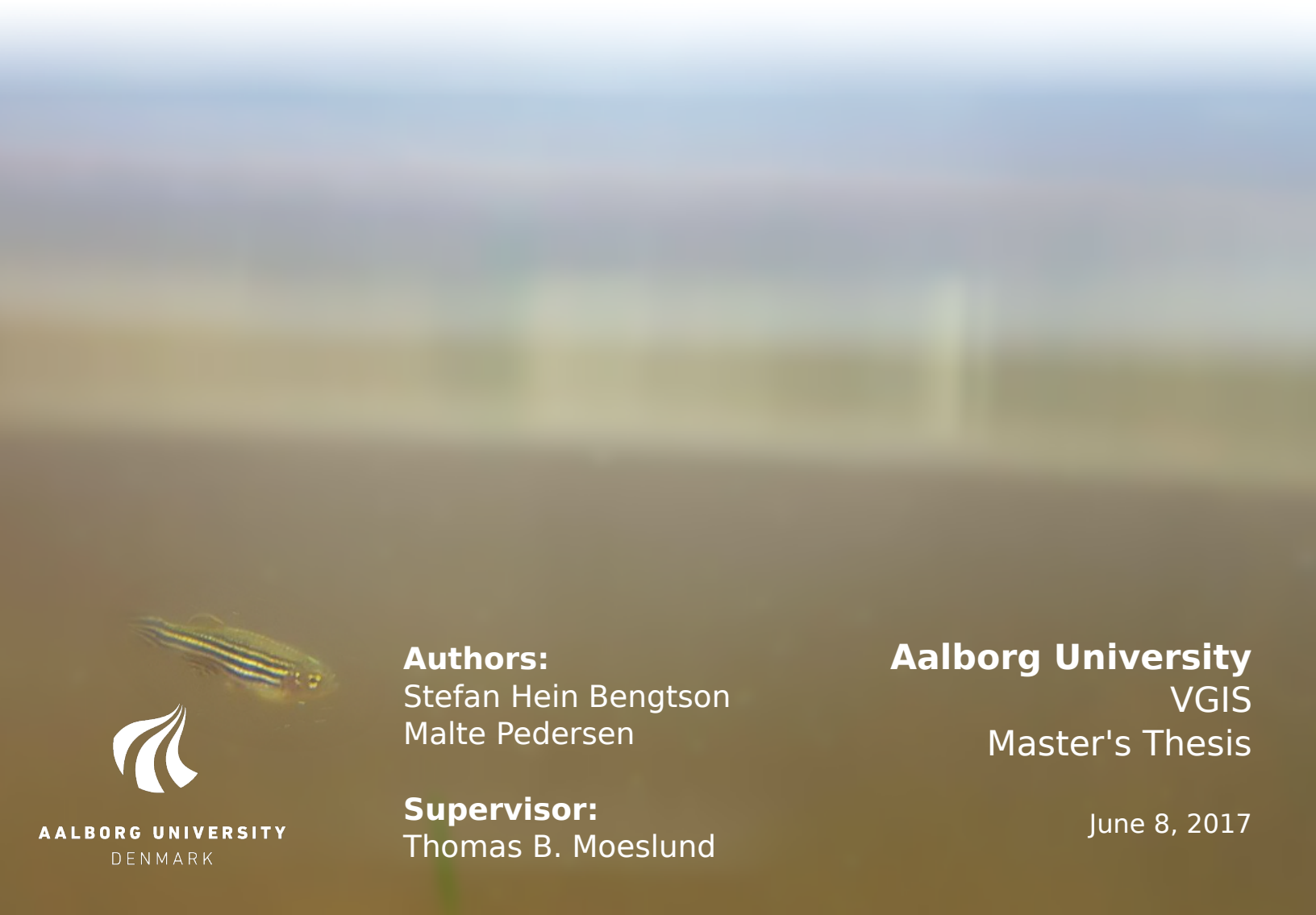
Tracking Zebrafish in 3D using Stereo Vision



Authors:
Stefan Hein Bengtson
Malte Pedersen

Supervisor:
Thomas B. Moeslund

Aalborg University
VGIS
Master's Thesis

June 8, 2017

AALBORG UNIVERSITY
DENMARK

AALBORG UNIVERSITY

Title:
Tracking Zebrafish in 3D Using Stereo Vision

Programme:
Vision, Graphics, and Interactive Systems

Project period:
1/2-2017 to 8/6-2017

Project group:
17gr1040

Participants:
Stefan Hein Bengtson
Malte Pedersen

Supervisor:
Thomas Moeslund

Number of pages:
100

The use of aquatic animals for research purposes has been increasingly popular in recent years and this is especially true for the zebrafish (*Danio rerio*), which is being used for genetic, environmental and pharmaceutical studies, among others. The motion trajectories of the fish can be an important metric and the studies are therefore often supported by a computer vision system, which automatically tracks the fish. However, the tracking is often only conducted on a single fish or in two dimensions, as most of this type of tracking software has been developed for mice, rats or similar. The focus of this thesis is hence to investigate the possibilities of tracking zebrafish in three dimensions.

The proposed system is made of off-the-shelf hardware and consists of a stereo-vision setup with two GoPro HERO5 cameras. The zebrafish are detected in each camera using the SURF keypoint extractor after which the 3D positions are estimated. The 3D reconstruction is based on ray-tracing in combination with Snell's law, in order to account for refraction. By using a Kalman filter and Munkres algorithm it is possible to assemble tracklets with an average length of 150 frames.

A final part of the system, responsible for linking the tracklets, has not been implemented in this iteration of the system.

Preface

This report is submitted as the final project at the Master's Programme in Vision, Graphics, and Interactive Systems at Aalborg University. The focus of the thesis is to design and implement a computer vision system capable of tracking fish in three dimensions for use in controlled test environments.

The project has been created in cooperation with Professor Niels Madsen from the Department of Chemistry and Bioscience at Aalborg University, which has been helpful in providing input and supplying materials for the test setup.

Reading Guide

Figures and tables are numbered sequentially within each chapter. Citations are written as [x] where x denotes the reference number used in the bibliography, which adheres to the IEEE citation standard.
The software developed during the project is written in Python 3.5 and is uploaded along with the thesis.

The first part of the thesis contains a review of different methods for extracting 3D information of underwater objects and how this process is affected by refraction. This is followed by a description of the gathered and annotated data used in the development of the system.

The main portion of the thesis consists of various design considerations for the modules of the system. The first module is concerned with detection of the fish in 2D, while the second module focuses on how to extract 3D information based on the detections. The last module is responsible for creating coherent tracks for each fish. Each of the modules are tested individually.

The thesis is concluded with a discussion of the system in its current state based on the findings from the module evaluations. Further improvements of the system are discussed as well.

Stefan Hein Bengtson

Malte Pedersen

Contents

1 Introduction

Microplastic, medical waste and other types of polution have been poured into the oceans for decades, but it is not until recently that it has gained attention of the people and media. Due to the lack of attention, the amount of research that has been put in motion to investigate how it affects the fish and marine environment has been scarce until a few years ago [1]. Microplastic is defined as pieces of plastic less than 5 millimeters long, which means that it can easily be consumed by marine animals and as the amount of plastic that is produced each year keeps increasing, this is an ungoing problem [2].

One of the methods to examine how microplastic and other types of polution affect marine life is to observe the behavioural patterns of fish in healthy environments and compare this to behavioural patterns observed in poluted environments. As this is a difficult, subjective and tedious task to do manually, software based solutions, such as automatic tracking of fish in video sequences, can be a helpful tool [3].

The behavioural patterns and locomotion of fish are also used in other types of research, such as pharmacology and genetics. Fish, and especially zebrafish (*Danio rerio*), have gained an increase in popularity and are used to test medicine and drugs as they are cheap, easy maintained, have a high fecundity and resemble humans in many ways [4] [5]. An example is that the social behaviour of fish has shown to be affected by certain pharmaceuticals that are flushed through the drains in large amounts, making the fish less social and more bold [6]. This could potentially alter the ecosystems that receive waste water from pharmaceutical facilities.

Software solutions have been used to map behavioural patterns of test animals for several years, however, a large part of the solutions have been developed for land living animals, such as mice [7] [8], or fish in shallow water [9]. The mapping of the behavioural pattern in such solutions has been confined to a single plane, however, this may be insufficient when looking at fish as behavioural deviance may be observed in all directions [10].

Beside looking at individual fish and abnormalities in their behavioural patterns, observing whole schools of fish may give indications of the health and condition of the respective groups and how they react on certain stress factors, such as predators, noise from ships or changes in temperature. Furthermore, this could potentially assist the fishing industry in the development of fishing methods that could minimize the amount of bycatch, which, in the U.S., is estimated to be around 20% of the overall catch per year [11].

The main scope of this thesis will be the development of a solution capable of observing and mapping the motion trajectories of multiple fish in three dimensions.

1.1 Tracking of Fish in 3D

As described in the introduction, the motion trajectory of fish in 3D can be used for various research scenarios and applications. However, it is not a trivial task to detect and track fish and the solution may vary to a large degree based on the application. Due to this, the scope of this project is limited to take basis in the following usage-scenario.

Usage-Scenario

The system should be a useful tool for research applications that focus on the behavioral patterns of one or multiple fish kept in a controlled environment. It should be capable of

measuring the 3D motion trajectories of the fish with no, or minimal, interference from the user and it should be easy-to-use for people without any specific technical skills.

Detection and tracking of the fish should be based on video sequences captured by one or several off-the-shelf cameras and it should be possible to use off-the-shelf light to illuminate the scene. The video sequences should be recorded by the user and submitted to the system, which in return should output a file consisting of information about the motion trajectories of every unique fish in the given scene.

1.2 Properties and Limitations

In order to keep the development of the system in a containable scope, a range of properties and limitations are proposed. They are based on the usage-scenario presented in Section *1.1 Tracking of Fish in 3D*.

Properties

1. **No real time** The system is not expected to process data in real time.
2. **Known amount of fish** The number of fish placed in the aquarium is known in advance.
3. **Off-the-shelf hardware** In order to make the system cheap and easy accessible.

Limitations

1. **Single aquarium** The system will be developed and tested on a single aquarium.
2. **One type of fish** The system will be developed to detect and recognize one type of fish.
3. **No interference** The system is not allowed to interfere with the fishes.
4. **No graphical user interface** Low priority on usability.
5. **Temporally aligned recordings** The system will not be able to synchronize recordings that are not temporally aligned.

The technical details needed in order to develop a system capable of tracking multiple unique fish in 3D will be described in the following chapter.

2 Technical Analysis

The purpose of this chapter is to analyze some of the areas of importance when developing a 3D tracking system. The first part will hence mainly be focused on how to extract 3D information and how this procedure may be affected by having to capture underwater objects. The last part will on the other hand concern different aspect of fish, such as their movement patterns.

2.1 3D Measuring Methods

There exist several types of vision systems that are capable of measuring depth. Three popular and widely used methods will be presented in this section along with a brief description and an overview of their capabilities. The sensors will be compared in regard to frames per second, depth precision, range, price and whether they are measuring depth actively or passively. Passive sensors use the light present in the scene while active sensors utilize specific sources of light to measure the depth. The comparison is made on basis of [12] [13] [14] and [15] and it is used as an overview on a general level as most of the specifications of the depth measuring systems vary to a large degree based on price, brand and the setup in which the measurements are gathered.

2.1.1 Time-of-Flight Depth Camera

Time-of-Flight depth cameras (ToF) measure depth based on the travel distance of pulses of light emitted from a known energy source [16]. The camera and light emitter are often placed as close to each other as possible, which means that the distance the light needs to travel, before the camera receives the reflection from an object, is twice the distance between the given object and the camera. The distance, d, can be described as

$$d = \frac{C \cdot t}{2k}, \tag{2.1}$$

where t is the time it takes from the light to leave the light source and be received by the camera sensor, C is the speed of light and k is the index of refraction of a given medium, such as water or air. An illustration of the reflection can be seen in Figure 2.1.

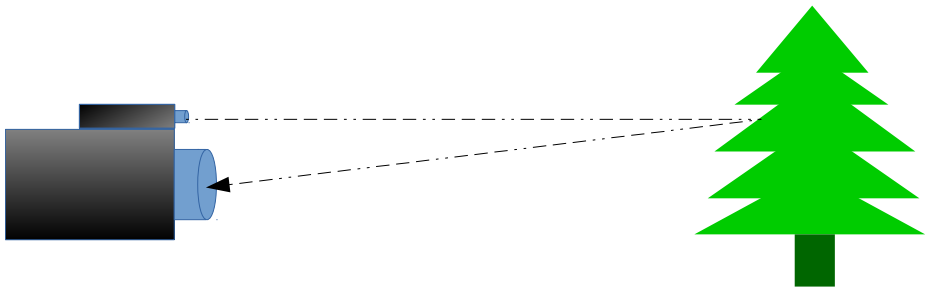


Figure 2.1: The light emitted from an energy source is reflected on the tree and received by the camera lens.

The time it takes for light to be reflected on an object placed one meter from the camera is ≈ 6.7ns. When the time constraints are so low, it takes highly specialized hardware in

order to make precise measurements and it is not possible to get ToF measurements with regular camera equipment.

There exist multiple variations of ToF cameras that uses different types of light emitters, sensors and methods to calculate the round trip time. Two common ways to measure the time is by using either continuous wave modulation or pulsed modulation. In continuous wave modulation the phase shift between the sent and received signal is measured and used to calculate the distance and in pulsed modulation it is the time delay between the sent and received signal that is used to calculate the distance.

Stereo Vision

Stereo vision refers to the use of multiple images covering the same object from different angles to generate depth information. The depth information is typically calculated using triangulation methods based on the intersection points that can be found in the respective images. Intersection points are points shared by two or more images captured from different angles. An often used stereo vision setup is to have two cameras aligned vertically but with a horizontal displacement or vice versa. An illustration of a stereo vision setup can be seen in Figure 2.2.

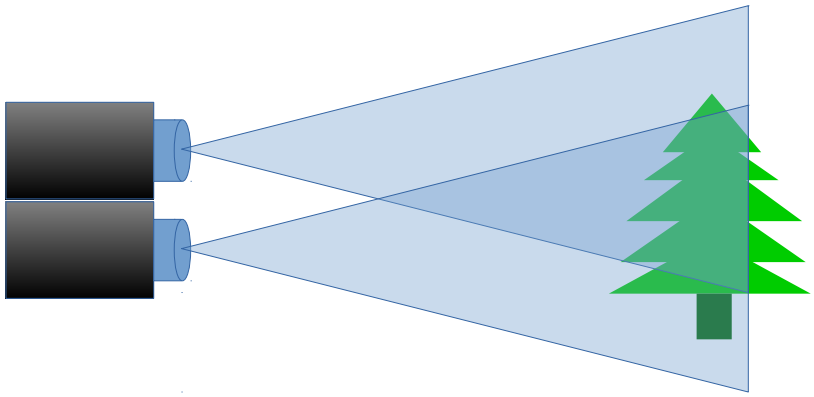


Figure 2.2: Typical stereo vision setup. Depth information can be found in the triangle shared by the two views.

There are no specific requirements to cameras or sensors used to acquire the images used in a stereo vision setup. However, the precision may be influenced by certain factors such as resolution, lens distortion and baseline between the cameras.

Plug-and-play stereo systems can be bought from various companies, but they often lack flexibility as the two cameras are interlocked. A downside of using regular cameras, on the other hand, is that calibration and synchronization must be customized to the specific setup, which, in some cases, can be a time-consuming task.

The amount of frames per second a stereo vision system can produce depends on whether the system is used in real time or not. If the system should be capable of measuring depth in real time, the bottleneck is most likely placed in the software that calculates the depth and not in the camera. On the other hand, if the system is not to be used in real time, the amount of frames per second corresponds to that of the camera, which is typically high compared to other 3D imaging systems.

The typical stereo camera is passive as it does not actively use a specific source of light to measure the depth in the scene. This can be an advantage in research installations where certain types of light may influence the experimental setup. However, this means that the stereo camera is dependent on light from other sources and generally produces the best depth measurements in well-lit conditions.

2.1.2 Structured Light

Structured light is a method utilizing at least one camera and a projector, that emmits light in a known pattern. The camera is placed at a known distance from the projector and the reflected pattern received by the camera is analysed and processed in order to calculate the depth based on distortions in the pattern. Triangulation is an often used method to measure the distance by calculating the geometric relationship between the camera, projector and a given point on the object. A typical setup of a structured light 3D scanner system can be seen in Figure 2.3.

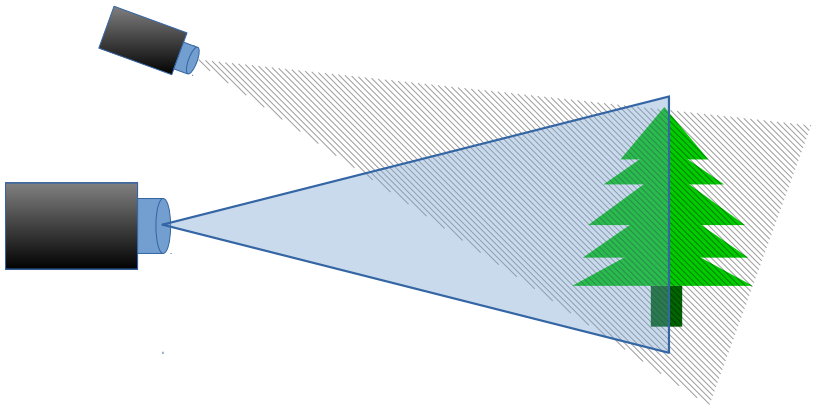


Figure 2.3: Structured light 3D scanner setup.

There exist many different types of patterns, light emmitters and image acquisition sensors that can be used to generate 3D data. Typical light emmitters include lasers, LCD and infrared projectors and depending on the scene, object and application, some setups may be preferred above others. However, in general, the scene must always be lit by an active controlled light source that inevitably interferes with the environment to some degree.

It is possible to make relatively simple structured light systems consisting of regular cameras and home-theater projectors, however, ambient light may have a negative effect on the performance of such systems. If the scene is lit by strong ambient light it may not be possible to recognise the pattern in the recorded image or video due to low contrast.

Some commercial systems utilize projectors emitting infrared radiation, but even though infrared light is invisible to the human eye it is still affected negatively by ambient light, degrading the performance of the system. Structured light 3D systems are primarily used in environments where ambient light can be controlled to some degree, although there exist methods that enhances the performance of the systems in sunlight [17].

The amount of frames per second is often, but not always, determined by the image acquisition sensor used in the setup. The precision generally resembles that of stereo vision systems as triangulation is used in both cases.

Summary of Depth Measuring Methods

The different depth measurement methods all have pros and cons dependent on the scene, environment and application. Generally, the price of commercial systems is coherent with performance, however, some cameras may have a narrow field of view, but a high precision and frame rate, while it is the opposite for others. The cameras vary a lot even if they are a part of the same type of 3D measurement system and this makes it difficult to create an overview in general terms. However, a summary of the information presented in the previous subsections has been in gathered in Table 2.1 to give a crude overview of three popular camera systems capable of acquiring depth information.

System	**FPS**	**Precision**	**Range**	**Light Source**	**Price**
Time-of-Flight	Medium	1cm	Short to long	Active	Medium-high
Stereo Vision	High	1cm	Medium	Passive	Low-medium
Structured Light	High	1mm	Short-medium	Active	Medium

Table 2.1: An overview of the capabilities of the described 3D scanner systems.

Stereo Vision is the chosen depth measurement method, as it has important characteristics that are requested in Section *1.2 Properties and Limitations*, which the other methods lack. This includes the no interference limitation, as both ToF and structured light use active light sources, that may interfere with the fishes. Another wanted property is that the system should consist of off-the-shelf hardware and this is also fulfilled in a stereo vision setup. The lack of precision of stereo vision compared to structured light is not seen as an obstacle as the application of the tracker system does not require highly precise measurements, as long as it is possible to create motion trajectories. The amount of FPS may play an important role in the system, however, the FPS of a stereo vision setup depends solely on the cameras that are used and can therefore be easily revised.

When using two or more cameras to estimate depth, it is essential that the recordings are temporally aligned. This is known as the synchronization problem and it will be described in the following section.

2.2 Synchronization of Multiple Cameras

One of the limitations to the system is that the recordings must be temporally aligned before being given as input to the system, as described in Section *1.2 Properties and Limitations*. This is because temporal alignment between the recordings is essential in order to be able to estimate the 3D position of the objects in the scene. The synchronization problem will be described briefly in this section, even though no modules will be designed to handle the problem.

Synchronization Problem

The problem of aligning two or more video sequences temporally consists of finding a single frame that refer to the same point in time. The problem is well explained in [18] as finding a frame, f, from each sequence, i, such that

$$T_i(f) \quad = \quad T_{i+1}(f) \quad = \quad ... \quad = \quad T_{i+n}(f), \tag{2.2}$$

where T_i is a function that maps from local to global time and $n-1$ is the number of sequences. This is known as the synchronization equality and if it is possible to find a subset of frames, that exactly solves Equation 2.2, the sequences are said to be perfectly temporally aligned.

However, this is often not possible unless the phase shift of the sequences exactly resembles the frame rate of the cameras. If this is not the case, the solution consists of finding a subset of frames that minimizes the difference in time between the sequences. An illustration of two phase shifted sequences can be seen in Figure 2.4. In a worst case scenario, the phase shift, ϕ, is given by

$$\phi = \frac{2}{\text{FPS}}, \tag{2.3}$$

which corresponds to half the time interval between to consecutive frames. It should be noted that the phase shift is constant through the entire sequence if the cameras have the exact same frame rate.

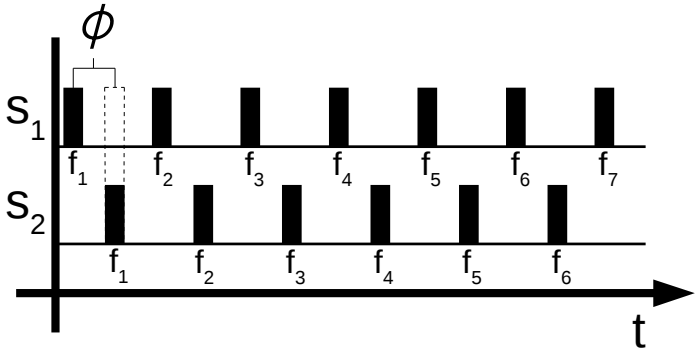


Figure 2.4: s_1 and s_2 are two sequences, which have been recorded with a temporal phase shift.

The synchronization problem will not be described in further details, but various methods to handle the problem will be mentioned in Section *8 Discussion*. The next section will be about camera theory and lights behavior when traveling between two media.

2.3 Capturing Images of Underwater Objects

The following is a concise survey of the basis camera and refraction theory. The section is commenced with a description of the pinhole camera model, as this is the prevalent model when dealing with cameras. This is followed by a description of how light behaves when moving from air to water.

2.3.1 The Pinhole Camera

The aim of the pinhole camera model is to provide a simple model for the mapping between a 3D space to a 2D plane, which is essentially what happens when a camera captures an image. The model is simple in the sense that a tiny aperture, i.e. pinhole, is used instead of a lens to control the light reaching the image sensor.

An example of the pinhole model is shown in Figure 2.5, where a sphere in the 3D world space is mapped to the 2D plane of the image sensor. The 3D space is spanned by the three vectors $\begin{bmatrix} x & y & z \end{bmatrix}$ while the image plane is spanned by the two vectors $\begin{bmatrix} x' & y' \end{bmatrix}$.

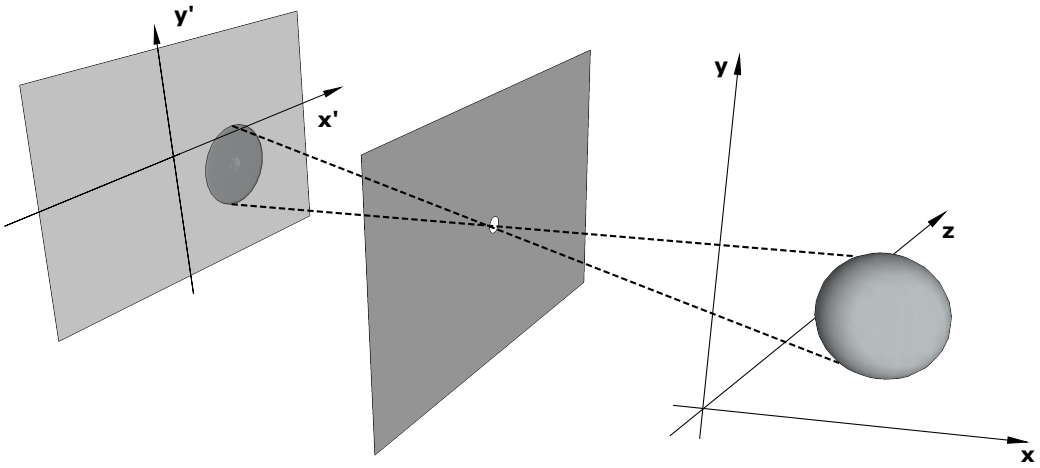


Figure 2.5: Example of the pinhole camera model where points in a 3D world space are mapped to a 2D image plane. The small pinhole aperture controls the light rays reaching the image sensor.

The mapping between 3D and 2D in Figure 2.5 can be described as a projective transformation, as shown in Equation 2.4. The matrix $P_{3\times4}$ encompasses the mapping from 3D to 2D and will be referred to as the camera matrix.

$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = P_{3\times4} \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} \tag{2.4}$$

It is possible to deconstruct the camera matrix P into intrinsic and extrinsic parameters such that:

$$P = K[R \mid t] \tag{2.5}$$

The intrinsic parameters, K, are given by

$$K = \begin{bmatrix} a_x & s & x_0 & 0 \\ 0 & a_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{2.6}$$

where (a_x, a_y) is the focal length in pixels, s is the skew coefficient and (x_0, y_0) is the position of the principal point.

The extrinsic parameters consist of a rotation, R, and translation, T, which describes a mapping from world coordinates to camera coordinates. An illustration of the mapping can be seen in Figure 2.6.

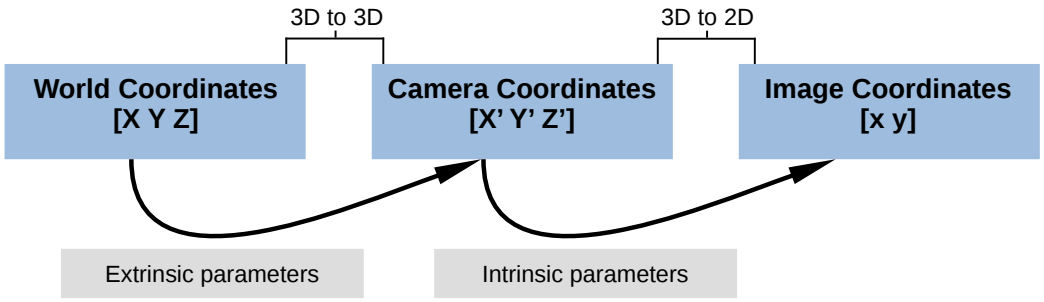


Figure 2.6: The mapping from 3D world coordinates to 2D image coordinates using the extrinsic and intrinsic parameters.

A position in world coordinates can hence be mapped to camera coordinates by applying the following transformation matrix:

$$\begin{bmatrix} R & T \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{2.7}$$

It should be noted that T does not correspond to the camera position, C, in world coordinates. C can instead be found as stated in Equation 2.8. The simplification: $R^{-1} = R^T$ is possible as the rotation matrix R forms an orthonormal basis, i.e. it consists of orthogonal unit vectors.

$$0 = RC + T \quad \rightarrow \quad C = -R^{-1}T \quad \rightarrow \quad C = -R^T T \tag{2.8}$$

2.3.2 Refraction

When recording video sequences where light travels between different media, a phenomenon, known as refraction, occurs. Refraction causes the light to change direction at the interface between two media, such as water and air. It can easily be observed by looking at an object partly submerged in water as shown in Figure 2.7.

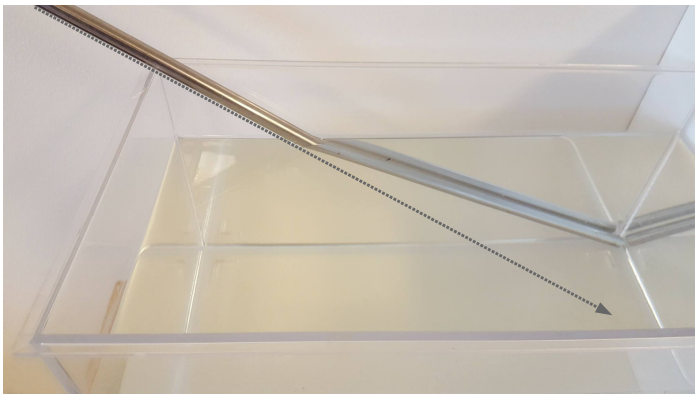


Figure 2.7: Example of refraction where a straight object appears to bend at the interface between air and water. The illusion of the object bending occurs as the way light propagates through waters differs from its propagation through air.

