d like to thank…

**Abstract**

Include an abstract for your project. This should be no more than 300 words.

**Contents**

1. Background, Analysis & Process 8

1.1. Introduction 8

1.2. Motivation 8

1.2.1. Collective learning 8

1.2.2. Visual tracking system for multiple fish in 2D videos 8

1.3. Aim 8

1.4. Objectives 9

1.5. Deliverables 9

1.6. Methodology 9

2. Research 10

2.1. Introduction 10

2.2. Experiments 10

2.2.1. Aim 10

2.2.2. Execution 10

2.2.3. Data collection 10

2.3. Research Questions 11

2.3.1. How does learning progress overtime? 11

2.3.2. How can previous experience improve learning time? 11

2.3.3. Can we quantify interactions among fish while all the learning is happening? 11

2.4. Similar systems 12

2.4.1. ID Tracker 12

2.4.2. Zoo Tracer 12

2.4.3. Tracktor 13

2.5. Technology Overview 14

2.5.1. OpenCV 14

2.5.2. Programming Languages 14

2.5.3. Motion Detection Algorithms 14

3. Design 16

3.1. Introduction 16

3.2. Overall Architecture 16

3.3. Development Choices 16

3.3.1. Programming Language 16

3.3.2. Development Environment 16

3.3.3. Video Tracking algorithm 17

3.3.4. Version Control 17

3.4. Class Design 17

3.4.1. FishTracker 17

3.4.2. BackgroundSubtractor 17

3.4.3. Globals 17

3.5. Manual Tracking 17

3.6. Automated Tracking 17

4. Implementation 18

4.1. Introduction 18

4.2. Dataset preparation 18

4.3. Sprints 19

4.3.1. Sprint 1 (30th of January – 6th of February) 19

4.3.2. Sprint 2 (6th of February – 13th of February) 19

4.3.3. Sprint 3 (13th of February – 20th of February) 19

4.3.4. Sprint 4 (20th of February – 27th of February) 20

4.3.5. Sprint 5 (27th of February – 6th of March) 20

4.3.6. Sprint 6 (6th of March – 13th of March) 20

4.3.7. Sprint 7 (13th of March – 20th of March) 20

4.3.8. Sprint 8 (20th of March – 27th of March) 21

4.3.9. Sprint 9 (27th of March – 3rd of April) 21

4.3.10. Sprint 10 (3rd of April – 10th of April) 22

4.3.11. Sprint 11 (10th of April – 17th of April) 22

4.3.12. Sprint 12 (17th of April – 24th of April) 22

4.3.13. Sprint 13 (24th of April – 4th of May) 22

4.4. Algorithms 23

4.4.1. Manual Fish Tracking 23

4.4.2. Background Subtraction 23

4.4.3. ROI Division 23

4.4.4. Data Analysis 23

5. Testing 24

5.1. Introduction 24

5.2. Preparation and strategy 24

5.3. Unit Tests 24

5.3.1. FishTracker 24

5.3.2. Config 24

5.3.3. MainProgram 24

5.4. Integration and Regression Tests 25

6. Results and Conclusions 27

6.1. Introduction 27

6.2. Results 27

6.2.1. Accuracy of background subtraction 27

6.3. Conclusions 27

6.4. Future Work 27

7. Critical Evaluation 28

7.1. Introduction 28

7.2. Aim 28

7.3. Objectives 28

7.4. Design 28

7.5. Project Management 28

7.6. Testing 28

7.7. Further Development 28

8. Appendices 29

A. Third-Party Code, Libraries & Technologies Used 29

B. Ethics Submission 29

C. Code Examples 29

Manual Fish Tracking 29

Create Background Model 29

Background Subtraction 29

ROI Division 29

9. Annotated Bibliography 29

**List of Figures**

[Figure 1. Screenshot of the video from the first experiment. 11](file:///C:\Users\OLIwi\Google%20Drive\Dissertation_\Report\Video%20Tracking%20and%20Data%20Analysis%20Software%20for%20Monitoring%20Group%20Behaviour%20in%202D%20videos.docx#_Toc512347540)

[Figure 2. Screenshot of the video from second experiment with ring displacement shown. 11](file:///C:\Users\OLIwi\Google%20Drive\Dissertation_\Report\Video%20Tracking%20and%20Data%20Analysis%20Software%20for%20Monitoring%20Group%20Behaviour%20in%202D%20videos.docx#_Toc512347541)

[Figure 3. Screenshot of idTracker taken from video on authors website [23] 13](#_Toc512347542)

[Figure 4. Screenshot of ZooTracer from authors website [24] 14](#_Toc512347543)

[Figure 5. Screenshot of application Tracktor taken from authors website [25] 14](#_Toc512347544)

[Figure 6. Picture before and after the video edit 19](file:///C:\Users\OLIwi\Google%20Drive\Dissertation_\Report\Video%20Tracking%20and%20Data%20Analysis%20Software%20for%20Monitoring%20Group%20Behaviour%20in%202D%20videos.docx#_Toc512347545)

[Figure 7. Screenshot of the GoProStudio with preset filter selected. 20](#_Toc512347546)

[Figure 8. Photo of initial Background Subtraction output window. 22](#_Toc512347547)

[Figure 9. Screenshot of initial Background Subtraction with assuming the fist frame is the background. 22](#_Toc512347548)

[Figure 10. Screenshots of background models for different values of threshold. 31](#_Toc512347549)

[Figure 12. Screenshots of background models for different value of waiting frames. 32](#_Toc512347550)

**List of Tables**

[Table 1. Test cases for the project. 26](#_Toc511649025)

Chapter 1

# Background, Analysis & Process

This chapter discusses my preparation, describes the background of the project, the objectives and processes followed to help structuring the work.

## Motivation

### Collective learning

Learning is important to adapt to dynamic environments. (What is learning and why is it important?)

Therefore scientists put a lot of effort to understand how animals learn.

Learning is complex and there are many different levels.

One of the basic learning paradigms is classical conditioning which has been used quite often in the past.

What is classical conditioning?

These experiments are usually done on isolated animals

What have been people studying? (molecular, neural, physiological and behaviour mechanism of learning). This is what previous research has focused on so far.

However, we know that animals usually live in groups and how group dynamics affect learning is much less understood.

Previous studies have provided compelling evidence that when animals occupy groups, they may have enhanced movement, sensing and decision making capabilities.

However, collective learning has not been investigated as much.

Be careful here because social learning has been investigated (information transfer from knowledgeable individuals to naïve ones)

One novel aspect of this study is to investigate when all group members are naïve to the task (no prior knowledge of the task) and they have to figure out the solution without a help.

And people have always been interested in how individuals within a group communicate with each other. In large groups one exciting idea is that individuals receive/transmit information indirectly by watching the movements of the neighbours (there is no direct, verbal or visual communication) This suggests that when one animal increases a movement speed, the others may follow the suit. The animals may infer what is going on by watching the movement patterns of the other neighbours.

We know a lot about learning, about collective behaviour and also learning on isolated individual. This learning paradigm was used many times before on multiple experiments, usually on individual animal. The experiments are starting after assuming that the subjects of experiment already learned what they needed or after teaching part of the group [1].

Additionally, the research on learning and collective behaviour was carried out on individuals, not in a group of animals, fish in this case.

We have lots of inspiring knowledge about how fish or birds move together in groups, how they make decisions. However, it’s much less studied about how the group as collective unit acquires the new information and encodes it.

This is important because usually animals are social and live in groups and rarely do things alone. This also applies in humans, we like doing things together, learning together. An example of that could be students in a lecture hall, they all sit together, learning and then meet up to share their knowledge between each other and discuss. Lots of interactions are changing the dynamics of learning.

These are much less understood compared to learning dynamics on individual. This is why this research may be really interesting and important. Learning new information about collective learning will not focus on only the fish, but it will hopefully be applicable to the bigger picture.

Many fish species, as with other animals, present collective behaviour, mainly to increase their survivability. Fish schooling is a group of fish in which they swim together in tight, synchronized fashion [4]. It is believed that this social behaviour serves, in part, as a protection against predation through increasing the detection of predators and decreasing the probability of individual capture [5].As first shown by McCann [6], shoaling preferences in zebrafish have been shown to be at least partly learned.

A basic animal instinct is to react after stimulation (or sometimes, lack of it). [2] Fish seem to actively perform in a way to achieve collective goals such as finding food. However, in regions with limited supply of food, it increases competition, fish want to get to the food as quickly as possible.

