Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Gaute Vatne Nybø

Estimation of Structural Geometry

Sensors and Software to Generate Real-time Maps of Structural Geometry in Fish Farms

Master's thesis in Cybernetics and Robotics
Supervisor: Annette Stahl, ITK
June 2019

NTNU
Norwegian University of Science and Technology

Gaute Vatne Nybø

Estimation of Structural Geometry

Sensors and Software to Generate Real-time Maps of Structural Geometry in Fish Farms

Master's thesis in Cybernetics and Robotics
Supervisor: Annette Stahl, ITK
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

PROJECT DESCRIPTION SHEET

Name:	Gaute Vatne Nybø
Department:	Engineering Cybernetics
Thesis title (Norwegian):	Estimatering av strukturgeometri
Thesis title (English):	Estimation of Structural Geometry

Thesis Description: The ARTIFEX project is developing technologies for future remote operations at fish farms where the main objective is to develop robots for regular remote inspection, maintenance and repair operations without onsite personnel.

The goal of this project is to develop reliable detection, 3D recovering and mapping algorithms enabling the ARTIFEX remote system to carry out a safe zone mapping of the environment based on the autonomous unmanned aerial vehicle (UAV) vision system.

The following tasks should be considered:

1. Review image processing algorithms capable of detecting the key structures in the cage system, given in the video recordings.
2. Review techniques available to preform recovering of 3D world coordinates.
3. Develop software capable detecting the geometrical structure given the estimated 3D world coordinates.
4. Define methods for carrying out safe zone mapping.
5. Test the developed algorithms on the data recordings.
6. Discuss results and error sources.
7. Conclude your results and suggest future work.

Start date:	2019-01-07
Due date:	2019-06-24
Thesis performed at:	Department of Engineering Cybernetics, NTNU
Supervisor:	Professor Annette Stahl, Dept. of Eng. Cybernetics, NTNU
Co-Supervisor:	Research Scientist Walter Caharija, SINTEF Ocean AS

Abstract

This thesis is a part of the Artifex research project at SINTEF Ocean AS. The project aims at developing a general-purpose system used for remote operations at fish farms in order to reduce the need for onsite personnel. The operations include remote inspection, maintenance, and repair operations. To achieve the remote operation ability, a method for observing the physical terrain needs to be developed. This study presents a method for mapping the terrain from digital images taken from air.

To enable these visual capabilities of the remote inspection system a UAV is used. The UAV is taking visual, internal, and GPS measurements. The visual measurements are acquired from the camera mounted at the bottom of the UAV, perceiving the location in a bird's eye view. Based on these measurements there is developed a computer vision based approach to create a digital map of the aquaculture production site.

The digital map can be then used to, for instance, analyze the structural integrity of the farm and to define safe operational areas on the surface as well as underwater for the Artifex USV and the ROV, respectively. This work proposes a method to define such safe operational zones for the USV.

To achieve this, a grid map representation method to model the sea surface is used, where the grid consists of 2-dimensional cells indicating if the cell is traversable or not. To keep this map of safe sones up to date, the map is continuously updated during operation.

The method proposed is based on dividing the problem into three parts. Firstly the objects of interest are detected based on known attributes. Second, the detected image points are projected down on to the water surface. Thirdly the world coordinates are used to generate a safety line, and in turn, updating the map.

Sammendrag

Denne oppgaven er en del av et forskningsprosjekt kalt Artifex, på SINTEF Ocean AS. Prosjektet tar sikte på å utvikle et generelt system som brukes til fjernoperasjoner på fiskeanlegg for å redusere behovet for tilstedeværende personell. Operasjonene inkluderer fjern-observasjoner, vedlikehold og reparasjoner. For å oppnå den fjerne operasjonsevnen må en metode for å observere det fysiske terrenget utvikles. Denne studien presenterer en metode for å kartlegge terrenget fra digitale bilder tatt fra luften.

For å aktivere disse visuelle evner i det fjernstyrte inspeksjonssystemet, brukes en UAV (fjernstyrt, ubemannet farkost). UAVen bruker visuelle, interne og GPS-målinger. De visuelle målinger er anskaffet fra kameraet montert nederst på UAVen som senser stedet i fugleperspektiv. Basert på disse målingene er det gjort en datasynbasert tilnærming for å lage dette digitale kartet over akvakulturproduksjonsstedet.

Det digitale kartet kan brukes til å analysere anleggets strukturelle integritet og å definere sikre driftsområder både på overflaten og undervann for henholdsvis Artifex USV og ROV. Dette arbeidet foreslår en metode for å definere slike sikre driftssoner for USVen.

For å oppnå dette benyttes et rutekart som representasjonsmetode for å modellere havflaten. Der rutenettet består av 2-dimensjonale celler som indikerer om cellen er traversabel eller ikke. For å holde dette kartet over sikre soner oppdatert, oppdateres kartet kontinuerlig under drift.

Metoden som foreslås er basert på å dele problemet i tre deler. Først oppdages gjenstandene av interesse basert på kjente attributter. Så projiseres de oppdagede bildepunktene ned til vannoverflaten. Til slutt brukes verdens koordinatene til å generere en sikkerhetslinje, og kartet blir oppdatert.

Preface

This thesis concludes my Master of Science degree at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology. The study has been completed in collaboration with SINTEF Ocean AS as part of an ongoing research project.

The gathering of aerial data was done in cooperation with Maritime Robotics AS. Where Maritime provided the drone and pilot to operate. The data were collected in the fall of 2018 as a part of this thesis pre-study. Description of the equipment used is included in Chapter 1.

The result of this study is largely in the form of software implementations. These implementations are to a large degree developed independently, with exceptions in the form of some third-party open source software libraries who are found listed in the introductory chapter.

I would like to thank my supervisors Annette Stahl and Walter Caharija for their guidance and help throughout the project.

Gaute Vante Nybø
Trondheim, 2018-06-24

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Aim of study and ARTIFEX

The ARTIFEX project is developing game changing technologies for future remote operations at fish farms where the main objective is to develop robots for regular remote inspection, maintenance and repair operations without onsite personnel. An Unmanned Surface Vehicle (USV) is used as a platform for carrying a Remotely Operated Vehicle (ROV) for underwater operations as well as a Remotely Piloted Aircraft Systems (RPAS) for airborne inspection tasks (see Fig 1.1). The USV will travel between different aquaculture sites and its land base. The project results will yield new and ground-breaking products and services that will unlock unmanned operations in aquaculture hence minimizing the risk for personnel. This will also expand the weather window for operations on remote and exposed sites. The various subsystems are cur-

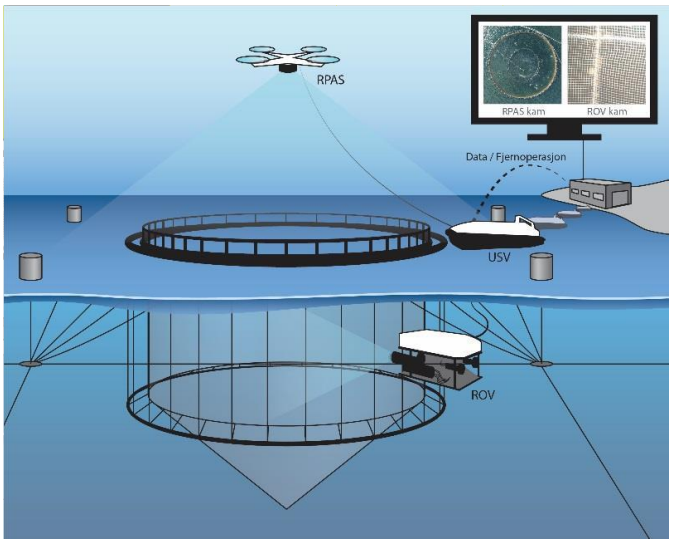


Figure 1.1: llustration of the ARTIFEX concept (A.M. Lien, SINTEF Ocean AS)

rently integrated into a prototype for full scale testing and validation. The project partners are: Maritime Robotics AS, Argus Remote Systems AS, Lerow AS, NTNU and WavEC.

This study is part of ARTIFEX and its main purpose is to generate a static map of the fish farm where the ARTIFEX USV is operating, by means of processing the images and videos obtained by the RPAS system. The overall goal of the visual system of the ARTIFEX project is to map the safe zones where the USV, the RPAS and the ROV can operate without the risk of collision or tethering. Therefore, the developed algorithm shall determine the georeferenced

coordinates of the key structures of the fish farm such as the buoys and the cages, that can be used for determining the position of the mooring lines underwater. The tasks of the project are closely coupled with activity 3.3 of the ARTIFEX project where the goal is to generate a high-resolution dynamic of 3 panorama picture and detect the objects of interest of the fish farm under inspection. A preloaded map is to be given to the USV prior to the commencement of the ARTIFEX operation describing the last known status of the geometry. The map is then updated upon arrival at the location and before the mission starts.

Figure 1.2: Rataren fish farm, SINTEF ACE full scale laboratory, picture taken from RPAS

In this context, the study is aimed at furthering development of the vision system of the ARTIFEX project. Based on the data recordings done at Korsneset 2018 (Fig 1.3). The study is aimed at enhancing understanding and enabling the visual capabilities of the system and builds on experience acquired from the specialization project done in fall 2018 [1].

Figure 1.3: Korsneset fish farm, west location, picture taken from ground

1.2 Structural geometry

To define the structural geometry of an aquaculture production site it can be convenient to look at the words making up the phrase. The geometrical shape is the geometrical information of an object that remains when location, scale, orientation, and reflection are removed from the description of geometric objects. While a structure is defined as the arrangement and organization of connected elements in a material object or system. Therefore it can be said that the structural geometry of an object or system is the structure of these geometrical objects. To obtain an understanding of the structural geometry of the relevant system it would be advantageous to investigate how a standardized fish cage system is structured. In Figure 1.4 a common way to structure the production site system can be viewed.

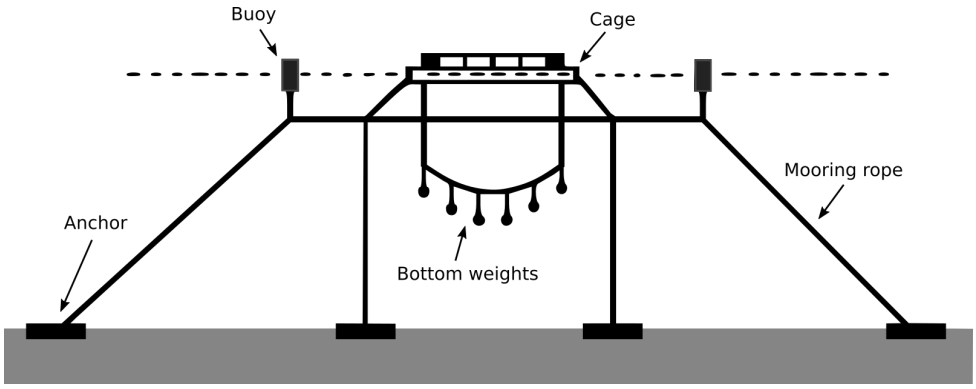


Figure 1.4: Structural cross section of the agricultural system

However, the objects found at one agricultural growing facility are vastly various. One cannot expect the geometries making up the structure of the system to explicitly be found at an arbitrary site, there might be several objects distorting the one's view of the structural scene. These types of objects can be maintenance boats (Fig 1.5) and the belonging personnel, birds or other objects floating in the sea. Hence there is a need to define what structures are of importance to the visual system. The objects omitted from this section will be considered to not belong to the scene. If present in the scene, it will be viewed as an unwanted disturbance to the vision system.

Figure 1.5: Korsneset fish farm, west location, picture taken from RPAS

There are mainly four structures that a given site consistently contains one considers the view from above. That is the buoy, cage, feeding pipe and feeding station as seen depicted in Figure 1.6. To further investigate what objects can be of importance we would need to define some define their characteristics. The objects can be divided into groups. Being objects containing the property of rigid bodies, and objects that deform with respect to time. This distinction is vital when choosing what structures to model in the map.