Refraction can be described using Snell's law, which states that

$$\frac{sin(\theta_1)}{sin(\theta_2)} = \frac{n_2}{n_1}, \tag{2.9}$$

where θ_1 is the angle of incidence, θ_2 is the angle of refraction, and n_2 and n_1 are the refractive indexes of the two media.

An example of the angle of incidence, θ_1, and angle of refraction, θ_2, is shown in Figure 2.8. Note that the rays of light travel from an object submerged in water, which are then observed from outside the water. The light will hence be refracted in the interface between water and air.

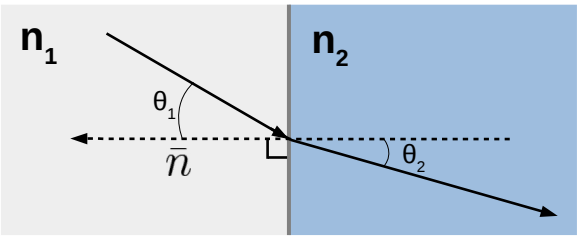


Figure 2.8: Example of refraction occurring as light travels from water to air.

The refractive indices for different media is listed in Table 2.2. The listed refractive indices are mainly the ones of interest for this project, i.e. air, water and glass and acrylic glass.

Medium	Refractive Index
Air	1.0
Water	1.326
Glass	≈ 1.5
Acrylic glass (PMMA)	1.481

Table 2.2: Refractive indices for different mediums. The values originates from [19].

An illustration of how refraction affects an observed scene is shown in Figure 2.9, which depicts the paths of several light rays, r, as they travel from water to air before hitting the lens of the camera. The refraction causes the camera to experience the scene as if it was observed from multiple viewpoints, vp. This observation is important as it essentially invalidates the pinhole camera model, described in Section *2.3.1 The Pinhole Camera*, as the rays are no longer focused in a single viewpoint. This strongly indicates that refraction needs to be taking into account when processing images where the light travels between two media and solutions to handle this will be described later. The following section will be about various stereo vision setups and their pros and cons.

2.4 Stereo Vision Camera Setup

The camera setup in a system based on stereo vision is not dependent on the cameras being aligned in a specific way, as long as there is an overlap between the views. This is an advantage, as it allows for multiple setups and viewing angles to be investigated.

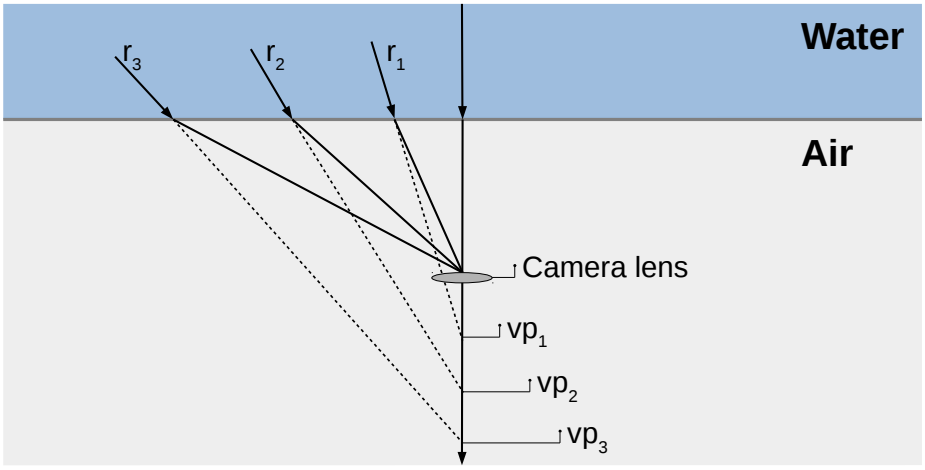


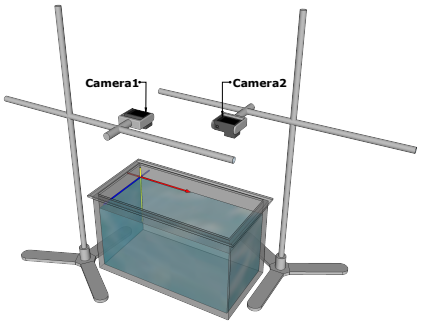
Figure 2.9: Illustration of how light rays are bent due to refraction before entering the aperture of a camera.

A common stereo vision setup consists of two cameras placed side-by-side, as known from the visual system of most mammals including humans. This type of setup makes sense to be used in two constellations: directly above or beside the aquarium. Another possibility is to place one camera above and one camera beside the aquarium, to utilize potential advantages from both angles.

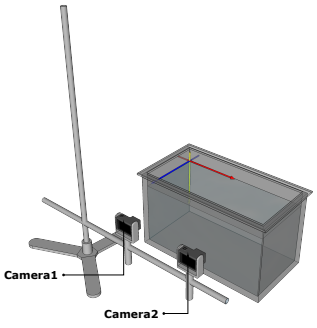
In this section, several stereo vision setups will be described and discussed in order to present pros and cons associated with the respective setups.

Side-by-Side

The side-by-side view is the most common type of stereo vision setup and is frequently used in commercial products. It is flexible in the way, that it can be moved around and placed in different positions around the aquarium. The two places, where it makes most sense to mount the cameras is either above, as seen in Figure 2.10a, or in front of the aquarium, as seen in Figure 2.10b. As the setup consists of two regular cameras, they are not restricted to



(a) Side-by-side - above the aquarium.



(b) Side-by-side - in front of the aquarium.

be aligned in a certain way or with a specific baseline distance and this makes the setup both

scalable and easy to mount. Furthermore, when the cameras are looking into the aquarium from the same direction, the light travels through the same media for both cameras. When looking down from above the aquarium, the light travels through air and water and when the cameras are placed in front of the aquarium, the light travels through air, glass and water.

A possible downside of this type of setup is that the risk of occlusion may increase. The two cameras share the same directional view, which means that multiple fish can easily occlude each other, such that they are either fully or partially occluded from both cameras. An example can be seen in Figure 2.11, where 3 fish are placed in the aquarium and the cameras are placed above. This scene is depicted in Figure 2.12a, seen from camera 1 and in Figure 2.12b, seen from camera 2. There is partial occlusion seen from both views and it is difficult to determine where the heads and tails from the fishes are located. Even though it is not illustrated, the occlusion problem occurs when both cameras are located in front of the aquarium as well. However, a positive effect of this setup is that the appearance of the fishes will be similar for both cameras and this could potentially make the detection algorithm more simple.

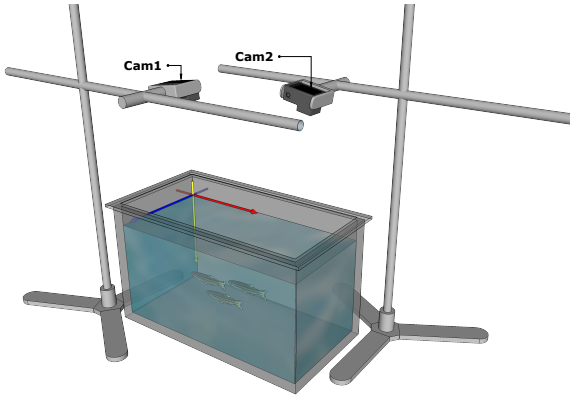
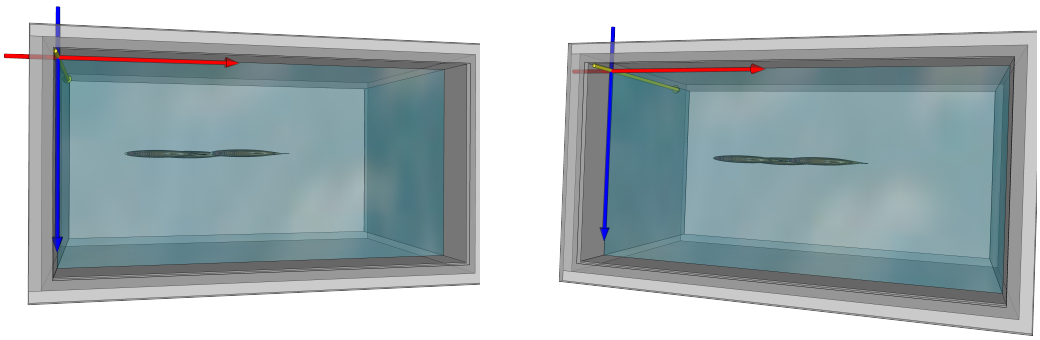


Figure 2.11: A scene with three fish and the cameras placed side-by-side above the aquarium.



(a) (Cam1) Partial occlusion.

(b) (Cam2) Partial occlusion.

Figure 2.12: This is the same scene as the one depicted in Figure 2.11.

Above and Beside

As mentioned earlier, there are no constraints on the alignment of the cameras, as long as they share the part of their view, where the 3D information is wanted. This allows for a setup where one camera is placed above the aquarium and one camera is placed in front of the aquarium. This kind of setup can be seen in Figure 2.13.

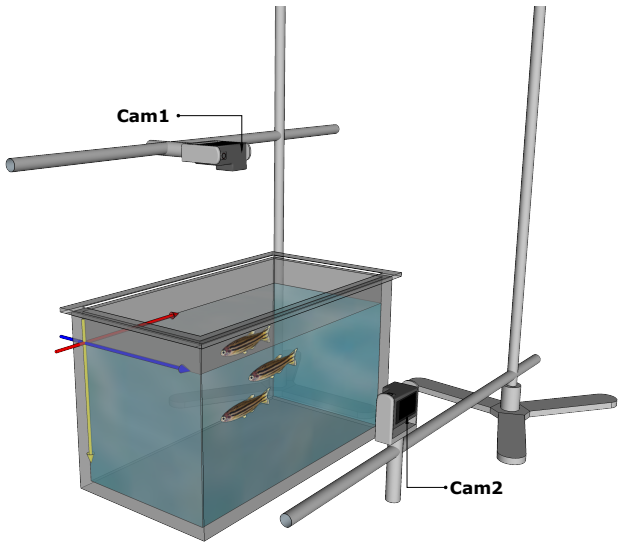


Figure 2.13: Camera setup with one camera above and one in front of the aquarium.

An advantage of such a setup is that the fishes are less likely to be occluded from both views at the same time, as illustrated in Figure 2.14. The scene depicted in the illustration is the same at the one scene in Figure 2.13, however, when seen from the two cameras, two things become clear. Seen from camera 1, there are only two visible fish, while there are three visible fish seen from camera 2. This means that a one fish is fully occluded for camera 1 and it must be beneath one of the visible fish. Furthermore, the fishes do not look alike between the two views. This may be a disadvantage as a more complex detection algorithm may be needed in order to detect the fishes in both views.

The light that is reflected from the aquarium only moves through water and air before entering camera 1, but needs to pass through glass as well before entering camera 2. Dependent on the depth of the glass, this may have to be taken into account and this introduces one more variable, and thereby uncertainty, into the system. Furthermore, this type of setup induces a restriction to the material of the test aquarium, as it must be transparent.

In the Water

Yet another setup involves the cameras placed into the water in order to avoid complications with refraction. However, when the cameras need to be submerged into the water, some constraints and complications are introduced.

The cameras need to be waterproof and it will be difficult to create a setup with the cameras submerged into the aquarium without introducing blind angles where the fish can hide for either one or both of the cameras. Beside the blind angles, the cameras may also

(a) (Cam1) Full occlusion.

(b) (Cam2) No occlusion.

Figure 2.14: This is the same scene as the one depicted in Figure 2.13.

cast shadows depending on where the light sources are placed. Furthermore, as explained previously, a minimum of interference with the fishes is wanted and the cameras may affect the fish in some way as they stand out in a controlled environment. No illustrations are shown as it is considered to be an undesirable setup and it will not be investigated further.

Besides the different camera setups, it may also be worth considered the actual behavior of the targets which are to be tracked. The next section will hence deal with the movement and behavioral patterns of zebra fish and fish in general.

2.5 Fish Movement and Behavioral Patterns

Many types of fish, including zebra fish, demonstrate group behavior to some degree, which are generally described by the two terms shoaling and schooling. Shoaling describes groups of fish that are together for social, defensive and foraging reasons, but without a joint movement pattern. Shoaling fish move independently, but stay near each other. Schooling is, on the other hand, used to describe fish that follow the same movement patterns with a common direction and movement speed [20]. There is a gradual transition between shoaling and schooling behavior and one specific group of fish may exhibit characteristics from both groups. A common measurement to determine the type of behavior a group of fish is demonstrating, is the degree of which the members of the group is moving in the same direction and their overall coordination.

It has been documented that zebra fish demonstrate both types of group behavior depending on the size of the group, among other things, and the type has influence on their general movement speed and distance to their neighbors [21]. In shoals, the movement speed is relatively high and the distance between individual members of the group is short. In schools, on the other hand, the individual members are positioned further from each other and the movement speed is generally lower. Whether a group of zebra fish demonstrates a shoaling or schooling behavior depends on both the group size and their familiarity with the environment. As the fish grow accustomed to their environment they tend to go from a schooling to shoaling behavior with more erratic and faster movement patterns.

The change in group behavior could have influence on the amount of occlusion and if this is true, tests could be conducted in order to determine for how long the zebra fish need

to stay in the test aquarium before they grow accustomed to their surroundings. However, this will not be investigated further.

The speed of the zebra fish varies to a large degree, depending on their surroundings, and they are capable of changing their travel direction within a short amount of time. To get an idea of the distances the fish are able to cover between two frames, a speed test has been conducted.

2.5.1 Zebra Fish Speed Test

The speed test is conducted in order to determine the maximum amount of pixels a zebra fish can cover between the acquisition of two consecutive frames. It is assumed that the fish is capable of reaching its maximum speed even though it only has the possibility of moving in a plane. Therefore, the test is conducted in shallow water, which only allows the fish to move in two dimensions and the sequence is acquired using a single camera.

Setup

The setup consists of the test aquarium described in Section *3* , with water corresponding to 4cm in height. The camera is mounted directly above the aquarium with a view of the entire floor. The camera is set to acquire images with 240 frames per second and a resolution of 1280x720, which is the highest resolution for the given frame rate on GoPro HERO5. The setup is illustrated in Figure 2.15.

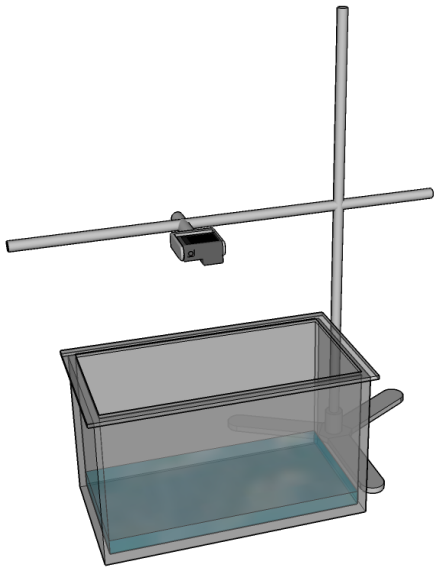


Figure 2.15: Illustration of the experimental setup for the zebra fish speed test.

Execution

A knock on the side of the aquarium is used to stress the fish, as it is assumed that the fish will move at full speed for at short amount of time if stressed by sudden moves or sounds

that may indicate danger. The test is executed once per fish with a total of 5 fish and consists of the following steps:

- Put one fish in the test aquarium.
- Record for 20 seconds.
- Knock on the side of the aquarium every few seconds.
- Put the fish into a bucket with water.

The above-mentioned is repeated for all 5 fish and when the last test has been executed, the fishes in the bucket are put back into the cozy aquarium.

Results

The speed varies to a large degree across the recorded sequences for all five fish. However, the same pattern is observed with high accelerations and short speed bursts when the fishes detects the finger approaching and knocking on the glass. A histogram showing the observed speeds of one of the fishes can be seen in Figure 2.16. The histograms of all five fish resemble each other with a maximum speed of ≈ 70-90$\frac{\text{cm}}{\text{s}}$ and an average speed of ≈ 9-13$\frac{\text{cm}}{\text{s}}$ in a stressful environment.

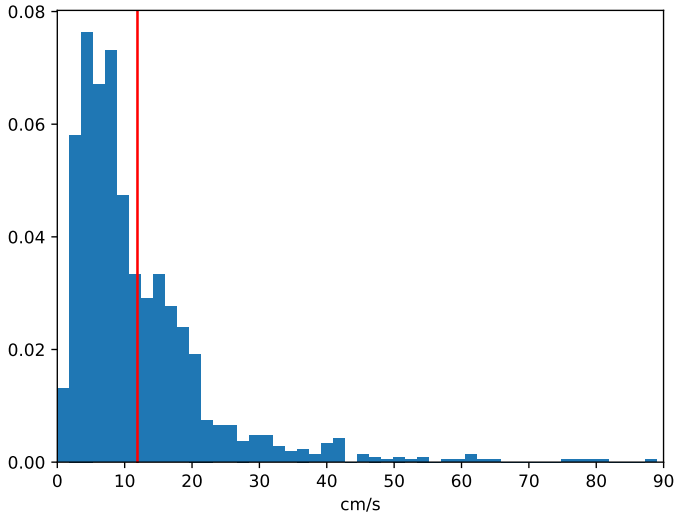


Figure 2.16: Histogram showing the speed variations of a zebra fish in a stressful environment.

In scenarios where the fish accelerates, a blur is observed despite a relatively high frame rate of 240 FPS. In Figure 2.17a, the fish is moving at a low speed, however, in the next frame shown in Figure 2.17b, the fish accelerates and becomes blurred. It is expected that this blur is enhanced when recording with a lower frame rate and it may reduce the detection rate to some degree.

The results of this test make it possible to calculate the amount of pixels a fish is able to cover in a single frame depending on the spatial resolution, frame rate and distance between the camera and fish.

(a) Moving slowly. **(b)** Acceleration.

Figure 2.17: Two consecutive frames, showing how the fish becomes blurred when it accelerates.

2.6 Problem Statement

The purpose of this section is to briefly highlight the main areas of focus in the thesis. This is done by formulating a problem statement based on the various observations and considerations made in the previous section. The problem statement consists of the following:

- How can multiple zebra fish be detected reliably in a controlled test environment?
- Is it necessary to account for refraction when employing stereo vision to extract the 3D positions of the zebra fish? If so, how can this be accomplished?
- How can coherent 3D tracks be formed for each zebra fish?
- Can the 3D information be incorporated in the process of creating tracks in a beneficial way? If so, how?

3 Experimental Setup

In the development of a system capable of tracking objects moving in three dimensions under water, a certain amount of data is required for test and validation. As the objects in this case are fish, there are several distinctive requirements that need attention before it is possible to gather data. The specifications of the experimental setup and the datasets used to develop the tracker system will be described in this section.

Fish

Tracking fish in 3D is the main topic of this project, but the specific type of fish is not crucial. However, the zebra fish is widely used for research purposes and it could therefore be favorable to make a system that is tested and validated on this specific species. As mentioned in Section *1 Introduction*, the zebra fish is one of the most used animals for research because they are cheap, small, easily maintained and they have a transparent embryo which makes it easy to examine how they are affected by diseases, drugs, medicine or gene manipulation [22]. Because of this, the zebra fish is chosen as the type of fish the system is tested on.

Cozy Aquarium

When keeping live fish for several months, it is necessary to have an aquarium where the fish can stay sound and healthy to avoid animal cruelty. The size of the aquarium is dependent on the type and amount of fish that are to be kept in it, however, as zebra fish are chosen as the test subjects, an aquarium of minimum 50 liters is enough to ensure a healthy environment for a group of 10-20 zebra fish [23]. The chosen aquarium is an Eheim Aquastar 96L, with a 5cm fine gravel substrate and multiple eel grass plants (vallisneria spiralis). Beside the zebra fish, some shrimps and catfish are kept in the aquarium to maintain a healthy environment. An illustration of the cozy aquarium can be seen in Figure 3.1 and the specifications of the aquarium are described in Table 3.1.

Length	**Width**	**Height**	**Volume**	**Material**
80cm	30cm	40cm	96L	Glass

Table 3.1: Cozy aquarium specifications.

Figure 3.1: Cozy aquarium.

Test Aquarium

The fish are placed in a smaller aquarium that contains nothing but water when they are recorded, as this allows for easier tracking and test reproducibility. The test aquarium has the specifications outlined in Table 3.2 and an image of the aquarium can be seen in Figure 3.2. The test aquarium is placed upon a white sheet of wood and in front of a white wall to keep the environment clean and simple.

Length	Width	Height	Volume	Material
40cm	20cm	25cm	20L	Plastic

Table 3.2: Test aquarium specifications.

Figure 3.2: Test aquarium.

Light

The light source used in the setup, to illuminate the aquarium, must be an off-the-shelf product according to the properties outlined in Section *1.2 Properties and Limitations*. It is important that the aquarium is well lit, but not all types of light is suitable to use. The light cannot be flickering with a low frequency, such as 50Hz, as this may interfere with the recordings and cause the luminance to change between each frame.

To ensure that the aquarium is well lit in the test setup, one 25cm LED strip has been mounted on each side of the aquarium connected to an AC/DC converter to avoid the flickering. The specifications of the LED strips used for the setup can be seen in Table 3.3.

Voltage	Current	Color Temp.	Light Color	LEDs
12V	1.5A/m	4000K	Neutral white	Samsung 5630

Table 3.3: Specification of the LED strips used in the test setup.

Camera

Two GoPro HERO5 cameras are used to acquire video sequences from two angles. The cameras have a wide range of configurations and are capable of recording with up to 240 frames per second (FPS). The resolution of the recorded sequences are highly dependent

on the frame rate and the highest spatial resolutions are only available for low frame rates. It is possible to adjust multiple camera settings manually, including ISO, shutter speed, white balance and more. However, as the system should be as user-friendly as possible, all settings, except for FPS and resolution, are left as default throughout the tests. The frame rate, maximum resolution and field-of-view (FOV) options can be seen for three chosen configurations in Table 3.4.

FPS	Max res.	FOV
240	1280x720	Narrow
120	1920x1080	Narrow/wide
60	2704x2160	Wide/medium/linear

Table 3.4: Relevant frame rates and matching resolutions of the GoPro HERO5 camera.

3.1 Description of Dataset

The datasets are important for the development of the system, as they serve as both test and validation for most of the modules. In order to be able to test different variations of the modules, two types of dataset have been acquired. They have been recorded from two different camera setups, which is the side-by-side placed above the aquarium, which is known as above-and-above from this point on, and the above-and-beside setup. Multiple variations of the two types of dataset have been acquired and used for preliminary tests, however only one set of each has been annotated and their specifications will be described in Table 3.5 along with a dataset used for measuring the speed of the fish and a dataset that has been used specifically for one part of the system, which will be described later.

In all of the datasets, the annotation is made on top of the head of the fish. When seen from above, the point is located between the eyes of the fish and when seen from the side, the point is located in the center of the eye. It is only this point which has been annotated as it responds to the keypoint found by the SURF algorithm, which the detection relies on. This is explained in further details in Section *5.1 Fish Detection using SURF*. It should be noted that the D14 dataset is from [24], while the rest has been recorded at Aalborg University by the authors of this thesis.

	AB-1080-60-6f	AA-1520-60-7f	Speedtest	D14
Setup	Above-and-Beside	Above-and-Above	Above	Above
Date	02-05-2017	15-05-2017	15-05-2017	09-03-2016
Resolution	1920x1080	2704x1520	1280x720	2040x2048
Length	61s	64s	26s	66s
FPS	60	60	240	30
Fish	6	7	1	14
Annotated	Yes	Half	Partially	Yes

Table 3.5: Specifications of the datasets used for testing and validating the performance of the system.

The naming of the two datasets are based on the setup, resolution, frame rate and amount of fish. The names for the above-and-above and above-and-beside setups are ab-

breviated to AA and AB, respectively. The specifications of the datasets are presented in Table 3.5, while an image from every dataset can be seen in Appendix - *Datasets*. The visual appearance of the zebra fish seen from above and beside will be shown and described in the following section.

3.1.1 Visual Appearance of Zebra Fish

The zebra fish is a lively and fast moving species, that is able to change direction within a short amount of time. When the fish moves around, its appearance may change to some degree, dependent on the viewing angle. A few snippets of different fish have been chosen from both top and front view recordings to illustrate the variations of appearance the fish may have. The images taken from the camera placed in front of the aquarium can be seen in Figure 3.3 and the images taken from above the aquarium are presented in Figure 3.4. All the images are taken from the AB-1080-60-6f dataset.

Figure 3.3: Different variations of how the fish may appear seen from the camera placed in front of the aquarium. All the images are taken from the AB-1080-60-6f dataset.

Figure 3.4: Different variations of how the fish may appear seen from the camera placed above the aquarium. All the images are taken from the AB-1080-60-6f dataset.

When the fishes are seen from above, the front part of their body tend to keep the same appearance with a broad head and a slightly pointed nose. The rear part of the body may twist and turn, but is visible at full length most of the time, except when the fish moves vertically in the water, which makes the body appear slightly shorter.

On the other hand, when the fishes are seen from the camera placed in front of the aquarium, their appearance may change relatively much as they move around in the water. A very distinct example is the rightmost image in Figure 3.3, where a fish is swimming towards the camera and appears as small round blob with only a few characteristics. Some of the details of the fish may also be lost due to a lower spatial resolution when the distance to the camera is increased.

3.2 Evaluation of Dataset

Dependent on the angle and distance between the cameras and the aquarium, the dataset may vary. This includes both appearance and spatial scattering of the fish. Because of that, the datasets are evaluated by a range of metrics in order to help determine pros and cons of the different camera setups.

3.2.1 Average Nearest Neighbor

In order to determine how scattered the fish are, the Average Nearest Neighbor ratio (ANN) is measured [25]. The ANN ratio describes the spatial relationship between the individual fish and their closest neighbor. It is a ratio between the observed and expected Nearest Neighbor Distance (NND), where the observed NND is the euclidean distance between a single fish and its nearest neighbor given by

$$d_i = min(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}), \tag{3.1}$$

where (x_i, y_i) is the position of the fish to be evaluated and (x_j, y_j) is the position of the other fish in the aquarium, where $j = \{1..n\}$ and n is the number of fish. The observed NND is calculated for every fish in a frame in order to find the mean and it is given by

$$D_O = \frac{1}{n}\sum_{i=1}^{n} d_i. \tag{3.2}$$

An example of NND for three fish is illustrated in Figure 3.5, where the colored lines represent the distance between the identically colored fish and its nearest neighbor.

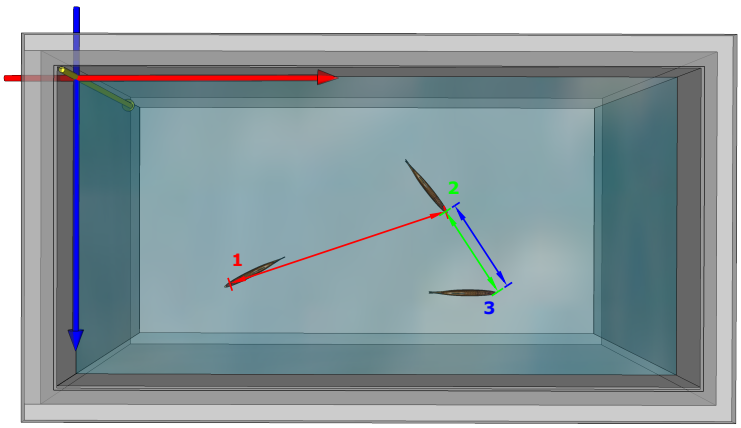


Figure 3.5: Illustration of nearest neighbors.

The expected NND, on the other hand, is calculated as the mean distance between the fishes, if they were evenly distributed in a random pattern across the aquarium, and it is given by

$$D_E = \frac{0.5}{\sqrt{\frac{n}{A}}}, \tag{3.3}$$

where A is the area in pixels seen by the camera, where the fish are able to swim. This area is dependent on the angle and distance between the camera and aquarium and may vary between the dataset. The ratio between the observed and expected NND is given by

$$R_{ANN} = \frac{D_O}{D_E}. \tag{3.4}$$

3.2.2 Inter-Individual Distance

The Inter-Individual Distance (IID) describes the spatial relationship between the individual fish and every other fish in the aquarium and is an expression of the degree of clustering for the whole group. As with the ANN, it is the ratio between the observed and expected IID, which is interesting.

