ECM3401 - Individual Literature Review and Project

Detection and Quantification of Shoaling Behaviour of Zebrafish in an Aquarium

James Vipond

Candidate Number: 099562

Abstract:

Zebrafish are social animals who swim in close proximity to one another in what is known as a shoal. Individual Zebrafish that do not shoal are called solitary fish. Solitary fish's abnormal behaviour is currently under research as it is unknown what causes it to occur. This project devises a method to detect and quantify the solitary behaviour of fish in a tank. In this report computer vision techniques are explored alongside clustering algorithms to develop a working program that successfully quantifies the shoaling in videos of Zebrafish inside of aquariums. The final program utilises the You Only Look Once method to detect the Zebrafish and a combination of Kmeans clustering and Davies Bouldin index to detect the shoaling.

I certify that all material in this dissertation which is not my own work has been identified

[illegible]

1. Introduction

1.1 Summary of Project

This project designs and executes a program to identify Zebrafish and return information that describes their shoaling behaviour. The project can be used as an aid or proof of concept that the manual methods currently used by scientists could be automated to prevent this time-consuming analysis hindering further study. To achieve this outcome, neural networks have been employed to create a model that isolates Zebrafish within a fish tank, permitting further algorithmic processing of data on their clustering behaviour to deliver a quantitative output.

1.2 Zebrafish Behaviour

Zebrafish are social animals and group close together in what is known as a shoal. Shoaling behaviour has several advantages for migration, in protection against predators and in reducing stress [1] [4]. When a predator is overhead the shoal can move together away from the threat. If one fish within the shoal spots the threat and reacts the shoal will move away from the predator. This evasive response makes shoaling important for the survival of the fish. Other benefits of shoaling include foraging efficiency, territorial defence and reproductive success. It is beneficial for the fish to be part of a large shoal. Laboratory experiments show that the general preference for social fish is to join larger shoals. However, there are instances of fish not joining a shoal and being solitary [2].

Individual fish choose whether to join a shoal based upon how attractive the potential shoal is. There are different factors that a fish looks for in a shoal including the size, speed and parasite status. The fish have a preference for larger groups [2] and slower shoals, this is on account of the fact when a shoal hasn't eaten in a while they tend to move faster in the search for food, also the parasite status of members in a shoal can put other fish off from joining [3].

In this project, I develop a program to detect and quantify shoaling behaviour of fish in an aquarium. Using the program, the fish that aren't part of a shoal can be identified. This project will help research that aims to discover what causes these solitary fish to not join a shoal. The fish that do not join a shoal have a lower survival rate in the wild and choose not to mate.

A shoal will be found in the algorithm as an area where the individual fish’s coordinates cluster together. In order to identify a shoal a clustering algorithm will need to be implemented.

2. Literature Review and Specification

2.1 Computer Vision Approaches

2.1.1 Mixture of Gaussian Background Subtraction

Mixture of Gaussians background subtraction can be used for the detection of any movement in the foreground of a video. Background subtraction is when an image is computed and any pixels that differ significantly from this background image are identified [8]. The algorithm relies on a camera being in a static position and the only movement of the image being in the foreground. The method was introduced by Friedman and Russel in 1997 [17], the mixture of Gaussians consists of an expectation-maximization algorithm and a Gaussian for each: foreground objects, object shadows and the background [7].

2.1.2 Haar Cascade Classifiers

A Haar Cascade classifier is a machine learning object detection algorithm that can be used to find objects in videos. The cascade function is trained by a few hundred positive images of the object and thousands of negative images where the object doesn't appear in the image [6]. This would have worked on Zebrafish by compiling many images where Zebrafish is present and images where there are no Zebrafish present. The classifier would be able to identify whether an image has a Zebrafish in it or not.

From my research there were not any Haar cascade classifiers available online for Zebrafish and so in order to use one on my project I would have had to train one myself. In the end I decided against the use of a Haar based classifier. The reasoning behind this decision was that the fish often partially cover one another, the Haar based classifier won't be able to recognise the fish in this instance because of the fact it uses edge and line features. The edge and lines had to be clear and that isn't always possible in the set up provided.

2.1.3 Convolutional Neural Network Techniques

Neural networks are a powerful technology for classification of visual inputs. A neural network is normally described as a network composed of a large number of simple processors that are: massively interconnected, operate in parallel and learn from experience [13].

Deep learning is a subfield of machine learning that looks at algorithms inspired by the brain, called artificial neural networks. For object identification of an image, a single function that maps the pixels to the object's identity is extremely complicated. The deep learning model for object identification requires breaking the desired complicated mapping into a series of nested simple mappings, each described as a single layer in the model [14]. There are layers to the model, the visible layer is the input and contains variables that can be observed, the hidden layers come next and they extract increasingly abstract features in the image, the more layers there are, the more abstract the features they extract.

A convolutional neural network (CNN) is a deep learning algorithm. It takes an input image and assigns learnable weights and biases to various objects in the image. It can differentiate one image of an object from another. Convolutional neural networks use layers with convolving filters that are applied to local features. The strategy of convolutional network is to extract simple features at a higher resolution and convert them into more complex features at a coarser resolution. Colours, edges and gradients are all examples of simple features that the algorithm could use. [9][10].

A standard CNN cannot work with this project because it would require a huge number of regions per frame to classify and although it would be accurate it would take too long to run. R-CNN is a method that uses a fewer number of regions (2000) from an image, where the regions analysed are selected using a selective search algorithm. For each region selected the CNN can classify the region as an object. After considering this method and researching how to implement it, I discovered per frame it would still take around 49 seconds, which is too long for the video length [18].

Faster neural network-based approaches include Fast RCNN, Faster RCNN and You Only Look Once (YOLO). Fast RCNN is similar to RCNN however instead of feeding 2000 region proposals into the CNN every time the convolution operation is done once per image and a feature map is generated [19]. The Faster R-CNN uses a separate network to predict region proposals instead of the selective search algorithm [20]. This cut the time down to 0.2 seconds per image as opposed to 2.3 seconds with Fast R-CNN. With these neural network approaches considered, the YOLO method was the fastest and therefore best suited for my project as I wanted the program to work as close to real-time as possible [11].

2.2 Cluster Analysis

Once the positions of the fish in each frame of the video are found, the fish need to be sorted into which are solitary and which are part of the shoal. In order to do this, I considered 3 different methods for whether they are suitable for the project.

2.2.1 Convex Hull

Graham scan algorithm is used to create a convex hull. A convex hull is the smallest convex set that contains all the points in a set S, where S is a set of (x,y) coordinates. The complexity of this algorithm with efficient sorting is $O(n\log(n))$. I considered using this with the clusters that are found in order to envelop the fish in each cluster. This was not necessary in the final product and would just increase the run time of the program.