Figure 1.6: Objects commonly found at the site perceived from an aerial view

The rigid body is a solid body in which deformation is zero or so small it can be neglected. The distance between a pair of two points in the body remains constant when external forces are exerted. This cannot be said for the feeding pipe, so we define it to be non-rigid. The deformation happening on the feeding pipe itself will make for a very challenging mapping problem. Thus, this object can be said to not be of interest in mapping of the scene.

The remaining three objects are considered to be of the rigid body type. The rigid body property can be argued to be approximately true for the purpose of this report. However, the assumptions are quite strong. The motivation for making these assumptions will be discussed in more detail in Chapter 2.

Moreover, the transformative behavior with respect to the world frame on the rigid bodies themselves needs to be considered. As seen from Figure 1.4 the cage and buoy are anchored. If we furthermore consider good weather conditions and small variations in the water currents we get small variations in the transformative behavior of the three remaining objects.

Having quantified properties of the objects the scene is inheriting. It is important to consider what topologies that the given set of objects has. From looking at different agricultural sites on Norgeskart [2] it can be seen that the buoys and cages have a topological order that occurs often. That is the four buoys line creates a rectangle with the cage in the middle. This structure is often extrapolated to form a collection of cages with a set distance relating to each other (Fig 1.7).

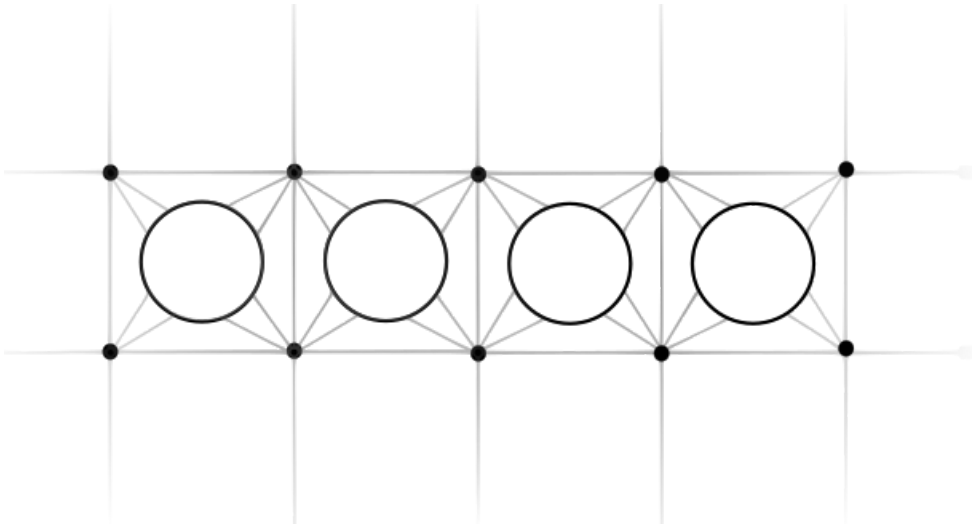


Figure 1.7: Model of the structural geometrical shape

The feeding station is hard to generalize in the same topological sense. Moreover, the feeding station is not of immediate importance to the remote operation system described earlier. Thus, we are not only omitting the feeding pipe as an object of interests but also the feeding station.

Until now it has been considered four objects that are redundantly perceived by at an aerial vehicle overseeing an aquacultural growing facility. Though these are four objects that are important to the daily operation at the site. It has been argued that they are only a subset of the objects that are of importance to the structural geometry of the fish caging system and the remote operation. These are the buoys and the cage. Furthermore, the uniqueness of this subset is that objects can be approximated as rigid body structures and having a static motion for the purpose of this study.

1.3 Safe zone mapping

The main idea behind the vision system outputting a map of the safe zones is to be able to communicate the world referenced map to the other vehicles involved in the operation. For this information to be useful, the aerial system needs to obtain a priori information of the terrain. This is done by making the UAV hover a distance in front of the USV and UAV. With the proper field of view and the proper operation altitude, the vision system can communicate a priori information of the farm's structure and its estimated zones of safety, so that the risk of collision is minimized.

Having identified objects making up the main structural geometry and their corresponding location, a map is to be constructed. Together with the safe zones of operation there exists an additional constraint in the perspective of the autonomous vehicles remotely operating. This is the power/ communication line. The constant being defined by the length of its cable. In Figure 1.8 both the cable constraint (painted in green for the AUV) can be seen together with the safe zone map. Combining these two constraints defines the set of the traversable cells that the vehicle is able to move freely within.

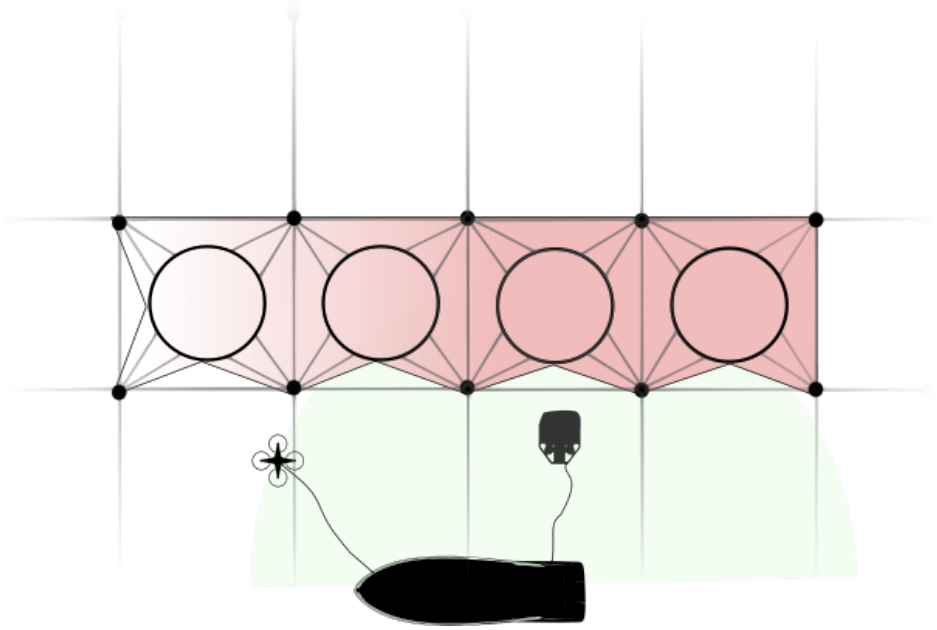


Figure 1.8: Model the real-time mapping

Considerable complications are involved in outputting such a map of high accuracy to the underwater vehicle, it can be argued that it may be impossible for the RPAS system to obtain such information with satisfactory confidence. However, there exists a possibility of constructing the underwater geometry by assuming some static model for the underwater geometry. Though, the practice is not standardized to the extent that could be viewed useful for the autonomous operational system [3] without loosing generally. However, there can be found a mooring line report that is available for every single aquaculture site, this would help in generalizing the underwater mapping system.

With that said, underwater geometry is not the focus of this study. The emphasis is aimed at getting georeferenced "anchor points" of the structures perceived. As pointed out before,

having this digitalized map, there exists a number of possible ways to build systems on top of this information structural georeferenced data.

1.4 Equipment and sensors

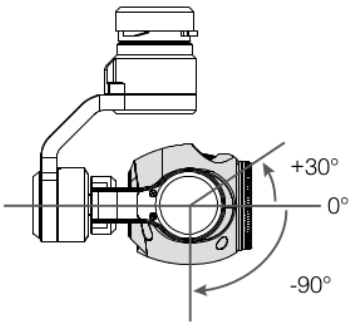
This section describes what equipment and sensors that were used for recording data at Korsneset aquacultural growing facility. The data set may differ from what is meant for the final version of the remote operation system, as the full system described in section 1.1 is in prototyping and not ready for full scale testing. The equipment specifications are obtained from the official DJI site, that contains comprehensive documentation [4, 5].

The equipment made it possible for the drone to move freely without the constraint given by a data/power line. The altitude of the drone could be set in the software during the flight, controlling the vehicle with considerable accuracy according to the sensor data. To control its longitude and latitude the drone operator was able to input velocity references in the direction of travel. Before the start of the recording, the drone was set to a fixed altitude and where approximately moving in the plane of the height specified.

Model name	MATRICE 600PRO
Max Pitch Angle	25°
Max Wind Resistance	8 m/s
Package Dimensions	525 mm × 480 mm ×640 mm
Weight	9.5 kg
Hovering Time	32-38 min

Figure 1.9: Drone used Korsneset 08/11/2018

The drone was equipped with a gimbal that was controlling the orientation of the camera in the three Euclidean rotational degrees of freedom. The gimbal was set to the direction aligning its z-axis. There where no change of setpoint of the gimbal during the time of flight. The visual



Model	Zenmuse X3
Controllable pitch range	-90° to +30°
Controllable yaw range	±320°

Figure 1.10: Gimbal used Korsneset 08/11/2018

data recorded was obtained by using the FC350 listed in Fig. 1.11. The device was set to video recording mode and data was outputted to an SD card of the format MOV(H.264). While the video data was being serialized, the drone continuously kept a full log of internal and GPS measurements made from the start to the end of the flight.

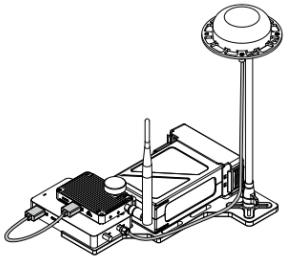
Previous to gathering the visual data the camera was set to log footage at its maximum capacity in regards to a resolution. This allowed for figuring out the optimal resolution with regards to resolution at a later stage.

Model	FC350
CMOS	Sony EXMOR 1/2.3
Resolution	4096x2160
FOV	94 deg
Lens	20mm
Vocal length	3.6mm

Figure 1.11: Camera used Korsneset 08/11/2018

The GNSS used was provided by DJI and it is of the RTK-GNSS. A system DJI is calling D-RTK high precision navigation and positioning system. The D-RTK is made customized and tuned for the DJI drones. This D-RTK positioning system is said to have centimeter-level positioning accuracy compared to a more standard GPS, IMU and barometer solution.

By its dual antennas design, its heading reference can be more accurate than a compass sensor. By providing a solution more robust to withstand magnetic interference from metal structures.



Positioning Accuracy	Horizontal: 1 cm + 1 ppm Vertical: 2 cm + 1 ppm
Orientation Accuracy	(0.2/R)° R is the baseline distance in meters.
Root Mean Square (RMS)	0.03 m/s
Frequency Used	GPS L1&L2, GLONASS F1&F2
Operating Temperature	0° to 45°C

Figure 1.12: GNSS system used Korsneset 08/11/2018

Moreover, barometers commonly found in UAV systems may suffer from altitude discrepancies when faced with fluctuations in airflow, such as braking, or after extended flying. The positioning system omits many of these discrepancies, by calculating high precision position estimates of its position.

1.5 Software

There is no shortage of libraries that contain an assortment of useful image processing algorithms. For processing the images there was a consideration of using the utilities and the convenience of these. It can save implementation time at the cost of not having full control of the implementation of how the algorithm runs. Therese libraries are quite extensive and are containing most of the most common algorithms. However, some specifics algorithms are not yet added. For these cases, the work Matlab has been used for its excellent debugging mode and ease of use, and later ported to either C++ or Python.

Figure 1.13: Image processing software, scimage and opencv respectively

OpenCV is an abbreviation for open computer vision . OpenCV is an open-source computer vision and machine learning software library. Officially launched in 1999, the OpenCV project has a long development history and is highly optimized in many aspects. Being originally developed in the C/C++, language, it has since release been opting for support in programming languages like Python and Java.

Skimage shorthand for Scikit-image, providing a library for image processing written for Python. The library is a part of the collection Scikit. Witch again is short for SciPy Toolkits holding libraries like Matplotlib, NumPy and other well-known libraries accessible within the Python environment.

In addition to these, a library for getting the GNNS readings and translate them into the global frame where used, called geoutilities [6]. This translation can be done through some simple equations but the convenience of using a library that has been under testing for some time seemed like a good way to go. In addition to this, there was a library that allowed for interaction between C++ and Gnuplot for real-time plotting during testing.