### Visual tracking system for multiple fish in 2D videos

Video tracking systems enable researchers to study behaviour in a reliable and consistent way and over longer time periods than if they were using manual recording

Although previous works in the literature [8][9][10] have similar titles or even motivation as this one, they targeted different computational problems and used approaches that are distinct from the one used and described in this project.

## Aim

The aim for this project is to produce a software solution that will take 2D videos of fish from 3 experiments carried out by Dr Otar Akanyeti and successfully track and count the number of fish in different sections of the image. The solution will also involve a software that will create graphs from gathered data and let us analyse it.

## Objectives

The objectives for this project include:

* Produce an algorithm for manual tracking of the fish in 2D videos.
* Produce an algorithm for an automated tracking of the fish in 2D videos.
* Produce multiple graphs from gathered data.
* Allow for storing collected data for possible future expansion of the project.

## Deliverables

The deliverables for this project include:

* The source code both for video tracking of the fish as well as for the analysis of collected data.
* This project report that discusses the motivation, research, design and critical evaluation of the project.

## Methodology

There are many processes that can be used during the development of the project, both agile and classic methods such as Waterfall. The methodology chosen for this project was an agile approach: Scrum.

The sprints for the project were one week long with Sprint Planning and Retrospective together with the tutor, Otar Akanyeti. During these the work done during the print was presented, reflected upon and new stories added to the current task board, the new sprint goal. The task board was created and updated every day or when necessary together with the Backlog with all the stories in it. More details will be described in Chapter 4.

Daily sprint meetings were carried out by writing down what have been accomplished and what has to be done the following day as well as looking back at the one written the day before.

Although Scrum is known to be used only for teams with a number of developers, I found that there is a number of people that use Scrum in their One-Person Operations[18]. Mike Cohn, founder of Mountain Goat Software shared his way of managing day to day work using scrum, in his recent blog[19].

**Chapter 2**

# Research

This chapter discusses the experiments, relevant research and technologies together with similar systems that already exists.

## Experiments

Figure 1. Screenshot of the video from the first experiment.

### Aim

The idea for the whole project came from experiments carried out by Dr Otar Ananyeti. He had been fascinated by collective learning of fish and decided to see how do they perform together and if they can associate a dipole with food.

In the trials fish were presented with a buzzer and after 10 seconds the food was dropped into the food ring.

The experiments were inspired by Pavlov’s Dogs. Ivan Pavlov was a Russian physiologist working on digestive processes. During his study on dogs, he noticed that when an assistant entered the room, his subjects would start to salivate. This process is now known as classical conditioning – a learning process that occurs though associations between an environmental stimulus and a naturally occurring stimulus. [11]



Figure 2. Screenshot of the video from second experiment with ring displacement shown.

The difference between the work that Pavlov carried out was that he was teaching the dogs individually, while Dr Akanyeti taught the fish at the same time.

### Execution

The experiment involved 24 zebra fish(See Fig. 1) in a 37,85 litre (10 gallon) tank. There were 3 experiments, each had a 5 week training period for the fish.

After first 5 weeks of experiment with good results, the placement of the food ring was changed (see Figure 2) for the second experiment that lasted another 5 weeks. After successful execution, the placement was changed once more for the final, third experiment lasting the same amount of time.

### Data collection

The fish were recorded from the bottom of the tank using a GoPro camera. The camera had a wide lens to capture the whole tank but created a fisheye effect. Fisheye is a visual distortion intended to create a wide usually hemispherical image. It uses special mapping which gives images non-rectilinear appearance. [3]

The fish were recorded twice a day during their feeding time. Each of the videos is at least 10 minutes long to capture 5 minutes before and after the feeding.

## Research Questions

There are a lot of questions these experiments have risen. Given more time, I would love to try to expand my research and try to focus on them and answer all of the questions listed below.

### How does learning progress overtime?

Looking at the experiments carried out, at first glance it may look simple. But when you actually look into it, you realise how complex it actually is.

There are multiple levels of learning. Fish need to find the location of the food source, they also need to find the buzzer. After that they need to associate the buzzer with food, learn that it is an alert that food is going to arrive. And additionally learn that they have to wait 10 seconds after the buzzer starts before the food is actually released.

One more layer is that they need to learn that they are not going to be fed until the next trial after few hours. Maybe they have a short term memory and will remember that they just ate in that location so they will stay around the food source long after the food is already dropped and eaten.

### How can previous experience improve learning time?

As said before, the placement of the food ring was changed in second and third experiments. This may raise a question if previously taught fish will be able to go to the new location much faster than in the first experiment. There is also a question if the fish are associating food with the food ring or maybe the location in the tank.

### Can we quantify interactions among fish while all the learning is happening?

An initial assumption from looking at the videos is that all of the group members associate the buzzer with food. But alternative, more possible hypothesis is that some fish learn the correct association and some acquire knowledge to follow the other fish that actually make the association.

.

## Similar systems

### ID Tracker

It is a video tracking software that keeps models of object to identify and track them throughout the frames of the whole video.

It requires Matlab Compiler Runtime 8.3 to run and runs on Windows 7 and XP.

It uses background subtraction, thus it requires static background. It also works best if it is uniform with a colour that gives big contrast with animals. A multitracking algorithm that extracts a characteristic fingerprint from each animal in a video recording of a group. IdTracker then uses these fingerprints to identify every individual throughout the video. Tracking by identification prevents propagation of errors, and the correct identities can be maintained indefinitely.

It was impossible to run the program on computer that was being used by the developer due to it crashing mid-way through. The cause of the problem was unfortunately not found but after some research, most probable reason is that the computer was too slow and it crashed while doing the process of segmentation.