In order to determine the observed IID, the euclidean distance is measured between a given fish and every other fish as illustrated in Figure 3.6. The distance between a fish and

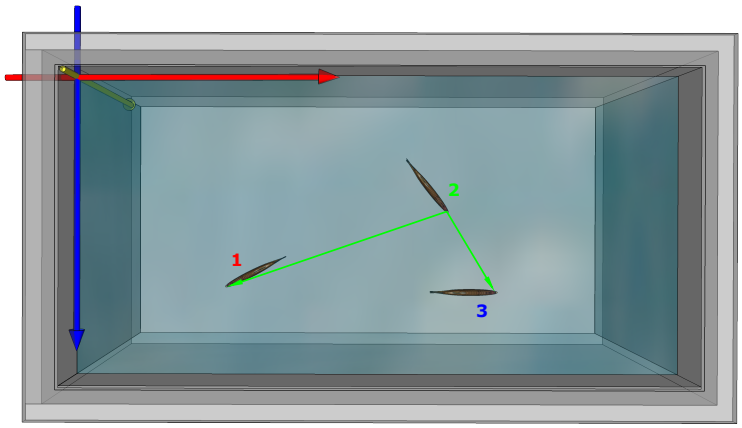


Figure 3.6: The distance between a fish and its neighbors. The mean of the distances is calculated for every fish in the frame in order to find the IID.

its neighbors is given by

$$d_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3.5}$$

and the IID for each fish is calculated as

$$I_i = \frac{1}{n-1} \sum_{j \neq i} d_{i,j}, \tag{3.6}$$

where $j = \{1..n\}$ and n is the number of fish. The mean of the observed IID is given by

$$I_O = \frac{1}{n} \sum_{i=1}^{n} I_i. \tag{3.7}$$

The expected IID is the mean distance between n random and evenly distributed fish across the aquarium and is therefore given by $I_E = D_E$. The IID ratio is given by

$$R_{IID} = \frac{I_O}{I_E}. \tag{3.8}$$

3.2.3 Spatial Evaluation

The evaluation is based on the AB-1080-60-6f dataset and the area, A, is found manually by measuring the region on the recordings where the fish are able to swim. The area is found to be:

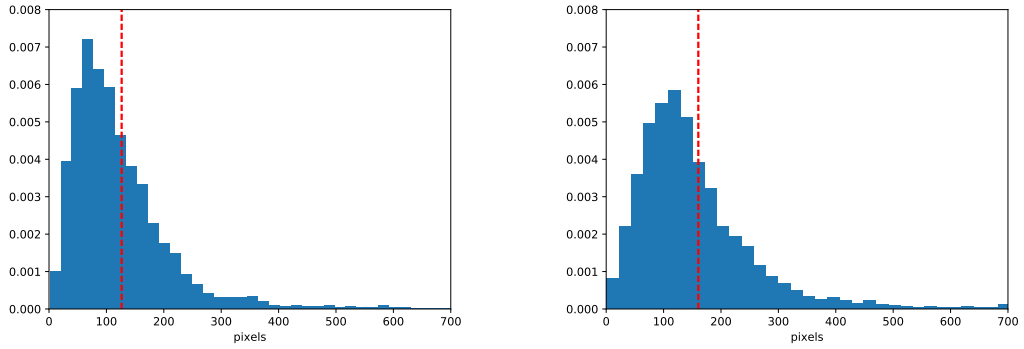
- $A_{front} = 1,579,812$px
- $A_{top} = 1,652,590$px

The area is used for calculating both the R_{ANN} and R_{IID} for the front and top camera views and the results are presented in Table 3.6. The ratio of the observed and expected values for both ANN and IID gives an idea of how the fish are scattered across the view. In case of $R = 0$, the fish are completely aggregated and as R increases, so does the scattering. This means that when R_{ANN} is high, the fish are generally close to a neighbor, however, this could be true for one large group or for an arbitrary number of smaller groups. A high R_{IID}, on the other hand, indicates that every fish is close to all its neighbors and reveals how closely grouped all the fishes are.

	$\mathbf{D}_O$	$\mathbf{D}_E$	R_{ANN}	$\mathbf{I}_O$	$\mathbf{I}_E$	R_{IID}
Front	126px	256px	0.49	228px	256px	0.89
Top	160px	262px	0.61	264px	262px	1.01

Table 3.6: The ANN, IID and ratios from the AB-1080-60-6f dataset for both top and front camera.

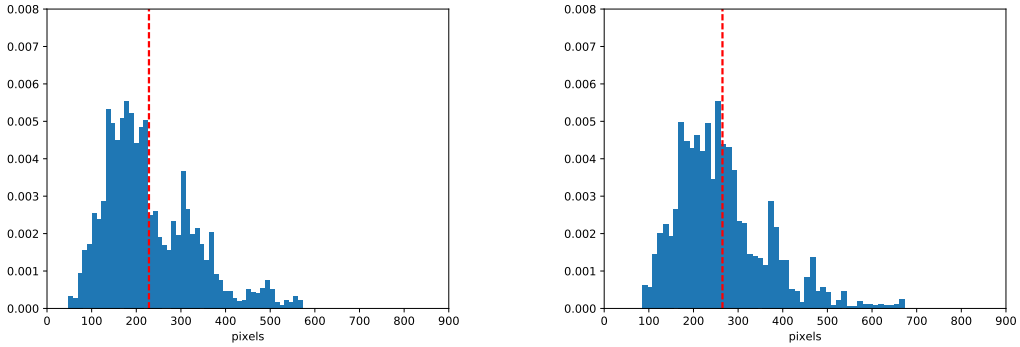
It should be noted that the R_{ANN} and R_{IID} are mean values of the entire recorded sequence and they are used as overviews of the general tendency of the respective view. However, the IID and NND have been calculated for every frame across the sequences and these results have been gathered in the histograms shown in Figure 3.7 and Figure 3.8.



(a) NND front. **(b)** NND top.

Figure 3.7: Histogram of the mean NND for every frame in the AB-1080-60-6f dataset.

According to the results, both R_{IID} and R_{ANN} are higher for the top view. This means that in general, when the fishes are seen from the top view, they are located further from each other both individually and group-wise, relative to the area of observance. The histograms of both IID and ANN resembles each other for both views, however, a slight right shift can be observed for both top view histograms, which supports the ratios.



(a) IID front.

(b) IID top.

Figure 3.8: Histogram of the mean IID for every frame in the AB-1080-60-6f dataset.

It can be concluded, based on the gathered results, that the fish are less grouped when seen from above the aquarium. This could indicate that the fishes tend to follow their neighbors and stay in the approximately same vertical level of the aquarium and move more freely in the horizontal directions.

The visual appearance of the fishes, when seen from the side, could cause problems for some detection algorithms. The contrast various depending on the distance between the fish and the camera and the bright stripes on the fish have a color that resembles the background to some degree. Furthermore, when the fish is swimming orthogonal to the viewing angle, the body becomes very small and difficult to detect.

The results of the IID and ANN tests and the visual inspection of the fish images indicates that the optimal position for the cameras are above the aquarium. It should be noted that only one dataset has been investigated, which makes it difficult to conclude whether it generally is the case that the fish are scattered the most, seen from the top view. However, this iteration of the system will be developed to track fish seen from two cameras placed side-by-side above the aquarium, based on the above findings. The overall design of the system will be outlined in the next chapter.

4 Design Overview

The purpose of this chapter is to provide an overview of the general structure of the system. The modules constituting the system are briefly described along with their respective interfaces and the data exchanged between them. The following will also include a reflection on the two main strategies considered for 3D tracking of the fish. The term used for the overall system is the Three Dimensional Fish Tracker (3D-FisT).

4.1 Overall System Design

Two main strategies, inspired by M. Betke and Zheng Wu [26], where considered in the design of the 3D-FisT. In both cases, the goal is to generate tracks in 3D, but the approaches vary to a degree and the system designs differs significantly. The two approaches that have been considered are:

- **Reconstruction-tracking** - Triangulate the fish detections, then generate tracks using the estimated 3D positions.
- **Tracking-reconstruction** - Track the fish in 2D using the detections, then triangulate the entire tracks.

A comparison of the two approaches was conducted in [27], where it is primarily found that the preferred approach is highly dependent on the scenario. The tracking-reconstruction approach will for instance have difficulties in scenarios with a high occlusion rate as it may prove difficult to create consistent 2D tracks for each view. The reconstruction-tracking, on the other hand, is likely to perform better in this scenario as the creation of tracks is based on 3D positions instead of 2D positions.

The main drawback of the reconstruction-tracking is the prerequisite that the 3D positions can be reconstructed with sufficient accuracy. This includes associating the correct 2D detections across the two camera views. Failure to correctly associate detections across views is problematic as it may create false positives in the 3D space if all possible associations are considered. Considering only the most likely associations, on the other hand, may be problematic as well, as it may result in a number of false negatives in the 3D space if the association between views is incorrect. A positive feature of the reconstruction-tracking approach is that it is possible to use the estimated 3D information to annihilate false positive, such as detections located outside of the aquarium.

The association across views is less of an issue in the tracking-reconstruction approach as it is performed on a track-by-track basis, thereby allowing the system to consider the temporal aspect when aligning across views. This approach uses the information carried by a tracklet, which is richer in information than a single 2D point. However, when occlusion happens frequently, the tracklets may be short and fragmented in either one or both of the views and this creates complications for the association procedure.

Based on the presented pros and cons, the reconstruction-tracking approach is chosen above the tracking-reconstruction approach. It is expected that the amount of occlusion can be high in some situations, as the zebra fish demonstrate both shoaling and schooling behavior. Furthermore, there may be reflections in the aquarium walls or elements located outside the aquarium, that are difficult to distinguish from the real fish and therefore are detected as false positives. By estimating the 3D positions, it should be possible to annihilate such detections before the tracking.

4.1.1 3D-FisT Design

The design of the 3D-FisT system and each of the respective modules will be described in this section. The main focus will be on the detection, 3D reconstruction and tracking modules, as the image acquisition module has been kept as simple as possible. An overview of the modules and how they are linked can be seen in Figure 4.1.

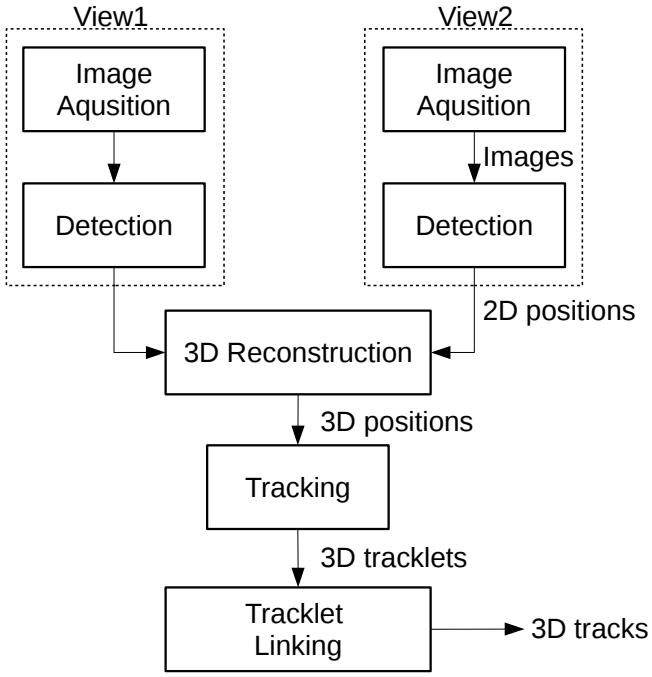


Figure 4.1: The reconstruction-tracking approach.

Image Acquisition

The first module in the system is the image acquisition module and it is responsible for receiving the temporally and spatially overlapping videos, breaking them down into images and passing them on to the detection module. As mentioned in Section *1.2 Properties and Limitations*, the videos must be temporally aligned before passing them into the system, as no module has been implemented to handle temporal synchronization in this iteration of the system.

- **Input:** Video recordings
- **Output:** Images

Detection

The detection module receives the images from the image acquisition module and processes them in order to detect the fishes. The main objective is to detect every fish, while keeping the amount of false positives as low as possible. It is expected that there will be false

positives, but this is tolerated as long as the amount of false negatives is kept at an absolutely minimum. The output from the detection module consists of a number of detections for each frame and camera.

- **Output:** .csv-file containing [Camera ID, Frame number, x, y] for each detection.

3D Reconstruction

When the 3D reconstruction module receives the 2D positions of the detections for both cameras, the main priority is to figure out which pairs of detections that matches between the cameras, in order to estimate the 3D positions of the fish. As the objects are under water, it is necessary for the 3D reconstruction module to take refraction into account. The output of the 3D reconstruction module is a range of estimated 3D positions for each frame.

- **Output:** .csv-file containing [Frame number, x, y, z] for each detection.

Tracking

The tracking module is responsible for associating the 3D positions in order to create 3D tracklets. The tracking module should be able to handle noisy data and outliers, as the output from the 3D reconstruction module can not be expected to be perfect. When the tracklets have been created, they are to be associated into complete tracks. The amount of final tracks should ideally be equal to the amount of fish present in the aquarium.

- **Output:** .csv-file containing [Frame number, Track ID, x, y, z] for each detection.

5 Fish Detector

The first step of tracking the fish in three dimensions is to detect the fish in both images. Dependent on the camera setup, the appearance of the fish may vary between the two images as described in Section *3.1.1 Visual Appearance of Zebra Fish*. In order to overcome this, the detector must be able to find the same spot on the fish independently of the viewing angle.

There exist several ways of detecting objects, however, not all may be equally suited to handle the given problem of detecting fish under water. In the following section, an overview of existing detection methods used for fish detection will be described. This will be followed by a short discussion of the suitability of the various methods with regard to the 3D-FisT system.

Related Work

There exists several examples of computer vision being used to streamline some of the tedious processes in the field of behavioral analysis of fish. An attempt of such is [28], where a system was designed to track Zebra fish in 2D using computer vision. The system is fairly simple as it relies on background subtraction and a user-defined threshold to extract fish BLOBs.

Background subtraction is a common method for detecting fish and has been used in the idTracker system[29] as well. The idTracker system is capable of tracking a variety of laboratory animals in 2D, including fish in low water.

Another common method is to use feature detection as in [9], where the detection of fish is handled by a BLOB detector relying on the determinant of the Hessian matrix (DoH) in a scale-space. This approach is somewhat similar to the interest point detector found in SURF[30], which utilizes the DoH and a scale-space as well. The authors are solely interested in detecting the heads of the fish as other parts of the fish, such as the tail, are deformable. The non-fish head detections are filtered away by imposing constraints such as the width of the head.

Several of the authors of [9] have continued their work on 2D tracking of fish in [24], with the focus being a more persistent tracking. The new approach continues to rely on DoH for the detection of the fish heads, but a new addition in this paper is the use of a Support Vector Machine (SVM) classifier to verify whether or not a detection is a fish-head. The SVM is pre-trained and is based on the Histogram of Oriented Gradient (HOG) descriptors, which encompasses both the appearance and shape of the fish heads.

Suitability of the Different Approaches

If the detection algorithm should be based on background subtraction, there are some problems that would need to be handled in order to get good detections. When the fish are swimming near the surface, the water level may become blurred, which makes the images captured from the camera placed above the aquarium unclear. Furthermore, preliminary tests have shown that when the camera is placed in front of the aquarium, parts of the zebra fish body become difficult to distinguish from the background.

Using a feature detector such as SURF could, on the other hand, have certain advantages as the setup is controlled and simple. The background is white and it is therefore expected

that most features will be found near the fish, independent of the viewing angle. This will be investigated and described further in the following sections as the detections will be based on keypoints found by the SURF detector.

5.1 Fish Detection using SURF

The fish wags its tail in order to create propulsion and move forward, which deforms the lower body to varying degrees, see Figure 5.1. The head of the fish is, on the other hand, pointing towards the travel direction and does not deform in the same manner as the lower body. The head is therefore considered a good candidate to use in the detection and recognition parts of the tracking system.

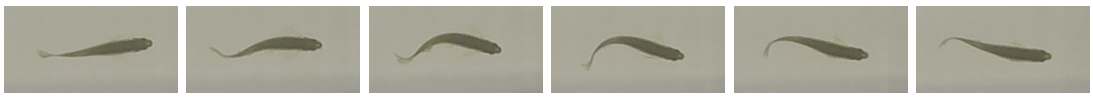
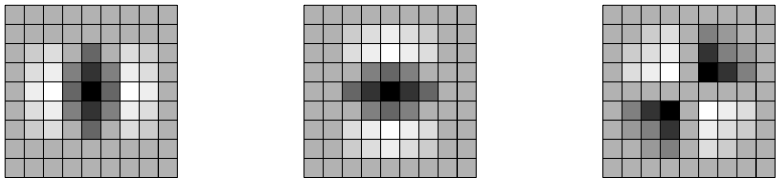


Figure 5.1: Six consecutive frames of a fish, wagging its tail to create propulsion.

Beside having a consistent form independent of how the fish is moving, the head is also shaped in a way that creates local second order partial derivative extrema points near the center of the head independent of the viewing angle. The Speeded Up Robust Features (SURF) algorithm is therefore an obvious choice to use as a feature detector. It uses an approximation of the Hessian matrix consisting of the second order partial derivatives, to detect interest points [30]. The Hessian matrix for a given point, $p = (x, y)$, is given by

$$\mathbf{H}(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix}, \tag{5.1}$$

where $L_{xx}(p,\sigma)$ is the convolution of a Gaussian second order partial derivative, $\frac{\partial^2 g(\sigma)}{\partial x \partial x}$, in a given image at scale σ. The same applies to $L_{yy}(p,\sigma)$ and $L_{xy}(p,\sigma)$. As Gaussian functions are continuous, they have to be cropped and discretized for practical reasons and this gives the type of filters seen in Figure 5.2. The presented example corresponds to a scale of $\sigma = 1.2$.



(a) $L_{xx}(p,\sigma)$ **(b)** $L_{yy}(p,\sigma)$ **(c)** $L_{xy}(p,\sigma)$

Figure 5.2: Cropped and discretized Gaussian filters.

The determinant of the Hessian matrix is calculated and used to find local maxima in the image. The determinant is given by

$$det(H(x,y)) = f_{xx}f_{yy} - f_{xy}^2, \tag{5.2}$$

where f_{xx}, f_{yy} and f_{xy} are the second order partial derivatives. If a stationary point is found where $f_x(x,y) = f_y(x,y) = 0$, the second partial derivative test reveals whether the given point is a minimum, maximum or saddle point. The point is a local maximum if both expressions in Equation 5.3 is true.

$$det(H(x,y)) > 0 \quad \text{and} \quad f_{xx} < 0. \tag{5.3}$$

Integral Images and Box Filters

The SURF algorithm uses integral images for multiple purposes, including detection of interest points. Every point in an integral image consists of the sum of all points in a rectangle from origo to the given point. An integral image can be calculated by running through the image once using the following equation

$$I(x,y) = i(x,y) + I(x,y-1) + I(x-1,y) - I(x-1,y-1) \tag{5.4}$$

where I is the integral image, (x,y) is the point to be evaluated and i is the original image. The sum of any rectangle can then be calculated by

$$I(x',y') = A + D - B - C, \tag{5.5}$$

where A, B, C and D are the corners of the rectangle as shown in Figure 5.3.

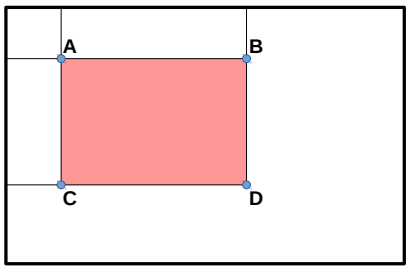
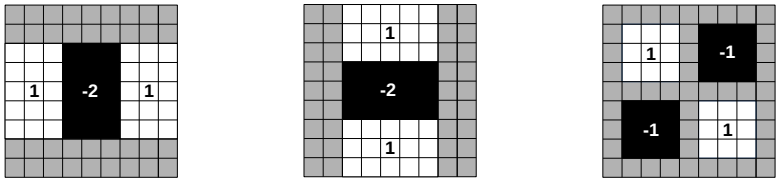


Figure 5.3: The sum of any rectangle in an integral image can be calculated using the values of its four corners.

To minimize the necessary processing power needed to find interest points in the image, box filters are used instead of the second order derivative Gaussian filters. The box filters are approximations of the Gaussian filters that have the advantage that they can be calculated as integrals. The box filters are denoted D_{xx}, D_{yy} and D_{xy} and an illustration of the filters at a scale corresponding to $\sigma = 1.2$ can be seen in Figure 5.4.



(a) D_{xx} **(b)** D_{yy} **(c)** D_{xy}

Figure 5.4: Box filter approximations of the discretized Gaussian filters.

As the fish are able to move in three dimensions, the detector must be scale invariant. The SURF algorithm again utilizes the fast computation of integral calculations by adjusting the size of the box filters to approximate varying scale spaces. An increase in the dimensions of a box filter corresponds to a lower spatial resolution of the input image and vice versa.

Orientation

The fish tend to change directions frequently, which means that the descriptor must be orientation invariant. SURF uses the Haar wavelet response of the horizontal, d_x, and vertical, d_y, direction to determine the orientation of the interest point. The Haar wavelet responses can also be computed quite fast as they are calculated using integral images. The Haar wavelet filters are illustrated in Figure 5.5.

(a) Vertical Haar wavelet filter, d_y. **(b)** Horizontal Haar wavelet filter, d_x.

Figure 5.5: Illustration of the Haar wavelet filters.

The horizontal and vertical response vectors are added and represented in a space with the horizontal and vertical magnitude on the abscissa and ordinate, respectively. The orientation of the interest point is estimated using a sliding window technique, where the sum of the responses in a window, corresponding to an angle of $\frac{\pi}{3}$, is found. The longest resulting vector is chosen as the orientation of the interest point as illustrated in Figure 5.6.

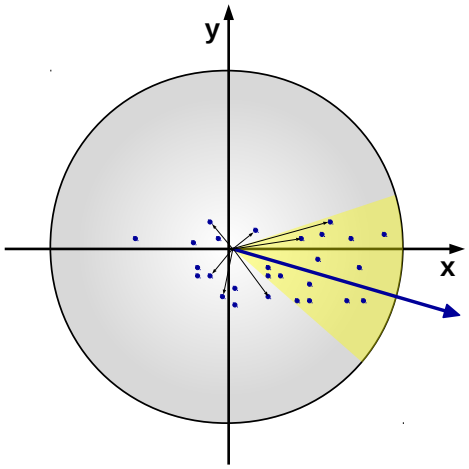


Figure 5.6: The response vectors in the window are summed to create an orientation vector.

Descriptor

The SURF descriptor is used to describe the point of interest found by the SURF keypoint detector. A square region corresponding to 20ϕ and with an orientation corresponding to the keypoint orientation is constructed. This region form the basis of the descriptor, which subsequently is split into 4x4 square sub-regions.

For each of the sub-regions, several features are produced with basis in the Haar wavelet responses once again. The vertical and horizontal responses, d_x and d_y, respectively, are calculated for multiple evenly distributed points for every sub-region. The responses are, once again, weighted with a Gaussian kernel centered at the keypoint, before they are summed up for each sub-region. Beside the regular responses, the sum of the absolute values, $|d_x|$ and $|d_y|$, are also computed. Therefore, each sub-region consists of a feature-vector given by

$$v_k = [\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|], \tag{5.6}$$

where $k = \{1..16\}$ are the given sub-regions, which produces the final feature descriptor of length 64. The descriptors are used to determine whether the keypoint is a fish head or noise. This is a classification problem and it will be described in more details in the following section.

5.2 Classifying Fish Head Descriptors

The keypoints found by the SURF detector are based on the DoH, which means that interest points will be found in a lot of places in addition to the fishes, unless the setup is completely empty. Furthermore, a keypoint found on a fish, should be considered false-positive, unless it is located directly above the head, as this has been chosen as the point of reference. Even though the setup is considered reasonable simple, the edges of the aquarium may also be considered good candidates for the SURF detector. All these possible pitfalls, calls for a classifier to minimize the false-positive detections and in order to keep the solution simple, the problem can be boiled down to be that a given descriptor either represents a fish head or not. A binary classifier, such as Support Vector Machine (SVM), should therefore be sufficient to use for eliminating potential false-positive detections.

SVM is a supervised learning model trained on labeled data. The training consists of fitting a hyperplane, that creates the largest separation between the two classes. The training data is therefore a crucial part of the SVM and is to some degree responsible for how well the SVM is able to classify unlabeled descriptors. There is no perfect amount of data needed in order to get a well-functioning binary SVM classifier, however, in order to avoid bias and overfitting, the data should consists of descriptors from different fish and sequences.

Labeled Training Data

The labeled data consists of SURF descriptors from manually chosen templates of fish heads and noise. The data is taken from views, where the camera is placed above the aquarium only. Different data-variations have been tested, where the data has been from a single video sequence, multiple sequences, varying lighting conditions and with varying ratios between

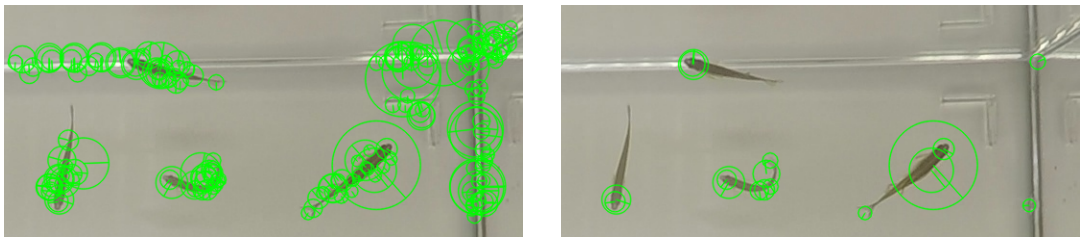
fish head and noise descriptors. An example of the labeled data can be seen in Figure 5.7, where the yellow dashed square represents the region of the SURF descriptor.

(a) Fish head. **(b)** Noise.

Figure 5.7: SVM training data. The dashed yellow squares illustrates examples of the templates used to generate SURF descriptors.

The SVM model, which seems to be the most robust, has been trained on 1,424 fish head descriptors and 518 noise descriptors, which is ≈ 2.75 times more fish heads than noise. Half of the fish heads were taken from the D14 dataset and the other half from the top view of the AB-1080-60-6f dataset, which are both described in Section *3.1 Description of Dataset*. The two dataset consist of different fish, camera, resolution, aquarium and lighting conditions, which is expected to make the model more robust. The noise, on the other hand, is only taken from the D14 dataset, as preliminary tests showed that when the SVM model was trained on noise descriptors from the AB-1080-60-6f dataset, the classifier performed poorly.

The SVM model and associated training algorithm used in the system is the `LinearSVC` class from the `sklearn.svm` module, which is well tested and documented. An example of classification performed by the above mentioned SVM model can be seen in Figure 5.8, where the left image shows the detections found by the SURF detector and the right image shows the keypoints classified as fish heads.



(a) SURF keypoints. **(b)** Keypoints classified as fish heads.

Figure 5.8: Keypoints before and after SVM classification.

As seen in the example, it is not all detections that are correctly classified. Occurrence of false-positive classifications is not crucial, however, it is undesirable if a keypoint located directly on top of a fish head is classified as being noise. In order to measure the performance of the SVM model, a precision-recall test has been conducted.

The descriptors of 50 keypoints per image have been gathered from a total of 20 images from multiple dataset. The images were picked with an interval of 100 frames and a total of 1,000 descriptors were collected. The descriptors were manually labeled as either fish heads or noise, making errors a possibility. However, the error caused by the manual labeling is assumed to be small and have no significant impact on the results. The predictions can be seen in Table 5.1 along with the precision and recall.

	P	**N**	**TP**	**TN**	**FP**	**FN**	**Precision**	**Recall**
02maj-t1	244	756	238	692	64	6	0.788	0.975
02maj-t2	247	753	230	661	92	17	0.714	**0.931**
15maj-2k60fps	43	957	43	953	4	0	0.915	1.000
15maj-720narrow	228	772	216	570	202	12	0.517	0.947
						Average	**0.734**	**0.963**

Table 5.1: Precision and recall for the SVM model. **P**=Positive **N**=Negative **T**=True **F**=False.

The precision, or positive predictive value (PPV), is given by

$$PPV = \frac{TP}{TP + FP}, \tag{5.7}$$

and it indicates the possibility for a predicted fish head descriptor to really be a fish head descriptor and not noise. The average PPV for the test cases is relatively low, which means that a fair amount of false positives are detected. Examples of the false positive classifications can be seen in Figure 5.9. A general tendency of the wrongly classified descriptors, is the presence of a dark elongated region surrounded by a brighter background. The leftmost image is a reflection of the LED light and is clearly miss-classified, however, the two rightmost images are difficult to identify due to low spatial resolution and may be fish heads, which have been miss-labeled. However, the relatively low precision is not crucial, as the SVM model is only used as the first step in annihilation of false positive detections.

Figure 5.9: False positive classifications.

A high recall, or true positive rate (TPR), is, on the other hand, essential for the system to be able to perform well. The recall indicates how likely it is that fish head descriptors are retrieved in the first place. It is given by the following equation

$$TPR = \frac{TP}{TP + FN}. \tag{5.8}$$

If a relevant fish head descriptor is missed, this leads to less information that can be used to assemble the tracks and thereby introduce potential errors. The lowest and average measured TPR is 0.931 and 0.963, respectively, which is not perfect, but considered satisfactory. Some of the classifications that impact the recall negatively can be seen in Figure 5.10.

A tendency among the false negative classifications, is that they have a relatively low resolution or contain noisy elements. It is expected that some false-negative classifications occur when the fish are moving near edges, near other fish, are moving at high speeds or similar.

The given precision-recall ratio is accepted as a large part of the noise detections are eliminated, while nearly all the fish head descriptors are preserved and classified correctly. Other parts of the system are expected to eliminate the remaining false-positive detections.