OpenCV has an extra repository (called OpenCV contrib) that contains extra modules. Containing modules that are quite new and often because of its age, has not undergone thorough testing. However, as the module matures and gains popularity it will often be moved over the main repository. Within the OpenCV contrib repository, the *aruco* marker library is found. This library was shown to be useful for building up the test environment. With using a single aruco marker the library was able to output an estimate of the cameras position and orientation.

Eigen, a library in C++ handling vectors and matrices was also used. While OpenCV has its version of many of the functionalities found in Eigen it made sense to not include any image processing library dependencies in the mapping module.

1.6 Dataset

There where done several recordings of data throughout the process of gathering data. Where it was tested flying at multiple different altitudes, both capturing photos at a set time interval and ones using the video recording setting on the camera system of the drone.

There where one flight of particular interest, found in the fourth flight made (flight 108). In this flight the drone was set to capture the structure of the site at an altitude of 100m. The altitude gave an satisfactory view of the fish farm cage structure while preserving the details. Making for several objects of interest being exposed fully to the camera lens at most times. Moreover, to shrink the amount of data available for testing reasons it was decided to only do an outtake of the data recorded. This was the data gathered while flying over the four first cages of the west end of the location. This flight is depicted in Figure 1.14, showing the location at large and displaying the path the drone took while capturing the set processed in this thesis.

Figure 1.14: Korsneset fish farm, picture taken from aerial vertical (photo by Norgeskart), plotted lines of the fourth flight of the drone, marking the path interest with respect to processing

The run was recorded in 4096x2160p at 24 frames per second with the encoding of MOV(H.264). The video file was converted to .png image files and downsampled to the resolution of 320x700. For the convenience of further processing, these images were converted to a grayscale representation as all the image processing algorithms reviewed uses the grayscale representation of the input image.

The corresponding internal measurement and GPS serialized to a CSV-file, containing data points corresponding to the time start of the flight. While the logging of the video, of course, started at the time of recording. Doing some hand calibration these data sets were synchronized.

Figure 1.15: Korsneset fish farm, north location, picture taken from ground

Figure 1.16: Korsneset fish farm, west location, picture taken from RPAS

1.7 Outline

As this specific case of safe-zone mapping is not widely researched as of today there is an emphasis on the surveying aspect. This would be trying to approach the problem with methods that agree with the problem description and run several tests capturing what might be worth looking into carrying out work in the future. This surveying aspect is especially true in the space of image processing as this is an inherently hard problem, thus needing attention. There are other aspects of this report that can be considered solved in a sense, within the reduced space where the assumptions are made.

In the background and theory chapter, there will be looking at the fields that are involved in solving the problem description. Brushing over some theory and background that are related to all the subfields.

The next chapter is about detecting the objects of interest. That is recognizing the objects that make of for the cage structural geometry seen from the aerial view. There will be looked at ways of both detecting the cage and buoy.

In Chapter 4 the problem of perceiving the 3D world will be looked at. This problem of perceiving the world through a lens might not intuitively seem hard, but this is one area that much research has gone into. The understanding of the extent of the problem arises when the loss of one dimension is considered. As going from the world to a digital image one loses the scale parameter. The world can be looked at as being infinite complex and capturing that from a finite-dimensional set of images is not a simple task [7]. Inferring the model of a scene would be impossible without imposing additional constraints on the problem. These methods will be looked at.

Chapter 5 is transferring the 3D landmarks observed to a grid map like structure. This encoded information is then used to generate what one may refer to as a "safety line", using this, and some extra readings the grid map utility are able to label safe zones.

Chapter 2

Background and Theory

There are several fields of study involved in solving the task of obtaining the objects of interest and constructing a map holding the safe zones of operation. To construct the map, mainly three topics of study are involved:

First, the classical field of digital image processing. There are no clear cut boundaries in the curriculum from image processing at the one end and computer vision on the other. However, a useful paradigm is to consider three types of image processing: Low-level processes that involve primitive operations such as reducing noise in an image or enhancing edges where both input and output can be characterized as being an image. Mid-level processing on images involves tasks such as segmentation (dividing the image up into regions or objects), description of those objects, and classification (recognition). Finally higher-level processing, "making sense" of an ensemble of recognized objects, here is normally where the field of computer vision starts, the same distinction will be made in this theory section.

Second, the wide field of computer vision. Within this field of study there are different subfields. An area that is of particular interest is the field of geometric Computer Vision. This includes the description of the way the appearance of objects changes viewed from different viewpoints as a function of camera parameters and the shape of the object (defined by [8]). The complexity of the world is infinitely superior to the measurements of its images. We cannot simply invert the image formation process and reconstruct the "true" scene from several images. We can reconstruct our best model, an "internal representation". This requires introducing assumptions, or hypotheses, on some unknowns about the environment, to infer the others. There are arbitrary no right or wrong way of doing so, modeling is a form of engineering art, which depends on the task at hand.

Third, the topic of building a Grid Map Utility, has become more and more relevant as the field of robotics is increasing in sophistication. A grid mapping utility is software that has the capacity for online surface reconstruction and terrain interpretation. When it comes to the navigation of areal and underwater robotics the third dimension, hight must be considered. Mobile ground robots are traditionally designed to move on flat terrain and their mapping, planning, and control algorithms are typically developed for a two-dimensional abstraction of the environment. The most popular approach is to build an elevation map of the environment, where each coordinate on the horizontal plane is associated with an elevation/height value. For simplicity, elevation maps are often stored and handled as grid maps. This will be the case for this study as well.

2.1 Camera motion and notation

The motion of the camera poses are related by the rigid body motion $SE(3)$, in many application of visual reconstctuions the notation is defined as the follwing

$$T_{k,k-1} = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix} \tag{2.1}$$

where $T_{k,k-1} \in \mathbb{R}^{4\times4}$ is the motion relating the two camera poses in the set $T_{1:n} = \{T_{1,0}, \ldots, T_{n,n-1}\}$. The set of camera poses $C_{0:n} = \{C_0, \ldots, C_n\}$ are the poses with respect to the initial corinate frame. The poses and transformations are related by the following relationship $C_n = C_{n-1}T_n$.

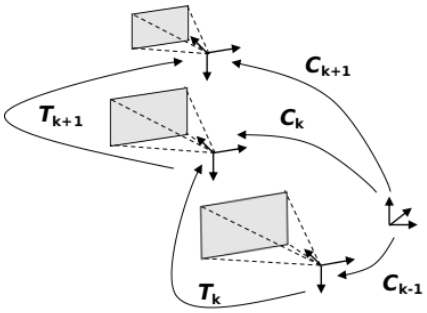


Figure 2.1: Camera motion

However, for the convenience of this work, there will be used a adjusted notation. In the image series of size N, the images is denoted by: $\{I^i \equiv I\,(t^i)\}\,, i \in [1, N]$. The related pose at a time t^i is denoted by T^i_{WC}. The 3D landmarks are denoted with $X^i = \{x(p) \forall p \in P^i\}$, $x\,(p^i_k)$ being the 3D landmark associated with each keypoint p^i_k.

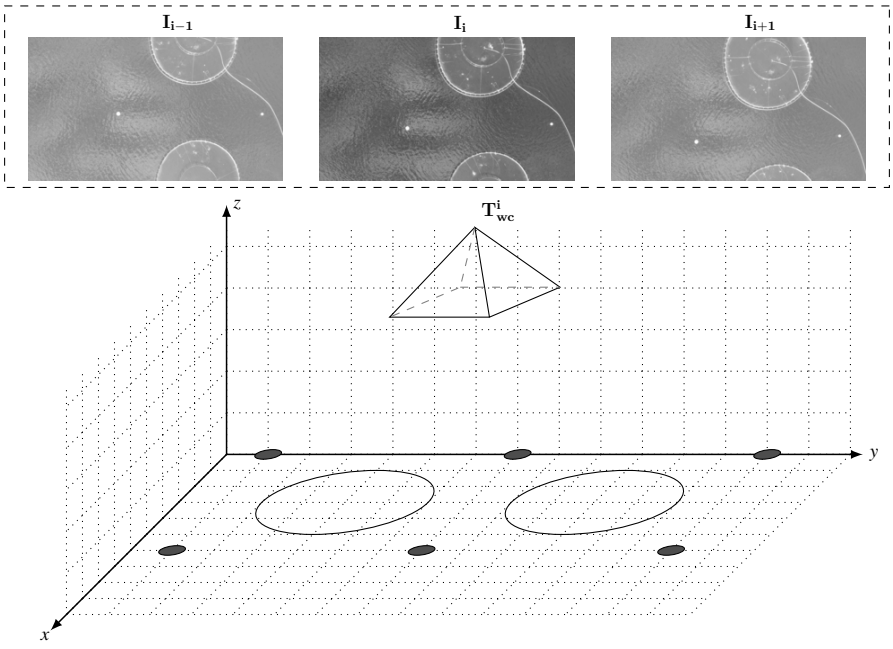


Figure 2.2: Notation defined corresponding

Having defined the notation of the cameras movement and its corresponding images in the discrete time i. There is a need for defining the objects perceived by the aerial system. As discussed in section 1.2 there are several objects to be expected at a site where the objects of interest for defining the underlying geometry of the overall structure is a subset of these objects. Namely buoys and cages. In many methods of detection, there has been assumed a model of what the vision system is expected to see and processing the image based on those assumptions. For making assumptions about the objects we need to consider the geometry of the real world object, their material characteristics and how they are perceived by the vision system.

2.2 Modeling objects of interest

As mentioned in the introduction the characteristics of both objects can be assumed to be of the rigid type. The type of plastic mostly used making the cages and buoys are High-density polyethylene [3]. Although the cage of the structure can be seen to be exposed for mechanical deformations these are small and can be considered to be zero. The buoy because of its typical nonhollow body it can be assumed to be rigid with high confidence.

Some further assumptions relating to the structures perceived by the camera. Considering the cage, it has a circular geometry while perceiving it from above the waterline. Although there exist growing facilities that make use of a cage geometry of the noncylindrical type, namely the rectangular steel cage, though this has not seen wide use. Therefore, for the purpose of this report, it is assumed that the site uses circular shapes for cages. Interpreting the cage object as a circle there are a few properties that it can be said at inherit. A point of origin and an associated circumference, which also can be described by its radius.

It is tougher to make strong assumptions about the buoy. In some cases, it can be resembling a cubical shape with curved edges. Other instances where the structure is more of a circular shape, within these it can be in some cases be conical or even spherical, more details can be found in the field handbook "Aquaculture operations in floating HDPE cages" [3].

Furthermore, floating objects perceived from the lens can be said to have a strong illumination. This is because of their surroundings. The sea surrounding the objects holds the property of absorbing the light hitting it. More precisely absorbing a wide specter of electromagnetic waves where their visible range lies. Thus making the contrast perceived by the image sensor large. Moreover, the objects can be said to have a reflective behavior by its design, making the structure more prone to being observed.

Objects can also be perceived by the lens in a deformed manner described by projective geometry. In a picture, we may see a square that is not a square or a circle that is not a circle. If the pose of the camera is not perpendicular to the plane of the object or the object somehow is rotated with respect to the pose this effect can be seen. However, for the purpose of the image processing detection algorithms, we will assume no to very small divinations from the pose aligning with the surface.

(a) (b)

Figure 2.3: Reference model the two main objects for the image processing algorithms seen at the site from aerial photo view

Figure 2.3 is showing the model to be used for the image processing algorithms. Marking black as the intensity and white as low. Now that there exists a model of the objects of interest and how the UAV is moving. A further investigation on how to obtain these modeled objects can begin.

Chapter 3

Detecting Objects of Interest

There are several approaches to processing an image depending on what information one wants to acquire and also what properties of the image series are expected to look like. In this section, there will be proposed a total of 5 methods of obtaining the shapes of interest. The methods will be described and later in the results section the outcome of the algorithms will be reviewed, tested and discussed.

Figure 3.1 is showing aforementioned algorithms. The rounded nodes are representations of the original RBG image, while the rectangular boxes are indicating algorithms. In outer right, it is shown what these algorithms are aimed at detecting. HiR made to be shorthand for High intensity Regions. Also, DoH made out to be the Difference of Hessian.