2.2.2 Hierarchical Clustering

Hierarchical clustering is a well-established technique in unsupervised machine learning [16]. The algorithms of hierarchical clustering all attempt to build a hierarchy of clusters. Agglomerative hierarchical clustering uses a bottom-up approach, this means each element starts in its own cluster and as you move up the hierarchy the clusters merge together in a nearest neighbour fashion. Eventually all the clusters and merges are shown in a dendrogram. The bottom of the diagram is one cluster, cutting the tree at a desired similarity level will yield appropriate number of clusters [15]. The time complexity of hierarchical clustering algorithms is O(kn^2) where k is the number of clusters to be formed and n is the number of elements to be sorted into clusters. [21]

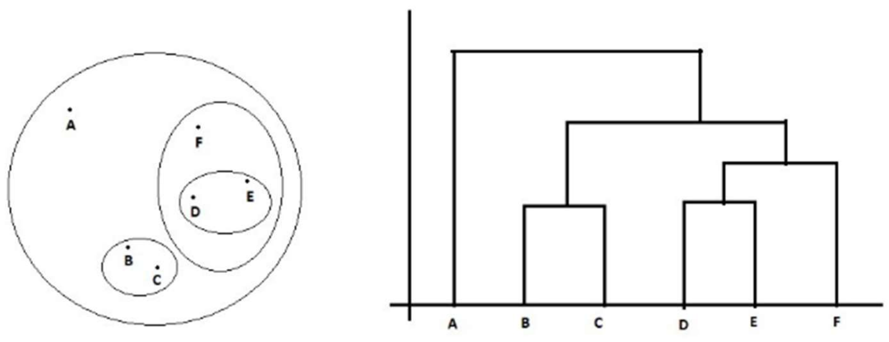


Figure 1: [22] Example of a dendrogram that agglomerative hierarchical clustering creates for points (A, B, C, D, E, F).

2.2.3 Kmeans

I decided against using hierarchical clustering because I thought that Kmeans clustering was a better fit for the project. Kmeans clustering is a method of vector quantization that can partition the Zebrafish coordinates into k clusters. For each Zebrafish it will assign it to the cluster with the nearest cluster centroid. Kmeans will minimize the variances from within the clusters, the variance is based on the squared Euclidean distances. The time complexity of the Kmeans algorithm is O(IKNT) where I is the number of iterations, K is the number of clusters, N is the number of tuples in the data set and T is the time to calculate distance between two data objects [23].

I considered using Kmeans clustering for values of k from 2 to (n-1), where n is the number of fish in the tank, this would work well for the small number of fish in the tank that I have been given, however with more fish, the algorithm would be working out too many unnecessary clusters, this would make the runtime of the program long and not very efficient.

2.3 Specification

The literature review detailed computer vision approaches to detection and quantification of shoaling of fish in an aquarium. The review explained and critically evaluated different computer vision techniques. It discussed whether or not each technique would be suitable for the project. Object detection is referred to as a method that can discover and identify the existence of objects of a certain class [5]. The object detection methods that were discussed include Haar Cascade Classifiers, Mixture of Gaussian Background Subtraction and various Convolutional Neural Network techniques.

MP4 video files will be sent from the Bio Science department at the University of Exeter and these will be used as an input for the algorithm. The algorithm will need to be able to locate the fish in the tank that are not part of a shoal. The important information to gain from the videos is the sum of the distance between the shoal to the solitary individuals. For this reason, the shoal will need to be identified as well.

Object classifiers and detectors are useful for obtaining the locations of specific objects, and therefore they can be used for this project which involves locating the Zebrafish. Certain situations may occur however in which they are not so useful, if the fish for example, are hiding behind one another then there is likely to be a decrease in the accuracy of detecting the fish. Object detectors can still be used to determine that the movement in the foreground detected is in fact a fish. The project requires by one method or another to locate the positions of the fish in the tank. Although it is not required it would be ideal for the algorithm to work as close to real time and so speed will be considered when deciding which fish locating method is best.

2.3.1 Functional Requirements

● The program must run using a video input. The video must not be corrupt, have an unobstructed view of the tank and load into Python.

● Isolate foreground objects within the video through a background removal technique.

● Distinguish the Zebrafish from other foreign objects and isolate only the Zebrafish.

● Record the coordinates of each fish.

● Classify the Zebrafish coordinates into appropriate clusters.

● Identify the solitary fish if and when they are present.

● Sum up the distances of the solitary fish from the nearest clusters.

● Produce and clearly label a graph representing the distance of the solitary fish per frame of the videos.

● The distance from the camera to the tank will be a variable that may change; therefore, the program has to account for this with the returned distance.

2.3.2 Non-Functional Requirements

● The graphs produced should ideally be in the form of a bar graph. The font must be easy to read, and the labelling must be understandable.

● Ease of use of the program is important. The program needs to be usable by people in the lab that may not be very knowledgeable about computers. A set of instructions on how to run the program should be provided.

● The speed of the program is important. Ideally it should work close to real time. The aim is that it takes less than a second per frame, so it is usable and does not take too long to run.

3. Project Development

3.1 Process

For my project to be a success I followed the following process:

- Obtain videos of the fish that I could create a detection model on.

- Identify and implement a computational method to detect the positions of the fish in the tank.
- Determine which fish if any are isolated from a shoal.
- Calculate the sum of distances between solitary fish and their nearest shoal.

One aspect that makes my project unique is that there is not a YOLO model available for Zebrafish online. So, for the detection part of this project a specific model for Zebrafish had to be trained. The model created was a convolutional neural network object detection classifier. This was a long process that required Tensorflow alongside OpenCV to train.

3.2 Obtaining the footage

At the start of the project I was in contact with the Bio-science researchers and requested some video footage of the fish. After receiving the initial video from the aquarium, it was apparent that a major issue was going to be the reflections on the inside and outside of the tank. The reflections were visible and when the fish were swimming close to the base and sides of the glass tank, when this happened two fish are clearly visible instead of one. The reflections on the outside of the glass posed an issue for the background removal techniques, the tank glass itself would be picked up as being in the foreground and the fish behind the reflections would be ignored. In order to reduce the effect of reflections, I asked for new videos with sand as a substrate on the base of the tank. The new videos had greatly reduced reflections inside the tank however the outside of the tank reflections couldn't be changed.

3.3 Mixture of Gaussians

After obtaining video of high enough quality the next step was to isolate foreground objects from the background. The first technique used to attempt this was a Mixture of Gaussian Background Subtraction. Implementing a Mixture of Gaussian as a computer vision Technique went as follows: The mixture of Gaussian background subtraction *OpenCV* function cv2.createBackgroundSubtractorMOG2()), was used to create a foreground mask. The mask would show up any movement in the tank as white and anything stationary as black. Using contouring the positions of all the fish and foreign moving objects in the tank could be found. The foreign moving objects in the tank were smaller than the fish and could be removed based on their area.