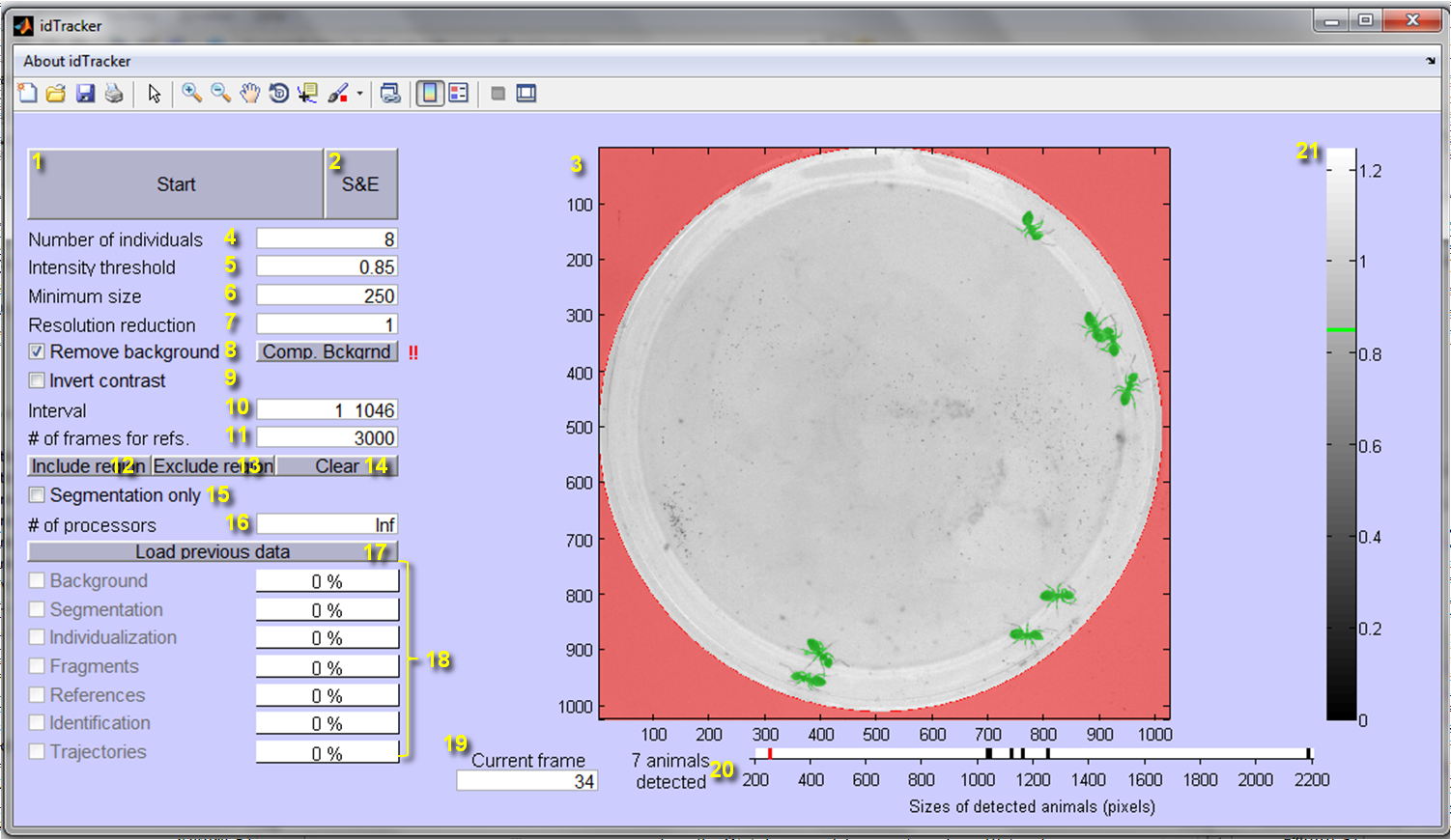


Figure 3. Screenshot of idTracker taken from video on authors website [23]

### Zoo Tracer

Zootracer is software developed at Microsoft Research [22], capable of accurately tracking multiple, unmarked, interacting individuals in arbitrary video footage. It implements an interactive algorithm from paper [21]. As the software is independent of recording devices, the user may provide recordings from any habitat type. With its intended ability to robustly cope with variations in lighting, camera movement and changes in object appearance, Zootracer represents a step forward in facilitating the collection and analysis of behaviour not just for laboratory experiments, but more importantly, for biologists in the field.

Requirements include 64bit computer with Windows Operating System. You also need to have a specific version of OpenCV, which is v.2.4.8. This version is no longer available, thus it was impossible to even run the application because it does not work with later versions.

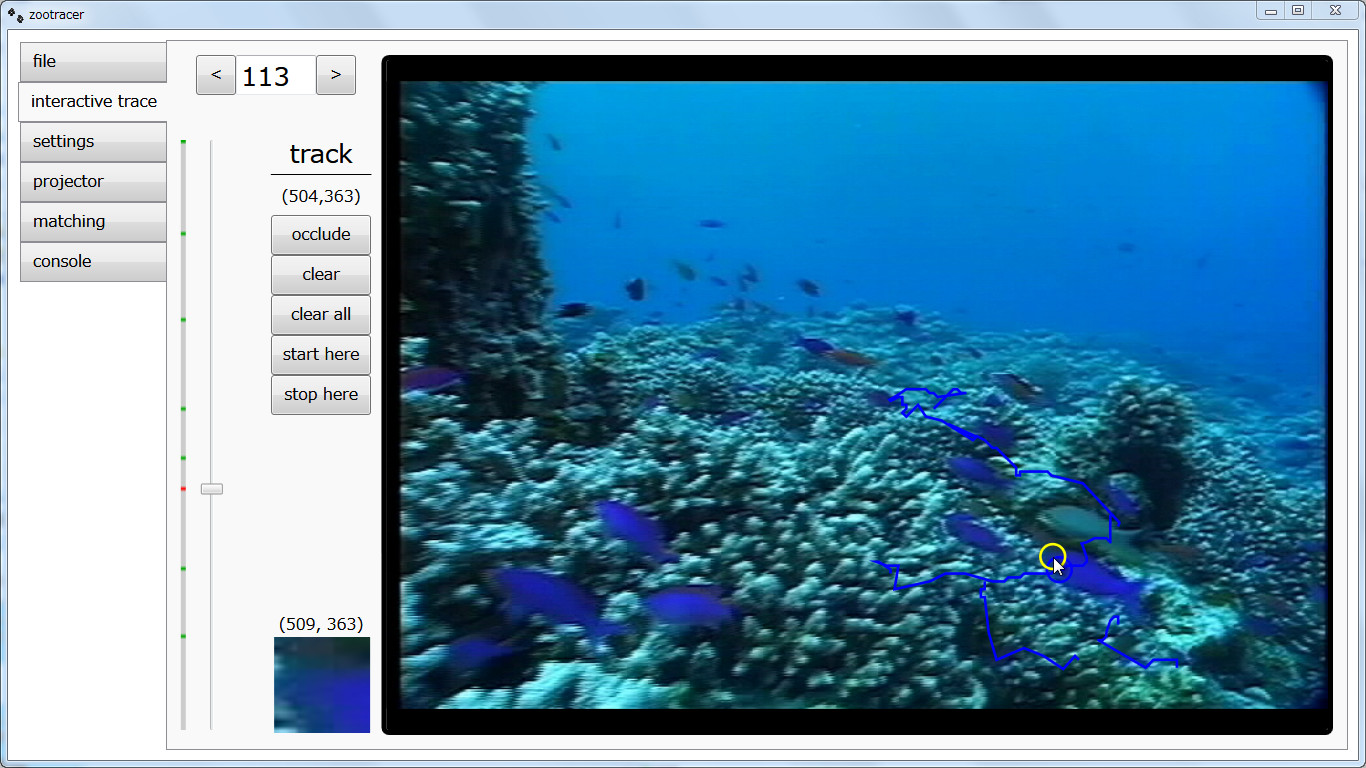


Figure 4. Screenshot of ZooTracer from authors website [24]

### Tracktor

It is an OpenCV based object tracking software. The software is able to perform single-object tracking in noisy environments or multi-object tracking in clear environments. In addition to tracking multiple objects simultaneously, tracktor is also able to maintain individual identities. [25]



Figure 5. Screenshot of application Tracktor taken from authors website [25]

## Technology Overview

### OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source library of functions used during image processing. It was released under BSD licence making is available for free for academic and commercial use [15]. It was released in June 2000 being initialised by Intel Corporation.

The library is cross-platform, supports Windows, Linux, Mac OS, iOS and Android. Written in C/C++, the library is available in few more programming languages not only C++, including Python and Java (see 2.5.2)

It is used in a big number of applications, including detecting faces, identifying objects, extracting 3D models of objects and tracking moving objects.

An applications can be created using tools and APIs from the library for an end-user and therefore, OpenCV has no direct contact to the customer or end user but it is there in most of the image processing applications.[17]

### Programming Languages

**C++** is one of the most popular programming languages especially in graphical applications, for instance the primary interface of OpenCV is in C++. It is a general-purpose OOP (Object-Oriented Programming) language. It was developed by Bjarne Stroustrup and it is an extension of the C language [16].

The main highlight of C++ is a collection of predefined classes, which are data types that can be instantiated multiple times. Classes can further accommodate member functions to implement specific functionality. Multiple objects of a particular class can be defined to implement the functions within the class. Objects can be defined as instances created at run time. These classes can also be inherited by other new classes which take in the public and protected functionalities by default.

**Python** was created by Guido van Rossum in the late 1980s[17]. Today it is developed by the Python Software Foundation. It is a multiparadigm, general-purpose, interpreted, high-level programming language. It allows use of different programming styles to create programs in a easily readable, almost human language way.

Python is an interpreted languages, the code does need to be compiled before executing a program because it is done in the background.

Although Python offers fast development times, it lags slightly in terms of execution time. Compared to fully compiling languages like C and C++, Python programs execute slower. But with the processing speeds of computers these days, the speed differences are usually only observed in benchmarking tests, not in real-world operations. Another thing is that it can be easily extended with C/C++. This allows programmers to write their code in C/C++ and then create a wrapper in Python, merging speed and readability. One of examples of that is OpenCV-Python, which is a wrapper over OpenCV. It also support Numpy (highly optimised library for numerical operations) as well as several other libraries like SciPy[17].

**Java** is a general-purpose, concurrent, strongly typed, class-based object-oriented language[20]. It is fast, reliable and secure. According to Oracle, Java runs on 3 billion devices worldwide.

Java is an OOP language which makes code reusable and makes it easier to maintain. It is also platform independent, using WORA (“write once, run anywhere”). There is no need for modification when changing the operating systems.

### Motion Detection Algorithms

There are two methods of motion detection: You can find the things that are not moving and ignore them or try to find the motion directly. Once the pixels representing motion are found, they can be grouped together into “objects” or “blobs”.