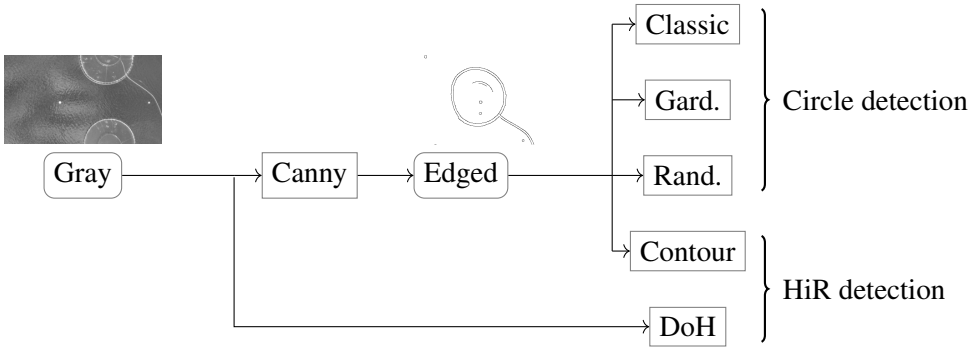


Figure 3.1: Proposed way of benchmarking the image processing algorithms

There will be proposed a way to obtain the structure in the image based on edge detection techniques. From the edged image, algorithms detecting objects of interests based on known attributes about their shape will be reviewed. Lastly, using the grayscale image a method of utilizing scale space will be looked at. The regions of interest are found by looking for large continuous intensity regions. The algorithms used are listed in Figure 3.1. The detection algorithms will be looked at in the order that starts from the outer right and in descending order.

3.1 Getting shapes: Edge Detection

Edge detection serves to simplify the analysis of the image, drastically lowering the amount to be processed, while at the same time preserving useful structural information in the image

boundaries. An edge in an image is defined by a point in which there has occurred noticeable change in intensity.

Local changes in intensity can be obtained by using derivatives, namely first and second order derivatives, which are particularly well suited for this purpose. This is illustrated in Figure 3.2 by the horizontal response of intensity.

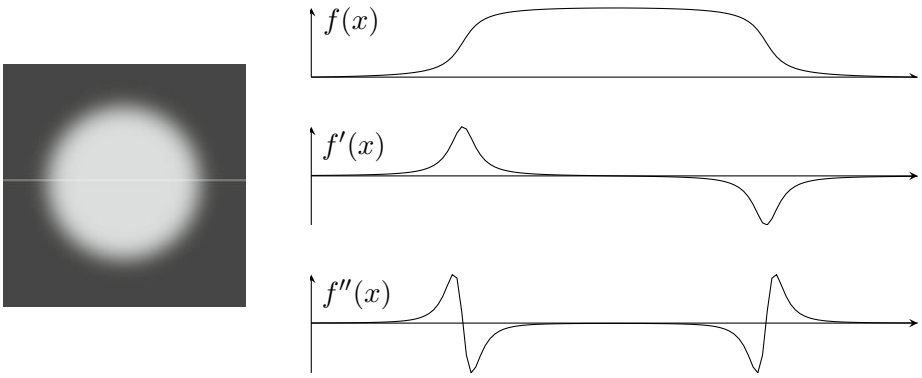


Figure 3.2: Motivation for deriving the discrete derivation in the discrete domain

Now, if we take this horizontal intensity and investigate what we get by taking the derivative of the function in a mathematical sense. The function $f(x)$ can be viewed as a continuous one, as being the function mapping image coordinates of a given row in the image to its corresponding intensity. However, the image is encoded as a collection of sampled signals along the horizontal lines. Thus, the function needs to be defined with respect to the noncontinuous samples u, in the case of the horizontal line.

$$\frac{df}{dx}(u) \approx \frac{f(u+1) - f(u-1)}{(u+1) - (u-1)} = \frac{f(u+1) - f(u-1)}{2} \tag{3.1}$$

The equation above is approximating the derivative of the smooth function $f(x)$. Dividing the domain of the function into discrete time steps u, the derivative of the discretized function is approximated. The equation (3.1) can be implemented by the linear horizontal filter

$$H_x^{\mathrm{D}} = \begin{bmatrix} -0.5 & 0 & 0.5 \end{bmatrix} = 0.5 \cdot \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \tag{3.2}$$

where the operations are applied in the horizontal neighborhood $I(u-1, v)$ and $I(u+1, v)$, weighting positive 0.5 and negative 0.5 respectively.

The same type of reasoning can be done to obtain the vertical gradient vertically, with the resulting filter

$$H_y^{\mathrm{D}} = \begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix} = 0.5 \cdot \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \tag{3.3}$$

Now that the derivation operator is defined. The convolution of this filter across all the horizontal and vertical lines of the image can be defined by the operator $*$. With this operator, we can define the image derivative in both x, and y-direction.

$$I_x = I * H_x \qquad \text{and} \qquad I_y = I * H_y \tag{3.4}$$

With these directive derivatives, it is possible to go further in analyzing the illumination response of the image. By taking the magnitude at each position of u, v we can get a measure of how strong the edge is in an absolute fashion.

$$E(u,v) = \sqrt{I_x^2(u,v) + I_y^2(u,v)} \tag{3.5}$$

It can also be calculated in which direction the gradient of the pixel pair u, v is pointing by applying the ArcTan over the image gradients I_x and I_y.

$$\Phi(u,v) = \tan^{-1}\left(\frac{I_y(u,v)}{I_x(u,v)}\right) = \operatorname{Arctan}\left(I_x(u,v), I_y(u,v)\right) \tag{3.6}$$

These operations stem from deriving and taking the directional derivatives the image. However, there are many different ways to obtain such an x,y derivative of the image I. There has man much research into the the optimal derivative filter, namely Prewitt [9] and Sobel [10] stands out. These works by expanding the neighborhood by using a matrix. These filters are said to counteract the noise sensitivity of the simple gradient operators. The Prewitt operator uses the filter kernels

$$H_x^{\mathrm{P}} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_y^{\mathrm{P}} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{3.7}$$

The filters for the Sobel operator are almost identical. However, emphasizing the current center inline.

$$H_x^{\mathrm{S}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad H_y^{\mathrm{S}} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{3.8}$$

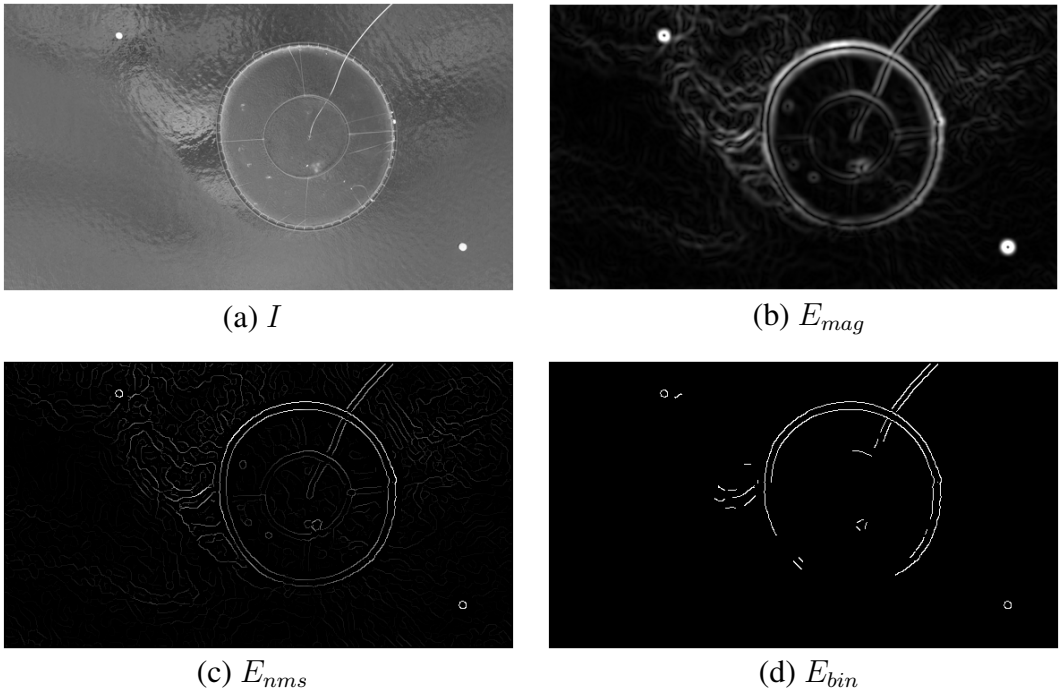
Canny

The operator proposed by Canny [11] is in wide use and is considered "state of the art" in edge detection. The method is opting to tackle three problems: Minimizing the number of false edges, achieve the good location of the edges and delivering a single line of where the curve contour is marked. These properties are not usually not achieved by the more simple edge detectors like Sobel or Pewit.

The pseudocode below will present the method in some detail. Depending on the impregnation there might be some deviations, but the concepts are still the same.

Algorithm 3.1 *Canny edge detector for grayscale images*

1: **procedure** PROCESSFRAME
2: $\bar{I} \leftarrow I * H^{G,\sigma}$ ▷ blur with Gaussian of width σ
3: $\bar{I}_x \leftarrow \bar{I} * [-0.5\ 0\ 0.5]$ ▷ x-gradient
4: $\bar{I}_y \leftarrow \bar{I} * [-0.5\ 0\ 0.5]^\intercal$ ▷ y-gradient
5: $(M, N) \leftarrow \text{SIZE}(I)$
6: $Createmaps:$
7: $E_{mag} : M \times N \mapsto R$ ▷ gradient magnitude
8: $E_{nms} : M \times N \mapsto R$ ▷ maximum magnitude
9: $E_{bin} : M \times N \mapsto 0, 1$ ▷ binary edge pixels
10: **for all** image coordinates (u, v) $\in M \times N$ **do**
11: $E_{mag}(u, v) \leftarrow [\bar{I}_x^2(u, v) + \bar{I}_y^2(u, v)]^{1/2}$
12: $E_{nms}(u, v) \leftarrow 0$
13: $E_{bin}(u, v) \leftarrow 0$
14: **for** $u \leftarrow 1, ..., M - 2$ **do**
15: **for** $v \leftarrow 1, ..., N - 2$ z **do**
16: $d_x \leftarrow \bar{I}_x(u, v), dy \leftarrow \bar{I}_y(u, v)$
17: $s_\theta \leftarrow \text{GETORIENTATIONSECTOR}(d_x, d_x)$
18: **if** $\text{ISLOCALMAX}(E_{mag}, u, v, s_\theta, t_{lo})$ **then**
19: $E_{nms}(u, v) \leftarrow E_{mag}(u, v)$ ▷ only keep local maxima
20: **for** $u \leftarrow 1, ..., M - 2$ **do**
21: **for** $v \leftarrow 1, ..., N - 2$ **do**
22: **if** $(E_{nms}(u, v) \geq t_{hi}) \wedge (E_{bin}(u, v) = 0)$ **then**
23: $\text{TRACEANDTHRESHOLD}(E_{nms}, E_{bin}, u, v, t_{lo})$
return E_{bin}



(a) I (b) E_{mag}

(c) E_{nms} (d) E_{bin}

Figure 3.3: Original image

In Figure 3.3 the different steps throughout the process can be seen. The Gaussian filtered image $\overline{I}$ (a) is not present, but the sigma was set to 3 for this particular run. The filtered image is then taken the gradient x and y derivative. From this, the magnitude gradients can be calculated (b). Also, noting the direction (sector) of decent from the derivative direction. Based on these two entities a thinning process can be conducted (c). Finally, a thresholding step is used to conserve the strong 8-connected edge responses, conserving the edge either on its own intensity response or its 8-connected neighborhood response (d).

Note to the reader. In this section, a method of mapping the grayscale image to a binary representation was reviewed. In the Figure 3.3 the edge pixels are rendered as white and the background as black. For the readability of the following text, this mapping is inverted making for clearer figures.