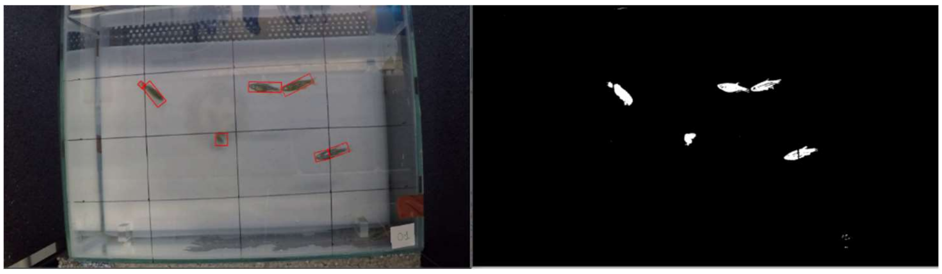


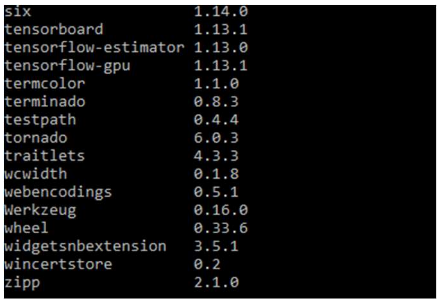
Figure 2: Left: From the input video, the Zebrafish are highlighted in red and their positions located using contouring. The fish on the far left and the far right are highlighted with two boxes, this is due to the lines that are drawn on the tank being identified as stationary in all frames.

Right: A mask is created from the mixture of Gaussian background subtraction demonstrating that this is a valid method for detection.

I decided against using this as my final model because on occasion the fish become stationary and appear invisible to the model. Some other times, the fish are covered by moving reflections outside of the tank. Furthermore, the fish that partially cover one another would be considered as a single fish. From reviewing literature, convolutional neural networks seemed a promising approach as there was potential that they could identify partially covered objects

3.4 Requirements Set up

Before training of the YOLO model could begin, I created a virtual environment for my project. The purpose of this virtual environment was to create an area specifically for this project that would not be affected by any other programs. The main thing that would affect the project is other packages installed on my personal computer. The virtual environment can be used as a way to track the packages used in my project. This feature is useful for running the program on other computers, because I can list all the packages, they need to install using Pip. In order to create the virtual environment I used python's package manager "Pip" [24] alongside the package "virtualenv". Virtualenv creates an isolated virtual python environment that when used behaves like a normal installation of python using pip to install packages required for the project. With the virtual environment created, it was now possible to begin development on the Zebrafish detection algorithm.



```
six                  1.14.0
tensorboard          1.13.1
tensorflow-estimator 1.13.0
tensorflow-gpu       1.13.1
termcolor            1.1.0
terminado            0.8.3
testpath             0.4.4
tornado              6.0.3
traitlets            4.3.3
wcwidth              0.1.8
webencodings         0.5.1
Werkzeug             0.16.0
wheel                0.33.6
widgetsnbextension   3.5.1
wincertstore         0.2
zipp                 2.1.0
```

Figure 3: An excerpt of the list of the packages used in the virtual environment which includes the package name followed by its version.

On my machine, I have a NVIDIA graphics card and so I was using CUDA (Compute Unified Device Architecture) for programming on it, alongside the library cuDNN which is used for deep neural nets. CUDA is a parallel computing platform and programming model that is used for general computing on NVIDIA graphics processing units. CUDA dramatically reduced the time it took to run the training process. The competitor for CUDA called OpenCL was considered however I decided against it due to the fact many deep learning frameworks do not support it.

The CUDA Deep Neural Network library (cuDNN) is a GPU accelerated library of primitives for deep neural networks. The library offers implementations for standard routines that I used in order to create the Convolutional Neural Network. Tensorflow Object Detection API, is an open source framework that is built on top of Tensorflow, I used this to aid in the construction, training and deployment of object detection models.
Tensorflow provides some object detection models, these are useful for inference if the object of interest is already in their data sets, however if the object is not then they can be used to initialise a model that can be used for training a specific object. The model I used was created by Microsoft called COCO (common objects in context).

3.5 Training

In order to train a model, I needed to gather suitable images of the Zebrafish. After talks with a University of Exeter Bio-science team, a tank was set up with sand on the base and 6 fish. Multiple pictures and videos were taken of the fish as they moved in the tank, in total 160 images were taken of the tank. This means that the model was trained on 960 fish. For each image the positions of every fish needed to be labelled and identified in a csv file. The images were separated into two different folders, the training images folder and about 20% the images put into a testing folder. For both folders a TFRecord file needs to be created, this has the naming information of the Zebrafish. A Labelmap was created as well so that the YOLO method can map the detected object to the object name.

The model took 4 hours to train, using a NVIDIA GTX 980 graphics card. The model created managed to locate the fish on each frame. Taking the frozen inference graph of the trained model I could run a detection algorithm on images achieving a high degree of accuracy. In order to test the accuracy of the detection algorithm I ran a video of 6 fish for 2,424 frames, for 100 percent accuracy (assuming all 6 fish are visible in each frame and none are hidden behind each other) the total number of fish detected should be 14,544. The model identified 13,441 fish within the video indicating a 92.4% accuracy. The accuracy figure further assumes no false positives, by determining the number of frames when more than 6 fish were identified, false positives were detected in 3.5% of the frames. It is worth noting that the majority of these false positives were caused by reflections of the fish rather than foreign objects; this accuracy could therefore be improved through using better camera equipment or non-reflective tanks.

After creating the YOLO model, it was apparent that the model would occasionally mistake foreign objects in the tank as being the Zebrafish. The issue of false positive results is that it can trick the model into thinking an antisocial fish is with other fish or that a foreign object is a solitary fish. Alternative detection methods, whilst more accurate also took longer to process. As the specification was looking for a model that worked in real time I decided to proceed with the YOLO model and seek a way to reduce the number and impact of false positive results. The two different methods I attempted were to train the model to recognise the false positives in the tank as their own object or to select the false positives manually.

The main false positive issue was caused by a mark on the fish tank glass that the detection algorithm mistook for a fish. After retraining the model to recognise the mark, I found it worked efficiently for that set up. This would work if all the experiments were done in the same tank. If the setup of the tank changes then it wouldn't be a suitable work around due to the fact it takes so long to train a new model.

I decided to add an optional prompt when each video is run showing the detections on the first frame of the video. Any objects that aren't the fish can then be selected and the small region of the tank that the centre of the object exists will then be ignored when the program is run. I decided to use this work around in the project because it made the program more versatile for other tanks and scenarios as the only requirement was for the camera to be stationary, which would be true in all cases.

3.6 Threshold Distance

In order to identify if a fish was solitary, the first thing I attempted was a threshold distance technique that was based on the average size of the fish in the tank. This method would check a threshold radius around each fish in the tank. If there were no fish present in the circle around the current fish, then it would be classified as solitary. This method was not used due to its complexity being O(n^2), not only was the algorithm too complex, but it needed an additional clustering algorithm implemented for the shoaling fish.

In figure 4 you can see the threshold distance is set to 2.5 times the average fish size. The large circles surrounding each fish have a radius equal to the threshold distance. The blue fish in this model would be identified as solitary and the fish in green would be in the shoal. The distance is based on the average size of the fish so that the distance scales as the camera is moved closer or further from the tank.