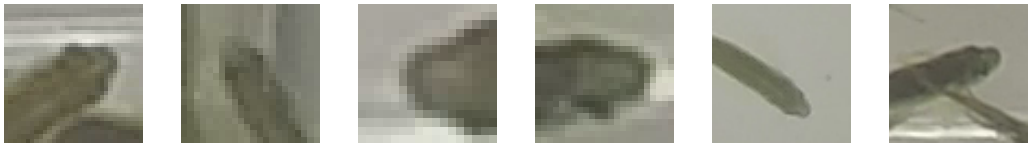


Figure 5.10: False negative classifications.

5.2.1 Non-Maximum Suppression

Even though a large part of the noise detections have been removed, there may still be multiple detections of the same fish located on top of each other. Non-maximum suppression (NMS) is used to reduce this in an attempt to end up having only one detection present for every fish head that has been located. The implemented NMS algorithm is loosely based on the method described in [31].

As a SURF descriptor is used to describe the interest points found on the fish heads, the region of interest around the point is determined by the scale of the descriptor. This scale determines the amount of pixels that are processed and several descriptors can be located near each other creating an overlap. Examples of overlapping descriptors can be seen in Figure 5.11, where the red squares illustrate the regions covered by the descriptors.

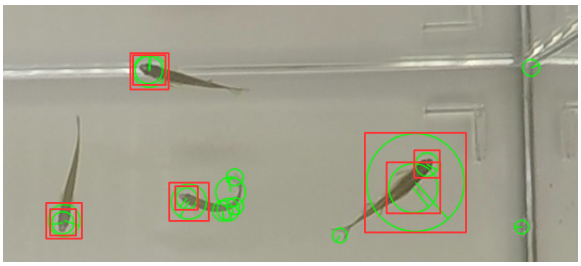


Figure 5.11: Partially and fully overlapping descriptors.

The interest points found by SURF is represented by a position, rotation and scale. The position and scale can be used to calculate the area of the descriptors and determine the amount of overlap between them.

A significant part of the algorithm is to dictate in which order the descriptors are investigated. One solution could be to sort the descriptors based on the y-coordinate of the lower right corner, as suggested in [31]. However, preliminary tests have shown that the algorithm tends to prefer large descriptors, which are not necessary located on top of the fish head. This problem is illustrated in Figure 5.11, where the large descriptor on the rightmost fish would be chosen, even though smaller and more precise candidates are present.

Instead, the area is calculated for every descriptor after which they are sorted in ascending order, with respect to their area. The algorithm is implemented in a greedy manner, where the first descriptor is picked and compared against the rest. If there are cases where the overlap exceeds a given threshold, those candidates are removed from the list, before preceding to the investigate the next descriptor in line. This will continue until every descriptor has been either investigated or annihilated. An example can be seen in Figure 5.12, where the overlapping descriptors have been removed by the use of NMS.

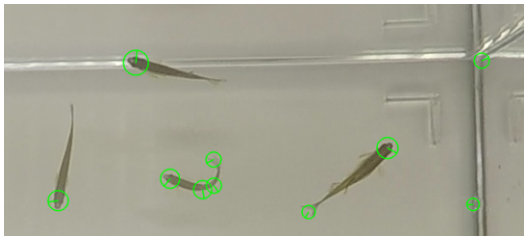


Figure 5.12: Overlapping descriptors have been removed.

	Positives	**TP**	**FP**	**FN**	**Precision**	**Recall**
Cam1	20973	13465	7508	528	0.64	0.96
Cam2	20914	13199	7715	794	0.63	0.94
Top	59955	19941	40014	447	0.33	0.98
				Average	**0.53**	**0.96**

Table 5.2: The precision and recall for the detection module test on detections from both cameras of the AA-1520-60-7f dataset and the top camera of the AB-1080-60-6f dataset.

5.3 Evaluation of the Detection Module

The performance of the detection module is evaluated by processing annotated data. The relationship between the detections and the annotations is investigated in order to produce the precision and recall. However, the annotations and detections are not expected to have the exact same positions and a maximum range is therefore introduced in order to identify situations where a detection is either missing or off-point.

The precision and recall are measured for the entire detection module, which includes detection of SURF keypoints, SVM classification and NMS. If a detection is within the allowed maximum range it is considered a true positive and if no detections are within the range of an annotated fish, it is considered a false negative. Detections that are outside the maximum range are all considered false positives.

The annotated sequences used for the module test consists of the first half of the AA-1520-60-7f recording and the top camera recording of the AB-1080-60-6f dataset. The precision and recall have been measured for the three sequences with regard to the detection rate and the results can be seen in Table 5.2. The maximum range have been set manually to 40 pixels, which is approximately twice the size of a fish head.

The recall of the three sequences are all relatively high and they are consistent with the results gathered in the SVM-test. The precision, on the other hand, varies a lot between the two dataset. In the high resolution dataset, the precision is ≈ 0.64, while it is only around half as good in the AB-1080-60-6f sequence. One of the main reasons for this low precision is assumed to be because of the specific setup, which results in the four bottom corners of the aquarium to be mistaken for fish heads continuously. In most of the frames, the four corners are detected as false positives, which increases the total amount significantly.

This type of false positives can occur in different types of setups, and can be caused by poor lighting conditions, fish feces and more. They are typically located in the same spot for every frame, which makes them easy recognizable. Nothing will be implemented to remove these types of false positives in the present iteration of the system.

6 3D Reconstruction

The following details the approach used to reconstruct the 3D positions of the fish based on their 2D positions in the two cameras. How the fish are detected is described in 5.1 and will not be covered in further details. Furthermore, the detected 2D positions are assumed to be temporally aligned, as this is outside the scope of this thesis.

The chapter is commenced by an overview of existing work related to 3D reconstruction while accounting for refraction. The overview will then be concluded with a discussion of the suitability of the different approaches with relation to this project. One of the approaches is selected and its implementation is described in more details. The chapter ended with a test of the selected approach where it is compared against other plausible approaches for 3D reconstruction.

6.1 Related Work

The following describes some of the existing approaches for dealing with refraction when capturing objects using multiple cameras. The different ways of handling refraction can generally be divided into two categories; approaches which directly tries to account for the physics of refraction (e.g. using Snell's law) and approaches which indirectly tries to account for refraction (e.g. relying on the SVP camera model to absorb the errors).

Indirect Approaches

The following describes some of the indirect approaches for counteracting the error caused by refraction. All of these approaches rely on the SVP camera model, which is advantageous due to its ease of use as 3D reconstruction relying on the SVP model is well documented and widely supported in many toolboxes. Its popularity is not without reason, as it is also simple to deal with mathematically, mainly because the camera is described using a projective transform, which is linear.

Examples of using the SVP model for underwater purposes can be found in [32] and [33]. The former does, however, try to account for the problem of refraction by mounting a dome port instead of a flat port in front of the camera. Doing so essentially focuses the refracted rays into a single viewpoint, as shown in Figure 6.1.

The corrective effect of the dome port is also quite clear when contemplating Snell's law from earlier (Equation 2.9), as the incidence angle will effectively be zero due to the surface normals of the dome port. The drawback of such a solution is the precision required to align the dome and the camera.

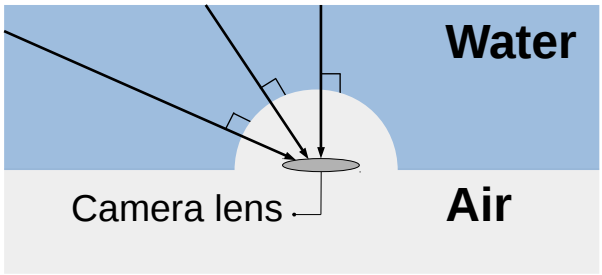


Figure 6.1: Illustration of how light rays are bent in a dome shaped interface.

The main problem of the approach in [33] is that the base assumption of SVP (i.e. single viewpoint) is wrong due to the refraction, as discussed earlier. It can, however, be argued that the error caused by refraction can be absorbed, to some extend, by the focal length adjustment and radial distortion correction commonly found in SVP models.

This idea is the scope of [34], which explores the performance of the SVP camera model for 3D underwater reconstruction. The paper tests different configurations of the SVP model with/without focal length adjustment and radial distortion correction. It is found that the adjustment of the focal length has the biggest impact on reducing the error caused by refraction.

A variation of this idea is to use multiple localized focal length adjustments instead of a single global focal length adjustment. An example of such can be found in [35] which utilizes a pixel-wise varifocal camera model. Another example is [36] which segments the scene into smaller partitions and calibrates localized SVP camera model for each partition.

Physic-based Approach

The second group of approaches are the ones which actively seek to counteract the refraction error by using Snell's law.

A straight-forward example of such an approach can be found in [37], where two cameras are used to track a single fish in 3D. The cameras are modeled using the SVP camera model and their intrinsic parameters have been found through calibration in air. The extrinsic parameters of the camera are found in relation to the corners of the aquarium.

The intrinsic and extrinsic parameters are then used to project two rays, one for each camera, into the aquarium. Knowledge of the aquarium's corners are used to calculate the intersection between the rays and the aquarium. The refracted rays originating from the intersections are calculated and used for triangulation.

A somewhat similar approach is described in [38], where four cameras are used to track a single seahorse in 3D. This approach relies on the same combination of ray tracing, Snell's law and the SVP camera model. However, it differs in the sense, that it includes an additional step where the intrinsic and extrinsic camera parameters are optimized. The position and orientation of the refractive interface (i.e. the side of the aquarium) is also optimized during this step. The optimization is based on the known length of a calibration frame.

The approach described in [39] performs a similar optimization step to refine the parameters used during ray tracing as well.

Suitability of the Different Approaches

One of the indirect approaches, relying on the SVP camera model, may be plausible to use, especially if expanding to one of the approaches relying on localized focal length adjustment. However, all these methods essentially tie the intrinsic camera parameters (namely the focal length) to a specific scene. This means that the calibration procedure, required to find the correct parameters for both focal length and radial distortion correction, will have to be repeated each time the scene changes, e.g. when moving the camera and/or aquarium.

This is a disadvantage, as it should be sufficient to only identify the intrinsic parameters for the cameras once and changes in the scene would normally only require an update of the extrinsic parameters. This decoupling between intrinsic and extrinsic parameters no longer

exists when relying on the SVP camera model to absorb the refraction error and this makes the setup a lot more tedious to deal with. It can also be argued that repeatability may be an issue if the calibration procedure has to be repeated before each test.

The advantage of the more direct approaches based on ray tracing is that the intrinsic parameters are independent of the aquarium. It is hence only necessary to calibrate the camera once and the calibration can be performed in air. This benefit does come at the cost of an increased computational overhead, consisting of the ray tracing and of calculating the refracted rays. However, it can be argued that this overhead is not of any importance, as there are no real time requirements for this project.

Based on the above considerations it is found prudent to focus on a solution based on ray tracing. Mainly as it avoids tying the intrinsic parameters to a specific scene, making the final setup less tedious to operate. Another contributing factor is that the SVP camera model can still be used to some extend, which is considered advantageous due to its widespread use.

6.2 Ray-tracing using Snell's Law

The implementation of the selected approach is mainly based on [37], where ray tracing has been used in combination with Snell's law to account for the refraction. The main steps of the approach is outlined in Figure 6.2 and are as follows:

1. Estimate the extrinsic and intrinsic parameters of both cameras.
2. Project the 2D detections for each camera into rays.
3. Calculate the point of intersection between the respective rays and the water.
4. Calculate the refracted rays originating from the points of intersection.
5. Triangulate the 3D position of the fish using the refracted rays.

How the different steps are performed is described in more detail in the following sections.

6.2.1 Step 1 - Camera Calibration

The intrinsic parameters, K, are found for each camera through calibration in air. The calibration is performed using OpenCV's `cv2.calibrateCamera(...)` method, which returns both the intrinsic parameters and lens distortion coefficients for each camera. The method is based on [40] and the Matlab Camera Calibration Toolbox [41]. A checkerboard is used for the calibration and approximately 25 images are captured for each camera.

Identifying the extrinsic parameters for each camera is essentially a matter of solving the Perspective-n-Point (PnP) problem, which seeks to determine both the orientation, R, and position, t, of the camera using n 3D-2D point correspondences. This is achieved using the OpenCV method `cv2.solvePnP(...)` with the `CV_P3P` flag, which solves the Perspective-3-Point (P3P) problem, i.e. $n = 3$. The method is based on [42] and requires a total of four correspondences, as the fourth point is used to solve ambiguities in the solution.

The four correspondences needed to solve the P3P problem is found through manual annotation of the intersections between the water surface and the corners of the aquarium.

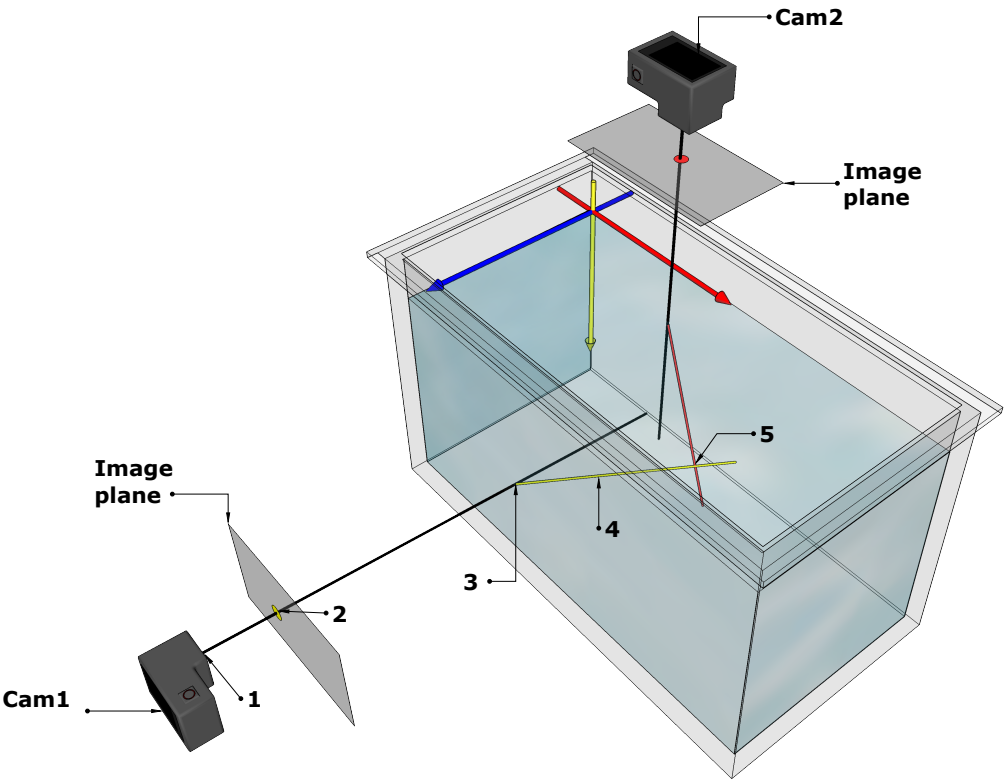


Figure 6.2: The main steps of the implemented 3D reconstruction approach. The approach is based on ray tracing and utilizes Snell's law to account for refraction.

The location of these four intersections are shown in Figure 6.3, along with origo and the chosen directions for both the $x-$ and y-axis. Note that a right-handed coordinate system is used and the z-axis will hence point downwards into the water.

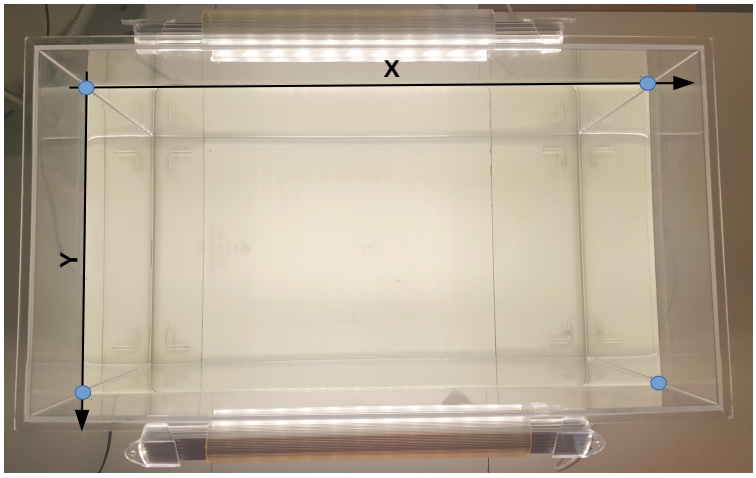


Figure 6.3: The four intersections between the aquarium corners and the water surface used in solving the P3P problem.

The above approach is mainly chosen because it is easy to manually annotate these points. Another advantage of this approach is that these four intersections will always exist when using a rectangular tank with water in it. The drawback, on the other hand, is that this approach may be more susceptible to noise, as only four points are used.

An alternative could be to solve the PnP problem, where n is the number of intersections which can be found on a dedicated calibration object, e.g. a checkerboard. Such a solution would likely be more resistant against noise, as more points are used. The main drawback of this method is that the extrinsic parameters, and hence the final reconstructed 3D positions, will be in relation to the coordinate space spanned by the checkerboard. The 3D positions may hence be less relatable than the P3P approach where everything is in relation to the corners of the aquarium.

Mapping from a space spanned by the checkerboard to a space spanned by the aquarium corners is of course a possibility, as it would just be a matter of an affine transformation, i.e. translation, rotation and possibly scaling. Doing so would, however, still result in the problem of precisely annotating the intersections illustrated in Figure 6.3.

Yet another possibility could be to utilize a checkerboard fitted to the size of the tank, but this would require a custom-made checkerboard for each tank. Relying on different calibration objects does also introduce a minor risk of user mistakes, where the user might forget to include the required calibration object.

6.2.2 Step 2 - Projecting 2D Point into a Ray

The 2D coordinates of the detected fish are back projected into a ray in the world space coordinate system. This step is repeated for each camera with each of the rays being characterized as

$$r(\lambda) = \lambda \bar{r} + r_0, \tag{6.1}$$

where $\bar{r}$ is the direction of the ray, r_0 is a point on the ray and λ defines all positions along the ray. The direction vector, $\bar{r}$, is found by

$$\bar{r} = R^{-1} K^{-1} \begin{bmatrix} u & v & 1 \end{bmatrix}^T, \tag{6.2}$$

where R^{-1} is the inverse rotation matrix, K^{-1} is the inverse intrinsic camera matrix and $\begin{bmatrix} u & v \end{bmatrix}^T$ are the image coordinates of the 2D point to back project into a ray.

The position of the camera, calculated from Equation 2.8 (Section 2.3.1), serves as r_0 as the back projected ray must pass through the center of the camera.

It should be noted that the OpenCV method `cv2.undistortPoint(points, cameraMatrix, distortionCoefficients)` is used when calculating $K^{-1} \begin{bmatrix} u & v & 1 \end{bmatrix}^T$ as it also accounts for lens distortion.

6.2.3 Step 3 - Identifying the Plane-Ray Intersection

Identifying the point of intersection between plane and a ray is essentially a matter of identifying λ, such that $r(\lambda) = p$, where p is the set of points in the plane. Such a plane can be defined as

$$(p - p_0) \cdot \bar{n} = 0, \tag{6.3}$$

where $\bar{n}$ is the plane normal and p_0 is a point in the plane. Combining Equation 6.1 and Equation 6.3 and isolating λ yields

$$\lambda = \frac{(p_0 - r_0) \cdot \bar{n}}{\bar{r} \cdot \bar{n}}. \tag{6.4}$$

Solving for λ and inserting the result into Equation 6.1 then yields the point of intersection, I, between the plane and ray

$$I = \lambda \bar{r} + r_0. \tag{6.5}$$

The above calculations require knowledge of the plane normal, $\bar{n}$, which can be found as $\bar{n} = \bar{v_1} \times \bar{v_2}$. The vectors $\bar{v_1}$ and $\bar{v_2}$ being two vectors in the plane, which can be found from three non-collinear points in the plane. However, due to step 1, the plane normal to the water surface can always be assumed to be $\bar{n} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$. This is due to the choice of aligning both the x- and y-axis with the plane of the water surface.

6.2.4 Step 4 - Calculating the Refracted Rays

The refraction of the ray $r(\lambda)$ at the intersection with the water is calculated using Snell's law. A refractive index of $n_1 = 1.0$ for air and $n_2 = 1.33$ for water is used during this calculation. The refraction caused by the glass/acrylic glass can be ignored, as the cameras are mounted pointing downwards into the aquarium.

The following describes the necessary steps to calculate the refracted ray, r', of an incoming ray, r, and is mainly based on [43]:

1. Calculate the cosine of θ_1 as

$$cos(\theta_1) = -\bar{n} \cdot \bar{r}, \tag{6.6}$$

where θ_1 is the angle between the incoming ray, $\bar{r}$, and the water surface normal, $\bar{n}$, as shown in Figure 6.4.

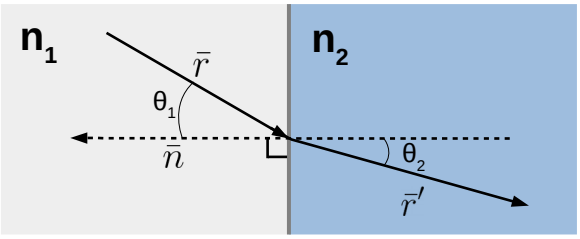


Figure 6.4: Calculation of the refracted ray, $\bar{r}'$, using Snell's law.

2. Calculate the cosine of θ_2, with θ_2 being the angle between the refracted vector, $\bar{r}'$, and the water surface normal, $\bar{n}$, as

$$cos(\theta_2) = \sqrt{1 - \left(\frac{n_1}{n_2}\right)^2 (1 - cos(\theta_1)^2)}. \tag{6.7}$$

Equation 6.7 is found by reformulating Snell's law (Equation 2.9 in Section 2.3.2):

$$\frac{sin(\theta_1)}{sin(\theta_2)} = \frac{n_2}{n_1} \rightarrow sin(\theta_2)^2 = \left(\frac{n_1}{n_2}\right)^2 sin(\theta_1)^2,$$

while replacing all sines such that

$$sin(\theta)^2 = 1 - cos(\theta)^2,$$

which is based on the trigonometric identity

$$sin(\theta)^2 + cos(\theta)^2 = 1$$

3. The final refracted vector, $\bar{r}'$, is then calculated as

$$\bar{r}' = \left(\frac{n_1}{n_2}\right)\bar{r} + \left(\frac{n_1}{n_2}cos(\theta_1) - cos(\theta_2)\right)\bar{n} \tag{6.8}$$

6.2.5 Step 5 - Triangulation using Rays

The final step is to triangulate the 3D position of the fish using the refracted rays, $r_1(\lambda)'$ and $r_2(\lambda)'$, for each camera. The refracted rays still adheres to Equation 6.1 and are formed using the refracted direction vector, $\bar{r}'$, from Equation 6.8 along with the position of the plane-ray intersection, I, from Equation 6.5. The intersection point, I, for each ray is used as the refracted rays will have to pass through this point.

The triangulation method employed in this thesis is commonly known as the midpoint algorithm. The idea is to identify the vector $\bar{m}$ between the two rays, $r_1(\lambda)'$ and $r_2(\lambda)'$, such that the length, $\|\bar{m}\|$, is minimized. The final 3D position is then found as the midpoint of the vector $\bar{m}$, hence the name of the algorithm.

Other ways of triangulating 3D positions do exists, e.g. the methods mentioned in [44]. The midpoint algorithm is, however, chosen over the others, as it operates on rays by default. Many of the other approaches utilizes the intrinsic and extrinsic camera parameters to triangulate a point, while minimizing the reprojection error instead. Such approaches assume that the pinhole camera model holds, i.e. a single viewpoint exists, which is not the case when dealing with refraction as discussed earlier.

The midpoint algorithm is based upon the observation that the length, $\|\bar{m}\|$, must be at its minimum when $\bar{m}$ is perpendicular to both rays. The main idea is hence to identify a vector, $\bar{m}$, such that

$$\begin{aligned} \bar{m} \cdot r_1' &= 0 \\ \bar{m} \cdot r_2' &= 0, \end{aligned} \tag{6.9}$$

where r_1' and r_2' are the direction vectors of the two refracted rays.

The vector, $\bar{m}$, is found by calculating the vector's start position, M_1, and end position, M_2, along the refracted rays. These two points can be calculated by

$$M_1 = I_1 + \bar{r}_1' \frac{-(\bar{r}_1' \cdot \bar{r}_2')(\bar{r}_2' \cdot I_1\bar{I}_2) + (\bar{r}_1' \cdot I_1\bar{I}_2)(\bar{r}_2' \cdot \bar{r}_2')}{(\bar{r}_1' \cdot \bar{r}_1')(\bar{r}_2' \cdot \bar{r}_2') - (\bar{r}_1' \cdot \bar{r}_2')(\bar{r}_1' \cdot \bar{r}_2')} \tag{6.10}$$

and

$$M_2 = I_2 + \bar{r}_2' \frac{(\bar{r}_1' \cdot \bar{r}_2')(\bar{r}_1' \cdot I_1\bar{I}_2) - (\bar{r}_2' \cdot I_1\bar{I}_2)(\bar{r}_1' \cdot \bar{r}_1')}{(\bar{r}_1' \cdot \bar{r}_1')(\bar{r}_2' \cdot \bar{r}_2') - (\bar{r}_1' \cdot \bar{r}_2')(\bar{r}_1' \cdot \bar{r}_2')}, \tag{6.11}$$

where $\bar{r}_1'$ and $\bar{r}_2'$ are the direction vectors of the refracted rays while I_1 and I_2 are the points of refraction.

The final 3D position, p', of the triangulation process, is then calculated as

$$P = \frac{(M_1 + M_2)}{2}. \tag{6.12}$$

The proof of Equation 6.10 and Equation 6.11 can be found in [45].

6.3 Spatial Association

The final stage of the 3D reconstruction module is to perform a spatial association, i.e. identifying which detection from one camera matches which detection from the other camera. The idea is essentially to reduce a given amount of detections to a few associated pairs.

The implemented approach for spatial association consists of the following sub-stages:

1. Remove 2D detections outside each cameras region-of-interest (ROI), as objects outside this region are unlikely to be fish. The ROI for each camera is extracted from the four manually annotated intersections between aquarium corners and the water surface (Section *6.2.1 Step 1 - Camera Calibration*).

2. Reconstruct the 3D position of any possible combination of detections between the two cameras. The result will be n^2 3D positions, with n being the maximum number of 2D detections present in any of the two cameras.

3. Remove all of the reconstructed 3D positions which resides outside of the aquarium. This is possible as the dimensions of the tank are assumed to always be known when dealing with a controlled test environment.

4. Identify the 3D points with the lowest error during reconstruction and discard the rest.

   The error for each 3D point were initially measured as the euclidean distance between the rays, i.e. $\|m\|$ from earlier (Section *6.2.5 Step 5 - Triangulation using Rays*). Preliminary tests did however indicate that this metric was not very reliable. The reprojection error is therefore used as a measure of the error associated with each of the remaining 3D points.

   The reprojection error for each 3D point is calculated as:

   $$|p_1 - p_1'| + |p_2 - p_2'| \tag{6.13}$$

   where p_1 and p_2 are the initial detections used for reconstruction of the 3D point P, while p_1' and p_2' are the reprojection of P onto the image plane of each of the two cameras. How this reprojection is performed is described in more detail in the next section.

   The best 3D points are selected such that the total reprojection error is minimized. This is done using the Munkres assignment algorithm, which is described in more details in Section *7.2.1 Global Minimization using Munkres*. The algorithm ensures that a 2D detection in any of the cameras is only used in the reconstruction of a single 3D point.

6.3.1 Reprojection: Accounting for Refraction

The following describes how reprojection is done, i.e. projecting a 3D point onto a cameras image plane, while accounting for refraction. Reprojection is commonly done by applying the inverse of the intrinsic, K, and extrinsic, $\begin{bmatrix} R & | & T \end{bmatrix}$, camera matrices to the 3D point. This is unfortunately not sufficient when dealing with refraction as the light does not travel in a straight line between the 3D position, P, and the camera position, S, as illustrated in Figure 6.5.

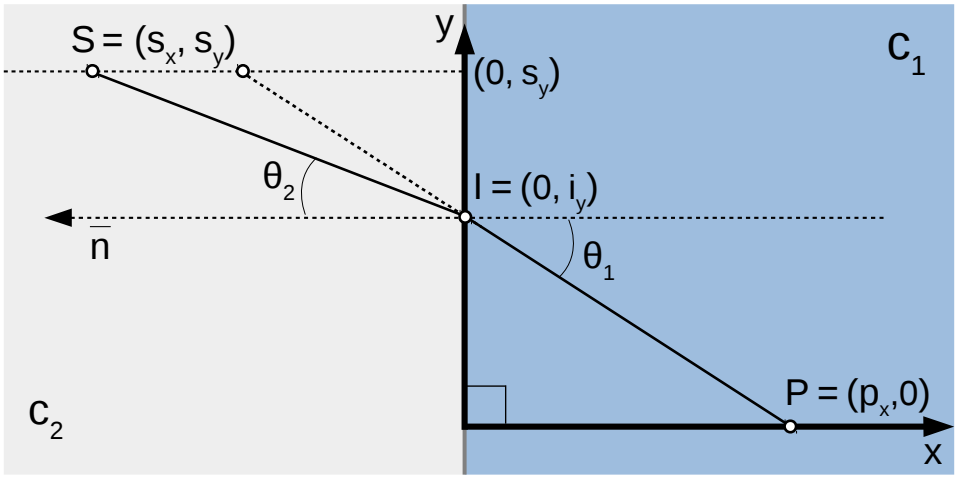


Figure 6.5: 2D view of a ray travelling between a known position, P, and the camera, S. The ray bends at the intersection point, I, due to refraction.