**Background Subtraction(foreground detection)** is a motion detection algorithm that created a model of a background and subtracts it from the current frame. It requires a static camera and makes a lot of assumptions including most of the scene being still and lighting not changing much throughout the video. Background subtraction works amazing if the background is static but has a lot of false positives when there is some flickering in the video (for example leaves blowing in the wind or shadows passing over the scene). Described in more details in section 3.3.

**Optical Flow** is a motion detection algorithm that detects the moving objects directly. It divides into two categories:

* Dense Optical Flow – it is appearance based, uses correlation coefficients on small image patches around each pixel. Assumes that points in image have about the same brightness, finds nearby points that satisfy this, work out direction of travel.
* Sparse Optical Flow – it is feature based. It first creates feature set in first frame, finds features in the next frame and then finds one that looks the most like the one in the feature set. It assumes objects do not move fast and that they move in fairly coherent way.

The advantage of the first is that it is generally faster while the second can give estimates for more pixels than the first.

**Chapter 3**

# Design

This chapter discusses the design and architecture of the program. It also includes the development choices in context of programming language and algorithms used.

## Overall Architecture

The process the application will follow goes as described:

* Take user input.

User can access the application from the terminal and provide appropriate flags described more in chapter whatever. User will be then prompt to choose a video file they want to open from their computer.

* Perform video tracking on the selected file

Depending on the flag, user can either run a manual tracking or do it automatically. The video will be playing with marks on each of the detected fish.

* Output the data

Application will output a csv file with all the data gathered for possible future work.

* Create multiple graphs (data analysis)

Application will output graphs from collected data for data analysis.

## Development Choices

### Programming Language

The decision for this project will be to use Python due to never using this language and this language being used throughout the industry. This programming language is widely used with OpenCV. The project is not including embedded devices, where C++ is widely used, or is not going to be used on Android/iOS devices, where Java is the main language.

Python is the most simple among the mentioned languages, its readability is also a plus in case a person wanting to use this application for any other research would like to expand the code. If in the future there will be need to create a website or server for the project, Python is also the most widely used language out of these three.

### Development Environment

The project will be developed using the PyCharm. It is an Integrated Development Environment developed by JetBrains. I am extremely familiar with products by this company, thus this IDE was extremely easy to understand and use.

PyCharm provides code analysis together with the normal text editor features. The programmer can use code completion as well as syntax and error highlighting. Another addition is the graphical debugger and integrated unit testing.

### Video Tracking algorithm

The decision for this project will be to use Background Subtraction because the background of the videos is static and Background Subtraction is less computationally expensive. However, there is some flickering in the video in the form of bubbles from the filter in the top left corner. Although, there are methods to reduce the false positives being detected in the video.

Like all background subtraction approaches, our proposed method is composed of a regularly updated background model and a similarity measure to compare a given frame with that background model.

### Version Control

The project was using Git as a software version control method. A private GitHub repository was created for the project which allowed for regular backup and code management. This prevented the need to keep multiple copies of code in case new code implementation breaks the solution. Another benefit of storing the code on GitHub is that in case of failure of a local computer, all of the code is still securely stored in a global solution on the website and can be easily accessed from a different machine.

## Program Design

### FishTracker

Description of the class

### BackgroundSubtractor

Description of the class

### Globals

Description of the globals

**Chapter 4**

# Implementation

This chapter describes the preparation and development of software during all 13 sprints of this project and algorithms developed and used in the program.

## Dataset preparation

The original videos from the experiments that were provided, were not the easiest to handle. Each of the videos were at least 2GB, they all had the audio included in them, even though it was unnecessary for the experiments. Additionally, since recorded on the wide lens on GoPro camera, there was a fish eye effect applied.

Thus, before any experiments were carried out, the videos had to be edited to make it available to video track and research it without carrying them on a portable hard disk (final result visible in Figure 6).

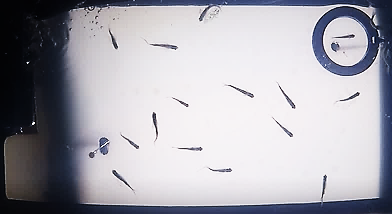


Figure 6. Picture before and after the video edit

Software used for edits was a free video editing software, GoProStudio, because there was an option to create a custom filter that encapsulated all the changes needed to be made (See Figure 7). That significantly decreased the time needed for editing the video since all it needed for most of the changes was just one click of the mouse.

The first step of editing prior to the filter was cutting the video into shorter one. Final videos were 2 minutes long with a minute before the buzzer and after. That step also included of removing the audio.

The custom filter created included of:

* Zooming in to get rid of the reflections of fish that would be problematic in automated detection of fish.
* Removal of the fish eye effect
* Changing the video from RGB to the greyscale
* Increase of the contrast and brightness to be able to distinguish the fish better.



Figure 7. Screenshot of the GoProStudio with preset filter selected.

## Sprints

### Sprint 1 (30th of January – 6th of February)

The purpose of the first sprint was purely research. I was away during the first week so I was unable to talk to my tutor (treated as Product Owner) about the software design yet.

The research included:

* Behaviour of fish in fish schools – gathered information about how and why they are behaving and what do we want to find during our research.
* Video Tracking tools for multiple agents – gather information about what tools are already available and how they perform in a case of our problem.
* Research about learning, collective behaviour and collective learning.

### Sprint 2 (6th of February – 13th of February)

The second sprint was when Otar and I had our first official meeting. He gave me videos of his experiments. We also talked about what we should focus on during out research. The initial idea was to use only videos from the fist experiment. We wanted to track each of the fish manually or in a semi-automated way to then later analyse speed or velocity of the fish and see if we can see any interesting patterns.

The videos were big files and had to be edited prior to being used in the video tracking. Thus, the tasks from this sprint included things already explained in Section 4.2. The videos had to be cut to make them shorter, they needed to be zoomed in and have the fisheye effect removed. Additionally, I increased contrast as well as converted them into greyscale. One last thing was to actually remove the audio to minimize the size of the files. I did all of this using the GoProStudio program.

### Sprint 3 (13th of February – 20th of February)

The third sprint was really busy. The previously researched video tracking algorithms had to be tested and analysed. Unfortunately, none of the video tracking tools were working, either due to crashing because my computer was too slow or need of version of OpenCV that was no longer available.

Therefore, next step taken was to install OpenCV on my computer that run Windows 10 and download PyCharm Community edition to be able to easily edit and run the code. There was a lot of settings that have to changed and libraries that had to be downloaded to enable everything to work together.

Next step was to create a private GitHub repository to back up my code and make it easily accessible from multiple devices.

One more thing done during this sprint was to research face transitions in context of fish tracking. But after a day, Otar and I decided to take a completely different approach and changed the aim of the project. Now we wanted to take into account all 3 experiments and analyse how fast the changing environment changes the behaviour and learning.

### Sprint 4 (20th of February – 27th of February)

The fourth sprint was when I was finally able to start coding. I got initial code from Otar to see how to use OpenCV. The code had a threading issue with having a setMouseCallback function inside the while loop. It was able to track a single fish by clicking at the fish manually every single frame. I refactored the code and ended up changing almost everything in the code. I made sure the code was changing the variables being passed in and the try catches were present in the appropriate places in the code.

The code was separated into a FishTracker class file and the MainProgram file. I was still able to gather output which was the coordinates of the fish, as well as create graphs of the fish during the process.

I also created a lot of unit test for the methods both used in main file and the class file.

### Sprint 5 (27th of February – 6th of March)

The fifth sprint brought a lot of changes and progress into the project.

I have researched the Regions Of Interest (ROI) and decided to separate the video into 6 ROI. This is further explained in section whatever. I created the function that took the video size and created the boundaries for each of the ROI to be initially stored in the FishTracker class.

Another important function created was the one which counted how many fish are in each of the ROI. It takes the coordinates of fish and then compares them to the boundaries of the regions. There is a list that stores how many fish are in each of the ROI.

### Sprint 6 (6th of March – 13th of March)

The sixth sprint was focused on starting to prepare myself for the Mid Project Demonstration I had on the 20th of March. I was cleaning up my code and creating the presentation to be used.

The rest of the sprint was occupied by finishing the Poster Assignment for another module I was partaking (CS38220). The assignment was closely linked to this project as it was to create a poster about the application and research I was carrying out. It was nice to do this together with preparing for the Mid Project Demonstration as they were both related.