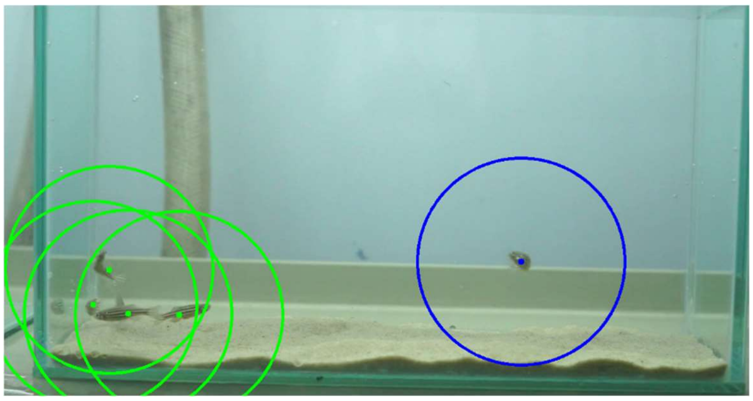


Figure 4: Example of threshold distance visualisation.

3.7 Determining the Shoals using Kmeans

In order to identify the solitary fish in the tank, the program needs to determine where the shoals are in the tank. I decided to use a "scikit-learn" module called sklearn.cluster [25] and import their method of Kmeans clustering. Scikit-learn is built on Numpy, SciPy and matplotlib and provides simple and efficient tools for predictive data analysis. This Kmeans method allows you to set useful arguments, the following arguments are important and led me to choose the scikit-learn package over other methods of working out Kmeans: the value k for the number of clusters to be produced, the method of initialisation for the clusters and the number of iterations that the Kmeans algorithm does in a single run. Sci-Learn libraries are widely used and are reliable for providing working algorithms.

Kmeans requires you to specify the number of K for which it will split the data into K clusters. On each frame of the video the program calls the algorithm 3 times. The method works by initially selecting K cluster centres and then iteratively refining them [12]. The fewer values of K evaluated the quicker the program runs. For this reason I chose only to evaluate the values of K equalling 2, 3 or 4 clusters. Observing the tank sizes that the bioscience team are working with, I concluded that once there are more than 4 clusters in the tank, there is no example of shoaling occurring in the tank. The Kmeans function returns a list of values from 0 to K. These values that are returned are the clusters the fish belong to respectively.

Once the data has been sorted into the 3 different clusterings they need to be evaluated and given a score. There is a minimum score required for a clustering so that in the instance that all the fish are in the same shoal, then it is not forced into separate clusters. These cluster options need to be ranked in order to be able to find the most suitable for the current frame. I created the function “findcluster”, which will return the clustering for the current frame.

3.8 Classifying the Solitary Fish

I created a Davies-Bouldin function that returns a score after inputting a list of cluster labels and the corresponding data. The score returned is the Davies-Bouldin index, this index is a function of the ratio of the sum of within cluster scatter to between cluster separation [26]. The Davies-Boulin index of cluster C is given by

$$\mathrm{DB}(\mathrm{C}) = \frac{1}{\mathrm{k}}\sum_{\mathrm{i}=1}^{\mathrm{k}} \overset{\max}{\mathrm{j} \leq \mathrm{k}, \mathrm{j} \neq \mathrm{i}}\, \mathrm{Dij}, \qquad \mathrm{k} = |\mathrm{C}|$$

$$\mathrm{Dij} = \frac{(\bar{\mathrm{d}}\mathrm{i}+\bar{\mathrm{d}}\mathrm{j})}{\mathrm{dij}},$$

Where Dij is the within-to-between cluster distance ratio for the ith and jth clusters. Where di (dj) is the average distance between every data point in cluster i(j) and its centroid, dij is the Euclidean distance between the centroids of the two clusters.

The lower the Davies-Bouldin index the more suitable the clustering is, I created a function for finding the most suitable clustering of the data. With all the clustering options scored each clustering is considered in numerical order. The thresholds were based on observation of each frame and the scores given. If the score for 2 clusters is below 0.4, 2 clusters get returned else if the score for 3 clusters is below 0.25 then 3 clusters are returned and finally if 4 clusters have a score below 0.15 then 4 clusters get returned. If none of the scores are below their thresholds, then the fish are all in one cluster and considered to be all shoaling. Deciding these scores was achieved through trial and improvement through observing the positioning of the fish.

In cases where there is no instance of a solitary fish it is important that we can test whether the clustering algorithm will still work. I added an argument that allows you to add a position in the tank that is treated like a fish. This position will be stationary throughout the video much like the solitary fish and will enable you to monitor how the program behaves under different scenarios. For example, with more fish in the tank will the program pick up on the artificial solitary fish, or if you position the artificial solitary fish near an existing solitary fish will the returned distance be minimised as these two fish won't be considered solitary anymore. After the clustering is decided, the solitary fish need to be identified. In this program a solitary fish is a coordinate that is in a cluster on its own. If there are other fish in the cluster then the fish are thought to be social. All the antisocial fish need to be identified because although rare, there may be instances of multiple solitary fish in the same tank.

3.9 Processing Data

The cluster variable is created which splits the fish position data into their respective clusters. The function called centroid, uses an input of the fish positions within a single cluster to locate the cluster's centroid. The centroid is the mean of all the positions in the cluster. In order to achieve this, I used the NumPy package's average function that can go through the list of coordinates and return the average X coordinate and the average Y coordinate.

In order to record the distance for each frame the solitary fish positions are compared against all the cluster centroids. The Euclidean distance that is the shortest between the solitary fish and the centroids is added to the list called "min_distances", this is a list of all the distances between solitary fish and centroids per frame. Throughout the video a list of lists is created, this includes the distances from every frame in the video. In order to get an average of this list, the total distance is divided by the length of list of the lists. This average distance value is plotted in a bar graph using matplotlib's pyplot.

4. Testing

The purpose of this project is to be able to input a video of Zebrafish and gain output information based on the positioning of the fish. Researchers at the Exeter University marine biology department are currently trying to see if there's a relationship between the conditions the fish are under and the number of antisocial solitary fish. Using this program, a graph can be produced comparing multiple videos of the fish and the average distances of the solitary fish. In figures 5, 6 and 7, scenarios from three different videos show a variety of situations that the program will have to analyse. These figures explain what is meant by solitary distance in a single frame.

The reason this program is useful is because it provides an easy way for the researchers to analyse whether or not the conditions in the tank have induced solitary behaviour. If you run the program with a control variable of an ordinary shoal in the same tank, alongside differing conditions, the graph produced will show up if there is a solitary fish. The three videos I used in testing can be observed as clear examples of distinct scenarios. In the first scenario (figure 5) the fish shoal together in one group, in the second scenario (figure 6) the fish shoal together with the exception of one fish that strays from the group and the third scenario (figure 7) the group is split up on different sides of the tank in multiple shoals.

4.1 Functional Testing

Functional testing provided instances when the program crashed, these include, use of videos that aren't of type mp4, and when the detection algorithm couldn't detect more than 3 fish in the tank. Fixes were introduced including a check that the file used ends in .mp4. For the instance of less than 4 fish crashing the program, I introduced a rule that no distance is added (it is all one shoal) when less than 3 fish are detected, these frames are rare and can occur when all the fish are shoaling behind one another, so the results aren't skewed by these anomalous and infrequent instances. It can be argued that in instances where only 3 fish are in the tank, the shoal isn't big enough for social behaviours to be observed. This is something to consider in future research.