However, it is possible to reproject 3D points under the influence of refraction by identifying the point of refraction, I, and use this position as the basis for the reprojection. The benefit of doing so is that the light will travel in a straight line between I and the camera, S, making it possible to perform reprojection using traditional methods.

The point of intersection, I, for a given 3D point and for a given camera can be found by solving a fourth order polynomial as described in [46]. The main idea is to identify the point, I, which minimizes the distance that the light will have to travel from the 3D position to the camera:

$$\frac{\bar{PI}}{c_1} + \frac{\bar{IS}}{c_2}, \tag{6.14}$$

where c_1 and c_2 are the speed of light in water and air, respectively.

The above observation can be reformulated into the distance function given by

$$dist(S,P) = \sqrt{p_x^2 + i_y^2} + \frac{c_1}{c_2}\sqrt{{s_x}^2 + (s_y - i_y)^2}, \tag{6.15}$$

where i_y and s_y are the only unknown variables, as s_y can be found by S projecting onto the y–axis using $\bar{n}$.

The minimum of this distance function, in relation to i_y, can then be found by solving the fourth order polynomial

$$Ni_y^4 - 2Ns_y i_y^3 + (Ns_y^2 + \frac{s_x^2}{r^2} - p_x^2)i_y^2 + 2p_x^2 s_y i_y - p_x^2 s_y^2 = 0, \tag{6.16}$$

where $N = \frac{1}{r^2} - 1$ and $r = \frac{c_1}{c_2} = \frac{n_1}{n_2}$.

The solution to the minimization of Equation 6.15 is the only root of Equation 6.16 which resides in the interval of $[0, s_y]$. More details and proof of this can be found in [46]. The above approach was successfully implemented and used as the basis for reprojecting 3D points in this thesis.

6.4 Evaluation of the 3D Reconstruction

The following serves to compare the performance of the implemented 3D reconstruction approach against two methods which do not directly account for the refraction. Some of the more sophisticated approaches (discussed earlier in Section 6.1) were not tested as they were not readily available.

The three tested approaches are as follows:

1. **Ray-tracing using Snell's law** - The implemented approach described in the previous section.
2. **Underwater calibration** - 3D reconstruction using cameras where the calibration object has been placed under water.
3. **In air calibration** - 3D reconstruction using cameras where the calibration object has been placed in the air.

The actual triangulation step remains identical for all of the three approaches, as each of them will perform the triangulation step using the algorithm described in Section *6.2.5 Step 5 - Triangulation using Rays*. The same calibration object is used throughout the entire test and consists of a calibration wand, featuring two brightly-colored ping-pong balls mounted on a rod, as shown in Figure 6.6. The ping-pong balls have a diameter of 4.0 centimeters and their centers are placed 10.2 centimeters apart.

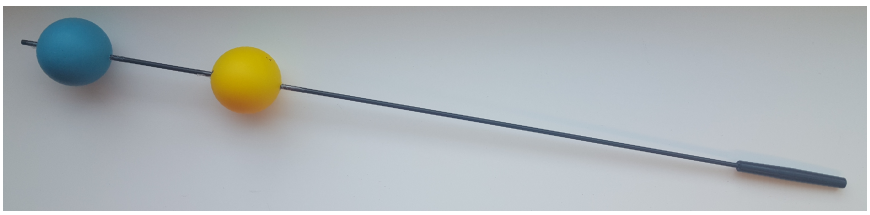


Figure 6.6: The calibration wand used during the test.

The different approaches are evaluated using two metrics:

- **Inter-distance error** - The inter-distance error, e_i, is calculated as the euclidean distance

$$e_i = d_w - |p'_1 - p'_2|, \tag{6.17}$$

where d_w is the known distance between the two ping-pong balls, while $|p'_1 - p'_2|$ is the euclidean distance between the two ping-pong balls according to the their triangulated positions. This error will hence provide a measure of the ability of each approach to correctly judge distances while capturing underwater objects at different locations in the aquarium.

- **Positional error** - The positional error, e_p, is calculated as

$$e_p = |p - p'|, \tag{6.18}$$

which is the euclidean distance between a triangulated point, p', and its known location, p. The purpose of the positional error is hence to supplement the shortcomings of the inter-distance error, as the latter will not detect any errors if all the triangulated points are translated equally.
The known locations used for calculation of the positional error are four position at the bottom corners of the aquarium, as shown in Figure 6.7. Their real world 3D location can be seen in Table 6.1. These four points are mainly choosen as they are easy to measure the location of and as they are easy to annotate in the image data.

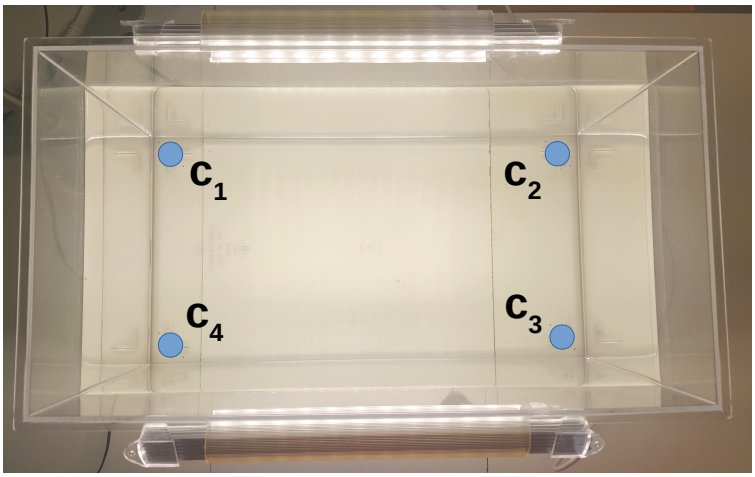


Figure 6.7: The four bottom corners of the aquarium used during calculation of the positional error.

c_1	c_2	c_3	c_4
(1.5, 1.5, 18.3)	(36.5, 1.5, 18.3)	(36.5, 18.5, 18.3)	(1.5, 18.5, 18.3)

Table 6.1: Positions of the four bottom corners of the aquarium. The stated positions are in centimeters.

The four bottom corners of the aquarium are also interesting as these locations are expected to have the biggest error due to refraction of any location within the aquarium. This expectation is based on the fact that these points are the furthest away from the cameras and with the highest incidence angle in relation to the water surface.

6.4.1 Test Setup and Procedure

The test is conducted using the setup shown in Figure 6.8, where two cameras are placed above the aquarium. The other camera configurations described in Section *2.4 Stereo Vision Camera Setup* has not been tested. Mainly as they are of little interest due to the difficulties experienced with the detection module when using the other camera configurations.

The test is executed as follows:

- **Step 1)** Gather data for calculation of the inter-distance error.

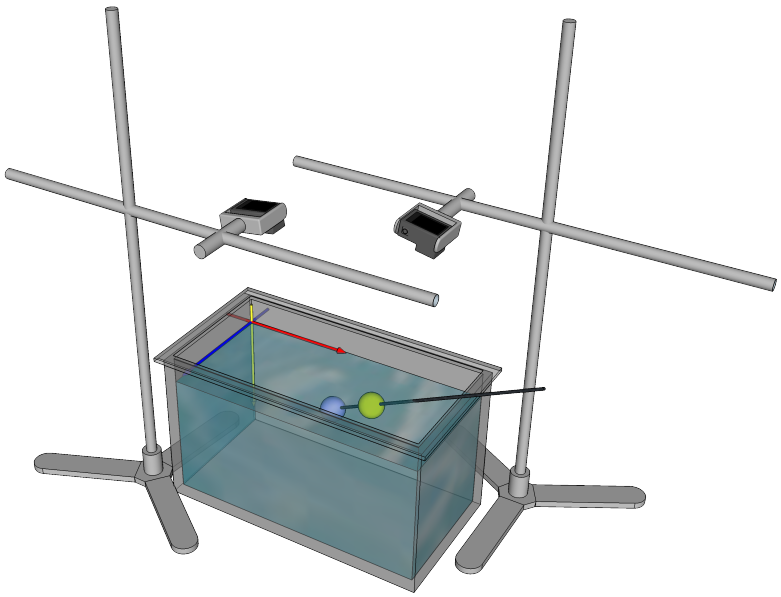


Figure 6.8: The setup used during evaluation of the different 3D reconstruction approaches.

  - Record a one minute video sequence with both cameras simultaneously while moving the calibration wand around in the water.
  - Temporally align the two video sequences manually.
  - Detect the positions of the ball center in each video frame-by-frame. The detection of the ball centers is done using a combination of color thresholding and the Hough Circle Transform found in OpenCV.

- **Step 2)** Gather data for calculation of the positional error.
  - Manually annotate the 2D positions of the four bottom corners of the aquarium, $(c_1, ..., c_4)$, from the view of both cameras.

- **Step 3)** Perform underwater camera calibration.
  - Record a one minute video sequence with both cameras while moving a checkerboard around in the water.
  - Extract 120 images from the video sequence and manually sort them until approximately 25 images remains. Blurred and distorted images are removed in this step.
  - Calibrate each camera as described in Section *6.2.1 Step 1 - Camera Calibration*.

- **Step 4)** Perform in-air camera calibration.
  - Repeat step 3 with the exception of moving the checkerboard around in free air instead of water.

- **Step 5)** Reconstructing 3D positions using the 2D positions from step 1 and 2.
  - Approach 1 (ray-tracing using Snell's law) and approach 3 (in-air calibration) rely on the camera calibration performed in step 4.
  - Approach 2 (underwater calibration) relies on the camera calibration performed in step 3.

- **Step 6)** Calculate both the inter-distance and positional error for each approach based on their reconstructed 3D positions.

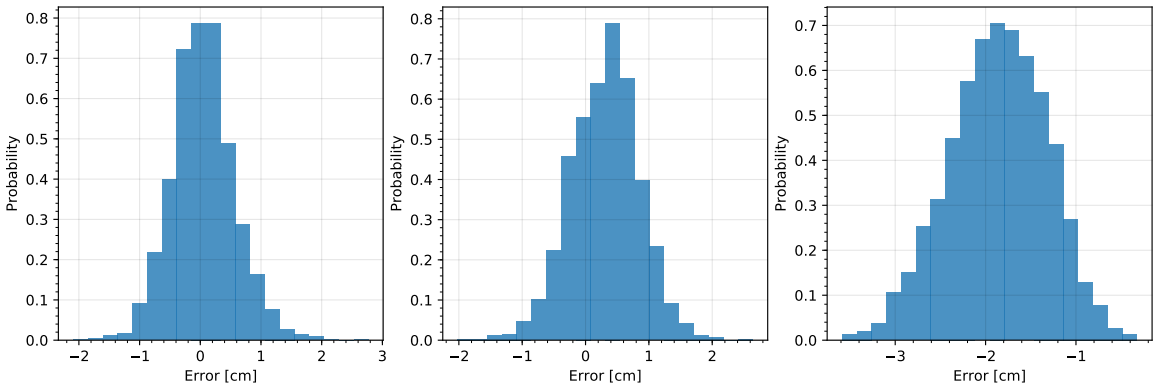
It should be noted that the above test approach suffers from certain uncertainties, such as the precision of the ball detections using the Hough Circle Transform and the precision of the distance manually measured between the ping-pong balls on the calibration wand. Comparing the absolute value of inter-distance error or the positional error to other tests of underwater reconstruction may hence not be a fair comparison.

The test is, however, valid in the sense that it serves as a comparison of the three approaches as they are all subjects to the same uncertainties.

6.4.2 Inter-Distance Error - Results

Histograms of the inter-distance error for all three approaches are shown in Figure 6.9. The errors appear to be normally distributed for all three approaches, with the ray-tracing approach performing the best with a mean error of $\mu = 0.05$. The approach which performs the worst is the one relying solely on the in-air camera calibration, which has a mean error of $\mu = -1.86$. This is not surprising, as this approach has no way of dealing with refraction. A negative mean error is sensible too, as it essentially amounts to the ping-pong balls appearing closer to each other than they really are. This agrees well with the general observation that objects appear closer when placed under water due to the refraction.

The approach relying on underwater camera calibration achieves a mean error of $\mu = 0.31$, making it better than the purely in-air camera calibration approach but worse than the ray-tracing approach. However, it is likely that a more thorough calibration of the aquarium volume could have reduced the inter-distance error further.



(a) Ray-tracing. ($\mu = 0.05$, $\sigma = 0.52$)

(b) Underwater calibration. ($\mu = 0.31$, $\sigma = 0.55$)

(c) Air calibration. ($\mu = -1.86$, $\sigma = 0.54$)

Figure 6.9: Histogram of the inter-distance errors for each the different approaches.

Another interesting aspect of the inter-distance error is its distribution in regard to the location of the triangulated ping-pong ball in the aquarium. This is sought illustrated in Figure 6.10, where each dot represents the midpoint between the reconstructed positions of the two ping-pong balls in the aquarium. The color represent the magnitude of the error,

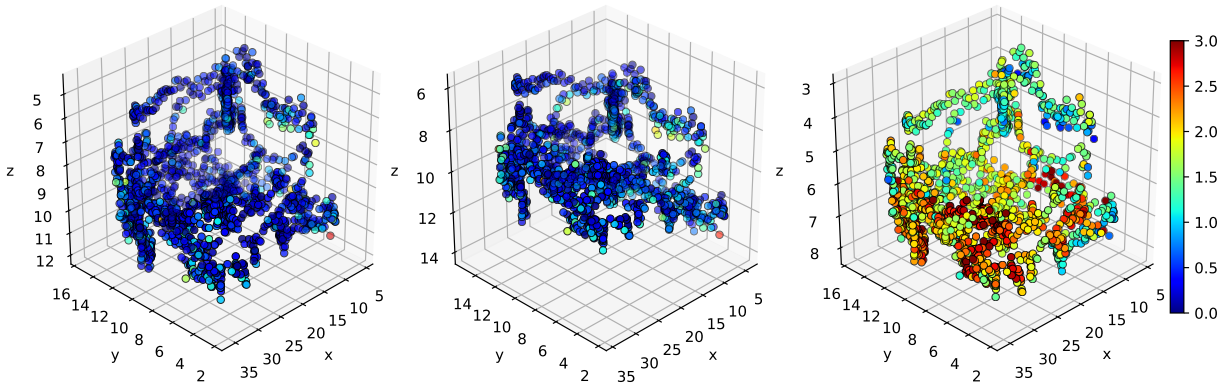
i.e. the absolute inter-distance error. The position of the midpoint, p'_m, is calculated as

$$p'_m = \frac{p'_1 + p'_2}{2}, \tag{6.19}$$

where p'_1 and p'_2 are the triangulated position of the two ping-pong balls.

The plots in Figure 6.10 reassembles the same pattern as in the histograms, with the in-air camera calibration performing the worst. The larger errors mainly appear at higher values of z (i.e. deeper in the water) and namely in the corners, i.e. the locations furthest away from the camera and with the highest incidence angles. The location of the errors are hence as expected.

It is also worth noting the different ranges for the x, y, z axes of the subplots in Figure 6.10, as all the points for the in-air calibration plot appear to be shifted toward the water surface (i.e. lower z values). This does once again correspond well with what is to be expected when looking a underwater objects without any correction for refraction.



(a) Ray-tracing. **(b)** Underwater calibration. **(c)** In-air calibration.

Figure 6.10: Plot of the inter-distance errors for each of the different approaches. Note that the color-bar is shared and depicts the absolute inter-distance error.

6.4.3 Positional Error - Results

The results of the positional error test for all three approaches are shown in Table 6.2. The results are somewhat similar to the previous test, with in-air camera calibration performing the worst while the two other approaches are nearly on par with each other. The difference between the ray-tracing approach and the approach based on underwater calibration could, as mentioned earlier, be attributed to the calibration of the latter.

6.4.4 Summary

The approach based solely on camera calibration in-air performed the worst of the tested approaches. This is not surprising as it has no way of accounting for the refraction. It performed especially horrible when looking at its positional error as it misjudged the depth of the aquarium by roughly 6 centimeters.

The two other approaches performed nearly identical in relation to both the inter-distance and positional error. Both approaches does hence appear suitably for the purpose

	c_1	c_2	c_3	c_4
Ground truth	(1.5, 1.5, 18.3)	(36.5, 1.5, 18.3)	(36.5, 18.5, 18.3)	(1.5, 18.5, 18.3)
Ray-tracing	(1.4, 1.7, 19.9)	(36.9, 1.7, 18.9)	(36.9, 18.6, 18.1)	(1.6, 19.1, 19.6)
Error [cm]	1.6	0.8	0.5	1.4
Average [cm]	**1.1**			
Underwater	(1.1, 1.2, 20.7)	(36.3, 1.6, 18.2)	(36.4, 18.4, 18.2)	(0.6, 18.8, 21.6)
Error [cm]	2.5	0.2	0.2	3.4
Average [cm]	**1.6**			
In air	(1.9, 1.9, 12.9)	(36.6, 1.8, 12.7)	(36.5, 18.4, 12.0)	(2.1, 18.9, 12.9)
Error [cm]	5.4	5.6	6.3	5.4
Average [cm]	**5.6**			

Table 6.2: The measured positions of the four bottom corners of the aquarium along with their positions in accordance to the 3D reconstruction methods. The error is calculated as stated in Equation 6.18.

of this project. The ray-tracing approach is, however, preferred due to its ease of use when changing the test configuration, as it would only require updating the positions used for calculating the extrinsic parameters of the cameras.

The approach based on underwater camera calibration would require re-calibration of the intrinsic parameters every time the setup is changed. Having to do so could quickly becomes tedious and may create issues with repeatability, as discussed in Section *6.1* . This drawback is further exacerbated by the current underwater calibration process as it is quite time-consuming to manually move a checkerboard around in the water. A more time-efficient approach would be to use a custom 3D calibration target manufactured to the size of the test aquarium.

7 Tracking the Fish

The following sections document the main ideas and implementation of the module responsible for associating the reconstructed 3D positions into tracks. The module consists of two parts; creating and linking tracklets. It is not a simple task to generate complete tracks based on the non-associated 3D detections, as there may be occlusion, missing detections and more, which creates complications. The idea is, however, that the incomplete tracks, known as tracklets, can ease the process as they carry more information than single detections.

The chapter is commenced with a brief review of what others have done to solve the above issue, followed by a description of the implemented method to generate tracklets. The chapter ends with an evaluation of the tracklets in order to identify potential issues and whether the tracklets could form the basis for further processing in the form of tracklet linking. A tracklet linking procedure is, however, not present in the system in its current state.

7.1 Related Work

This section seeks to highlight some of the existing systems and approaches for tracking moving objects. The following will mainly focus on approaches developed for fish or other animals in a controlled environment, as this is the focus of the thesis. Approaches for tracking in 2D will be addressed in addition to tracking in 3D as some of the main points are similar.

Tracking in 2D

A simple way of tracking Zebra fish in 2D can be found in [28]. The tracks are essentially generated by associating detections across frames while minimizing the total travel distance of the fish between successive frames. Missed detections and occlusions are handled manually by the user. The latter is, however, a bit counter-intuitive as the idea of employing computer vision was to avoid manual labor.

The system described in [9] is targeted towards 2D tracking of fish in shallow water and requires far less intervention from the user. A Kalman filter is applied to each of the tracked fish, which serves to predict the motion of the fish in case of occlusion. The fishes are associated across consecutive frames using several features such as the width, area and a gray-scale histogram of the fish.

The system relies on the Kalman filter to predict the location of a fish when it becomes occluded. This process is repeated either until the fish is no longer occluded or until a predefined amount of frames have elapsed. In case of the latter, the track will be marked as dead and a new track will be spawned once the fish is no longer occluded. Frequent occlusion may therefore result in a lot of fragmented tracks. The latter is handled by a final tracklet linking step, where tracklets are linked together into tracks if they adhere to certain spatial and temporal constrains.

Another way of solving the tracking problem can be found in [29], which describes the *idTracker* system, designed for tracking of animals in 2D for research purposes. The main focus of the system is to learn the visual appearance of each individual fish and then use this information to generate the final tracks.

The appearance of the fishes are learned in periods where the system estimates a low risk of occlusion. The appearance of the fishes are encompassed as 2D histograms where one axis is the intensity difference between two pixels and the other axis is the euclidean distance between the pixels. A nearest neighbor classifier is used to distinguish the fish using the 2D histograms.

Tracking in 3D

The following will try to outline some of the existing approaches for tracking of objects in 3D, with the main focus being on animals. The discussed methods will not be limited to the reconstruction-tracking approach but will include tracking-reconstruction approaches as well, as they might still provide some inspiration for how to perform the tracking.

Looking at the domain of 3D tracking, it is possible to find a commercially available solution offered by *Noldus*[47]. The base system is targeted at 2D tracking of laboratory animals, but it is possible to purchase an add-on allowing for 3D tracking of fish, flying insects and more. The exact price of such a setup is unknown, but the base package (without the 3D tracking add-on) is advertised at a starting price of $5,850.00.

The *Noldus* system is, however, limited to tracking of a single fish and does therefore not have to account for any of the issues related to tracking of multiple fish simultaneously. This limitation was confirmed in an email-correspondence with a representative of *Noldus*. This system is hence of no interest and will not be discussed further.

An example of tracking multiple targets in 3D can be found in [48], which focuses on tracking several hundred fruit flies using two cameras. The tracking procedure employed in this system is essentially divided into three assignment problems. The first one being the assignment of detections between subsequent frames to form 2D tracks for each camera. A Kalman filter is applied to each of these tracks and new detections are assigned based on the euclidean distance between the position of the new detection and the position predicted by the Kalman filter. The actual assignment is handled using the Hungarian/Munkres algorithm [49].

The second assignment problem consists of matching the 2D tracks across the two cameras. This matching solely relies on the position of the targets, as visual information was hard to extract due to the small size of the flies being tracked. The Hungarian algorithm is once again used during assignment, with the cost function being based on the *'epipolar motion length'* between the tracks. This length is essentially the disparity between the positions of the 2D tracks when projected to a common plane. The approach hence assumes that both cameras are calibrated and temporally synchronized.

The matched 2D tracks are used to construct 3D tracks by triangulating the positions of the 2D tracks. The last assignment problem consists of a final trajectory linking step, which serves to bridge any discontinuities in the 3D tracks. This assignment is primarily done based on kinematic and temporal constrains.

An approach for a similar scenario, i.e. tracking fruit flies, is described in [50]. This approach relies on particle filters to infer the most likely 3D position of the tracked flies by spawning multiple hypothesis, i.e. particles, which are weighted based on observations from the cameras. Furthermore, a kinematic model of the flies' velocity is learned using an RNN (Recurrent Neural Network) in the form of an LTSM network (Long Short-Term Memory). The kinematic model is used in the weighting of the particles as well.

Suitability of the Different Approaches

When looking at the different approaches described above, there appears to be two main categories of cues used during the association process; spatial cues (e.g. Kalman filtering) and visual cues (e.g. texture-features).

The approaches for tracking targets in 3D mainly appear to favor the spatial cues, while employing a cost function based on euclidean distance. This decision seems sensible, as it should in theory be impossible for two targets to occupy the same space in 3D. Assuming of course that the targets location is known precisely.

The 2D tracking approaches, on the other hand, appear more likely to employ either the visual cues or a combination of both types of cues. This is likely in an attempt to do without the additional information offered by tracking in 3D, as it is possible for multiple targets to occupy the same position when limited to 2D.

Based on the above observations it is decided to implement a relatively simple tracklet creation step based on spatial cues, much like [48] but in 3D instead of 2D. The resulting tracklets should then be linked into complete tracks by a subsequent tracklet linking step, which could for instance employ visual cues, for instance such as the *idTracker* system mentioned above.

The implementation of the tracklet linking procedure is, however, outside the scope of this thesis, as mentioned earlier. The main ideas and the implementation of the tracklet creation step is described in the next section.

7.2 The Assignment Problem

The tracklets, forming the basis of the tracking module, are to be created using 3D positions for each frame received from the reconstruction module. The task at hand is essentially an assignment problem, where the detections in a given frame, $f = i$, are to be associated with the detections in the following frame, $f = i + 1$. Such a scenario can often be contemplated an optimization problem, where the goal is to minimize the global cost with regard to some cost function when associating the detections across frames.

A common assumption in such scenarios is to presume that the objects are located in nearly same position in two consecutive frames. A suitable cost function would hence be the euclidean distance between the detections at $f = i$ and the detections at $f = i + 1$. This method can be efficient in scenarios where the objects to be tracked do indeed not move a lot between consecutive frames. However, problems can occur if multiple objects are located close to each other and if they move long distances between the frames.

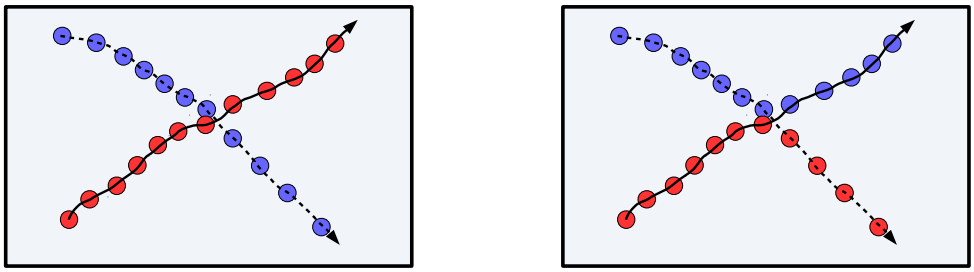
A more sophisticated solution to the assignment problem is to use a predict-match-update approach, which takes previous knowledge about the tracked object into account. This could be velocity, acceleration, direction or similar. This is a common method to use when the behavior of the object is relatively simple and, to some degree, known on beforehand.

An example of a scenario where a predict-match-update approach might be advantageous is depicted in Figure 7.1. The lines depicts the true path of the two fish, while the colored circles represent the paths of the fish as perceived by the tracking module.

In Figure 7.1a, the tracking module will solve the intersection correctly as the fish moves in a similar fashion before and after the intersection. The idea of using prior information

to make predictions will hence work well in this scenario, which is the idea of the predict-match-update approach.

The approach in Figure 7.1b may on the other hand fail to solve the intersection correctly, as it does not account for any prior information except for the last known position of the fish.



(a) Tracking with a predict-match-update approach.

(b) Tracking assuming near to no movement between consecutive frames.

Figure 7.1: Scenario where the path of two fish intersects. Both fish moves in a linear fashion, i.e. in a straight line.

A way of performing the prediction step in a predict-match-update approach is through the use of a Kalman filter. The filter uses previously obtained information about the tracklet to predict where the next detection could be. Detections close to the prediction are weighted higher than detections further away and the accepted detection is used to update the filter, in order to adapt to potential changes in velocity and direction.

The Kalman filter has several advantages as it is relatively easy to implement and use and it is, furthermore, cost effective in terms of computational cost, as it is mainly a matter of performing a few matrix-vector and matrix-matrix multiplications.

The main drawback of the Kalman filter, at least in the context of tracking fish, is that its underlying motion model is inherently linear. This trait means that large and abrupt changes in either velocity or direction are difficult to handle and may cause the tracklet to be cut short prematurely, even though correct detections are available.

An example of such is illustrated in Figure 7.2, were the tracklet, according to the Kalman filter, is illustrated using the red colored dots, while the correct path of the fish is illustrated using a solid black line. The blue dots illustrate the beginning of a new tracklet.

The Kalman filter is nevertheless employed in a predict-match-update approach in this thesis, despite the issue mentioned above. The main reason for doing so is the ability of the Kalman filter to smooth out the tracklets, making the system more resistant towards outliers and irregularities in the 3D positions from the 3D reconstruction module.

7.2.1 Global Minimization using Munkres

The actual process of assigning detections between frames is handled by the Munkres assignment algorithm, also known as the Hungarian algorithm[49]. The main reason for using

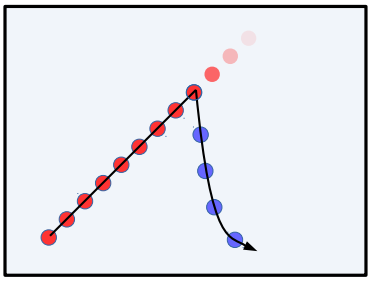


Figure 7.2: Erratic movement, where a fish changes direction within a short amount of time.

Munkres algorithm is to ensure that the assignment is performed such that the global cost is minimized, as opposed to a greedy approach, for example.

An example of why Munkres is preferred above a greedy approach is illustrated in Figure 7.3, where the circles represent the detections in the first frame and the crosses represent the detections in the consecutive frame. The dashed lines illustrate the associations and what is worth noting is the cross associated to the yellow circle. Even though that specific cross is closer to the green circle, the lowest global cost is obtained by associating it with the yellow circle.