3.2 Matching linear stuctures

Finding simple shapes, such as lines and circles in images may look like a simple task but computational issues coupled with noise and occlusions require some nonnative solutions. In spite of the apparent diversity of lines and areas, it turns out that common approaches to the detection of linear structures can be seen as an efficient implementation of matched filters. The section describes how to compute salient image discontinuities and how simple shapes embedded in the resulting map can be located with the Hough transform [12].

3.2.1 Generialised Hough transform

The Hough transform has been around for a long time [13]. In the formal analysis, there is a way to generalize the transform into arbitrary shapes. To start with, one can consider the parameterization the curve described in the plane:

$$v(\theta) = x(\theta) \begin{bmatrix} 1 \\ 0 \end{bmatrix} + y(\theta) \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.9}$$

Generalizing the transform with the equation

$$\omega(\theta, b, \lambda, \rho) = b + \lambda \mathbf{R}(\rho) v(\theta) \tag{3.10}$$

where the parameters $b = (x_0, y_0)$ is the location, λ scale factor and $\mathbf{R}(\rho)$ rotation matrix. Thus, making it possible to capture the shape in the plane by considering the shapes translation, rotation, and change of scale relative to the image at hand.

Ellipse Hough transform

As discussed earlier, the detection method is reliant on modeling the shape of the cage as a circle. Yet depending on the angle of view or structural deformation, it might not precisely be a perfect circle. In fact, it is more like an ellipse. However, matching ellipses is a non-trivial task. The shape described by its parameter space of a high magnitude. The parameter space is made up of observing the equation (3.11). The equation has 5 unknowns: x,y scale S, orientation by ρ and translation by (t_x, t_y).

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\rho) & \sin(\rho) \\ -\sin(\rho) & \cos(\rho) \end{bmatrix} \begin{bmatrix} S_x \\ S_y \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{3.11}$$

Although there has been made effort into reducing the time complexity of the ellipse detection, these methods will not be considered.

3.2.2 Classical Hough circle transfrom

Using the parametsarion of an circle for $x(\theta)$ and $y(\theta)$. Then rearegning (3.10) and solving for x_0, y_0 gives

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x - r\cos(\theta) \\ y - r\sin(\theta) \end{bmatrix} \tag{3.12}$$

With this equation, one is able to describe every circle in the plane. The intuition behind why the Hough circle transform is iterating through a large set of the set of possible circles in the

image. Limiting the translation x, y by the edge pixels in the image and radius by fixing it to the radius of the radius by the edged image. Denoting where the pixels in the image each circle is passing through. This will result in the accumulation of values in the center of the true circle.

To clarify, we can consider Figure 3.4. Taking the three different edge pixels p_1, p_2, p_3 and their corresponding x, y. Furthermore, assuming r to be fixed. Then (3.12) leaves one unknown θ, this is the free parameter and it defines the trace of the curve. Collecting the accumulated values generated by the traces C_1, C_2, C_3 it is clear that one would see a max accumulation in the center of the circle.

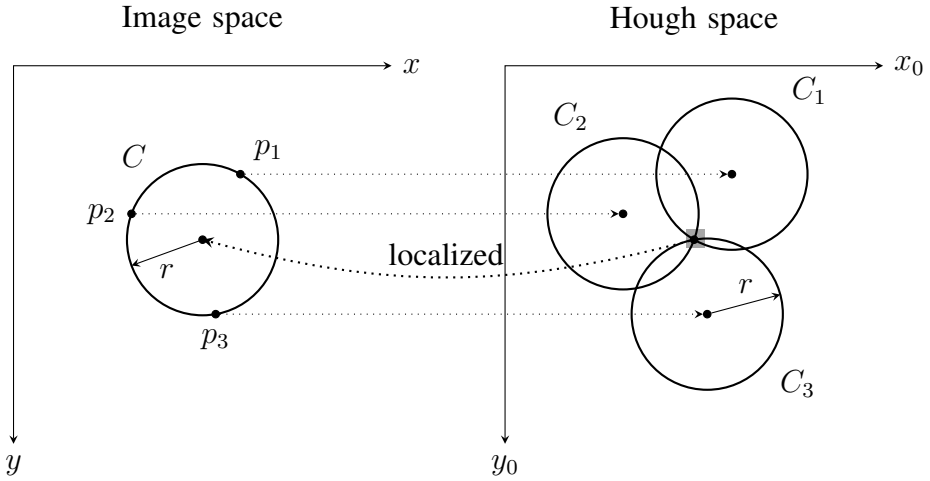


Figure 3.4: Hough circle transform

If r is not known we can make the transform include a third dimension to the parameter space. Making the search space not only to include x_0, y_0, but also a r (visualized in Fig. 3.5). By searching the 3D space for the highest value we obtain the center. By denoting where in the r space the maxima were detecting we obtain the circles corresponding radius.

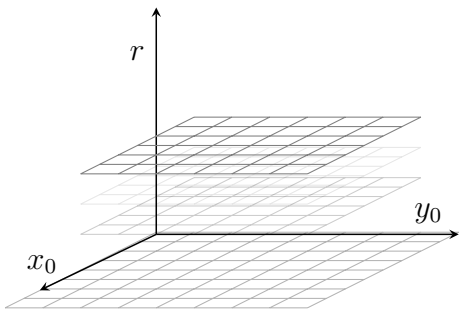


Figure 3.5: The three dimentional accumulator space for the classical Hough Circle Transform

The accumulator space grows big in this brute force manner, especially for large images. Throughout the years there have been works on how to reduce the time complexity of this technique. The next sections will present two ways of tackling this time complexity.

3.2.3 Classical Hough circle arameter reduction

The Hough transform gives the same (optimal) result as template matching and even though it is faster, it still requires significant computational resources [14]. One way to reduce the computation time is by looking at the gradient information in the edged image. In this section, the technique of parameter space reduction will be reviewed. There are several ways of using gradient information to reduce computational time. Using the gradient information in the detection of circles goes back to as early as 1975 [15].

In this section, the parameter space of the Hough circle transform is reduced utilizing the relationship between points of the circle and its derivatives.

By considering the parameterized definition of the circle

$$x(\theta) = x_0 + r\cos(\theta); \quad y(\theta) = y_0 + r\sin(\theta) \tag{3.13}$$

One can find its derivatives

$$\begin{aligned} x'(\theta) &= -r\sin(\theta); & y'(\theta) &= r\cos(\theta) \\ x''(\theta) &= -r\cos(\theta); & y''(\theta) &= -r\sin(\theta) \end{aligned} \tag{3.14}$$

Figure 3.6 shows the geometrical interpretation of these derivatives

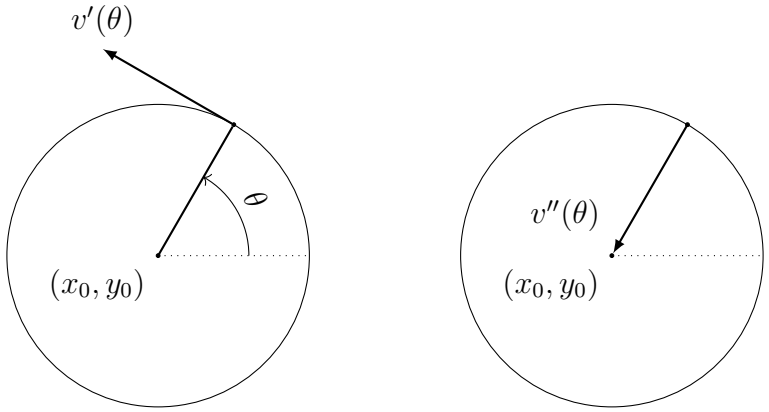


Figure 3.6: First secound derivative of the parametarized circle

The behavior of this directional trigonometry is the one the gradient-based the approach tries to exploit. It is dependent on the following reasoning:

By considering the directional derivatives and its properties:

$$\phi'(\theta) = \frac{y'(\theta)}{x'(\theta)} = -\frac{1}{\tan(\theta)} \qquad \phi''(\theta) = \frac{y''(\theta)}{x''(\theta)} = -\frac{1}{\phi'(\theta)} \tag{3.15}$$

Overserving the (3.13) we notice that $-r\sin(\theta) = y(\theta) - y_0$ and $-r\cos(\theta) = x(\theta) - x_0$

$$\phi''(\theta) = \frac{y''(\theta)}{x''(\theta)} = \frac{y(\theta) - y_0}{x(\theta) - x_0} \tag{3.16}$$

rearranging the equation

$$y(\theta) = \phi''(\theta)\,(x(\theta) - x_0) + y_0 \tag{3.17}$$

Using the earlier definition of $\phi''(\theta)$ we get

$$y_0 = y(\theta) + \frac{x(\theta) - x_0}{\phi'(\theta)} \tag{3.18}$$

By looking at Figure 3.6, we observe that the derviative of the circle can be approximated by taking a pair of two close points along the the circle $v(\theta)$. In (3.19) we use this two point approximation and then the relation of $\phi''(\theta)$ in equation (3.15) to obtain

$$\phi'(\theta) = \frac{y_2 - y_1}{x_2 - x_1} \qquad \phi''(\theta) = -\frac{x_2 - x_1}{y_2 - y_1} \tag{3.19}$$

Inserting the approximated $\phi'(\theta)$ into the equation (3.16), we obtain

$$y_0 = y(\theta) + \frac{(x(\theta) - x_0)\,(x_2 - x_1)}{(y_2 - y_1)} \tag{3.20}$$

Figure 3.7 the line of the equation 3.20 is illustrated.

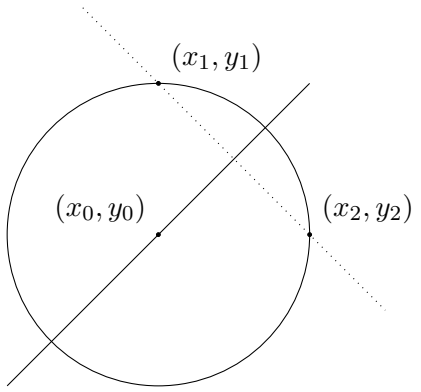


Figure 3.7: Estiamted geometric gradient of the circle

However, $x(\theta), y(\theta)$ cannot exactly be known, this must be approximated. This being the intersection point, of the two lines of Figure 3.7, obtained by simply taking the mean of the (x_i, y_i) $i \in \{1, 2\}$ pair. Now, (3.20) having two unknown variables, iterating one of the variables x_0, y_0 the other can be calculated and the position in the accumulator space can be found and incremented.

For this algorithm to work, its clear that edge points (x_2, y_2) and (x_1, y_1) need to be found. This can be done, iteration through the edge points of the binarized image and start the accumulative process if there are found edge points in the corner of the window.

The gradient-based Hough circle algorithm is various in this implementation. But they are often based on the same principals shown in this section. Moreover, these implementations often add a step where the radius is extracted after accumulating the candidates for the center. Because of the loss of the radius dimension while using this technique, there is often used a histogram where each pixel close to the candidate centers length is calculated, based on the histogram the associated radius can be determined.

3.2.4 Randomized circle detection

The technique described in this section depends on gathering evidence and is a quite recent find taking into account the history of circle matching methods. The method has its roots in the popular method of randomized Hough transform [16]. Where the main idea is to pick three random edge pixels from a given binarized image and computing the circle's properties, then denoting it in an accumulator space. Given a sufficiently large number of these circle candidates, it can be determined by voting where the true circle is most likely to have appeared.

The theory in this section builds on this technique of finding a circle given three edge pixels. However, suggests a fourth edge pixel to be picked [17]. Making this new pixel responsible for an evidence collection step in the process. The method can mathematically be described by first considering the equation of the circle:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{3.21}$$

If we take the equation (3.21), adding in the circle $a = x_0$ and $b = y_0$

$$2xa + 2yb + d = x^2 + y^2 \tag{3.22}$$

The distance $d = r^2 - a^2 - b^2$ and then the three randomly picked edge pixels in the image can be written $v_i = (x_i, y_i)\,, i = 1, 2, 3$.