The load time is mainly affected by the length of the video needed to be quantified. It runs on a frame by frame basis so the more frames in the video the longer it will take. There is an initial unavoidable time at the beginning for Tensorflow to load, this time is only a few seconds and should not drastically impact the user experience.

Unit testing occurred in the design process, after each function was created the unit was tested with stand in arguments to ensure the specific function returns a correct variable. All the individual components in this project at code level work just as they were designed to. The components work independently and so when each function was integrated together very few changes were required.

4.2 System Testing

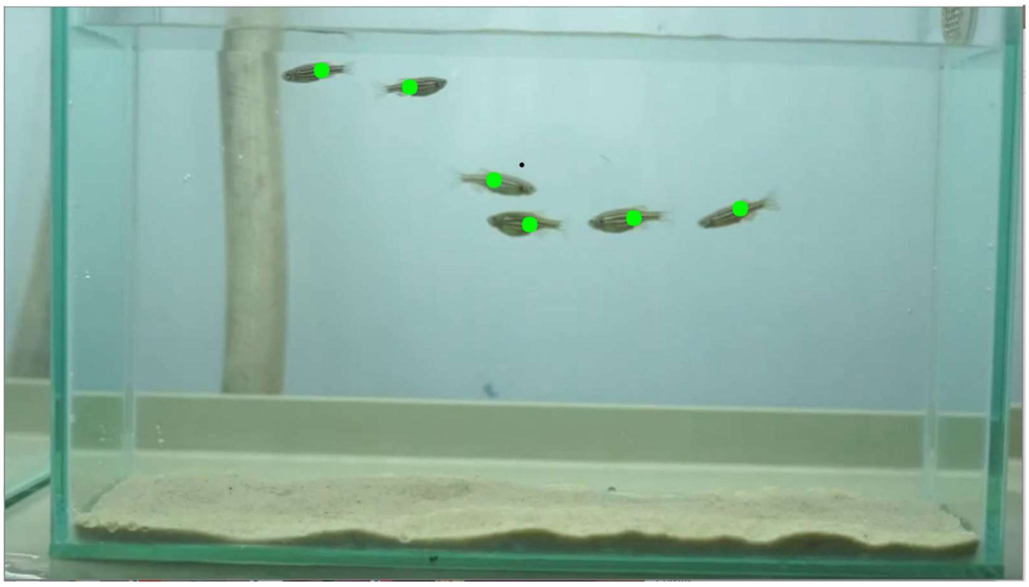


Figure 5: Example frame from the video called “shoaling”. This video shows a clear example of all the fish shoaling therefore the total distance returned from the algorithm should be low.

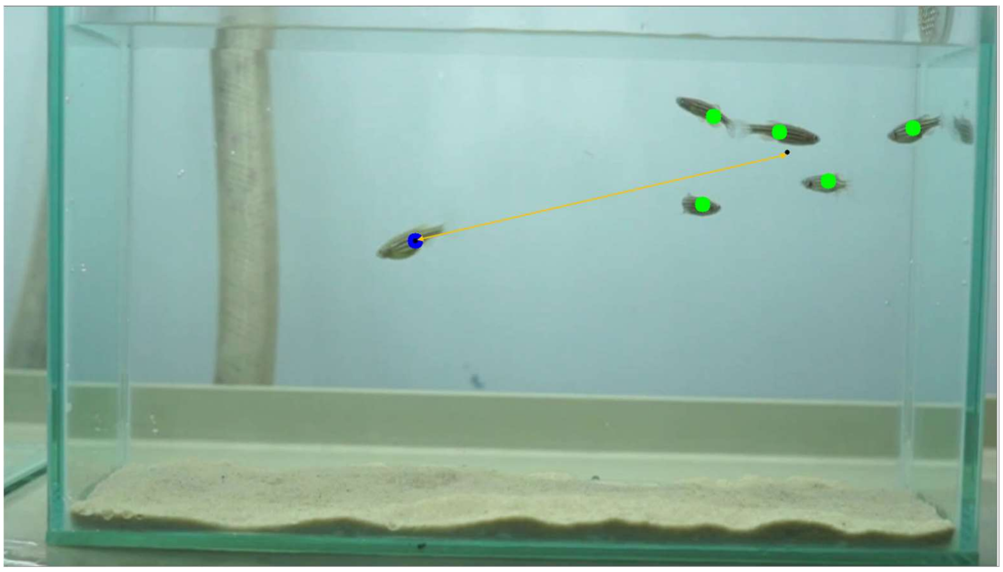


Figure 6: Example frame from the video called “antisocial”. This video shows a clear example of one fish straying from the shoal and a good example of how an antisocial Zebrafish would act. The distance from the blue fish to the centroid of the green shoal will be added to the total distance

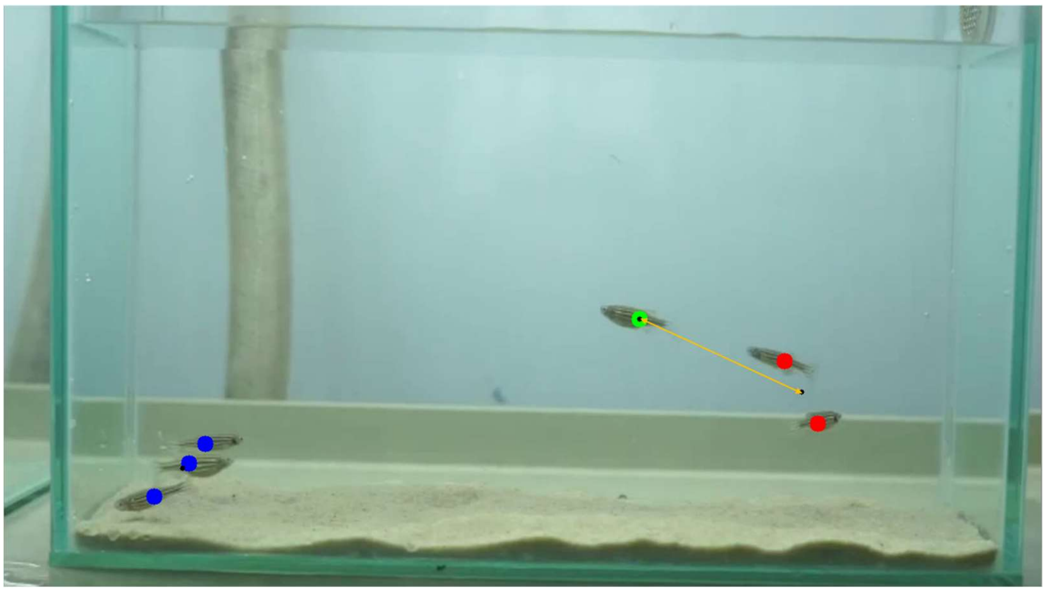


Figure 7: Example frame from the video called “split_shoaling”. In this video, the fish are in groups but are separated from one another, this leads to them occasionally registering as antisocial, but the distances recorded in comparison to the “antisocial” video are small. The green fish here is identified as antisocial as it is not part of either group, the distance recorded will be the distance between the green fish and the centroid of the red fish, this is because the red fish centroid is closer to green than the blue fish centroid.