### Sprint 7 (13th of March – 20th of March)

This sprint was intended to research the automated motion detection algorithms available in OpenCV. I had to install libraries mentioned in Appendix A.A and finally be able to successfully run background subtraction on the videos (see Figure 7 below).

For the rest of the week, the research about noise cancelling was carried out. There was a lot of noise, especially in the left upper corner as bubbles from the filter were present there.



Figure 8. Photo of initial Background Subtraction output window.

### Sprint 8 (20th of March – 27th of March)

In sprint 8, I decided I will write the background subtraction code myself instead of using the one built in the OpenCV. I managed to create a simple algorithm and get actual output of coordinates of detected objects out. The initial code I have written was assuming the first frame is the background model and then subtracting the current frame from the initial one. This was not giving good results (see Figure 8) as the first frame still had the fish in.

The rest of the week I had to spend finishing a Job Application assignment for a different module (CS38220).

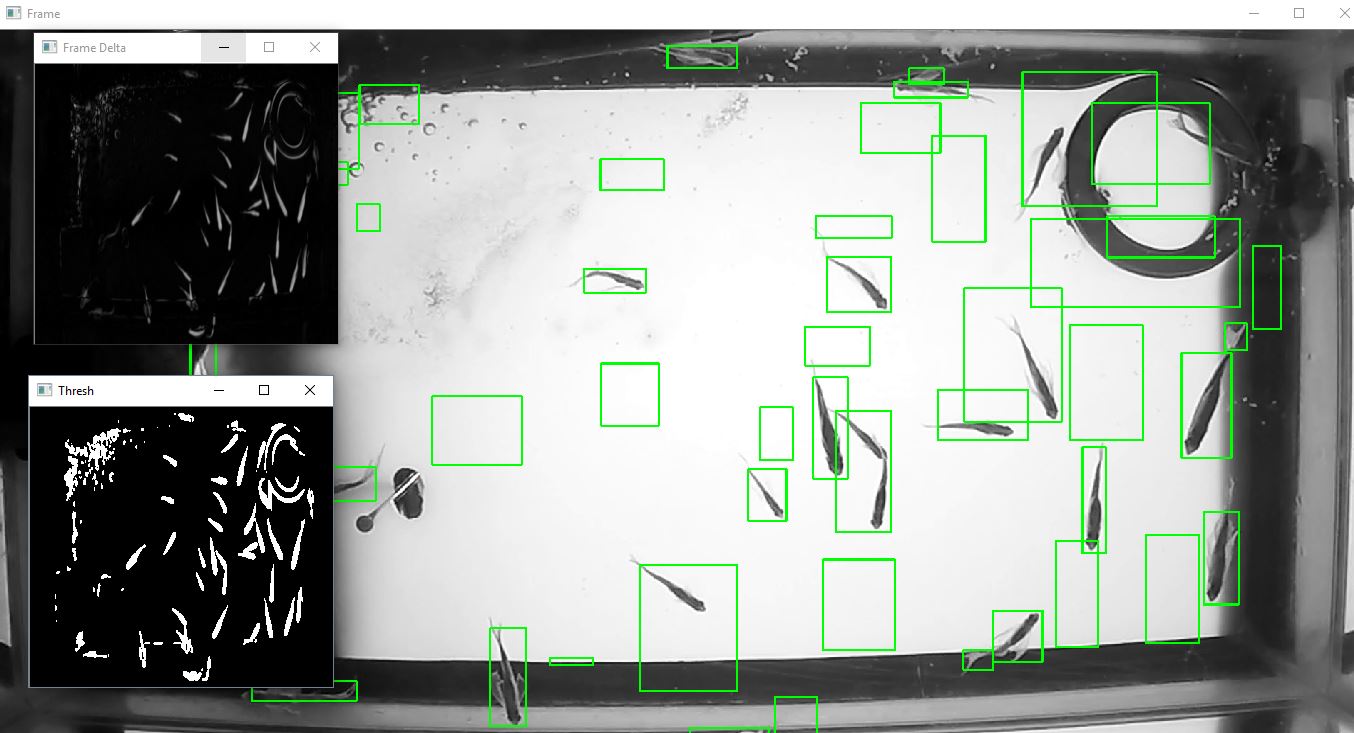


Figure 9. Screenshot of initial Background Subtraction with assuming the fist frame is the background.

### Sprint 9 (27th of March – 3rd of April)

This sprint was mostly dedicated to fixing the Background Subtraction algorithm. I initially changed the code to use moving average. It was working really well but it was taking a very long time until the model was good enough for detection to be almost perfect.

Therefore, the next step I have taken was to change the behaviour of the algorithm into fist creating the background model and then doing the actual detection of fish. This is further explained in Section 4.4.2. I got actually good results (see figure) and made the number of frames customizable as it was working better if the time to create the model was longer (further explained in Section 4.4.2)

After this I spent some time researching cancelling out noise in the algorithm. I ended up changing the minimum area size of the object detected and got fairly good results. Although, the bubbles were still being detected as there was a lot of it.

This was also when I started writing my report. I have written a draft of my motivation, description of experiments and the research questions.

### Sprint 10 (3rd of April – 10th of April)

The 10th sprint was dedicated to merging the Background Subtraction code with the main code I have already written. This was very difficult as both of the applications were working in a slightly different way. I had to divide the new code into much smaller and concise functions.

Additionally, the new code was doing just the detection and tracking of the fish. I had to add the functionality of dividing the video into ROI and counting the fish.

Another thing that I have done this sprint is creating the config file where I have put all of the global, customisable variables as well as functions used in multiple places.

The last thing done this week was editing the videos from experiment 2, as the only videos I had so far were the ones from the first experiment. So I had to use GoProStudio to zoom in, remove fisheye and edit the videos.

### Sprint 11 (10th of April – 17th of April)

In this sprint I was mainly focused on writing the report. I have edited previously written information and fixed the sectioning of the file. Finally, I have entirely finished writing chapters 1, 2, 3 and 4. I have also started on writing the draft of chapter 5.

One more thing done this week was fixing small issues, bugs in the code. There were few error catches that I have not thought about as well as some small errors in output writing.

Finally, I have written more unit tests, including ones for background subtraction and config file.

### Sprint 12 (17th of April – 24th of April)

This sprint was where I have run a lot of manual testing and got lots of output files. I have created multiple graphs, all presented in section Results and Conclusions.

I have also written the chapter 6 and 7 and re-read the previous ones. I have also sent the draft to my tutor to give me feedback and changed few parts of the report accordingly.

### Sprint 13 (24th of April – 4th of May)

The main purpose of the final sprint was to perform final bug fixing, cleaning up the code and making sure all of the comments are making sense.

I added the functionality for user to choose the location for the output files to be stored in in comparison to previously hard coded path.

The final part of the sprint was proof reading my report to make sure everything was easy to understand and correct.

## Algorithms

The code for all the algorithms are presented in the Appendix A.C.

### Manual Fish Tracking

Draw another or undo

Press ‘n’ to go to next frame

Yes

Yes

No

Is manual?

Read next frame

Is frame ok?

Click to draw a point

What’s next?

No

### Background Subtraction

Yes

No

Yes

Is manual?

Read next frame

Is frame ok?

Convert to greyscale and blur

No

Compute absolute difference

Find contours

Draw points

Yes

Yes

No

Is size of contour right?

Is point(x,y) darker than background?

Get coordinates of contour from moments

No



### ROI Division

### Data Analysis

**Chapter 5**

# Testing

This chapter discusses the preparation and strategy adopted for testing the program. It provides the description of Unit and Manual Testing of the software on number of videos.

## Preparation and strategy

Before testing, the suite of programs were turned into executable files. This was able to be achieved thanks to pyinstaller and turned all the .py files into versions that are able to run on any operating system without the need of Python to be installed.

## Unit Tests

Unit testing was carried out throughout the whole duration of the project. This was to ensure the methods written were performing correctly and functionality remained the same. The tests used in Unit testing were contained within the Test folder in the source code.

### FishTracker

The tests for FishTracker class were contained in test\_fishTracker.py file. Since the methods in the class rely heavily on the OpenCV, the cv2 had to be mocked. The methods tested were:

* draw\_point which had one test for checking that the function was setting the right values and calling the mocked function with correct attributes.
* get\_no\_fish\_for\_ROI which had one test for checking that the function creates the list of ROI with correct values. The values for the coordinates of borders for ROI were mocked using patch() method from mock library.