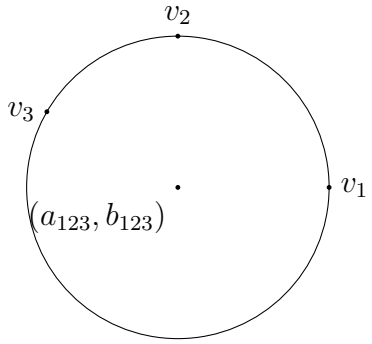


Figure 3.8: Three points sadifiying and defining the equation of the circle

The constraint on v_i can be written into

$$2x_i a_{123} + 2y_i b_{123} + d_{123} = x_i^2 + y_i^2, \qquad (i = 1, 2, 3) \tag{3.23}$$

These can be arranged to matrix from

$$\begin{pmatrix} 2x_1 & 2y_1 & 1 \\ 2x_2 & 2y_2 & 1 \\ 2x_3 & 2y_3 & 1 \end{pmatrix} \begin{pmatrix} a_{123} \\ b_{123} \\ d_{123} \end{pmatrix} = \begin{pmatrix} x_1^2 + y_1^2 \\ x_2^2 + y_2^2 \\ x_3^2 + y_3^2 \end{pmatrix} \tag{3.24}$$

By rearranging by Gaussian elimination and applying Cramer's rule for linear system it can be obtained that

$$a_{123} = \frac{\begin{vmatrix} x_2^2 + y_2^2 - (x_1^2 + y_1^2) & 2(y_2 - y_1) \\ x_3^2 + y_3^2 - (x_1^2 + y_1^2) & 2(y_3 - y_1) \end{vmatrix}}{4((x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1))} \tag{3.25}$$

and

$$b_{123} = \frac{\begin{vmatrix} 2(x_2 - x_1) & x_2^2 + y_2^2 - (x_1^2 + y_1^2) \\ 2(x_3 - x_1) & x_3^2 + y_3^2 - (x_1^2 + y_1^2) \end{vmatrix}}{4((x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1))} \tag{3.26}$$

Now that the radius of the circle is determined, the radius can be found by picking an (x_i, y_i) pair and computing $r_{123} = \sqrt{(x_i - a_{123})^2 + (y_i - b_{123})^2}$. If the center of the circle is inside of the width and height of the image, the last step can be performed and the evidence collation step can be calculated. If we take this pair and apply (3.27), the measurement of the last pixel to the circle can be found. Based on the distance d_4 there is a possibility of determining if the circle is likely to have appeared in the image .

$$d_{4\to123} = \left| \sqrt{(x_4 - a_{123})^2 + (y_4 - b_{123})^2} - r_{123} \right| \tag{3.27}$$

3.2.5 Accumulator space non-maximum suppression

The methods of matching the linear structures described above will accumulate circle center candidates in one way or another. There is a need for suppression of these candidates if there exists more than one local maxima. This can be done with a simple 2D windowed maxima finding. By dividing up the image/ accumulator space in several regions it is possible to obtain the local absolute maxima of each region. For this simple approach, there needs to be a distance check wherever there has been a bad cut in the window operation. For example, it can be considered that one extrema region has been divided into two or more regions. One way of catching these cases is to perform a distance check wherever there are many absolute regional maxima located in the same area in the image/ accumulator space.

3.3 Detecting high interisity regions

3.3.1 Contour selection by topological structural analysis

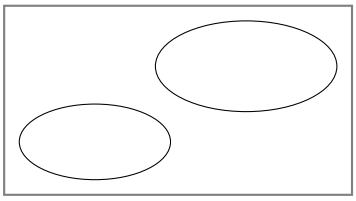
There are several methods of obtaining structural information in a binary image. A simple approach may be the connected components labeling and decide on the connectivity. Fortunately, OpenCV provides a sophisticated method of extracting regions not only based on the intermediate connectivity but also provides the hierarchy of structures in the binarized image. This method is based on border following.

Border following is one of the fundamental techniques in the processing of binarized image images. It requires a sequence of coordinates or chain codes from the border between a connected component of a 1-valued pixel (component) and a connected component of the 0-value (background). In [18] it was shown that one could efficiently extract the full topological structure of an image. Later, this method has been implemented in OpenCV, making it possible to extract the topological structure of a binary image, preserving the hierarchy in a treelike structure with a corresponding set of coordinates. This kind of topological extraction is desirable, allowing for the structure of the shapes to be extracted and further analyzed. The goal is to divide regions into segments with a unique label to it while preserving information about their hierarchy. This can be viewed as a more sophisticated connected components algorithm for the purposes of this report. OpenCV returns the structure in the following manner running the algorithm on the black and white image:

$$[Next, Previous, First\_Child, Parent] \leftarrow \text{COMPUTECONTOURS}(I)$$

$$\begin{bmatrix} -1 & -1 & 1 & -1 \\ 3 & -1 & 2 & 0 \\ -1 & -1 & -1 & 1 \\ -1 & 1 & 4 & 0 \\ -1 & -1 & -1 & 3 \end{bmatrix}$$

(a)



(b)

Figure 3.9: Showing the full structure hierarchy of the binarized image. (a), displaying the hierarchy found in the image (b), where every row representing a border with this corresponding child and parent, these combined makes up regions within the image.

In Figure 3.9 there are in total 5 borders where the borders of the second hierarchy are the ones that are describing the lines in black. The fourth and fifth (3,4 array indexed) are the nodes of the inner contours, with no corresponding child.

Computing region properties

Given that the regional topological structure and its relating contours it is possible to compute properties of the regions surrounded by the contours. This enables for further analysis of the image.

But first, we need need to use the moments of the contour (node). The moment can be used to

compute the properties.

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \tag{3.28}$$

In particular, for this application we are interested in the center of mass of the contours, denoted by (u_c, v_c) and is given by the 1st moments of the object:

$$\{u_c, v_c\} = \left\{ \frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right\} \tag{3.29}$$

The other properties of the contour are listed below with their corresponding calculation, given in the table:

Centre	=	$\{u_c, v_c\}$
Circularity	=	$\frac{4*\pi*Area}{perimeter^2}$
Aspect Ratio	=	$\frac{Width}{Height}$
Solidity	=	$\frac{Contour\ Area}{Convex\ Hull\ Area}$
Mean intensity =	=	$\frac{Contour\ Area}{Convex\ Hull\ Area}$

Table 3.1: Common shape features

Testing contours

Having the structure of the binarized image, and the regions corresponding properties. There can be made some assumptions of what is to be expected to see.

Several observations can be made by looking at the edged image set in (Appendix A.2). Beginning with the hierarchy. Considering the hierarchy given by the algorithm and it can be of value to preserve only the outermost border. One can see that the region of interest almost always lies within the lowest hierarchy. This property tells us that the contour containing the pixels of the buoy is most likely to appear in the lowest hierarchy. Thus, the node of the contour should not have any child .

Furthermore, by inspecting the grayscale images in (Appendix A.1), one can find that the intensity of the buoys is object to little change of intensity. Also, they tend to be the most illuminative. Based on this fact it is possible to set a threshold of the illumination in the given pixel region.

Some other properties that can be filtered on are area and circularity, with the same thresholding method used when filtering contours based on illumination.

Figure 3.10 showes the filtered contours of the binarized image.

Figure 3.10: Showing the contours candidates in the canny filtered image

Table 3.2 showes the computed properties of the filtered contours.

Table 3.2: Computed properties of b_i in Figure 3.10

	b_1	b_2	b_3
Mean intensity	234.9	233.2	200.7
Circularity	0.85	0.84	0.85
Area	69.5	66.0	77.5
Hierarchy	0th	0th	0th

The splash from the fishes may be very similar to the canny obtained buoys. By knowing the hierarchy of the contour it can be performed a check to see if it has an outer border/ contour surrounding it. Ideally, this would give a good indication. However, the cage is almost never outputted from the canny algorithm as one long connected edge, making this process not possible. Yet the splashes in the cage may be subject to several edges entangled making filtering on the inner edge advantageous.

3.3.2 Trajectories of critical points in scale-space

Background and the Hessian matrix

The method is taken from the technique of finding corresponding patches in two images [19]. Borrowing from the concept of how two images correlate it can be used to see how an image is correlated to itself. To obtain this autocorrelation the one first needs to consider the equation of correlation between two images.

$$E(\mathbf{u}) = \sum_i w\left(\mathbf{x}_i\right)\left[I_1\left(\mathbf{x}_i + \mathbf{u}\right) - I_0\left(\mathbf{x}_i\right)\right]^2 \tag{3.30}$$

The same concept can be used, by comparing the image patch to itself, also known as the autocorrelation

$$E_{\mathrm{AC}}(\Delta\boldsymbol{u}) = \sum_i w\left(\boldsymbol{x}_i\right)\left[I_0\left(\boldsymbol{x}_i + \Delta\boldsymbol{u}\right) - I_0\left(\boldsymbol{x}_i\right)\right]^2 \tag{3.31}$$

Now the image function $I_0\left(\boldsymbol{x}_i + \Delta\boldsymbol{u}\right)$ can be approximated using the first Taylor series expansion $I_0\left(\boldsymbol{x}_i\right) + \nabla I_0\left(\boldsymbol{x}_i\right)$, making it possible to approximate the auto-correlation surface as

$$\begin{aligned} E_{\mathrm{AC}}(\Delta\boldsymbol{u}) &= \sum_i w\left(\boldsymbol{x}_i\right)\left[I_0\left(\boldsymbol{x}_i + \Delta\boldsymbol{u}\right) - I_0\left(\boldsymbol{x}_i\right)\right]^2 \\ &\approx \sum_i w\left(\boldsymbol{x}_i\right)\left[I_0\left(\boldsymbol{x}_i\right) + \nabla I_0\left(\boldsymbol{x}_i\right)\cdot\Delta\boldsymbol{u} - I_0\left(\boldsymbol{x}_i\right)\right]^2 \\ &= \sum_i w\left(\boldsymbol{x}_i\right)\left[\nabla I_0\left(\boldsymbol{x}_i\right)\cdot\Delta\boldsymbol{u}\right]^2 \\ &= \Delta\boldsymbol{u}^T\boldsymbol{A}\Delta\boldsymbol{u} \end{aligned} \tag{3.32}$$

Then the autocorrelation at the point x_i, y_i can be computed by taking the derivatives and generalizing for the image I_0 we get

$$\mathbf{A} = w * \begin{pmatrix} I_x^2 & I_{xy} \\ I_{xy} & I_y^2 \end{pmatrix} = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \tag{3.33}$$

where w is the filter kernel used. Pixelwise will the determinant of the autocorrelation A respond strongly at edges of the image.

$$\det A = ac - b^2 = \lambda_1\lambda_2 \tag{3.34}$$

In the next section, the convolutional filter w will be discussed.

Gaussian scale space

Conspicuously this is motivated by real-world observations. Considering waking up without the glasses one can argue that only the largest contours of the objects surrounding us are available for our perceiving. This "blurring" factor is what introduces the scale in the image. Dividing the image into several scales, thus making the image not contain only 2d coordinates but a third one, making it a 3D structure. This concept is known as the scale space [20].