Out of the three videos we would expect the results to return “shoaling” as having the smallest distance because the fish stuck together in a group throughout the video. The next smallest distance is expected to be “split_shoaling”, because the fish are split into two groups and are more separated than “shoaling”. “antisocial” is expected to return the greatest distance value as it shows a clear example of how a solitary fish behaves, there is a shoal on one side of the tank and one fish that strays from the shoal and moves to the opposite side of the tank.

System Testing Results

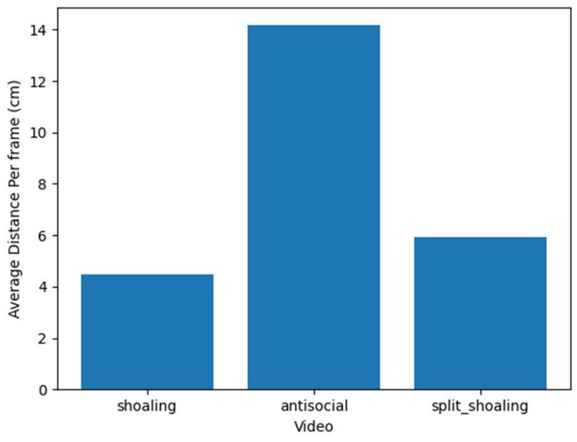


Figure 8: Output of the program for running the three video examples. The results show the greatest distance in for “antisocial”, followed by “split_shoaling” this was the expected results for the 3 example videos.

Based on the requirements of the original problem, the fitness for purpose is best demonstrated in the results. The videos chosen, were chosen specifically as examples with predictable results. After running the program, the actual results show the same trend that a successful analysis would produce. The examples demonstrate the capability of the program and shows that the Davies Bouldin threshold value is set at a good balance to yield good results.

During this project it was observed that when detecting more than 4 clusters, the fish were not demonstrating shoaling behaviour. Without the shoal there is no evidence of solitary fish, and so the decision to limit the detected clusters to 4 not only stops the program from returning a distance for a video without shoals but it also reduces the time spent per frame clustering the fish. The original requirements didn't state that the clusters should have a limit of 4 however from the observations making the project it was clear it needed to be done.

4.3 Load Testing

To test and see if the program could work with more fish, a video was used of a tank containing 13 fish. The program detected the fish when they were separated, and although it detected most fish, on occasions, when the fish were covering up one another, some fish failed to be detected. The video only demonstrated shoaling throughout, the program worked as expected returning a small average distance, however in order to see if a solitary fish would affect the results, I ran the results with the original video followed by a video with a fish at (30,690). The detection was successful, and most fish were detected. In the video provided it is clear that all fish are shoaling and no fish are solitary.

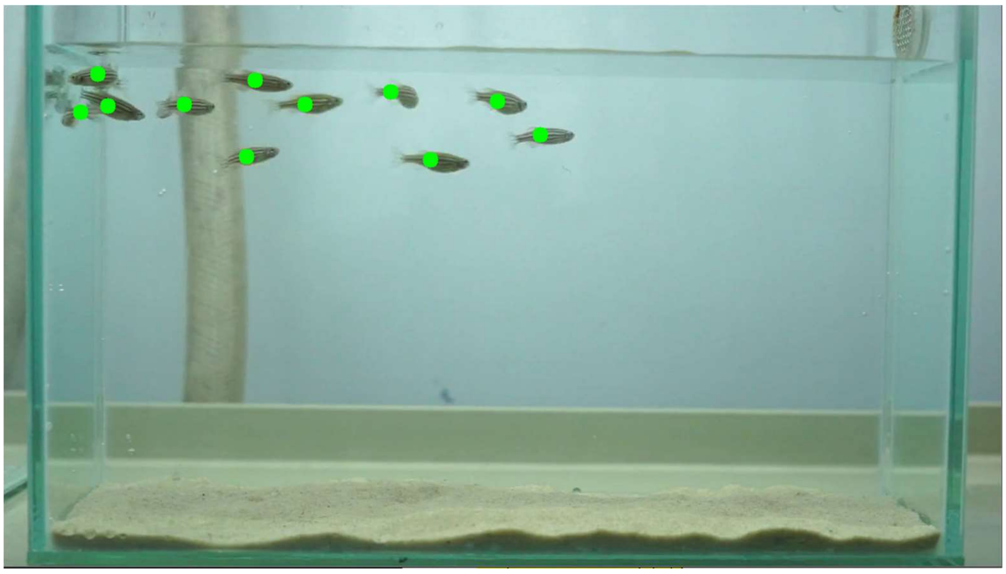


Figure 9: An example of a tank with 13 fish, In this frame most of the fish are detected, with the exception of a couple that are hidden behind one another. These are all showing shoaling behaviour.

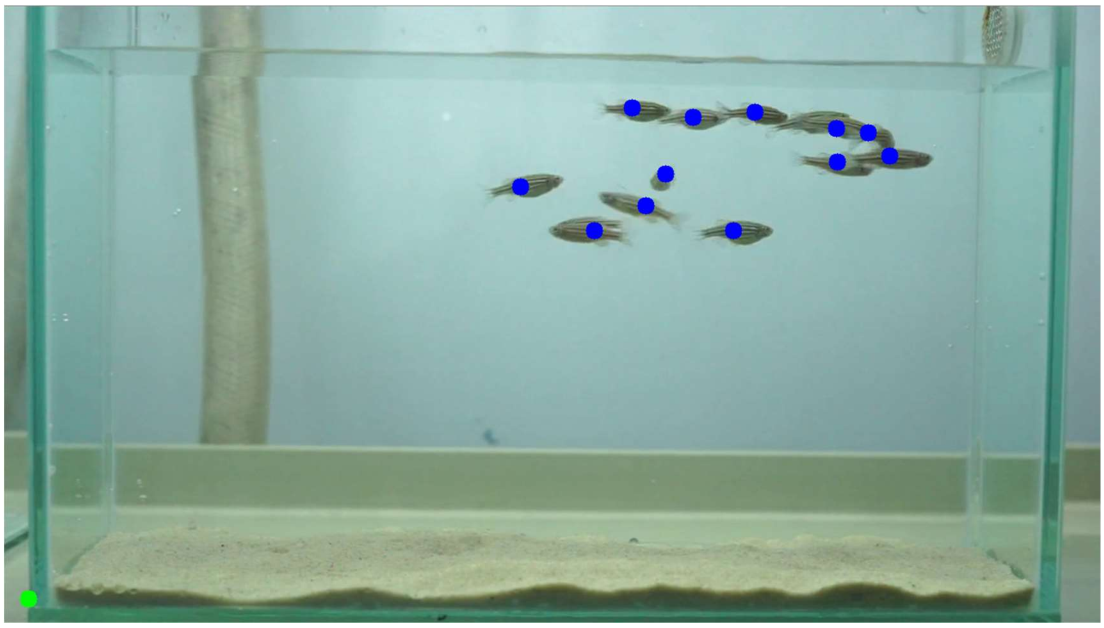


Figure 10: Example frame of 13 fish but with an artificial solitary fish added at the position (30,690). The green dot in the bottom left acts as the artificial fish and the blue fish are in a shoal together. The distance recorded will be the distance from the green dot to the centroid of the fish shoal.