### Config

The tests for the config file were contained in test\_config.py file in the source code. The methods tested were:

* log which had two tests. First one checks that if the DEBUG flag in the file is set to True, the logs are being printed. The second test checked that if the flag is set to False, the logs are not printed. It required the system output being mocked.
* construct\_argument\_parser had one test. It checks that calling the method creates an argument parser and sets its values to ones stored in the config file as defaults.
* get\_array\_increments had three tests. First one checks the correct functionality of the function returning array increments according to the passed arguments. The second one checks that passing arguments that are too big, returns an empty array. The last test checks that the ValueError is thrown if the values passed are equal to zero. These tests contain error catching.
* is\_not\_string had one test but it calls the function multiple times. It checks that it returns a right Boolean value depending on arguments passed.
* roi\_video had two tests. First one checks that passing right arguments results the right functionality. The second test checks the error catching for AttributeError.

### MainProgram

The tests for the main file were contained in the test\_mainProgram.py in the source code. The main program was using a capture from the OpenCV, therefore two classes were created act as mocks that return the values needed for the tests. The methods tested were:

* print\_frame\_rate had two tests. First checks the error catching for TypeError and the second one uses mock for the print statement to check the correct functionality.
* calculate\_frames had two tests. They check the correct functionality and error catching with TypeError.
* calculate\_video\_duration had two tests. They check the right values being set and error catching with TypeError.
* get\_name\_from\_path had two tests. The first one checks for the correct file name to be extracted from the path passed in. The second one checks TypeError catching.

## Integration and Regression Tests

The Integration tests are verifying that the interactions of components function as designed.

The Regression tests are verifying that the unchanged module does not cause any defect.

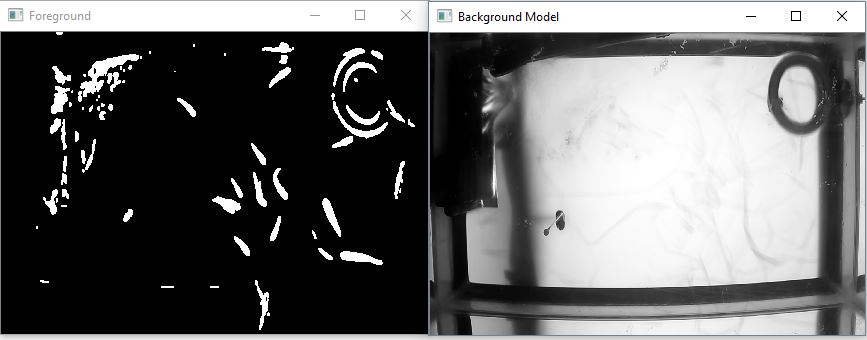
Both of the testing were carried out manually. Below all test cases used in both testing are listed and described (see Table 1). The performance of the test is described in the table whatever.

Table 1. Test cases used for testing of the project.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Type | Title | Description | Expected Result |
| 1 | File dialog | Choose ‘avi’ video | When file dialog opens, choose the video with ‘avi’ format. | File opens |
| 2 | File dialog | Choosing ‘mp4’ video | When file dialog opens, choose the video with ‘mp4’ format. | File opens |
| 3 | File dialog | Choosing unsupported format of file | When file dialog opens, try opening a different file format. | Unable to select the file. No errors. |
| 4 | File dialog | Closing the file dialog | Do not select any file and close the dialog. | Dialog reopens. No errors. |
| 5 | Argument Parser | Run with no arguments passed | Run the program in terminal with no arguments passed in. | Program runs with default settings. |
| 6 | Argument Parser | Run with arguments passed | Run the program in the terminal with arguments passed. | Program runs with new settings. |
| 7 | Argument Parser | Run with unknown arguments | Run the program in terminal and pass in arguments that are not specified. (example ‘-flag inexistent’) | Program returns an error and does not run. |
| 8 | Flags | Run with DEBUG set to True | Run the program with the DEBUG flag set to true. | Prints logging information. |
| 9 | Flags | Run with DEBUG set to False | Run the program with the DEBUG flag set to false. | No logs displayed. |
| 10 | Flags | Run with MANUAL set to True | Run the program with the MANUAL flag set to true. | Runs manual version of the application. |
| 11 | Flags | Run with MANUAL set to False | Run the program with the MANUAL flag set to false. | Runs background subtraction. |
| 12 | Manual | Run manual program | Run manual version correctly on at least 3 fish. | No errors logged. |
| 13 | Manual | Right click to undo one tracking point. | Run manual version. Click on the fish, right click to undo the selection. | No errors, the selection disappears and is not logged. |
| 14 | Automated | Run automated program with default values. | Run automated version of the program with default values and watch the selections. | Background model created, the fish are being detected, no errors logged. Opened windows are: Background, Frame, Foreground and Motion. |
| 15 | Automated | Press ‘q’ to exit. | Run the automated program. Wait for background model to be created and when the tracking starts, press ‘q’ to exit the program. | The program is exited, no errors logged. |
| 16 | Output | Coordinate output file is created. | Run both manual and automated version, check the coordinate file is created with right values. | Output file is created, no errors. |
| 17 | Output | Coordinate output not created. | Run automated version and manual and stop before it finishes. | No output file created. |
| 18 | Output | Fish count file created. | Run both manual and automated version, check the fish count file is created with right values. | Output file is created, no errors. |
| 19 | Output | Fish count file not created. | Run automated and manual version and stop before it finishes. | No output file created. |
| 20 | Data analysis | Run data analysis on output files. | Run data analysis on the output files created before. | Graphs created, no errors |

Add another table for the performance of how well they are performing with additional comments.

Add screenshots and short description of what videos I have used.



**Chapter 6**

# Results

This chapter discusses problems encountered during the time spent working on the project as well as result and conclusions drawn from carrying out the experiments.

## Variable selection

### Min-area

### Waiting-frames

This variable is used as the number of frames used to calculate the background model. The numbers of frames included: 200, 400, 600, 800, 1000 and 1300 (see Figure 12).

Since the videos were 60 frames per second, the time it took for the background model to be created were respectively: 3.33, 6.66, 9.99, 13.33, 16.66 and 21.66 seconds. There was a clear trade-off between the quality of the background model and the time it took to create it.

The quality of the background subtraction did not seem to differ between values 800 and 1000 as you can see in Figure 11. Therefore, the final value for waiting-frames variable was selected to 800.

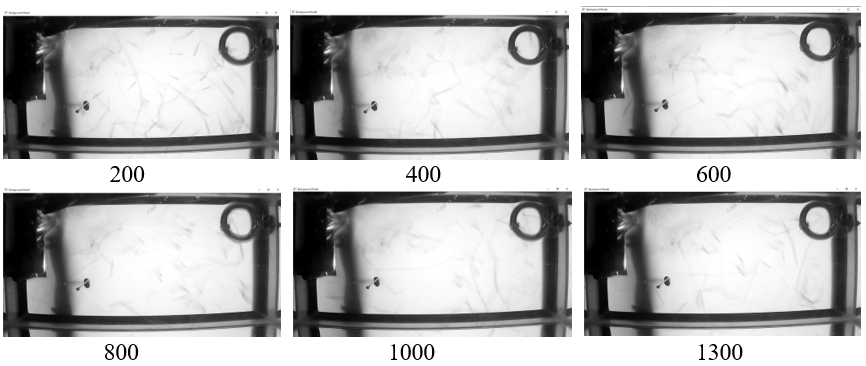


Figure 11. Screenshots of background models for different value of waiting frames.

### Threshold

Threshold is a variable in Background Subtraction that sets the value used in creating the background model.

Normally in using background subtraction, we get the model of background. This is not a case in this project. There is no clean frame with no fish in it. Therefore, Background Subtraction using moving average was used. This is used prior the motion detection to get the model in advance because it takes good few seconds to get the model that is good enough to be used.

In most cases, thresholding is not easy and a case-by-case real-time evaluation of optimal threshold may be required. Here, the value of threshold was selected based on the background model it was producing after using moving average for 800 frames where videos had about 60 frames per second. The lowest value is 0.01 and as shown in Figure 10, it clearly resulted in creating the best model for the program. The rest of the models resulted in creating many False Positives as they were detecting movement, where the fish no longer exists.