In particular, we can have the Gaussian scale space, with its associated m levels of filtering.

$$\mathrm{G}_m = I * H^{\mathrm{G},\sigma_m} \tag{3.35}$$

The representation of scale space is advantageous for getting interested points at multiple scales. We define the discrete Gaussian scale space representation of an image I as a vector of M images, one for each scale level m:

$$\mathrm{G} = (\mathrm{G}_0, \mathrm{G}_1, \ldots, \mathrm{G}_M) \tag{3.36}$$

Associated with each level G_m is a corresponding σ_m. This amounts to the image I being converted into a series of G_m, each representing a blurred version of the original image. Therefore the scale representation could be thought of as a three-dimensional representation of the image. The scale ratio between the different images are defined as

$$\Delta_\sigma = \frac{\sigma_{m+1}}{\sigma_m} \tag{3.37}$$

And can be considered is set to ba a constant in this study. The base scale is defined as σ_0 which is defined as the point of the start of the linspace. In Figure 3.11 it is shown how a scale space image seires might look like. With base scale at $\sigma_0 = 1$ and incrementions of $\Delta_\sigma = 1$

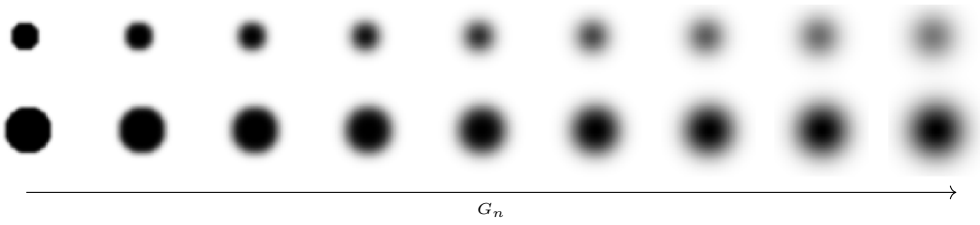


Figure 3.11: The gausian scale space of two intensity regions

Regional local maxima

Having defined the images scale space of an image and the hessian operation. These two can operations can be combined, taking the hessian of each image in the scale space. Effectively turning the image into a cube-like representation of the hessian with each of its own corresponding sigma. It has been shown that interest points, being high-intensity regions do exert a strong response in the cube. These strong responses can be found by non-maximum suppression. There are several implementations of this technique, the one popular method is looking at the 3x3x3 neighborhood of the cubed image [21] .

Testing gausian scale space

To test the scale space method a simple script was comprised toghter. applying a Gaussian filter. Then the gradient was taken in both directions, obtaining the Hessian of the filtered image. Lastly, there was applied a simple smoothing of the Hessian matrix. Then the max value is found to seek out the strongest response of each image. The of calculating the hessian is found in the pseudocode below:

Algorithm 3.2 *Computing determinant*

1: **procedure** DOH ▷
2: $[G_x, G_y]$=GRAD(G_m)
3: $[G_{xx}, G_{xy}]$=GRAD(Gx)
4: [Gxy, Gyy]=GRAD(Gy)
5: $H_{det} \leftarrow G_{xx} * G_{yy} - G_{xy}^2$

The values of sigma were fined tuned to be in the area where the determinant got the highest response. From these tests, it was concluded that the buoy's make of the highest intensity most condense regions in the image. Although the splashes from the fish cages seemed to have some quite large responses as well, they were not at the same magnitude of the buoys.

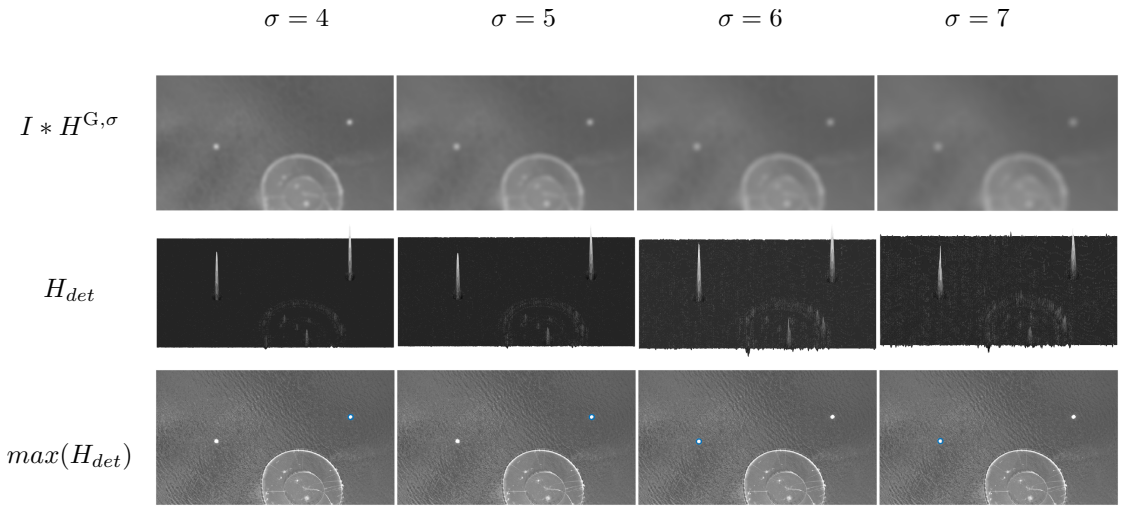


Figure 3.12: Gaussian and Hessian response of the scale space

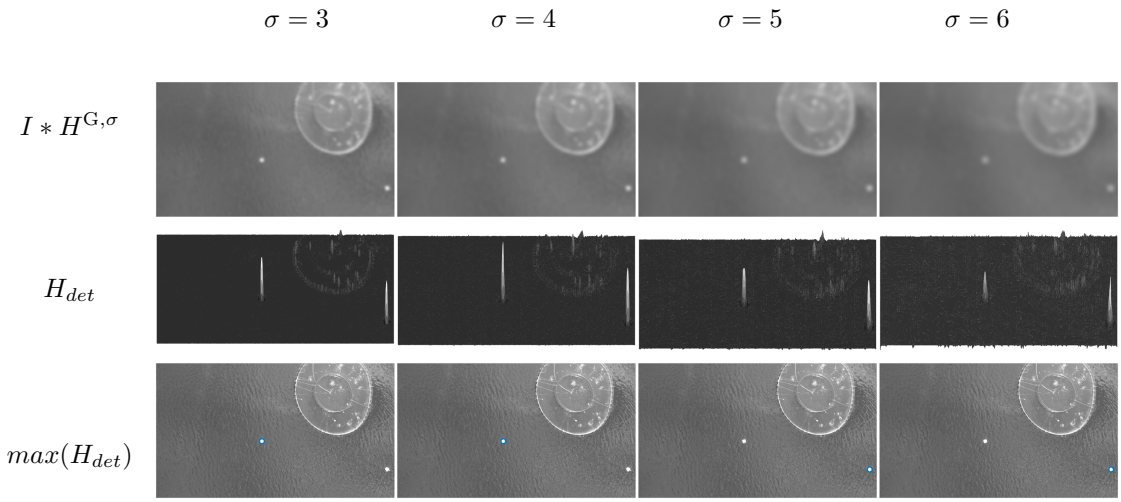


Figure 3.13: Gaussian and Hessian response of the scale space

In Figure 3.12 and 3.13 one can see the simple three-stage process described in the previous paragraph. Where it tends to be strong response surroundings the large connected regions of

high intensity. The figures are also picking up in the scale-invariant property of the scale space, making for multiple regions of different sizes being pickup up in the same space.

Chapter 4

Geometric Computer Vision

Geometric computer vision comes in to play when we have the points of interest (keypoints) obtained by the image processing module. In this section, there will be a review of how the camera is chosen to be modeled. There will also be proposed methods of solving the problem of how to recover the detected pixel points in 3D space. Going from 2D points in the image plane, one can think of the operation as rays being projected into the world. The scale is what defines the position of the detected object in the world frame. There are two main methods proposed for doing this. Firstly, by assuming the world perceived world as a plane and then finding where the point of the ray cast is intersecting the plane. This makes for solving for the unknown in the rays projectile, the scale. Secondly, there will be a review of how to use point correspondence from two separate images and triangulating these to obtain the final point. This method, in particular, is estimating the point of intersection of the two rays cast from each image.

4.1 Pinhole model

The camera can be approximated by a projective model, often called the pinhole projection model. Which considers a planar imaging plane or focal plane z $= f$ that the light is hitting. The simplest representation of a camera is a light sensible surface (sensor): an image plane, a lens (the projective projection) at a given position and orientation in space. This representation can be seen in Figure 4.1.

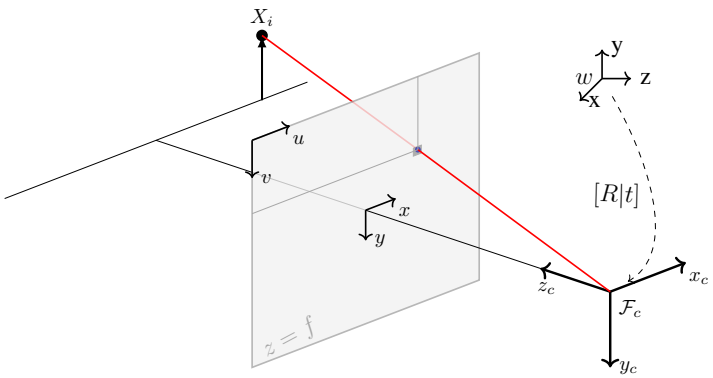


Figure 4.1: The pinhole camera model

This model uses a camera Matrix, often referred to as the projection matrix is written as

$$P_{3\times4} = K[R|t] = \begin{bmatrix} f*k_u & & c_u \\ & f*k_v & c_v \\ & & 1 \end{bmatrix} \begin{bmatrix} & & t_x \\ & R_{3\times3} & t_y \\ & & t_z \end{bmatrix} \tag{4.1}$$

The pinhole camera geometry models the projective camera with two sub-parametrizations, intrinsic and extrinsic. The optic component is described by the intrinsic denoted by K below and the extrinsic as the orientation in space. 3D points is projected in an image with the following formula:

$$x_i = PX_i = K[R|t]X_i \tag{4.2}$$

$$\begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} = \begin{bmatrix} f*k_u & & c_u \\ & f*k_v & c_v \\ & & 1 \end{bmatrix} \begin{bmatrix} & & t_x \\ & R_{3\times3} & t_y \\ & & t_z \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ W_i \end{bmatrix} \tag{4.3}$$

However, solving this equation for X_i, one still lacks the scale parameter, this will be looked into in the following sections.

4.2 One view geometry

In this section, a method of projecting the observed point in the camera on to the ground plane will be reviewed. This method is dependent on having a planar model of the scene. In the case of the mapping of the fish farm scene, we can define the plane as aligning with the world coordinate system.

Line-plane intersection

The line-plane intersection is a simple solution. This depends on having a model of the plane. Also, having a line which passes through two points P_1, P_1. In the case of the camera, this will be the line starting from the optical center and intersecting the keypoint the distance f from the optical center .

The two points are denoted $P_1\,(x_1, y_1, z_1)\,, P_2\,(x_2, y_2, z_2)$. With $P(x, y, z)$ being point on the plane, illustrated in Figure 4.2.

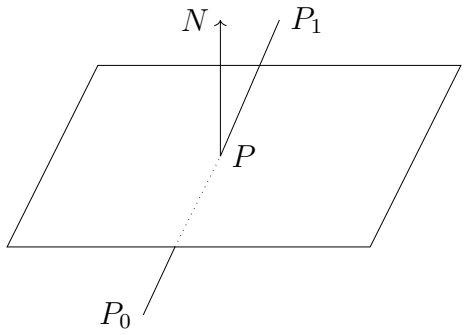


Figure 4.2: Line intersection with plane

Firstly the plane can be described by the equation:

$$Ax + By + Cz + D = 0 \tag{4.4}$$

substituting for the parameterized line passing through the points $P = P_1 + t(P_2 - P_1)$

$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_1 + at \\ y_1 + bt \\ z_1 + ct \end{bmatrix} \tag{4.5}$$

the substituting P back into the plane equation and solving for t gives

$$t = -\frac{(Ax_1 + By_1 + Cz_1 + D)}{Aa + Bb + Cc} \tag{4.6}$$

Finally inserting back into the equation of P

$$P_0 = \begin{bmatrix} x_1 - \dfrac{a\,(Ax_1 + By_1 + cz_1 + D)}{Aa + Bb + Cc} \\ y_1 - \dfrac{b\,(Ax_1 + By_1 + cz_1 + D)}{Aa + Bb + Cc} \\ z_1 - \dfrac{c\,(Ax_1 + By_1 + cz_1 + D)}{Aa + Bb + Cc} \end{bmatrix} \tag{4.7}$$

Making this relevant for our case we know that the plane (equation (4.4)), in the world frame is described by $z + D = 0$ depending on where the world coordinate frame is defined. Simplifying the point P_0 to

$$P_0 = \begin{bmatrix} x_1 - \dfrac{a\,(cz_1 + D)}{c} \\ y_1 - \dfrac{b\,(cz_1 + D)}{c} \\ D \end{bmatrix} \tag{4.8}$$

Relating this equation to the flight at Korsneset, we can know that D is dependent on where the world coordinate system is defined to be. If it is defined to be at zero altitude we get $D = 0$, Simplifying the equation even more. Also we see that one can obtain a, b with the simple matrix operation $K^{-1}x_i$. Lastly, the point P_1 is interpreted as the translation vector of the UAV corrected for the camera.