Load Testing Results

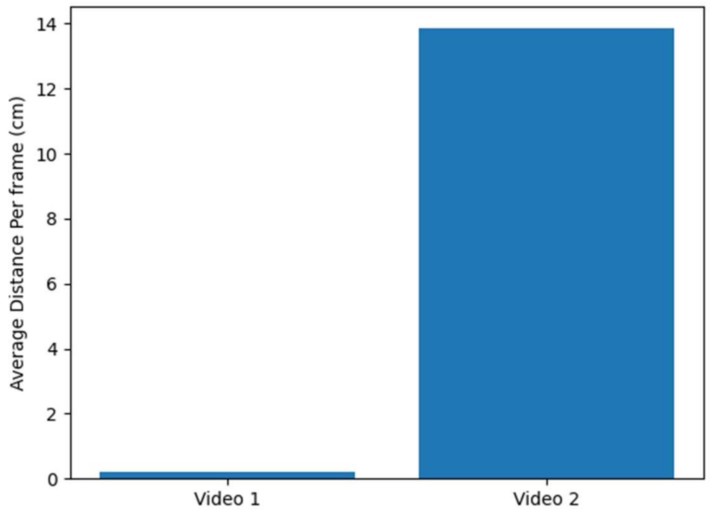


Figure 11: The two examples in figure 9 and 10 output different distances. Video 2 has a solitary fish (figure 10) and the distance returned is over 13 cm, whereas the distance of video 1 (figure 9) is less than 1 cm. This demonstrates that the program can identify solitary fish and produce accurate results with more fish in the tank.

In order to load test, the performance of the program, I used the video provided by the bioscience team with the most fish in the tank. Testing whether the program can detect solitary fish with more fish in the tank was a success based off the results with an artificial solitary fish. This is useful for the bioscience team as they are not limited to using just a small sample of fish.

5. Evaluation

This project has provided me with an insight into how neural networks are trained and used. The bioscience team at the University of Exeter are wanting to research what causes Zebrafish to become antisocial and solitary. The hypothesis they need to investigate is that the fish can become solitary due to the conditions they are under, these may be the salinity of the water, the temperature or the amount of microplastic in the water. This program created is a useful aid as it can be used for the Zebrafish research to save time of the researcher who would have to do this task manually. It is also useful as a proof of concept as something that can be developed to be more accurate and reliable. Computer vision is capable of aiding bioscience and this project demonstrates how we can speed up the time consuming low skilled tasks, giving more time for effective study into the results. In this aspect the program could also provide a financial benefit.

5.1 Customer Feedback

After sharing the project results with a PhD student working with the Zebrafish, the feedback was positive, the student responded saying that the program performed in accordance of the initially proposed project plan, the student also stated that the program would be useful to aid the research. To expand the use of the current program, the student would like functions to be added that could record: the number of shoals, the mean group density and the number of solitary fish. I think that in order to create a program that can have these features, the current model and clustering techniques applied in this project could be used.

5.2 Development Evaluation

It is worth noting that during the development of the project I decided to steer away from the specification of using a background subtraction method. Using a YOLO model did not require the use of background subtraction; however, I did experiment with using background subtraction initially. I think that in order for background subtraction to be a more viable option for this project, the aquarium set up would need to have no reflections on the glass.

This project differs from other work that I have seen online in that I have had to train a new model for detecting Zebrafish. The use of this project is unique as it is specifically designed for ongoing research. Currently, there is not a detection and clustering program available for this research that would be fit for purpose. The use of clustering to find the solitary fish was one of the first ideas that I decided to use, the Kmeans method of clustering isn't unique on its own however its application to this problem is.

From this project I have learnt how convolutional neural networks work from a technical and practical perspective. There are different methods of using neural networks that have their individual advantages and disadvantages, where one may have a greater accuracy of finding an image, the runtime of the method is unusable for videos. For this project I successfully found a method that balanced accuracy and run time.

I had to use the existing model COCO to train a new model for Zebrafish as none were available online for use. Another example of applying of existing methods to a new problem includes finding the best fit clustering for the data on each frame. I used Davies Bouldin method to filter what clustering was best fit for the individual frame and then validated the clusters found using Kmeans method, as to whether or not there is a solitary fish present.

I think that a skill I have shown from doing this project is the ability to a new area of computer science and apply it to a novel problem. Before undergoing this project, I didn't know anything about computer vision. After research and independent learning, I have managed to produce a program using computer vision techniques that works. I have learnt in this project and gained an understanding of how clustering methods differ in complexity and the importance of practicing efficiency in your code.

I have learnt that a program isn't just about the returned result, other factors that aren't solely functional such as the runtime are important so that users are willing to use your project. If the program produced takes hours to run on a short video for example, the researchers would choose to use the manual methods instead. Therefore in order to make the program usable it is important to keep in mind these factors in the development of a program.

I have shown how I can collaborate with customers that aren't part of the computer science field of study, and work with them to produce a product that is fit for their needs. Throughout this project I have had meetings and emails with computer scientists and university staff in order to make sure my program meets their needs. I have learnt that communication throughout the development process is helpful for keeping on track and getting feedback to improve upon.

5.3 Critical Assessment

The primary goal of the project was to develop a program to detect and quantify shoaling behaviour of fish in an aquarium. Based on this goal this project is a success, from the examples given you can see that the tanks with antisocial fish are distinguishable from those that don't have solitary fish. My approach to this project required planning and research into which methods would be best to use. I think that the methods used in the final project are suitable for the footage obtained and the research environment that's used in University of Exeter Labs.

5.3.1 Improvements and Further work

The program is only trained for Zebrafish so it would need a new model for other fish species. The program likely wouldn't be suitable in all set ups. In the University of Exeter's set up, the reflections on the base of the glass tank are removed due to the sand. Furthermore the set up's combination of plain background and static camera allows the detection algorithm to work to a high degree accuracy.

Much of the time spent on this project was attempting to get the detection of the fish working. Initially I thought that background subtraction followed by a Haar cascade classifier would be most suitable, however after poor results from the mixture of Gaussian background subtraction and further research into Haar like features, I took the decision to investigate a neural network approach. Creating the model for the Zebrafish was a steep learning curve and required me to devote a lot of time into understanding what each training stage was doing. Namely any time an error occurred in setting up the environment or training the model, finding a work around was very time consuming. In hindsight, I would initially create a virtual environment and plan which versions of each installation I would use before executing the training. This would have prevented the many bug fixes that needed to be done when creating the inference graph.