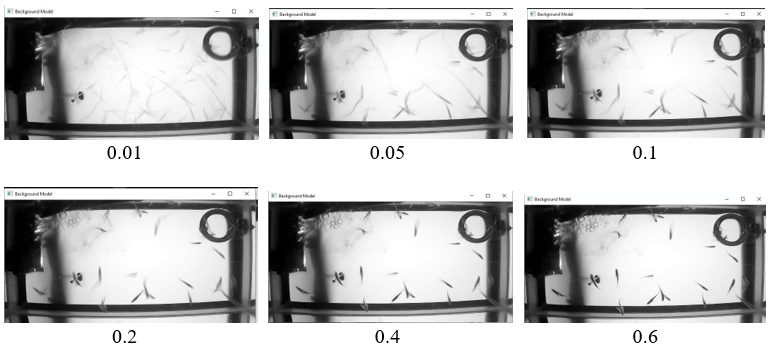


Figure 10. Screenshots of background models for different values of threshold.

## Results

### Accuracy of background subtraction

Received accuracy of around 81.3%. There were some frames where the automated algorithm seemed to be missing even up to 10 fish, although, in vast majority of time, it seemed to be performing rather well.

On average, the amount of missed fish was 3.6586 where average of fish detected was 19.65.

It is believed, the program would work much better if the experiments were redone. That is because there are many problematic areas, for example sides of the frame were dark, almost the same colour as the fish. There was also big filter and the heater in a way, that not only was similar colour as the fish, but also was blocking a lot of the view where fish could be detected. All these are clearly visible in the Figure 12.

With video tracking of the fish, overlapping is a well known problem (AS SHOWN IN PAPER). It was extremely difficult in the case of this project, as the fish were not all the same size, the tank was not shallow and the only thing available was 2D videos. To nadrobic for that, the programmer has added a functionality that counts the fish as two, if the detected area of the contour was bigger than pre-set value. For visual effect, the contour colours have changed from red to bright blue when this situation occurred, as shown in Figure 13.

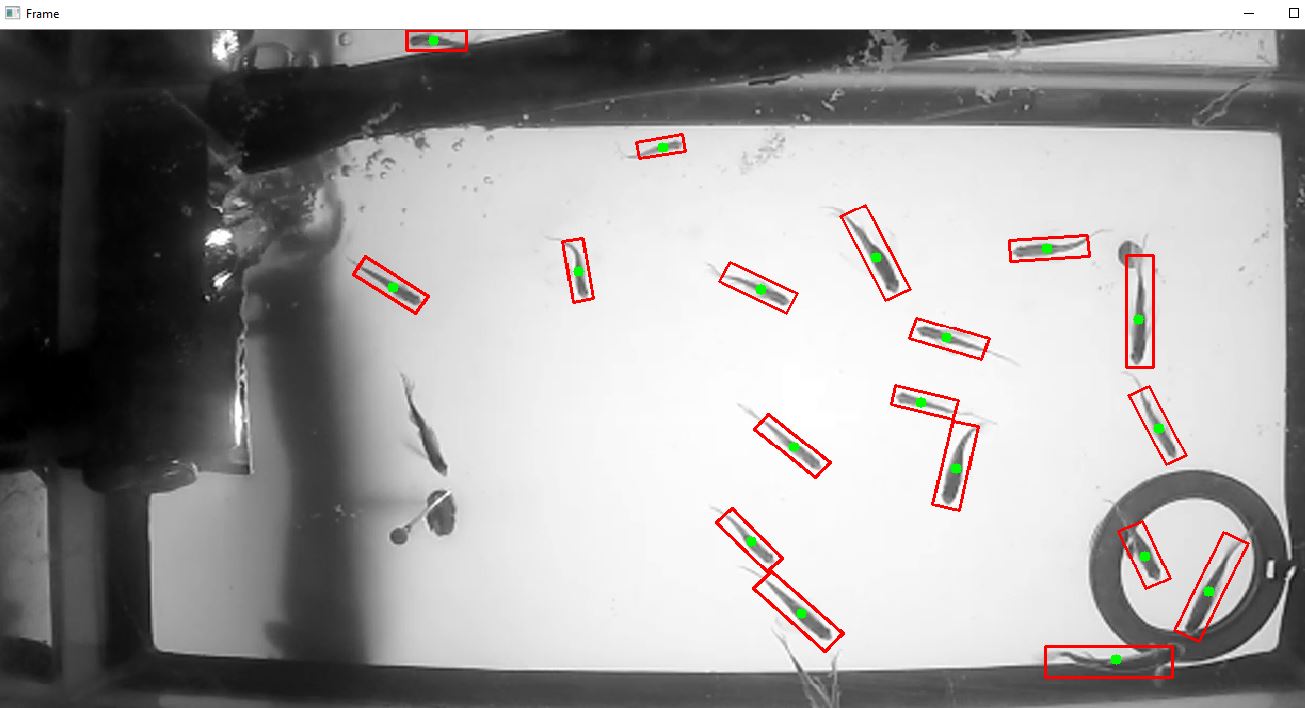


Figure 12. Screenshot of the program running with few fish not being selected due to colour changes.

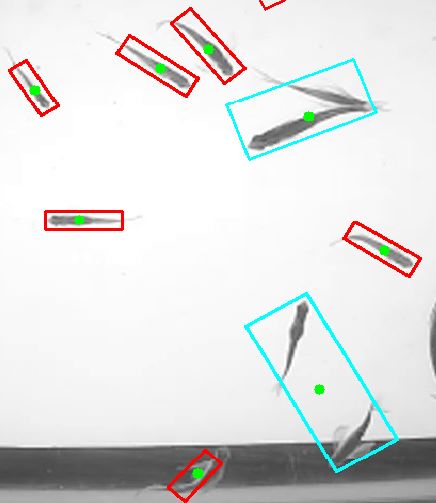


Figure 13. Screenshot of multiple fish overlapping and detected as one fish.

## Achievements

**Chapter 7**

# Critical Evaluation

This chapter reviews the aims and objectives listed and described in the first chapter and evaluate how well they were met. This gives a valuation and how the work was carried out and how the project could be improved upon.

## Aim

## Objectives

## Design

## Project Management

## Testing

## Further Development

# Summary

# Appendices

* 1. Third-Party Code, Libraries & Technologies Used

**OpenCV** (Open Source Computer Vision Library) - released under a BSD license. Version used 3.4.0. The library was used without modification. Can be downloaded from [15].

**NumPy** - the fundamental package for scientific computing with Python. NumPy is licensed under the BSD license. The library was used without modification.

**Python –** the language used for the program. Used version 2.7.0. With exception of libraries listed below, all the code was written by developer.

**Python Standard Library** – library distributed with Python. It is Open Source with PSF license. The library was used without modification.

This library includes:

* os library – this library was used to get the filename from file path acquired from the user. Used without modification.
* sys library – this library was used in the main try catch block to write the raised exception in the command line before raising it. Used without modification.
* Tkinter, tkFileDialog – the libraries were used to create Graphical User Interface to ask user to choose the video file they wish to upload to the program. It allows for the best usage for the program for users of various abilities. Used without modification.
* argparse library – this library was used as an addition to be able to change the modifiable global variables from command line. Used without modification.

**PyInstaller** – the library used to create executable Python files. Used version 3.3.1. The library was used without modification.

**GoProStudio** – free video editing software. Version 2.0.0.285 used.

PyCharm – the Integrated Development Environment used for Python language. Created by JetBrains. Used Commercial edition under Apache License.

* 1. Ethics Submission

Ethics number: 9590

Copy of ethics form below.

**AU Status**Undergraduate or PG Taught

**Your aber.ac.uk email address**  
okm@aber.ac.uk

**Full Name**Oliwia Karolina Marek

**Please enter the name of the person responsible for reviewing your assessment.**  
Reyer Zwiggelaar

**Please enter the aber.ac.uk email address of the person responsible for reviewing your assessment**rrz@aber.ac.uk

**Supervisor or Institute Director of Research Department**  
cs

**Module code (Only enter if you have been asked to do so)**  
CS39440

**Proposed Study Title**  
Video Tracking and Data Analysis Software for Analysing Group Behaviour in 2D videos

**Proposed Start Date**  
29/01/2018

**Proposed Completion Date**04/05/2018

**Are you conducting a quantitative or qualitative research project?**Mixed Methods

**Does your research require external ethical approval under the Health Research Authority?**No

**Does your research involve animals?**Yes

**Does your research involve human participants?**No

**Are you completing this form for your own research?**Yes

**Does your research involve human participants?**No

**Institute**IMPACS

**Please provide a brief summary of your project (150 word max)**The aim of the project is to analyse the behaviour of fish in videos taken over few weeks to try to find out about collective learning among the group members. During this time, the fish were tested to see if they can associate a dipole (buzzer or light source) with food. In the project I want to use video tracking to see the movement patterns of fish, the possible grouping of the school of fish, their speed to then later analyse it and recognise collective behaviour and collectve learning. The data drawn from the experiment will give a new

view of the behaviour of animals, treating them not individually, but as a group.