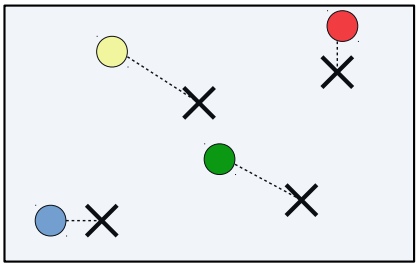


Figure 7.3: The circles represent the detections from the first frame and the crosses are detections from the second frame.

An example of a cost matrix is shown in Equation 7.1, where the tracklets t and detections d contain the positions of the detected fish in the first and second frame, respectively. The positions are noted t_j and d_i, where i is the detection number and j is the track number. Each index (i, j) of the cost matrix C will contain the cost of assigning detection d_i with tracklet t_j. The current implementation utilizes the euclidean distance, denoted $dist(t_j, d_i)$, as the cost function when constructing the cost matrix.

$$t = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{bmatrix}, \quad d = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix}, \quad C = \begin{bmatrix} dist(t_1, d_1) & dist(t_2, d_1) & dist(t_3, d_1) & dist(t_4, d_1) \\ dist(t_1, d_2) & dist(t_2, d_2) & dist(t_3, d_2) & dist(t_4, d_2) \\ dist(t_1, d_3) & dist(t_2, d_3) & dist(t_3, d_3) & dist(t_4, d_3) \\ dist(t_1, d_4) & dist(t_2, d_4) & dist(t_3, d_4) & dist(t_4, d_4) \end{bmatrix} \tag{7.1}$$

The cost matrix, c, will be padded by a constant value in a number of rows corresponding to the amount of detections that are missing. In an example where there are 2 detections,

4 tracklets and a padding value of p, the cost matrix would have the appearance seen in Equation 7.2.

$$c = \begin{bmatrix} dist(t_1, d_1) & dist(t_2, d_1) & dist(t_3, d_1) & dist(t_4, d_1) \\ dist(t_1, d_2) & dist(t_2, d_2) & dist(t_3, d_2) & dist(t_4, d_2) \\ p & p & p & p \\ p & p & p & p \end{bmatrix} \tag{7.2}$$

Only two of the tracks will have a true distance value, the other two will have a padding value of p. The padding value should be considerable higher than the distance the fish are able to cover between two frames to avoid the tracks to shift between the fish. Based on the test described in Section *2.5 Fish Movement and Behavioral Patterns*, the padding value should correspond to a minimum of $90\frac{cm}{s}$.

If there are fewer tracks than detections, new tracks should be spawned in order to accept all the detections. Furthermore, the amount of tracks should always exceed the amount of detections to ensure that there are padded tracks available for false-positive detections to attach to. Using this approach may result in a number of short tracklets mainly consisting of false positives, however, by introducing a minimum length, a number of these can be annihilated without further intervention.

Every track will have a counter assigned to it, which keeps track of the amount of frames since a new detection has been assigned to it. In cases where there are fewer detections than tracks, the tracks that are not assigned a new position will have their counter increased by one. If this counter reaches a certain threshold, the track is killed.

7.3 Evaluation of Created Tracklets

The purpose of this following is to evaluate the tracklets generated by the module described in this chapter. The first part of the evaluation is mainly focused on testing the impact of the maximum cost value which is used when creating the tracklets.

The second part of the evaluation will try to identify cases where the created tracklets appear less than ideal for the subsequent tracklet linking step. These cases are investigated further to identify the cause of the subpar tracklets.

7.3.1 Test Metrics

The created tracklets are evaluated by the mean tracklet length and visual inspection of fragmentation maps. These two metrics will be described in the following.

Mean Tracklet Length (MTL) - The average length of the intersection between the evaluated tracklets $T = \{T_1, T_2, ...\}$ and the annotated data $A = \{A_1, A_2, ...\}$, as shown in Figure 7.4. The metric is inspired by the "Mean Time Between Failures", which were proposed in [51] as an alternative to the widely used CLEAR MOT metrics [52].

The intersection $A \cap T$ is found by employing the same sequence extraction step as described in [51], which consists of associating the set of annotations, A, with the tracklets, T, for each frame $f = \{1, 2, .., n\}$, where n is the total number of frames.

The matching is executed such that the total euclidean distance between the two sets A and T is minimized for each frame f (using Munkres algorithm). Matches which deviate too much from the annotated data are removed by discarding any matches with a cost greater

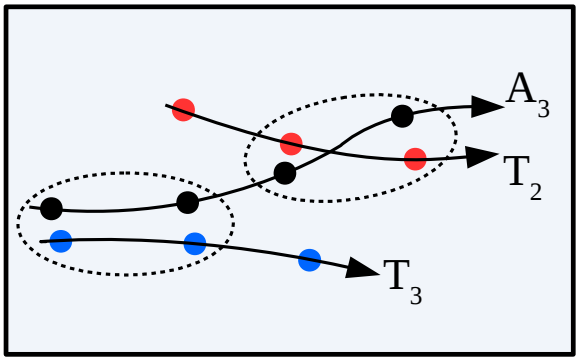


Figure 7.4: Example of the intersection between two tracklets and a track from the annotated data. The idea of identifying these intersections, and their length, is the underlying basis of the proposed MTL metric. The shown example will have a MTL score of 2.

than 1 cm. The result is then a new set of matches $M = \{[A_i, T_j]_1, [A_i, T_j]_2, ..., [A_i, T_j]_f\}$, where $[A_i, T_j]_n$ signifies a match between tracklet j and track i from the annotated data at frame f. The MTL score is then calculated as the average length of the matches M, where the assigned tracklet remains unchanged between frames.

The MTL metric is chosen over other metrics, such as counting ID switches (part of MOTA from the CLEAR MOT metrics), due to multiple reasons. The first one being its tendency to favor longer and more consistent tracklets, as long as they agree with the annotated data.

Another positive trait of the MTL metric is that it will directly punish ambiguous tracklets, as such tracklets would reduce the length of $A \cap T$ and hence the MTL score. Solely relying on counting ID switches would for instance fail to encompass the duration in which the tracklets agrees with the annotated data.

Fragmentation Map - A visual representation of the tracklets needed to recreate each of the tracks found in the annotated data. The fragmentation map can essentially be viewed as a visualization of the MTL metric, as it depicts the intersection between tracklets, T, and the annotated data, A. An example of fragmentation maps for two tracks are shown in Figure 7.5.

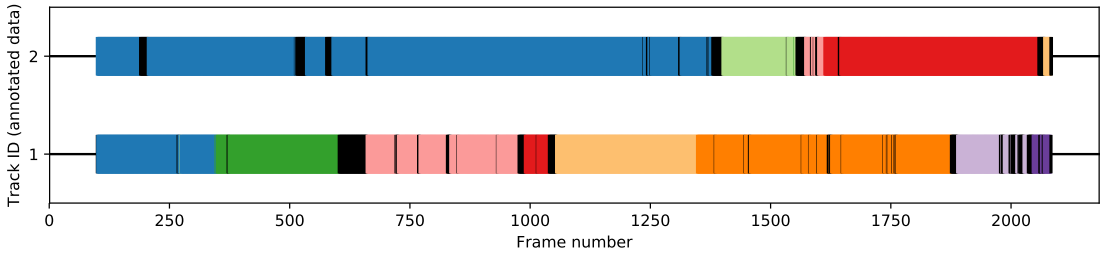


Figure 7.5: Example of fragmentation maps for two arbitrary tracks from the annotated data.

The black gaps in the fragmentation maps signifies periods where no intersection could be found between the track from the annotated data and the created tracklets. The color changes, with the exception of black, signifies that a different tracklet intersects with the track from the annotated data. An example is track 2 in Figure 7.5, which intersects with five different tracklets as indicated by the five color changes in its fragmentation map.

It should be emphasized that it is the color changes which is of importance in the fragmentation map. The colors do **NOT** uniquely identify certain tracklets, i.e. the same color being used in two fragmentation maps does not necessarily mean that it is the same tracklet in both maps. This limitation is a result of the limited range of clearly distinguishable colors.

7.3.2 Test Data

Ground truth: The annotated dataset, A, used in this test originates from the AA-1520-60-7f dataset, described in Section *3.1 Description of Dataset*. The manually annotated 2D positions from this dataset are passed through the 3D reconstruction module (Chapter *6 3D Reconstruction*) to generate their corresponding 3D positions, while still retaining the IDs of the fish. The tracklets are evaluated against these 3D positions.

Test data: The tracklets, T, used in this evaluation are formed by the procedure described in Section *7.2 The Assignment Problem*. The tracklets are generated from 3D positions from the 3D reconstruction module in the same manner as the ground truth. The 3D positions used for generation of the tracklets, T, are found using detections found by the detection module (Section *5 Fish Detector*).

Any errors in relation to the 3D reconstruction module are hence not a part of this test, as the two datasets being compared have both been processed using this module and are hence exposed to the same errors.

7.3.3 Impact of the Max Travel Distance

The first part of the evaluation is conducted by providing the same dataset to the tracklet creation module multiple times, while varying the max travel distance used during association. The result of this test is shown in Figure 7.6, where the MTL score is plotted as a function of the maximum travel distance.

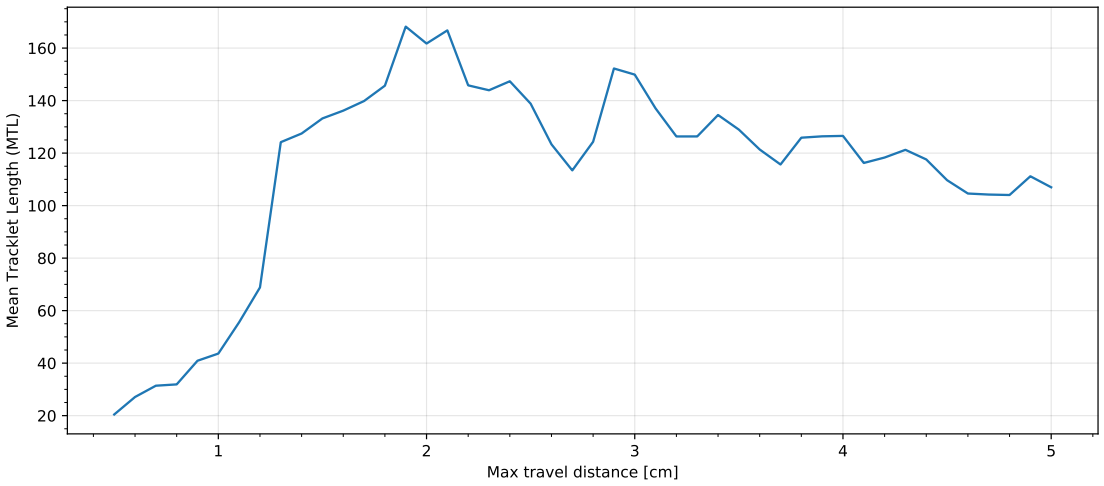


Figure 7.6: Result of the maximum travel distance test.

It can be seen that highest MTL score is achieved at a maximum travel distance of ≈ 2cm and that values larger than this appear to have a negative impact on the MTL score.

This observation is sensible, as the assignment step in the tracklet creation will become increasingly more random if there is no maximum travel distance to bound it. A maximum distance of ≈ 2cm per frame fits to the results found in Section *2.5 Fish Movement and Behavioral Patterns*, where are maximum speed of $\approx 90\frac{cm}{s}$ were observed. This corresponds to 1.5cm per frame, when the frame rate is taken into consideration and this allows for a margin of 0.5cm per frame.

A maximum travel distance value of 2.0 cm is hence used for the remainder of this evaluation.

7.3.4 Tracklet Failures

The second part of the evaluation serves to identify scenarios where subpar tracklets are created, i.e. tracklets resulting in a low MTL score. Such scenarios are found by contemplating the fragmentation map for each fish in the annotated data, which are shown in Figure 7.7.

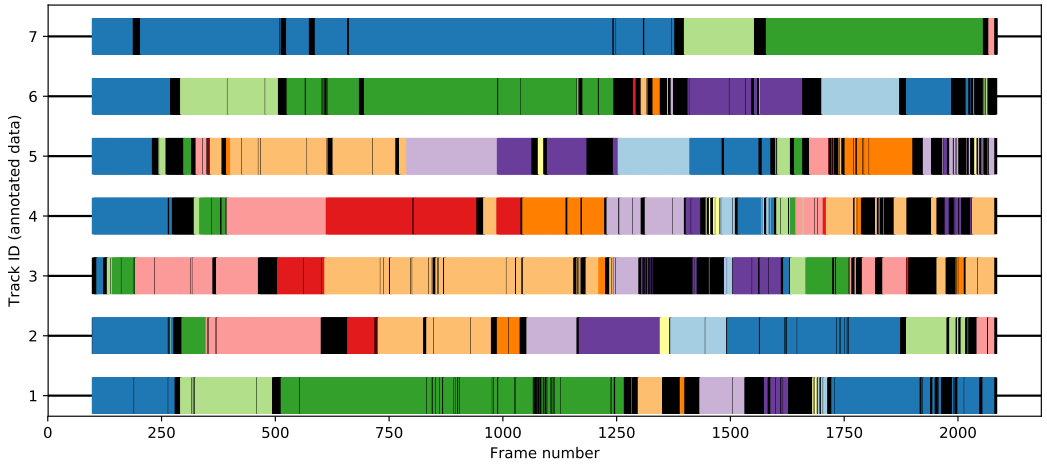


Figure 7.7: Fragmentation maps for each of the seven fish in the annotated dataset. Black spaces in the fragmentation map are periods where no intersection can be found between the annotated data and the created tracklets.

Several regions in Figure 7.7 shows certain features of the tracklets and are particularly interesting. Some of those will be highlighted and described in further details in the following sections.

Tracklets Failures - Abrupt Tracklet Change

A notable thing about the fragmentation maps is the abrupt tracklet changes which occurs with relatively short intermediate gaps in the data. Examples of such cases have been highlighted in Figure 7.8.

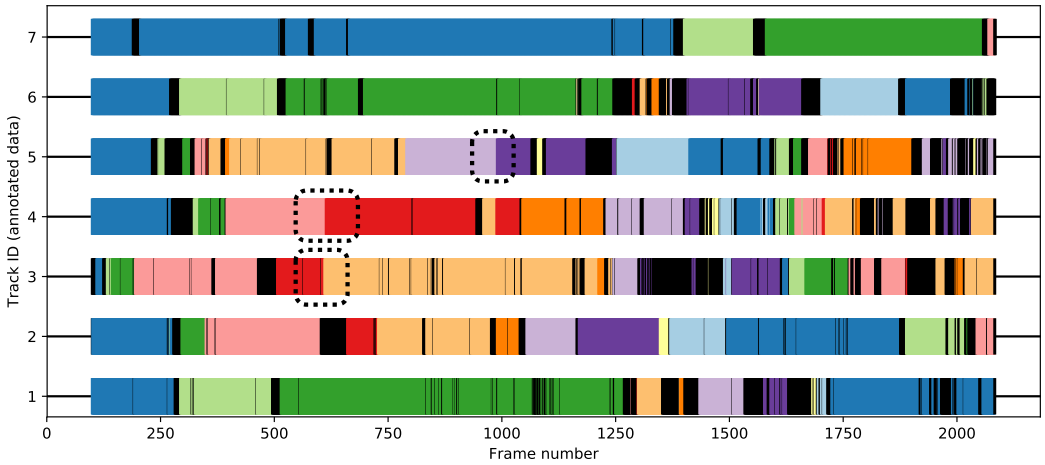
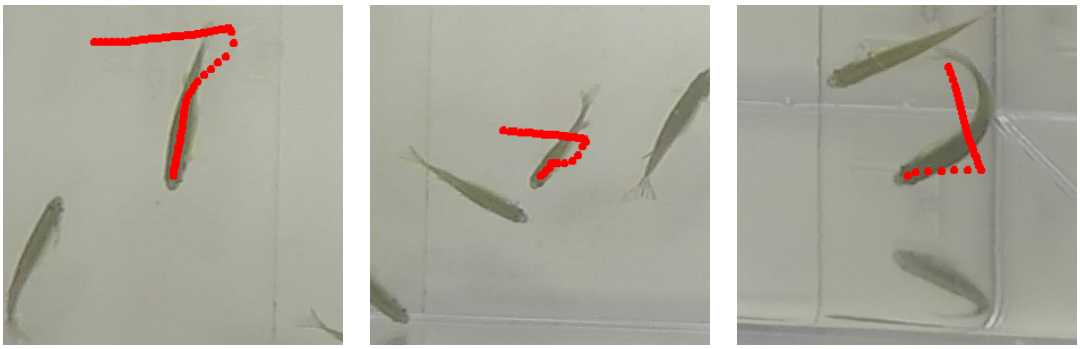


Figure 7.8: Fragmentation maps for each of the seven fish in the annotated dataset. Cases with abrupt tracklet switches are highlighted using dotted boxes.

By looking in the image data from each camera it is found that these abrupt tracklet changes mainly happens when a fish makes a sudden change of direction, as shown in Figure 7.9. The cause is likely the Kalman filter as it simply cannot keep up with the sudden change in direction, causing the tracklet to propagate linearly until the tracklet is killed.

Re-adjusting the parameters of the Kalman filter is a possible solution. However, doing so would make it more susceptible to outliers, thereby counter-acting the reasoning of having the Kalman filter in the first place. Another possibility is to accept that such sudden movements will break the tracklets and have the subsequent tracklet linking step to deal with it. The reasoning being that the resulting tracklets are quite close both spatially and temporally, and should hence be ideal candidates for the tracklet linking step.



(a) Fish 3, frame 570 to 635. **(b)** Fish 4, frame 590 to 620. **(c)** Fish 5, frame 1000 to 1030

Figure 7.9: Extracted regions from the image data depicting scenarios where an abrupt tracklet change is spotted in the fragmentation maps. The red dots depicts the path of the fish according to the annotated data.

Tracklet Failures - Gaps in the Data

Another notable occurrence in the fragmentation maps is the fairly large gaps, where no tracklets intersects with the annotated data. Examples of such gaps are shown in Figure 7.10.

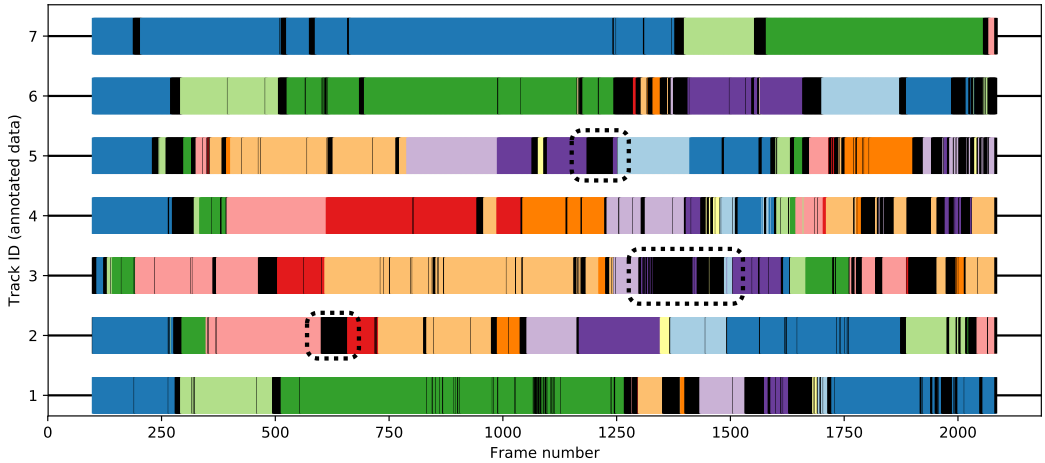


Figure 7.10: Fragmentation maps for each of the seven fish in the annotated data. Cases with large gaps in the fragmentation maps are highlighted using dotted boxes.

The image data is once again inspected and it is found that these gaps typically appear when the 3D reconstruction module incorrectly associates a detected fish in one camera with a reflection in the other camera, as shown in Figure 7.11.

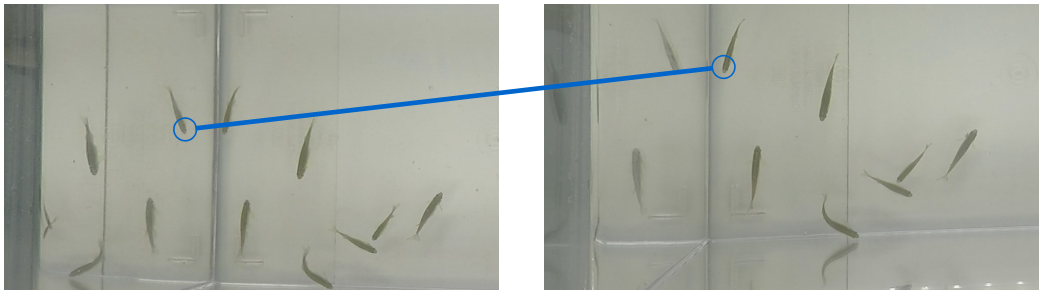


Figure 7.11: Fish incorrectly associated with a reflection by the 3D reconstruction module. The association for the other fish has been omitted in the figure to avoid clutter.

The result of the incorrect associations is quite clear when contemplating the resulting tracklets, which are shown in Figure 7.12. The tracklets of interest are mainly tracklet 7 and 9, with tracklet 7 being the result of the wrong association depicted in Figure 7.11. Tracklet 9, on the other hand, is for the period where the fish is once again correctly associated by the 3D reconstruction module. In Figure 7.10, tracklet 7 and 9 corresponds to the black area marked by the dotted square on track ID 2 and the red colored area, respectively.

Another thing worth noting in Figure 7.12 is the tracklet pairs which appear to belong to the same fish. Namely the pair consisting of tracklet 2 and 8, and the pair consisting of tracklet 3 and 6. The observation is of interest as it corresponds well with the observations in Figure 7.9a and Figure 7.9b from earlier. Both pairs can be observed in Figure 7.10 as

track ID 3 and 4.

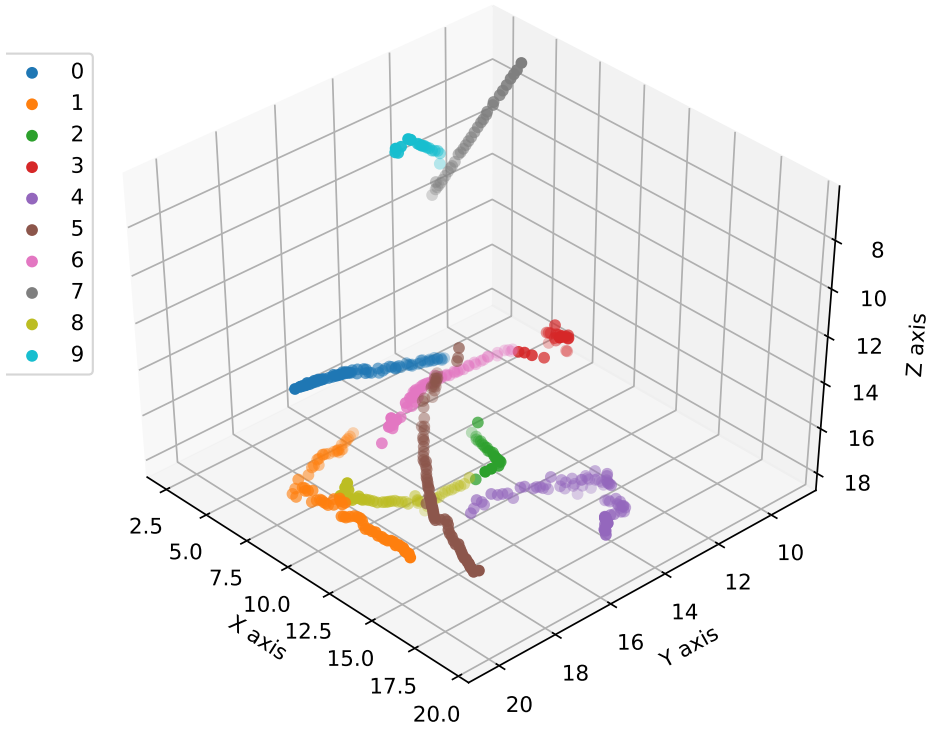


Figure 7.12: 3D tracklets for frame 590 to 670. Tracklets shorter than 20 frames are not plotted.

A way to remedy the gaps in the fragmentation maps may be to improve the spatial association performed by the 3D reconstruction module. The current implementation, see Section *6.3 Spatial Association*, utilizes Munkres algorithm to select the most likely candidates, while discarding the rest, which may not be the most optimal solution. Another possible method could be to actively look for reflections in the detections and discard them prior to the spatial association by the reconstruction module. This may be possible as there are physical laws describing when and where such reflections can occur, much like Snell's law used for the correction of the refraction. However, the most straightforward solution would be to eliminate reflections altogether by using another type of tank while recording the fish. A tank with opaque sides could for instance be a solution, depending on the requirements of the test being conducted.

Tracklet Failures - High Tracklet Frequency

The last scenario which will be discussed is the regions of the fragmentation maps with a high density of unique tracklets. Examples of such cases are marked in Figure 7.13.

By inspecting the 2D detections it is discovered that periods with a high tracklet density are characterized by many false positives. For instance, the tail and body of the fish being incorrectly labeled as a fish head, as shown in Figure 7.14.

There are multiple potential solutions for the problem of false positives being detected on the fish body. This could be by simply training a more suitable binary classifier or by implementing a multi-class classifier, which is capable of distinguishing between head, body

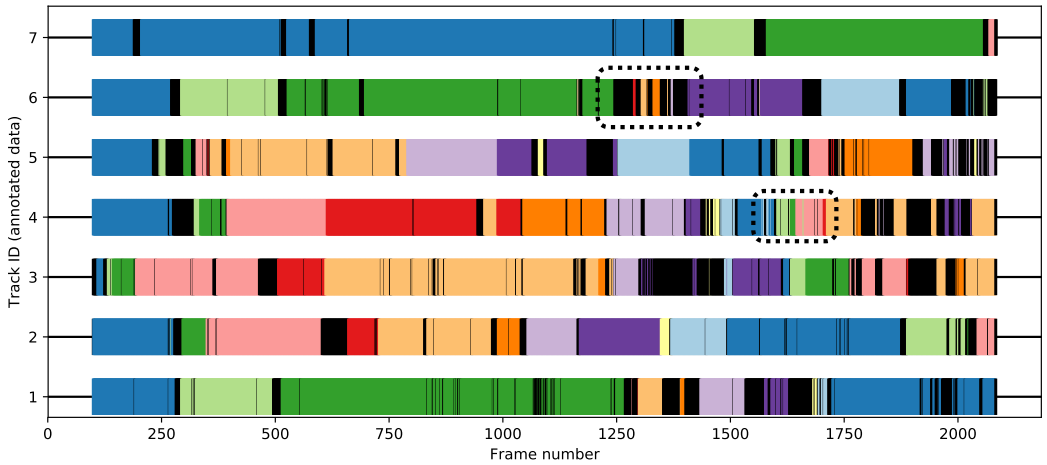
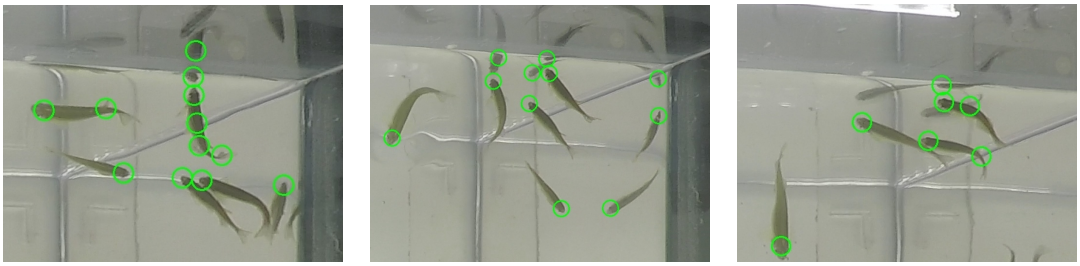


Figure 7.13: Fragmentation maps for each of the seven fish in the annotated dataset. Regions with a high density of unique tracklets are highlighted using dotted boxes.



(a) Frame 1562. **(b)** Frame 1590. **(c)** Frame 1608.

Figure 7.14: Extracted regions from the image data depicting scenarios with an increased number of false positives. The green circles depict detections found by the detection module.

and tail of the fish. This could give more information about how the fish is swimming and potentially reveal if the fish becomes partially occluded. No investigation has been made in this area and it will not be discussed further.

Another method could be to use a sort of re-identification, which could also be used for the linking process. If a detection were found at the tail or center of the body, it would be easy to distinguish from a fish head. The idea of using re-identification will be discussed in greater details in Section *8 Discussion*.

7.3.5 Summary

A simple method to create tracklets has been implemented, which is mainly based on the euclidean distance between detections in consecutive frames. The solution creates relatively long tracklets in scenarios where the fish follows a linear path without being interfered. However, problems, such as reflections, false positives, and false negatives, may occur when the fish moves erratically, are near other fish or swims close to the aquarium walls.