The created model worked well, however, there is room for further development. If I were to continue the development of the project, I would improve the training of the model by using a different tank setups. The tank example used for training had a plain background, use of a background with more noise would have led to a more accurate inference graph. Despite this the work around strategy I employed for the false positive results will only affect the final results in very rare cases. For this reason I would deem the detection algorithm part of the project an overall success. The rare instance would easily be avoided in the labs by cleaning the glass or using a plain background like in the training videos. The model created had an accuracy of 92.4%. Kulkarni's work in "Deep Learning Based Object Detection Using You Only Look Once" achieved a mean average precision of 91.28% for their COCO dataset [27]. The difference between my accuracy and Kulkarni's could be explained by the false positive results caused by the reflections on the glass. Otherwise I can conclude the model works to a suitable degree of accuracy.

The clustering algorithm used is deemed a success. On reflection I think that the decision to cap the number of clusterings worked out per frame to 4 was a good decision in order to reduce runtime. Time spent implementing different methods of identifying the solitary fish was significant in developing this project. The Davies-Bouldin function created worked efficiently, it works as expected by returning the index value. The threshold values of Davies-Bouldin index were adjusted based on observation. The values had to be low enough so that when the fish were all shoaling, they weren't forced into separate clusters. From the sample videos I received I think the values chosen demonstrate this well.

5.4 Conclusion

This project was chosen because computer vision research and development fascinate me. I wanted to make a program that used existing techniques as a solution to a new and unique problem. This problem was challenging because I had to explore multiple different computer vision techniques to find the most effective solution to the problem.

This project devised a method to detect and quantify shoaling behaviour of Zebrafish in an aquarium. I used a combination of convolutional neural networks, Kmeans clustering technique with Davies Bouldin index in order to create a program that automatically detects the solitary fish and the distance away from the nearest shoals.

The overall aim of the project has been achieved and the results allow the researchers to quantify any solitary fish behaviour that is occurring in the tank. I feel that this program could be used to supplement the research, if many videos need to be analysed. This program can be used to identify the tanks with solitary fish and then those tanks can be further observed by the researchers. This saves their time in finding out which tank set ups are of interest.

In conclusion the detection model created does work for the purpose of this project. Further research in this topic can lead on to tracking and recording the spatiotemporal distribution of the Zebrafish. In further development and future research, I think that an improved model would be required in order to make the analysis more objective. The methods used in this project can be replicated with minor alterations in order to produce the foundational functions for future projects. To conclude I am proud of the results my program has produced and I am optimistic to see how it facilitates the interdisciplinary study of the social behaviours of Zebrafish.

References

[1] M. Litvak, "Response of shoaling fish to the threat of aerial predation", Environmental Biology of Fishes, vol. 36, no. 2, pp. 183-192, 1993. Available: 10.1007/bf00002798.

[2] V. Pritchard, J. Lawrence, R. Butlin and J. Krause, "Shoal choice in zebrafish, Danio rerio: the influence of shoal size and activity", Animal Behaviour, vol. 62, no. 6, pp. 1085-1088, 2001. Available: 10.1006/anbe.2001.1858.

[3] I. Barber and H. Wright, "How strong are familiarity preferences in shoaling fish?", Animal Behaviour, vol. 61, no. 5, pp. 975-979, 2001. Available: 10.1006/anbe.2000.1665.

[4] D. Hoare, "Body size and shoaling in fish", Journal of Fish Biology, vol. 57, no. 6, pp. 1351-1366, 2000. Available: 10.1006/jfbi.2000.1446.

[5] S. Soo, "Object detection using Haar-cascade Classifier", Seminar, University of Tartu, 2014. Available:
https://pdfs.semanticscholar.org/0f1e/866c3acb8a10f96b432e86f8a61be5eb6799.pdf?_ga=2.17384057.1145791255.1587997312-1284592716.1585920417 [Accessed 27th April 2020]

[6] Y. Wang, "An Analysis of the Viola-Jones Face Detection Algorithm", Image Processing On Line, vol. 4, pp. 128-148, 2014. Available: 10.5201/ipol.2014.104.

[7] T. Bouwmans, F. El Baf and B. Vachon, "Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey", Recent Patents on Computer Science, vol. 1, no. 3, pp. 219-237, 2008. Available: 10.2174/2213275910801030219.

[8] N. Friedman and S. Russell, "Image Segmentation in Video Sequences: A Probabilistic Approach", Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, 1997.

[9] D. Specht, "The general regression neural network—Rediscovered", Neural Networks, vol. 6, no. 7, pp. 1033-1034, 1993. Available: 10.1016/s0893-6080(09)80013-0.

[10] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way", Medium, 2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutionalneural-networks-the-eli5-way-3bd2b1164a53 [Accessed 27th April 2020].

[11] J. Redmon, S. Divvala, R. Girshick and A. Farhad, "You Only Look Once: Unified, Real-Time Object Detection", 2016. Available: 10.1109/CVPR.2016.91

[12] K. Wagstaff, C. Cardie, S. Rogers and S. Schödl, "Constrained Kmeans Clustering with Background Knowledge", '01 Proceedings of the Eighteenth International Conference on Machine Learning, pp. pp.577-584, 2001.

[13] A. Jain, Jianchang Mao and K. Mohiuddin, "Artificial neural networks: a tutorial", *Computer*, vol. 29, no. 3, pp. 31-44, 1996. Available: 10.1109/2.485891.

[14] I. Goodfellow, Y. Bengio and A. Courville, Deep learning. MIT Press, 2016, pp. 5-7.

[15] O. Maimon and L. Rokach, Data mining and knowledge discovery handbook. New York: Springer, 2010, pp. 278-279.

[16] D. Mullner, "fastcluster: Fast Hierarchical, Agglomerative Clustering Routines for R and Python", *Journal of Statistical Software*, vol. 53, no. 9, 2013.

[17] N. Friedman and S. Russell, "Image segmentation in video sequences: A probabilistic approach", 1997.

[18] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[19] R. Girshick, "Fast R-CNN," in IEEE International Conference on Computer Vision (ICCV), 2015.

[20] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149, 2017. Available: 10.1109/tpami.2016.2577031.

[21] S. Bhattacharyya, U. Maulik and P. Dutta, *Quantum Inspired Computational Intelligence*, 1st ed. Morgan Kauf, 2016, pp. 361-389.

[22] C. Patlolla, "Understanding the concept of Hierarchical clustering Technique", *Medium*, 2020. [Online]. Available: https://towardsdatascience.com/understanding-the-concept-of-hierarchical-clustering-technique-c6e8243758ec [Accessed 27th April 2020].

[23] V. Patel and R. Mehta, "Impact of Outlier Removal and Normalization Approach in Modified Kmeans Clustering Algorithm", International Journal of Computer Science Issues, vol. 8, no. 5, 2011.

[24] Pip package installer, 2019. URL https://pypi.org/project/pip/ [Accessed 27th April 2020].

[25] sklearn Kmeans, *Scikit-learn.org, 2020. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.Kmeans.html* [Accessed 27th April 2020].

[26] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650-1654, 2002. Available: 10.1109/tpami.2002.1114856.

[27] S. Kulkarni, and G. P, "Deep Learning Based Object Detection Using You Only Look Once", *International Journal of Research in Advent Technology*, vol. 7, no. 4, pp. 9-12, 2019. Available: 10.32622/ijrat.74201902.