**Where appropriate, do you have consent for the publication, reproduction or use of any unpublished material?**Not applicable

**Will appropriate measures be put in place for the secure and confidential storage of data?**Yes

**Does the research pose more than minimal and predictable risk to the researcher?**Not applicable

**Will you be travelling, as a foreign national, in to any areas that the UK Foreign and Commonwealth Office advise against travel to?**No

**Please include any further relevant information for this section here:**

**If you are to be working alone with vulnerable people or children, you may need a DBS (CRB) check. Tick to confirm that you will ensure you comply with this requirement should you identify that you require one.**Yes

**Declaration: Please tick to confirm that you have completed this form to the best of your knowledge and that you will inform your department should the proposal significantly change.**Yes

**Please include any further relevant information for this section here:**

* 1. Code Examples

### Manual Fish Tracking



### Create Background Model



### Background Subtraction



### ROI Division



# Annotated Bibliography

1. A. Strandburg-Peshkin et al. Visual Sensory networks and effective information transfer in animal groups. In Current Biology, Volume 23, Issue 17, R709 - R711

This paper is about research about information transfer in fish schools, otherwise known as collective thinking. They introduced a small proportion of the fish already associating the light to food, to an uninformed group.

1. Carmelo J. et al. A Novel Search Algorithm based on Fish School Behavior. In International Conference on Systems, Man and Cybernetics, Recife, Brazil, 2008.

This paper describes an algorithm they used for searching in highdimensional spaces taking into account behaviours drawn from fish schools. It also describes a lot of behaviour of fish in fish schools.

1. R. Kingslake, A history of the photographic lens, San Diego, US: Academic Press, 1989-10-28.

This book describes the history of photographic lenses, including the fisheye lens. It explains what is an fisheye effect.

1. A. Roberts, B. Bill and D. Glanzman, "Learning and memory in zebrafish larvae", *Frontiers in Neural Circuits*, vol. 7, no. 1662-5110, 2013.

This article describes the behaviour and memory in zebrafish.

1. C. Peichel, "Social Behavior: How Do Fish Find Their Shoal Mate?", *Current Biology*, vol. 14, no. 13, pp. R503-R504, 2004.

This text describes the social behaviour in fish inside the fish shoal.

1. L. McCann, D. Koehn and N. Kline, "The Effects of Body Size and Body Markings on Nonpolarized Schooling Behavior of Zebra Fish (Brachydanio Rerio)", *The Journal of Psychology*, vol. 79, no. 1, pp. 71-75, 1971.

This text describes shoaling preferences in zebra fish.

1. J. Cachat et al. Three-dimensional neurophenotyping of adult zebrafish behavior. PLoS ONE, 6(3), e17597.

This article shows the use of zebrafish in neurobehavioral research. They are using video tracking and data mining techniques on videos of fish that may be useful for my work.

1. Noldus, L.P.J.J., Spink, A.J. & Tegelenbosch, R.A.J. A versatile video tracking system for automation of behavioral experiments. In Behavior Research Methods, Instruments, & Computers (2001) 33: 398. <https://doi.org/10.3758/BF03195394>

This article describes work on video tracking software for multiple animals that can later be used for analyzing the tracks and further research.

1. Andrew S. Kane, James D. Salierno, Geoffrey T. Gipson, Timothy C.A. Molteno, Colin Hunter. A video-based movement analysis system to quantify behavioral stress responses of fish. In Water Research, Volume 38, Issue 18, 2004, Pages 3993-4001, ISSN 0043-1354.

This paper describes work on video tracking application that can track up to 12 fish in big tanks and the analyzing of the work done on looking at how the fish react to stressors.

1. Fang-fang Luo, Guo-long Chen and Wen-zhong Guo. An improved "fish-search" algorithm for information retrieval. In IEEE, Wuhan, China, 2006.

It describe the use of dept first search algorithms in World Wide Web that are based on analysis of flocking behaviour of fish schools.

1. Various. Simply Psychology blog. <https://www.simplypsychology.org/pavlov.html>, 2007. Accessed February 2018.

This blog described the experiment by Ivan Pavlov about Pavlovian conditioning.

1. Kao AB, Miller N, Torney C, Hartnett A, Couzin ID. Collective Learning and Optimal Consensus Decisions in Social Animal Groups. PLoS Comput Biol 10(8): e1003762. <https://doi.org/10.1371/journal.pcbi.1003762>

This paper explores the idea that individual's learned preferences influence what others experience. Describes collective thinking taking into account size of group, environmental *cues.*

1. "idTracker: tracking individuals in a group by automatic identification of unmarked animals", *Nature Methods*, vol. 11, no. 7, pp. 743-748, 2014.

This article describes the implementation and background behind the application idTracker.

1. "Using Track3D for fish", *Noldus.com*, 2018. [Online]. Available: http://www.noldus.com/animal-behavior-research/products/track3d/track3d/fish. [Accessed: 29- Feb- 2018]..

This describes why fish have to be tracked in 3D and a tool that may be used for it.

1. "OpenCV library", *Opencv.org*, 2018. [Online]. Available: https://opencv.org/. [Accessed: 14- Apr- 2018].

The website describes OpenCV and its history.

1. "What is the C++ Programming Language? - Definition from Techopedia", *Techopedia.com*, 2018. [Online]. Available: https://www.techopedia.com/definition/26184/c-programming-language. [Accessed: 14- Apr- 2018].

This website describes C++ and its history.

1. "Introduction to OpenCV-Python Tutorials — OpenCV-Python Tutorials 1 documentation". [Online]. Available: <http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_setup/py_intro/py_intro.html#intro>.

This website describes OpenCV as well as Python.

1. "Scrum Of One: How to Bring Scrum into your One-Person Operation". [Online]. Available: <https://www.raywenderlich.com/162654/scrum-one-bring-scrum-one-person-operation>.

This website describes how Scrum can be used in one person project.

1. M. Cohn, "How I Work, and Use Scrum Personally", *Mountain Goat Software*, 2018

This blog describes how Scrum was used in one person project.

1. "Java Programming Language", *Docs.oracle.com*, 2018. [Online]. Available: https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html. [Accessed: 14- Apr- 2018].

This website describes Java Programming Languge.

1. A. Fitzgibbon and A. Buchanan, "Interactive Feature Tracking using K-D Trees and Dynamic Programming", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

This paper describes an algorithm used in ZooTracer.

1. R. Knies, "ZooTracer: Setting a Track Record", *Microsoft Research Blog*, 2018

This text describes the application of ZooTracer.

1. "Automatic tracking of multiple animals with occlusions", *YouTube*, 2018. [Online]. Available: https://www.youtube.com/watch?time\_continue=4&v=MDMRcyJKzdU. [Accessed: 14- Apr- 2018].

This video illustrates the idTracker. Used screenshot from the video as a Fig.

1. "Microsoft Research、映像内の動物を自動追跡して軌跡を表示できる「ZooTracer」", *窓の杜*, 2018. [Online]. Available: https://forest.watch.impress.co.jp/img/wf/docs/636/893/html/image1.jpg.html. [Accessed: 15- Apr- 2018].

This image shows a screenshot of a program ZooTracer.

1. "Python and OpenCV based object tracking software", *GitHub*, 2018. [Online]. Available: https://github.com/vivekhsridhar/tracktor. [Accessed: 15- Apr- 2018].

This GitHub repository holds information about an application Tracktor.

1. Neil Taylor, “MMP: Final Report and Technical Work”, 2017 (Online) Available at: <http://blackboard.aber.ac.uk/> Accessed 26th April 2017.

A document that outlines information about the marking guide for the Final Report and Technical Work. This document was referred to as Structure of the Final Report before academic year 2016-2017. This is published in the Assignments folder. If you are logged in to Blackboard, you can access the folder using <http://jump.aber.ac.uk/?sxxpt>.