By evaluating the tracklets using the fragmentation map, it has been possible to determine where, when and why these problems occur and this can be a help in developing solutions that are capable of handling the given problems. This, of course, is based on the

assumption that longer and more consistent tracklets are desirable for the tracklet linking step, which has not yet been implemented.

Beside inspecting the fragmentation map, the optimal maximum travel distance, in relation to the MTL metric, was found to be ≈ 2cm for the given test recording. However, more tests are needed in order to make conclusions on whether this is the generally most optimal maximum travel distance.

8 Discussion

The following chapter contains a discussion regarding the implemented modules, their performances, pros and cons and possible enhancements. The temporal alignment problem, briefly mentioned in the thesis, will also be discussed along with possible solutions. The latter is included in the discussion in spite of previous delimitations as temporal alignment is considered an important part of a fully functional system. The discussion will include thoughts and ideas of how a tracklet linking step could be implemented as well.

Temporal Alignment

The synchronization problem has been explained in Section *2.2 Synchronization of Multiple Cameras*, but no solution has been offered as it was outside the scope of the thesis. The following seeks to rectify this omission by offering different methods to take this issue into account.

In the literature, several methods have been proposed to solve the problem including the use of light sources, specific synchronization hardware and feature tracking. In [53], a single LED light source is used to synchronize multiple cameras based on binary patterns, which allow them to obtain sub-frame precision. Pseudo random sequences are used to avoid phase shifts that may results in ambiguity and the two cameras are assumed to be identical to ensure an equivalent frame rate.

Another approach is offered in [54], where two video sequences are synchronized using trajectories of moving objects. The authors claim that their method is capable of obtaining sub-frame temporal synchronization between the video sequences as long as the cameras are stationary or moved together, keeping the same internal and relative external parameters, under the acquisition phase. In a tracking-reconstruction algorithm, this would be an obvious method to use, as the 2D tracklets can be used to calculate the trajectories, which can then be compared with tracklets in the other view in order to identify similar trajectories and thereby find the temporal shift. In an early stage of the 3D-FisT system, the tracking-reconstruction approach was considered and the use of curvatures for temporal alignment was investigated (see Appendix - *Spatio-Temporal Alignment using Curvatures*).

A reconstruction-tracking approach, as employed in this thesis, does however seem less suited for a synchronization approach like this. This is mainly due to the fact that tracklets are not available until after the 3D reconstruction, which in turn requires the temporal shift to be known. A possibility is of course to create 2D tracklets along with the 3D tracklets and then use the former in the calculation of the temporal shift.

An approach more suited for the current implementation could be to utilize the re-projection error to identify the most optimal temporal shift. The main idea would simply be to reconstruct 3D positions using 2D detections with varying temporal shifts. The most optimal shift would then be the one resulting in the lowest total re-projection error. This is of course based on the assumption that the found minima is indeed the most optimal temporal shift. Further testing is required to prove or disprove such a correlation.

It is also possible to find commercially available solutions for real-time synchronization of cameras. An example of such is the GoPro HERO dual [55], which allows the user to start two GoPro HERO3+ Black Cameras at the exact same time. The cameras are then continuously synchronized in real-time by a small cable interconnecting the internal clock of the cameras. However, such a solution is highly inflexible as it only works for a specific

type of camera. Even the new GoPro HERO5 models, used in this thesis, are not officially supported. The length of the cable connecting the two cameras is also a limiting factor, as it only allows the cameras to be separated with a few centimeters apart.

Detection

The implemented detection algorithm is simple, but performs well, as nearly every fish are found in the images. However, it is likely that other approaches could enhance the performance by either increasing the amount of true detections or decreasing the amount of false positives.

An approach which could most likely outperform the combination of using SURF keypoints and a binary SVM classifier is a well trained convolutional neural network (CNN). If a CNN were implemented and trained, it would probably require a large part of fish heads as training data, in order for it to work probably. However, if it was possible to get enough labeled data, it could potentially give a significant enhancement boost to the detection algorithm with less false-negatives and false-positives. Furthermore, it may even open up for the possibility of not just detecting, but also recognizing the specific type of fish species, which could make the 3D-FisT system more applicable.

Another and more simple method that could enhance the performance of the detection module, is to introduce better lighting. The amount and type of lighting used for the test-setups may not be the most optimal. If the setup was lit by a stronger and more uniform source of light it would be possible to record with a higher frame rate and shutter speed. This could help on a range of problems observed through this project including

- **Blur** caused by accelerations
- **Loss of features on the fish** caused by dim light
- **Lack of detections** caused by the frame rate

When the fish turns or accelerates, it is often by a rapid movement, that render the fish difficult to detect due to blur. This is bad in multiple ways, as there is a high probability that the fish will not be detected for several consecutive frames and it is at a time, where the fish is moving at high speeds or with abrupt changes in direction.

If it is possible to detect the features of the fish, it may be possible to learn the features of each unique fish and use re-identification to improve 3D reconstruction accuracy and tracking. However, this makes demands to the placement and type of light, as it needs to have a certain luminous intensity and be placed in a way, that allows the light to hit the fish from above.

More light and better illumination makes it easier to record at higher frame rates and if the frame rate is higher, the negative impact caused by missing detections is most likely reduced.

3D Reconstruction

The implemented 3D reconstruction module, relying on ray tracing in combination with Snell's law, did generally perform well in comparison to the two other tested approaches. The approach relying on ray tracing achieved a positional error of 1.09 centimeters and an inter-distance error of 0.05 centimeters.

The next best results were obtained by the approach relying in underwater camera calibration, which achieved a positional error of 1.59 centimeter and an inter-distance error of 0.31 centimeters. The worst performer of the tested approaches was the one relying solely on in-air camera calibration, obtaining a positional error of 5.69 centimeters and an inter-distance error of -1.86 centimeters.

Whether the reconstruction errors experienced in the ray tracing approach is acceptable depends on the intended use. The positional error of ≈ 1 centimeters can for instance be disastrous if it was used as the basis for automatic manipulation of objects, e.g. a robot handling delicate objects. It can, however, be argued that such errors are perfectly fine in the case of behavioral analysis, as the reconstructed position mainly serves to provide an idea of where the fish are in the aquarium.

In the context of behavioral analysis, the main drawback of an imprecise 3D reconstruction is the risk of mixing up two fish as they get close to each other. The risk of doing so would, however, appear minimal in the current solution, as the inter-distance error was found to be quite low.

Adding an optimization step, as described in [38], could be a possible way of improving the current solution, if need be. The idea of the optimization would be to optimize the camera parameters (intrinsic and extrinsic) along with position and orientation of the water surface.

An additional way of testing the reconstruction module could be through the use of a simulated scenario. The main advantage of such a simulation being knowledge of the precise position of everything in the scene, i.e. the points being reconstruction, the cameras, the water surface and so forth.

The idea of using a simulated scene to test the reconstruction module was initially pursued and some basis fish A.I along with the aquarium were modeled in the Unity game engine [56] (see Appendix - *Fish Simulation*). The simulation was ultimately discarded due to difficulties with the implementation of the refraction effect, as Unity was found to be ill suited for such a task.

In hindsight, a program capable of photo-realistic rendering, such as Blender[57], would probably have been a better choice. The advantages of photo-realistic rendering being that effects such as refraction, reflections and so on are portrayed correctly. Game engines, such as Unity, will often prioritize real-time performance over realism. The resulting effects may hence look correct to a person playing a game but it does not necessarily abide by the laws of physic, e.g. Snell's law.

Tracking

A significant amount of the miss-associations made by the tracking module are caused by erratic movements and sudden changes in direction, which causes the Kalman filter to make wrong predictions. It could be investigated whether there are forewarnings or certain tendencies, that could help predict when the fish is about to make a drastic turn or acceleration. As the zebrafish demonstrate different types of group behavior, there may be a correlation between the group dynamics and the individual fish or it may be possible to find clues in the motion trajectories.

Another, and maybe more straightforward, solution could be to implement a prediction-update algorithm, which is not linear like the Kalman filter. This could be an Extended Kalman filter or Particle filter, which are both better suited for nonlinearities, which the trajectories of the fish certainly are.

Tracklet Linking

The improvements mentioned in the previous section should improve the overall mean tracklet length (MTL). However, the tracklets are likely to still remain tracklets, i.e. they will not span the entire video sequence, hence prompting the need for a subsequent tracklet linking step. Ideas and thoughts on such a linking procedure will be described in the following.

A relative simple way of performing tracklet linking would be to link all tracklets adhering to certain spatial and temporal constraints, as proposed by [9]. These constraints simply states that two tracklets should be linked if they are both spatially and temporally located within a pre-defined distance of each other. This approach is hence quite inflexible due to the usage of pre-defined values.

Another approach relying on the spatial cues can be found in [58], which proposes a similarity measurement between tracklets based on the resulting complexity of the linked tracklets. The main idea is that each tracklet can be described as an nth order polynomial with the order of the polynomial being an expression of the complexity of the tracklet. Then similarity of two tracklets is then judged using the ratio

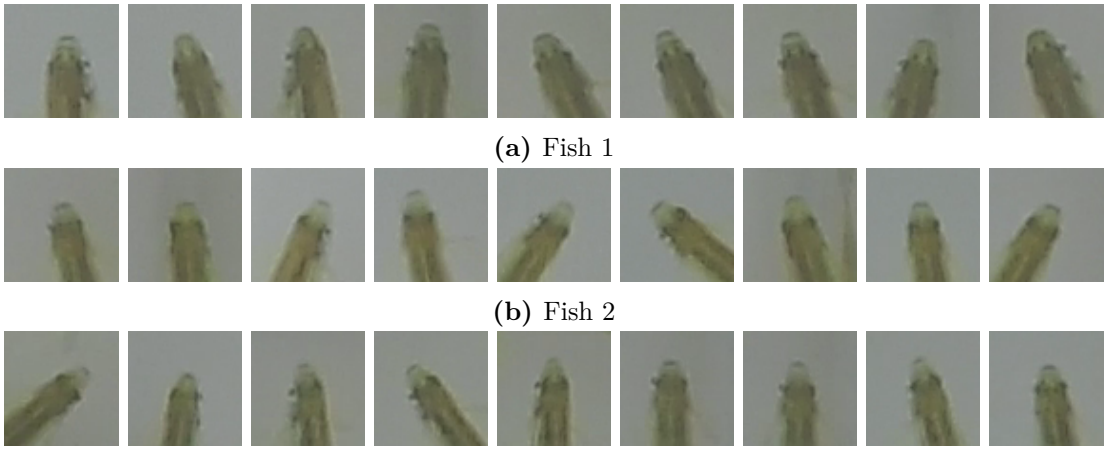
$$sim(t_i, t_j) = \frac{order(t_i) + order(t_i)}{order(t_{ij})}, \tag{8.1}$$

where t_i and t_j are the two tracklets to be compared, while t_{ij} is the tracklets linked together.

The above is based on the underlying assumption that a linked track should not have a higher complexity than its part if the linked tracklets are indeed part of the same track. The remaining part of the paper primarily deals with clever ways of utilizing Hanklet matrices to determine the tracklet complexity in a cost-efficient way.

However, the problem of linking the tracklets could also be treated as a re-identification problem, with each of the fish being an individual to identify. An advantage of such an approach would be its independence of the temporal separation between the tracklets to be linked. I.e. linking tracklets which are temporally separated by 2000 frames should, in theory, be identical to linking tracklets separated by 200 frames. This is as opposed to linking using spatial cues, as these approaches would be increasingly likely to fail as the temporal separation is increased.

The main question regarding the use of re-identification is whether it is possible to actually distinguish between the fish as they look somewhat similar, at least to the human eye. An example of fish heads from different fish can be seen in Figure 8.1.



(a) Fish 1

(b) Fish 2

(c) Fish 3

Figure 8.1: Example of fish heads from three different fish. The shown images are spaced with roughly 50 to 100 frames between them.

Even though it may be possible for the human eye to distinguish the fish from each other, it should be possible. At least according to [29] and [24], which both relies on re-identification for tracking fish in 2D. The former approach mainly relies on a texture features based on 2D histograms gathered from the entire fish. The histograms would on one axis contain the intensities between two pixels, i.e. $|i(x_1, y_1) - i(x_2, y_2)|$, while the other axis would contain the distance between them.

The approach in [24] did, on the other hand, make use of a CNN to distinguish between the different individuals solely using the head of the individual fish. The CNN was pre-trained using 300 samples for each individual but the approach also included a strategy for adding new samples continuously while tracking.

Yet another possibility, in terms of re-identification, could be to utilize local binary patterns (LBP), which have been successfully used for re-identification of persons using their palm-prints [59]. The idea of using LBP in relation to the fish heads would be its emphasis on texture, as it may be able to pickup the minor differences between the individual fish.

Regardless of the selected re-identification approach, it should be possible to obtain both negative and positive samples for each of the tracklets generated earlier. This could be accomplished by considering all instances of fish within that tracklet as positive samples, while all instances not within the tracklet, but within the same span of frames, are regarded as negative samples.

Given a mean tracklet length of ≈ 150 frames, the above approach should on average provide $2 \cdot 150 = 300$ positive samples and $2 \cdot 150 \cdot (n - 1) = 300n - 300$ negative samples, where n is the total number of fish in the aquarium. This is assuming a system consisting of 2 cameras.

Another way to engage the problem of tracklet linking could be to utilize the reprojection capabilities of the implemented reconstruction module to check the validity of any links

between tracklets. The main idea of this approach is illustrated in Figure 8.2, which depicts three tracklets, t_1, t_2 and t_3, in both 3D and 2D. Two possible links are illustrated; $t_1 \leftrightarrow t_2$ and $t_1 \leftrightarrow t_3$.

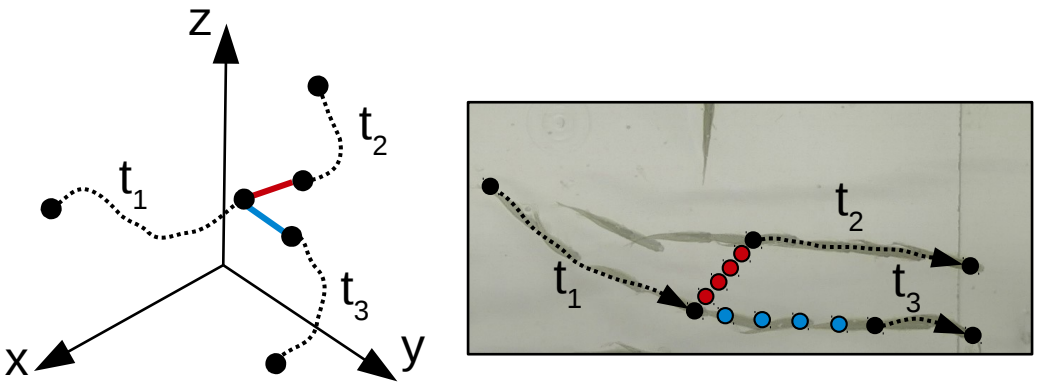


Figure 8.2: Example of checking links between 3D tracklets. The possible links are reprojected back into the image plane of the cameras to check whether the reprojected position would corresponds to a fish.

The intermediate points in each of the two links are interpolated in a linear fashion, as illustrated in the 3D plot of the tracklets. These interpolated points are then reprojected back into the image plane of the camera, where it is checked whether the reprojected points overlaps with a fish. The link consisting of $t_1 \leftrightarrow t_3$ is hence considered more likely than the $t_1 \leftrightarrow t_2$ link, as the latter does not overlap with any fish in the image plane.

Exactly how to determine the overlap between the reprojected points and the fish in the image plane is still an open issue. However, a possible solution could be to use the existing detection module around the immediate area of the reprojected point.

The effectiveness of the above approaches and suggestions is still untested but they form a basis for future work.

9 Conclusion

The problem of tracking multiple objects in water in three dimensions is a challenging problem, which has not yet been investigated thoroughly. Existing solutions are either expensive, highly customized to certain setups or limited to one target. The purpose of this thesis has been to investigate the possibilities of tracking multiple zebrafish in 3D using off-the-shelf equipment.

Different sensors for acquiring 3D information were reviewed and a stereo vision setup was found to be the most suitable solution. The advantages of a stereo vision setup are the possibility of using passive sensors and the ability to easily assemble such a setup using off-the-shelf hardware. Active sensors, such as time-of-flight and structured light, were deemed less suitable as they may disturb the test subjects and the refraction caused by the water may interfere with the sensors.

A stereo vision setup with two GoPro HERO5 cameras, placed side-by-side looking down into an aquarium, was chosen as the setup for the system.

Detection

The detection of the fish is primarily achieved using a SURF keypoint detector as it was found to be effective at detecting the head of the fish, which is desirable as the rest of the fish deforms noticeably when it moves. It is assumed that the controlled test environment made it possible to get satisfactory results with a detection module based primarily on the SURF keypoint detector, as it caused the fish to stand out. However, even though the tests were conducted in a controlled environment, non-fish detections were still observed. A part of those were removed by employing a linear binary SVM classifier in combination with non-maximum suppression.

The detection module achieved a recall of $\approx 95\%$ and a precision of $\approx 40 - 60\%$, which was deemed satisfactory, as the rest of the false positives should be annihilated in other parts of the system.

3D Reconstruction

During the initial analysis it was found that the commonly used pinhole camera model is not applicable when capturing underwater objects. This is due to the refraction of light at the interface between air and water which invalidates the assumption of a single viewpoint. This observation was confirmed during a test in which an average error of -1.86 centimeters were measured when reconstructing underwater positions.

3D reconstruction using the pinhole camera model specifically calibrated for underwater objects were tested as well, yielding an average error of 0.31 centimeters. This approach did, however, include a rather inconvenient calibration procedure, where a checkerboard had to be moved around under water in the confined space of the aquarium. A major drawback of this approach is that the calibration procedure would have to be repeated often, as the intrinsic camera parameters are tied to the entire setup and not only the camera.

The best results in terms of 3D reconstruction was achieved by employing ray-tracing in combination with Snell's law, which achieved an average error of 0.05 centimeters when capturing underwater objects. An advantage of this approach was a less restrictive calibration procedure, as the camera could be calibrated in air. The precision of the 3D reconstruction

was hence found to improve noticeably when actively accounting for the errors induced by refraction.

It should be noted that all the tested approaches for 3D reconstruction achieved a similar standard deviation of ≈ 0.5 centimeters.

Tracking

The implemented tracking procedure relies on a reconstruction-tracking approach where the reconstructed 3D positions are associated into tracks. This approach was mainly chosen as it allows the system to impose certain restrictions. This include discarding 3D positions outside the volume of the aquarium and discarding reconstructed points with high reprojection errors. The reprojection of each 3D position was found by solving a quadratic polynomial in order to account for refraction.

The actual association of the 3D positions was handled as an assignment problem and solved using Munkres algorithm, where the cost of each assignment was the euclidean distance between the positions. A Kalman filter was added to smooth the tracks and functioned as part of a predict-match-update approach during tracking.

The average length of the tracks were found to be ≈ 150 frames, which equals 2.5 seconds when accounting for the frame rate of the test recording. The tracking module is hence capable of generating tracklets, i.e. incomplete tracks, which may form the basis for a subsequent step that links them into complete tracks.

A tracklet linking step has, however, not been implemented, but it could be treated as a re-identification problem using visual cues to learn an appearance model for each tracklet. The content of each tracklet would then function as positive samples while all other detections, within the same timespan, would be treated as negative samples. Using spatial cues, such as movement or curvatures, is a possibility as well but may prove problematic due to the somewhat erratic movements of the fish.

Bibliography

[1] National Ocean Service, U.S Department of Commerce. What are microplastics? [Online]. Available: http://oceanservice.noaa.gov/facts/microplastics.html

[2] UN Environment. UNEP Year Book 2014 emerging issues update, plastic debris in the ocean. [Online]. Available: http://www.unep.org/yearbook/2014/PDF/chapt8.pdf

[3] E. Fontaine, D. Lentink, S. Kranenbarg, U. K. Müller, J. L. van Leeuwen, A. H. Barr, and J. W. Burdick, "Automated visual tracking for studying the ontogeny of zebrafish swimming," *Journal of Experimental Biology*, vol. 211, no. 8, pp. 1305–1316, 2008. [Online]. Available: http://jeb.biologists.org/content/211/8/1305

[4] E. Burke. (2016) Why use zebrafish to study human diseases? [Online]. Available: https://irp.nih.gov/blog/post/2016/08/why-use-zebrafish-to-study-human-diseases

[5] R. National Centre for the Replacement and R. of Animals in Research. (2014) Five reasons why zebrafish make excellent research models. [Online]. Available: https://www.nc3rs.org.uk/news/five-reasons-why-zebrafish-make-excellent-research-models

[6] T. Brodin, J. Fick, M. Jonsson, and J. Klaminder, "Dilute concentrations of a psychiatric drug alter behavior of fish from natural populations," *Science*, vol. 339, no. 6121, pp. 814–815, 2013. [Online]. Available: http://science.sciencemag.org/content/339/6121/814

[7] S. Ohayon, O. Avni, A. L. Taylor, P. Perona, and S. R. Egnor, "Automated multi-day tracking of marked mice for the analysis of social behaviour," *Journal of Neuroscience Methods*, vol. 219, no. 1, pp. 10 – 19, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165027013002070

[8] N.-W. Lo, H. T. Chang, and J.-Y. Chang, "Caged mice mating behavior detection in surveillance videos," *Journal of Visual Communication and Image Representation*, vol. 25, no. 5, pp. 755 – 762, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1047320314000285

[9] Z.-M. Qian, X. E. Cheng, and Y. Q. Chen, "Automatically detect and track multiple fish swimming in shallow water with frequent occlusion," *PLOS ONE*, vol. 9, no. 9, pp. 1–12, 09 2014. [Online]. Available: http://dx.doi.org/10.1371%2Fjournal.pone.0106506

[10] J. Cachat, A. Stewart, E. Utterback, P. Hart, S. Gaikwad, K. Wong, E. Kyzar, N. Wu, and A. V. Kalueff, "Three-dimensional neurophenotyping of adult zebrafish behavior," *PLOS ONE*, vol. 6, no. 3, pp. 1–14, 03 2011. [Online]. Available: https://doi.org/10.1371/journal.pone.0017597

[11] A. Keledjian, G. Brogan, B. Lowell, J. Warrenchuk, B. Enticknap, G. Shester, M. Hirshfield, and D. Cano-Stocco. (2014) Wasted catch: Unsolved problems in u.s. fisheries. [Online]. Available: http://oceana.org/sites/default/files/reports/Bycatch_Report_FINAL.pdf

[12] D. J. Wang, D. Senicic, M. Agrawal, and M. Nadeski, *3D Machine Vision Reference Design Based on AM572x With DLP Structured Light*, Texas Instruments, 2016.

[13] Paul Hvass. 3d camera survey. [Online]. Available: http://rosindustrial.org/news/2016/1/13/3d-camera-survey

[14] L. Li, "Time-of-flight camera - an introduction," Tech. Rep. SLOA190, January 2014. [Online]. Available: http://www-cs.ccny.cuny.edu/~wolberg/capstone/kinect/ToFBasicsTI14.pdf

[15] B. Langmann, K. Hartmann, and O. Loffeld, "Depth camera technology comparison and performance evaluation," in *Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods*, 2012, pp. 438–444.

[16] P. Zanuttigh, G. Marin, C. D. Mutto, F. Dominio, L. Minto, and G. M. Cortelazzo, *Time-of-Flight and Structured Light Depth Cameras - Technology and Applications*. Springer, 2016.

[17] M. Gupta, Q. Yin, and S. K. Nayar, "Structured light in sunlight," in *2013 IEEE International Conference on Computer Vision*, Dec 2013, pp. 545–552.

[18] A. Whitehead, R. Laganiere, and P. Bose, "Temporal synchronization of video sequences in theory and in practice," in *Proceedings of the IEEE Workshop on Motion and Video Computing (WACV/MOTION'05) - Volume 2 - Volume 02*, ser. WACV-MOTION '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 132–137. [Online]. Available: http://dx.doi.org/10.1109/ACVMOT.2005.114

[19] M. N. Polyanskiy, "Refractive index database," https://refractiveindex.info, accessed on 2017-04-13.

[20] M. Zheng, Y. Kashimori, O. Hoshino, K. Fujita, and T. Kambara, "Behavior pattern (innate action) of individuals in fish schools generating efficient collective evasion from predation," *Journal of Theoretical Biology*, vol. 235, no. 2, pp. 153 – 167, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022519305000056

[21] N. Miller and R. Gerlai, "From schooling to shoaling: Patterns of collective motion in zebrafish (danio rerio)," *PLOS ONE*, vol. 7, no. 11, pp. 1–6, 11 2012. [Online]. Available: http://dx.doi.org/10.1371%2Fjournal.pone.0048865

[22] FishForPharma. (2017) Why zebrafish? [Online]. Available: http://www.fishforpharma.com/why-zebrafish-17

[23] Matthew Seymour. Zebra danio - the care, feeding and breeding of zebra danio fish. [Online]. Available: http://aquariumtidings.com/zebra-danio-care/

[24] S. H. Wang, J. W. Zhao, and Y. Q. Chen, "Robust tracking of fish schools using cnn for head identification," *Multimedia Tools and Applications*, pp. 1–19, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11042-016-4045-3

[25] P. J. Clark and F. C. Evans, "Distance to nearest neighbor as a measure of spatial relationships in populations," *Ecology*, vol. 35, no. 4, pp. 445–453, 1954. [Online]. Available: http://www.jstor.org/stable/1931034

[26] M. Betke and Z. Wu, "Data association for multi-object visual tracking," *Synthesis Lectures on Computer Vision*, vol. 6, no. 2, pp. 1–120, 2016. [Online]. Available: http://dx.doi.org/10.2200/S00726ED1V01Y201608COV009

[27] Z. Wu, N. I. Hristov, T. H. Kunz, and M. Betke, "Tracking-reconstruction or reconstruction-tracking? comparison of two multiple hypothesis tracking approaches to interpret 3d object motion from several camera views," in *Proceedings of the IEEE Workshop on Motion and Video Computing (WMVC)*, December 2009, pp. 1–8.

[28] N. Miller and R. Gerlai, *Automated Tracking of Zebrafish Shoals and the Analysis of Shoaling Behavior*. Totowa, NJ: Humana Press, 2012, pp. 217–230. [Online]. Available: http://dx.doi.org/10.1007/978-1-61779-597-8_16

[29] A. Perez-Escudero, J. Vicente-Page, R. C. Hinz, S. Arganda, and G. G. de Polavieja, "idtracker: tracking individuals in a group by automatic identification of unmarked animals," *Nature Methods*, vol. 11, pp. 743 – 748, 2014.

[30] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2007.09.014

[31] "(faster) non-maximum suppression in python," http://www.pyimagesearch.com/2015/02/16/faster-non-maximum-suppression-python/, visited 2017-28-05.

[32] G. Bianco, A. Gallo, F. Bruno, and M. Muzzupappa, "A comparative analysis between active and passive techniques for underwater 3d reconstruction of close-range objects," *Sensors*, vol. 13, no. 8, pp. 11 007–11 031, 2013. [Online]. Available: http://www.mdpi.com/1424-8220/13/8/11007

[33] A. P. Silvatti, F. A. S. Dias, P. Cerveri, and R. M. Barros, "Comparison of different camera calibration approaches for underwater applications," *Journal of Biomechanics*, vol. 45, no. 6, pp. 1112 – 1116, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S002192901200019

[34] L. Kang, L. Wu, and Y.-H. Yang, "Experimental study of the influence of refraction on underwater three-dimensional reconstruction using the svp camera model," *Appl. Opt.*, vol. 51, no. 31, pp. 7591–7603, Nov 2012. [Online]. Available: http://ao.osa.org/abstract.cfm?URI=ao-51-31-7591

[35] "Dynamic 3d capture of swimming fish by underwater active stereo," *Methods in Oceanography*, vol. 17, pp. 118 – 137, 2016, special section on Novel instrumentation in Oceanography: a dedication to Rob Pinkel. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2211122015300074

[36] Y. Kwon and J. B. Casebolt, "Effects of light refraction on the accuracy of camera calibration and reconstruction in underwater motion analysis," *Sports Biomechanics*, vol. 5, no. 2, pp. 315–340, 2006. [Online]. Available: http://dx.doi.org/10.1080/14763140608522881

[37] K. Müller, J. Schlemper, L. Kuhnert, and K. D. Kuhnert, "Calibration and 3d ground truth data generation with orthogonal camera-setup and refraction compensation for aquaria in real-time," in *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 3, Jan 2014, pp. 626–634.

[38] S. Henrion, C. W. Spoor, R. P. M. Pieters, U. K. Müller, and J. L. van Leeuwen, "Refraction corrected calibration for aquatic locomotion research: application of snell's law improves spatial accuracy," *Bioinspiration and Biomimetics*, vol. 10, no. 4, p. 046009, 2015. [Online]. Available: http://stacks.iop.org/1748-3190/10/i=4/a=046009

[39] P. D. V. Buschinelli, G. Matos, T. Pinto, and A. Albertazzi, "Underwater 3d shape measurement using inverse triangulation through two flat refractive surfaces," in *OCEANS 2016 MTS/IEEE Monterey*, Sept 2016, pp. 1–7.

[40] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, Nov 2000.

[41] J. Y. Bouguet, "Camera calibration toolbox for Matlab," http://www.vision.caltech.edu/bouguetj/calib_doc/, accessed on 2017-04-19.

[42] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 930–943, Aug. 2003. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2003.1217599

[43] A. S. Glassner, Ed., *An Introduction to Ray Tracing*. London, UK, UK: Academic Press Ltd., 1989.

[44] R. Hartley and P. Sturm, "Triangulation," 1996.

[45] "Closest point between two rays," http://morroworks.com/Content/Docs/Rays%20closest%20point.pdf, visited 2017-05-06.

[46] G. Glaeser and H.-P. Schröcker, "Reflections on refractions," *Journal for Geometry and Graphics*, vol. 4, no. 1, pp. 1–18, 2000.

[47] Noldus. Using Track3D for fish. [Online]. Available: http://www.noldus.com/innovationworks/products/track3d/fish

[48] H. S. Wu, Q. Zhao, D. Zou, and Y. Q. Chen, "Automated 3d trajectory measuring of large numbers of moving particles," *Opt. Express*, vol. 19, no. 8, pp. 7646–7663, Apr 2011. [Online]. Available: http://www.opticsexpress.org/abstract.cfm?URI=oe-19-8-7646

[49] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. [Online]. Available: http://dx.doi.org/10.1002/nav.3800020109

[50] X. E. Cheng, S. H. Wang, and Y. Q. Chen, "3d tracking targets via kinematic model weighted particle filter," in *2016 IEEE International Conference on Multimedia and Expo (ICME)*, July 2016, pp. 1–6.

[51] P. Carr and R. T. Collins, "Assessing tracking performance in complex scenarios using mean time between failures," in *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016, Lake Placid, NY, USA, March 7-10, 2016*, 2016, pp. 1–10. [Online]. Available: https://doi.org/10.1109/WACV.2016.7477617

[52] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *J. Image Video Process.*, vol. 2008, pp. 1:1–1:10, Jan. 2008.

[53] Q. Zhao and Y. Q. Chen, "High-precision synchronization of video cameras using a single binary light source," *Journal of Electronic Imaging*, vol. 18, no. 4, pp. 040 501–040 501–3, 2009. [Online]. Available: http://dx.doi.org/10.1117/1.3247860

[54] Y. Caspi, D. Simakov, and M. Irani, "Feature-based sequence-to-sequence matching," *Int. J. Comput. Vision*, vol. 68, no. 1, pp. 53–64, 2006.

[55] GoPro. (2017) Dual hero system. [Online]. Available: http://gopro.com/camera-accessories/dual-hero-system

[56] Unity. Unity - Game Engine. [Online]. Available: https://unity3d.com/

[57] Blender. Blender Homepage. [Online]. Available: https://www.blender.org/

[58] C. Dicle, O. I. Camps, and M. Sznaier, "The way they move: Tracking multiple targets with similar appearance," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[59] X. Wang, H. Gong, H. Zhang, B. Li, and Z. Zhuang, "Palmprint identification using boosting local binary pattern," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, ser. ICPR '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 503–506. [Online]. Available: http://dx.doi.org/10.1109/ICPR.2006.912

[60] C. Rao, A. Yilmaz, and M. Shah, "View-invariant representation and recognition of actions," *International Journal of Computer Vision*, vol. 50, no. 2, pp. 203–226, 2002. [Online]. Available: http://dx.doi.org/10.1023/A:1020350100748

[61] J. M. Rubin and W. Richards, "Boundaries of visual motion," 1985. [Online]. Available: http://hdl.handle.net/1721.1/5616

Appendices

A Fish Simulation

The following contains the worksheets for the discarded fish simulation scene.

A.1 Fish Simulation

The following section details the setup used for simulation of fish moving in a 3D space. The fish simulations are used to evaluate the performance of the tracker in addition to evaluation based on data from real fish.

An advantage of the simulated data is that the exact 3D position (i.e. x,y,z) for each fish is known at all times, even in cases of occlusion. The simulated data is hence suited for evaluation of the precision of the 3D position of the fish as estimated by the tracking algorithm. Doing a similar test with real data would be cumbersome as it is hard to manually annotate 3D bounding boxes. Another benefit of avoiding manual annotation is the time saved.

Another benefit of using a simulation is the full control of both environment of test subjects. It is for instance possible to vary the visibility of the water with a few clicks in the simulation whereas this process would be harder to control in a real aquarium. It is also possible to easily change between different test setups, e.g. changing the amount of fish and/or the placement of the cameras. Yet another benefit is full control of the behavior of the fish, e.g. their movement speed, which would be near impossible to control in real life.

The main drawback of relying on a simulation for evaluation purposes is the completeness, or rather incompleteness, of the simulation. This incompleteness can either be the result of some real world phenomenons being to complex to model or simply overlooking certain aspects in the simulation.

An algorithm may therefore perform better when testing on the simulated data, as problematic aspects may have been omitted from the simulation. For instance the movements in the water surface caused by the fish may not be a part of the simulation due to the complexity of modeling this behavior. Relying solely on simulations for evaluation purposes is hence not without its problem. This is also the reason why real data is used during evaluation as well in this report.

The simulations of fish are created using the Unity 5 game engine. Unity 5 is chosen as the basis for the simulations as it is free, widely used and have good community support.

A.1.1 3D Fish Model

The simulated fish are based on 3D model of a sardine from Unity's asset store. The model was mainly chosen as it was free and due to its fluid animations. The used model contains animations for both turning left and right. The mouth and fins of the fish are animated as well.

Variation in the simulated fish is created by varying their size and their shade of coloration. The size is varied in all three dimensions, resulting in both thin and more stout specimens. Show examples of different specimens?

A.1.2 Fish A.I.

A simple A.I. (artificial intelligence) is implemented for the simulated fish. The main purpose of the implemented A.I. is to ensure that the fish moves around within a virtual

fish tank in order to generate a basic for tracking.

The movement is created by having the A.I. in each fish randomly pick a new destination, which it will start moving towards. The A.I. will select a new destination with a probability of 10% and the fish may therefore change destination before arriving at its current target destination. The speed at which the fish moves about and the speed of its animations are randomly chosen when spawning the fish.

The fish A.I. does also include collision avoidance to ensure that the fish stays within their virtual tank and to avoid collisions with other fish. The main purpose of the collision avoidance is to prevent clipping issue, where a fish may pass through another fish or through the sides of the tank. An example of a fish clipping through another fish is shown in Figure A.1.

Figure A.1: Example of a clipping issue resulting in an unrealistic scenario where two fish intersects.

The collision avoidance is implemented using Unity's built-in box colliders, which are applied to the fish and the sides of the virtual fish tank. Intersecting box colliders will trigger a response in the fish A.I. causing it to veer away from the colliding object. The box collider for a fish can be seen in Figure A.2.

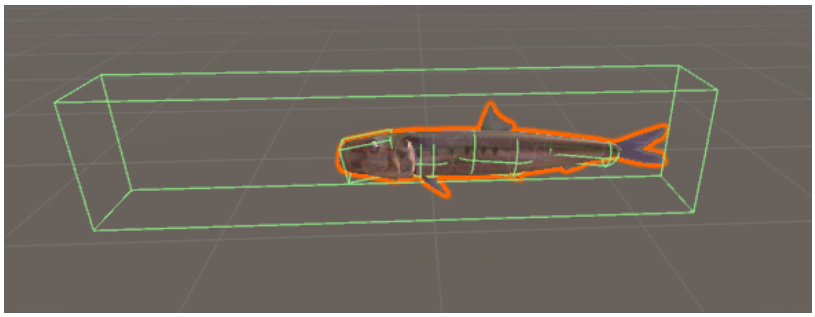


Figure A.2: The box collider for the 3D fish model.

Note that the box collider is extended at the front/head of the fish model. This is done to allow the the collision avoidance to look ahead so the fish can start turning before it collides with an object.

An example of a CSV snippet from a simulation is shown in Table A.1. Note that only

two ids (-263058 and -263792) are present in the CSV snippet as the simulation contained two fish. The fish ids are based on the instance id of each fish object in Unity, which are guaranteed to be unique.

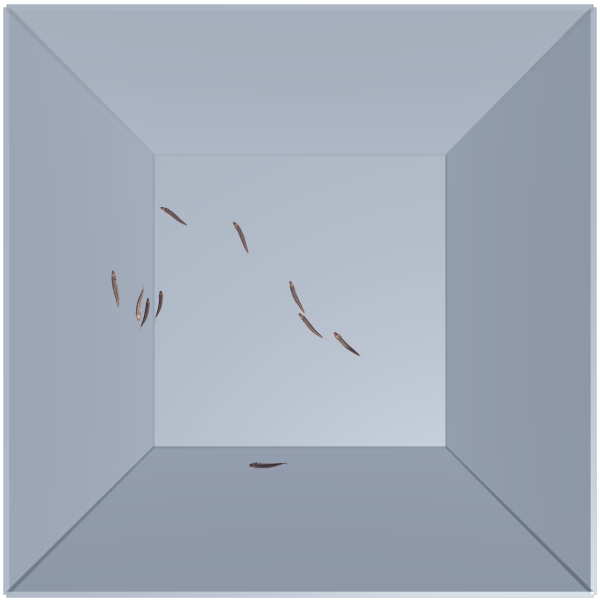
Frame count	Fish id	x	y	z
33	-263058	-1.6	0.0	0.3
33	-263792	0.0	0.0	0.0
34	-263058	-1.6	0.0	0.3
34	-263792	0.0	0.0	0.0
35	-263058	-1.7	0.0	0.4
35	-263792	0.0	0.0	-0.1

Table A.1: Snippet from CSV file created during a simulation of two fish.

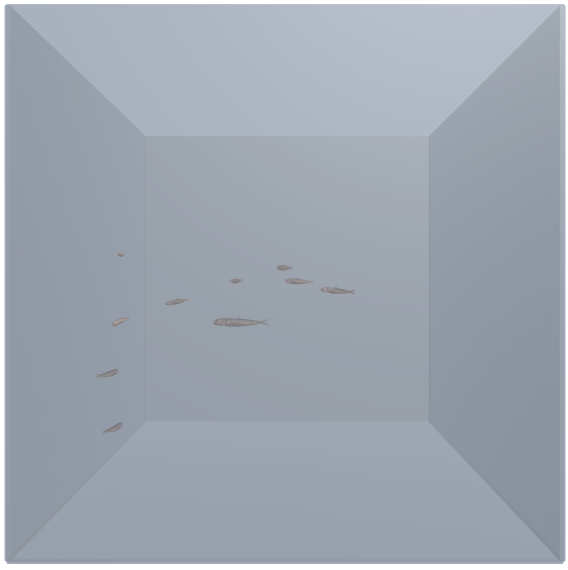
The video feed for each camera is generated by having the each camera render to a texture instead of rendering to the screen. The content of this render texture is then saved to an image file (.jpeg) before the next frame is rendered. The result is an image sequence for each camera, where each image corresponds to a certain frame in the simulation.

An example of a single frame from a simulation, but viewed from two different cameras, is shown in Figure A.3. The left-most image originates from a camera looking down into the tank and the right-most image is from a camera looking at the side of the tank.
The video feeds from the cameras (i.e. the image sequences) and the data in the CSV file are aligned using the frame counter in Unity. This is possible, as each entry in the CSV file contain the frame count (as shown in Table A.1) and the frame count appears in the naming of the images from the cameras.

The framerate of the simulation is controlled as well. This is done using the `targetFrameRate` parameter found in Unity.



(a) Camera with top-down view.



(b) Camera with side-looking view.

Figure A.3: Example of the same frame from a simulation viewed from two different cameras.

B Spatio-Temporal Alignment using Curvatures

This is the worksheets for a spatio-temporal alignment module for a system using the tracking-reconstruction approach. It was not implemented as the reconstruction-tracking approach was chosen instead.

B.1 Spatio-Temporal Alignment

The motion of an object can be described by its change in position over time and according to [60], the variations in velocity and acceleration are view invariant and can be used for recognition. This gives the possibility for aligning the recorded fish sequences spatio-temporally if the motion trajectory of the same fish can be extracted from both views.

The motion trajectory of a fish in three dimensions can be represented by a vector

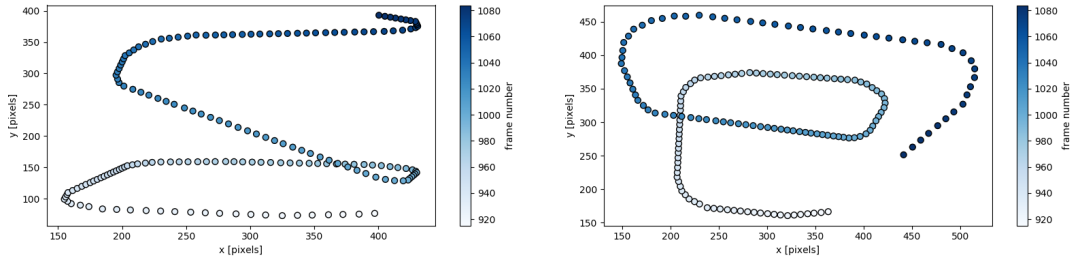
$$r(t) = [x(t) \quad y(t) \quad z(t)], \tag{B.1}$$

where x, y and z represent the position of the fish in the respective dimensions at time t. The motion trajectory captured in a sequence of images is the 2D projection of a 3D trajectory and it is therefore not possible to precisely determine the position in all three dimensions based on images from a single view-point.

However, every point in the projection represents the position of the object at a given time and a spatio-temporal trajectory can therefore be expressed by the equation

$$p(t) = [x(t) \quad y(t) \quad t], \tag{B.2}$$

where $z(t) = t = \{1..n\}$ is the frame number in the given sequence of images and n is the total number of frames. An illustration of a 2D trajectory seen from two different angles is shown in Figure B.1. In Figure B.1a, the camera is looking into the aquarium from the side and in Figure B.1b, the camera is looking down directly above the aquarium.



(a) Cam front trajectory. **(b)** Cam top trajectory.

Figure B.1: Trajectory

The motion trajectory of the fish can be described by its curvature, which in the given case is an expression for the changes in velocity and acceleration [61]. For a three dimensional case, the curvature is given by

$$k = \frac{\sqrt{(z''y' - y''z')^2 + (x''z' - z''x')^2 + (y''x' - x''y')^2}}{(x^2 + y^2 + z^2)^{\frac{3}{2}}}, \tag{B.3}$$

where $x = x(t)$, $y = y(t)$ and $z = z(t)$. However, as the first and second order derivative of p is equivalent to velocity and acceleration, respectively, the expression can be written as

$$k = \frac{\sqrt{y''^2 + x''^2 + (y''x' - x''y')^2}}{(x'^2 + y'^2 + 1)^{\frac{3}{2}}} = \frac{\|p' \times p''\|}{\|p'\|^3} = \frac{\|v \times a\|}{\|v\|^3}, \tag{B.4}$$

where $v = p'$ is the velocity, $a = p''$ is the acceleration and $\times$ symbolise the cross-product.

However, as it is not possible to detect movement in the direction that is orthogonal to the image plane, the 2D representation of the curvature may not be optimal due to the camera setup, which is illustrated in Figure B.2.

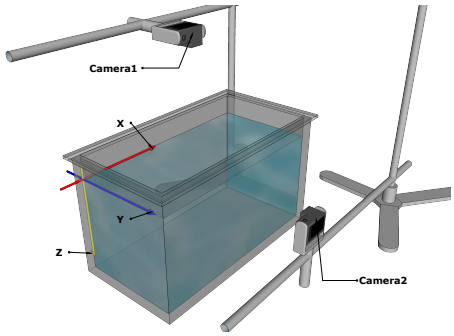
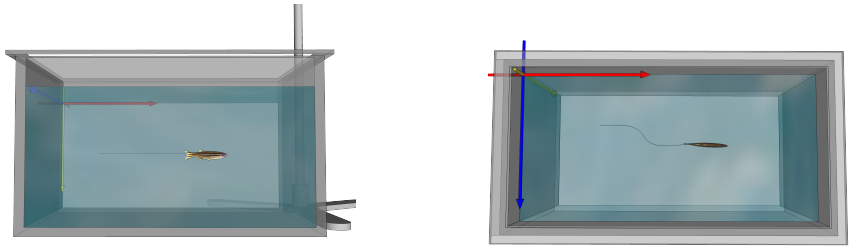


Figure B.2: Above and beside camera setup.

The camera placed above the aquarium, $cam1$, captures information with regard to the x- and y-axis of the coordinate system spanning the aquarium. This is illustrated in Figure B.3a, where the motion trajectory of a fish is seen from above by $cam1$. On the other hand, the camera looking into the aquarium from the side, $cam2$, captures information with regard to the x- and z-axis. Therefore, the two cameras only share information about the movement of the fish along the x-axis as seen in Figure B.3b, which resembles the same trajectory seen from the side.



(a) View from $cam1$.

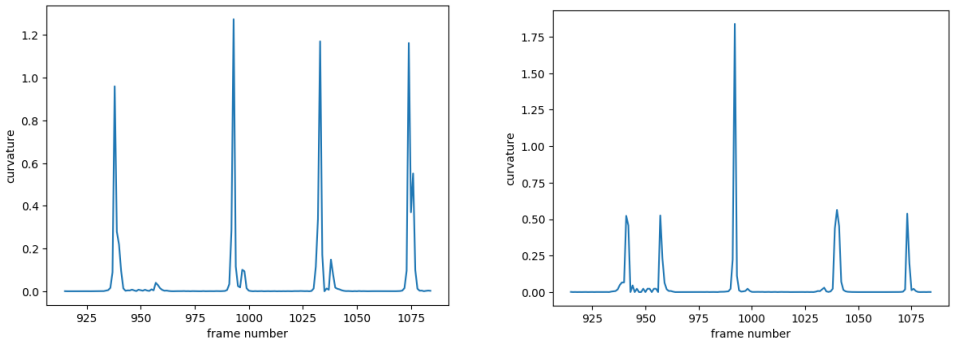
(b) View from $cam2$.

Figure B.3: The motion trajectory seen from the two cameras.

The relation between velocity and acceleration of the respective dimensions are kept between the two views. However, as explained above, the most pronounced variations can be found in the x-direction. Due to this, the one dimensional temporal curvature with regard to the x-axis is found using the equation

$$k_x = \frac{|x''|}{(x'^2 + 1)^{\frac{3}{2}}}, \tag{B.5}$$

which is derived from Equation B.4, by letting $y(t) = 0$. An example of the one-dimensional curvature can be seen in Figure B.4, where the two graphs illustrates the same motion trajectory seen from the front and top, respectively.



(a) 1D curvature - front camera. **(b)** 1D curvature - top camera.

Figure B.4: Curvature

The temporal alignment of the video sequences can be determined using cross-correlation to measure the similarity. The cross-correlation is given by

$$(k_t \star k_f)[i] = \sum_{n=0}^{N} k_t[n] k_f[n+i], \tag{B.6}$$

where k_t and k_f are the curvature of the top and front camera, respectively, $i = \{0..N\}$ is the displacement and N is the length of the vector, k_t or k_f, with most parameters.

The cross-correlation of the two curvatures given in the example above can be seen in Figure B.5, where the horizontal axis is the frame displacement and the vertical axis is the similarity between the two vectors. In the given example, the similarity is highest when the displacement is equal to half the length of the longest vector, which means that the temporal alignment between the two vectors fit. If this was not true and the similarity, for example, was higher at $\frac{N}{2} - 10$, one of the trajectories should be shifted 10 frames, before they were temporally aligned.

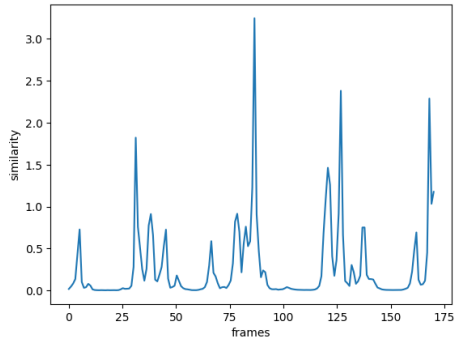


Figure B.5: Correlation.

C Datasets

C.1 AB-1080-60-6f

Figure C.1: Frame 254. Top camera.

Figure C.2: Frame 254. Front camera.

C.2 AA-1520-60-7f

Figure C.3: Frame 100. Camera 1.

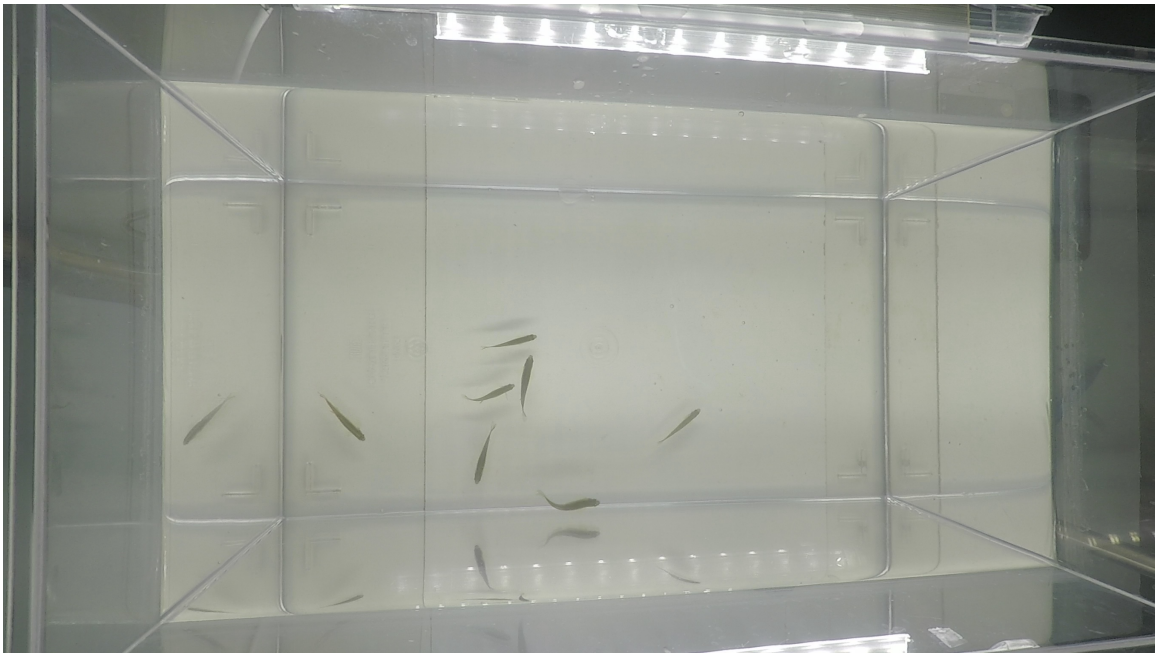


Figure C.4: Frame 100. Camera 2.

C.3 Speedtest

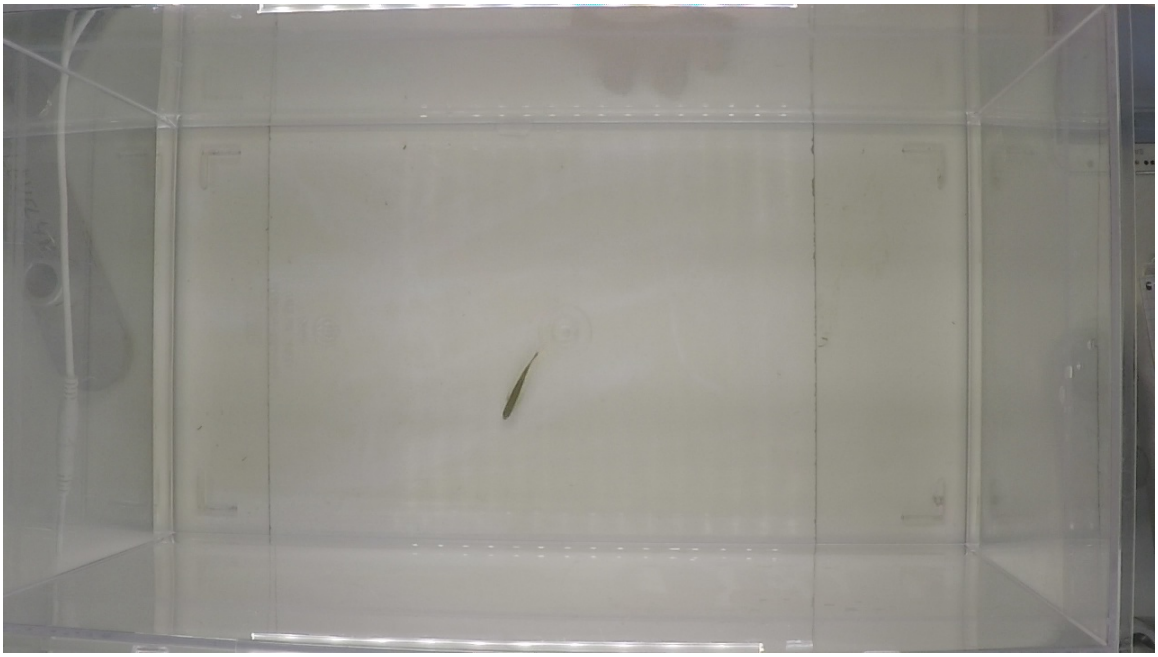


Figure C.5: Frame 2500.

C.4 D14

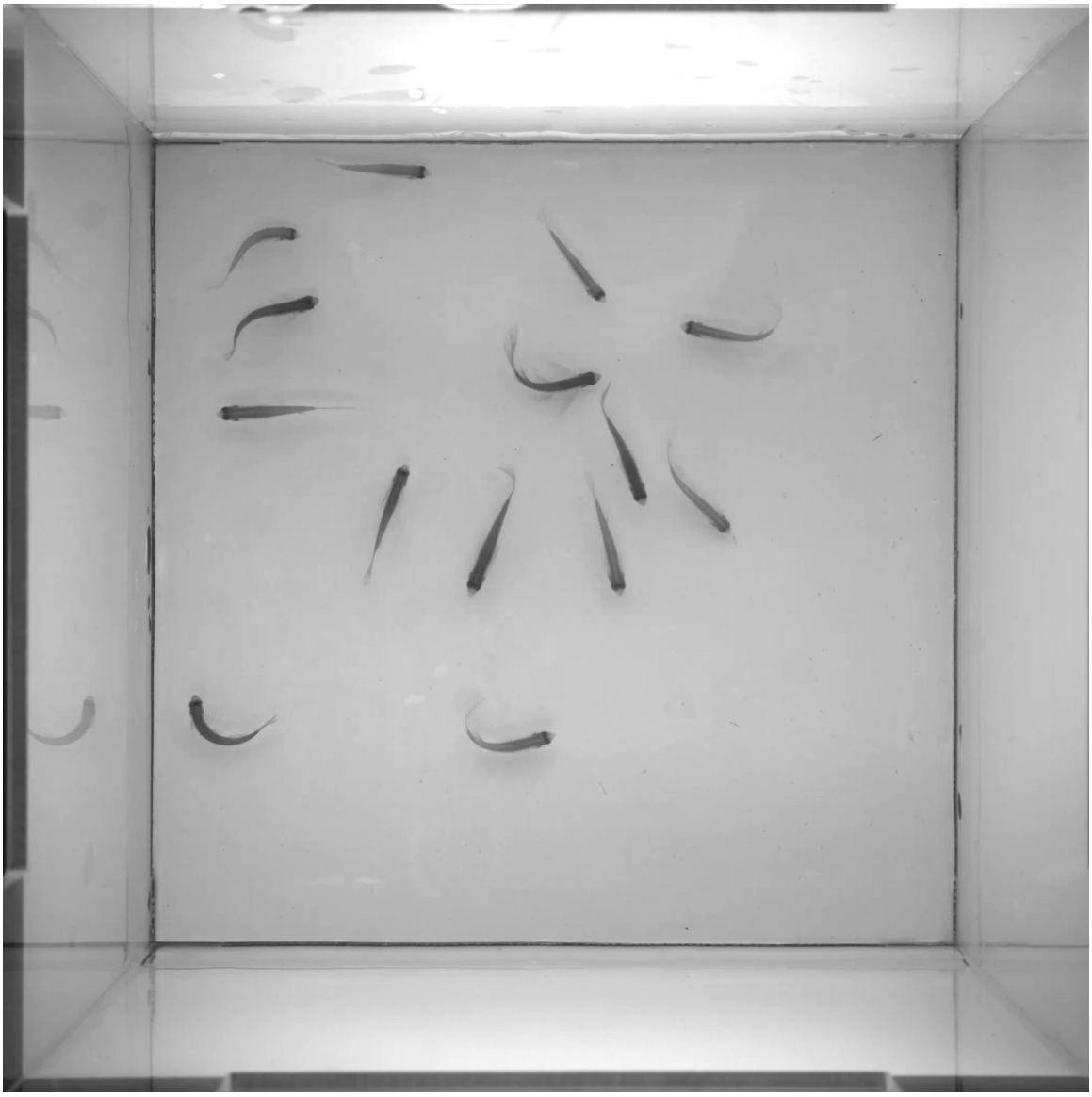


Figure C.6: Screenshot at 14 seconds.