4.3 Two view geometry

Triangualtion

Given the two camera matrices P and P' with two points identified x and x'. There is a line (ray) that can be cast starting at the optical center of the camera and intersecting the corresponding point x in the image plane. These points identified satisfies the relation $\mathbf{x} = \mathrm{PX}$, $\mathbf{x}' = \mathrm{P}'\mathbf{X}$. But because of noise related to the camera and the image processing algorithm, these relations will not hold exact. In fact, even if the ground truth of the poses are known, there is some the error associated by the feature extraction algorithms and camera model that result in imperfectly measured x, x'. Thus, the world coordinate $\mathbf{X}$ and the image plane x needs to be estimated and the equation

$$\hat{\mathbf{x}} = \mathrm{P}\widehat{\mathbf{X}} \qquad \hat{\mathbf{x}}' = \mathrm{P}'\widehat{\mathbf{X}} \tag{4.9}$$

can be used to describe this relation. $\widehat{X}$ is tried estimated from the measurements x and x'. In Figure 4.3 this estimation is visually illustrated.

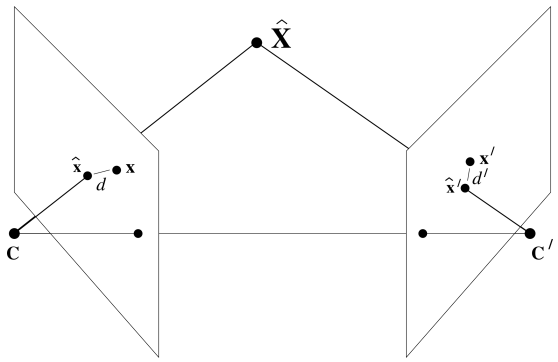


Figure 4.3: Ray intersection

There is a need for solving for the landmark, given the two equations in (4.9). There are several methods solving this equation, in the next section there is a way of composing the equations into a system om linear equations and solving by minimizing the algebraic least-squares error.

Linear triangulation

Trying to solve the two equations of the unknown X. The homogeneous scale factor is eliminated by a cross product to give three equations for each image point, of which two are linearly independent. For the first equation, this gives $\mathbf{x} \times (\mathrm{PX}) = \mathbf{0}$. Writing this out gives

$$\begin{aligned} x\left(\mathbf{p}^{3\top}\mathbf{x}\right) - \left(\mathbf{p}^{1\top}\mathbf{x}\right) &= 0 \\ y\left(\mathbf{p}^{3\top}\mathbf{x}\right) - \left(\mathbf{p}^{2\top}\mathbf{x}\right) &= 0 \\ x\left(\mathbf{p}^{2\top}\mathbf{x}\right) - y\left(\mathbf{p}^{1\top}\mathbf{x}\right) &= 0 \end{aligned} \tag{4.10}$$

Where $\mathbf{p}^{iT}$ are the rows of P.

These equations can be combined into a system which is linear around $\mathbf{X}$, making for the

equation $AX = 0$. Written out this becomes

$$A = \begin{bmatrix} u\mathbf{p}_3^T - \mathbf{p}_1^T \\ v\mathbf{p}_3^T - \mathbf{p}_2^T \\ u'\mathbf{p'}_3^T - \mathbf{p'}_1^T \\ v'\mathbf{p'}_3^T - \mathbf{p'}_2^T \end{bmatrix} \tag{4.11}$$

In the ideal case, one would expect this equation to hold exactly. But due to image noise and error in the camera matrices, there will not be an exact solution to the system that can be solved by SVD to get a least squares estimate of the 3D point.

4.4 Projection geometry validation

Before the computer vision module can be carried out and be applied to them from the data from the site there was a need to validate the method in a smaller environment where more information about the scene was known. Such as the measurable distance from the origin to a given keypoint. This also provides the opportunity to have several, inexpensive, data gathering stages. The test environment had no internal sensors to directly get an estimate of the pose of the camera such as on the UAV of the ARTIFEX project. So the pose is be estimated with the use of an aruco marker [22, 23] . A program was written that could retrieve the transform of the camera with respect to the world frame. The program noted down the picture with its corresponding position and orientation received from the aruco module. Originally the aruco module will define the coordinate frame of the marker as pointing upwards, this was modified to the pointing down making it resemble NED. The test environment is depicted in Figure 4.4.

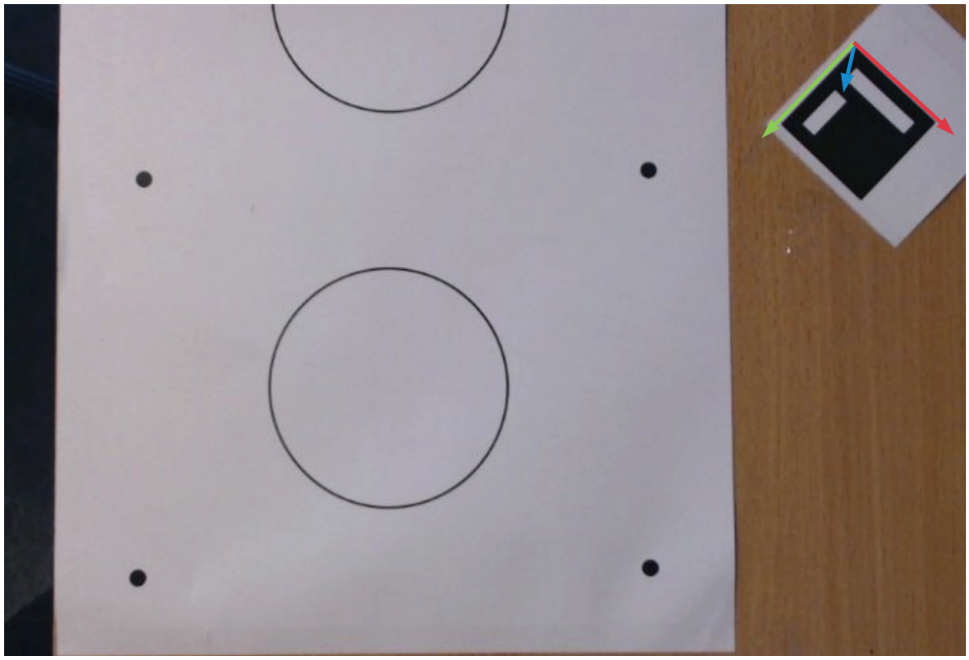


Figure 4.4: Picture of the test environment taken from above

In the first stage, the poses were read in with its belonging image. The Viz module of OpenCV was to create a visual debugging environment [24] and to evaluate the methods used. Furthermore, the buoy closes to the origin were hand-labeled and added in the code. This allowed a ray to be cased out from optical center projecting it on to the image plane and further outwards into the scene, this can be seen in Figure 4.5.

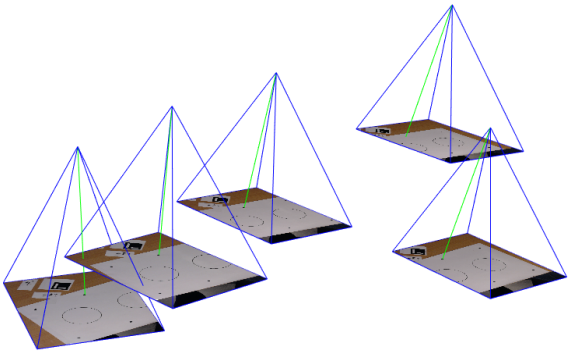


Figure 4.5: Screen capture of the poses and their poses (blue), with each related image projected onto the image focal plane of the camera. The green line showing the ray coming from the optical center intersecting the same keypoint in each image.

This was a validation of the intrinsic parameters of the calibration process. An interesting observation that was made in this stage was that the intrinsic parameters obtained from running OpenCV own calibration program found at its repository online yielded better projection error making the parameters representing the principal point half of the image size.

To validate that the estimation of the position and orientation given by the aruco markers was useable the rays were projected further and observed that they meet. The rays are not expected to meet exactly but somewhat in the same area would give a good indicator of the validation of the aruco module.

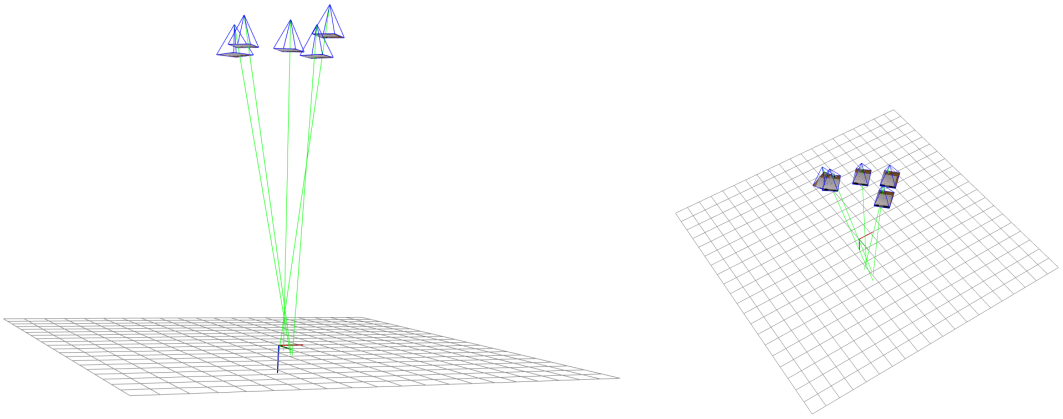


Figure 4.6: Rays projected down to the plane of the wold cordinate system, grid cell size 25mm

Having observed the correctness up until now, testing of the methods described in section 4.3 and 4.2 could be carried out.

4.4.1 Line-plane intersection

In order to test this functionality there were made an image processing function for the program to call automatically detecting every single keypoint in the images. This allowed for the hand-labeled keypoints to be gone. All these were projected onto the plane seen in Figure 4.7.

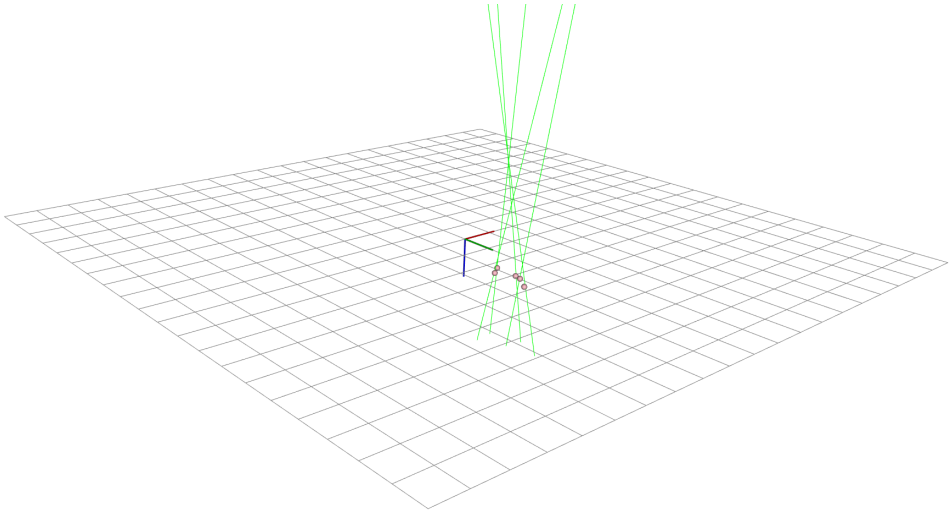


Figure 4.7: Rays projected from the opical centere, line-plane intersection method used to obtained the points on the plane

To see how the method compared to the real world scene, the measured ground through, it was measured the distance to the object of interest in milli meters. To get an understand of the spread size the a circle added was added with a radius of 30mm. There was also added A4 paper that enclosed all the objects.

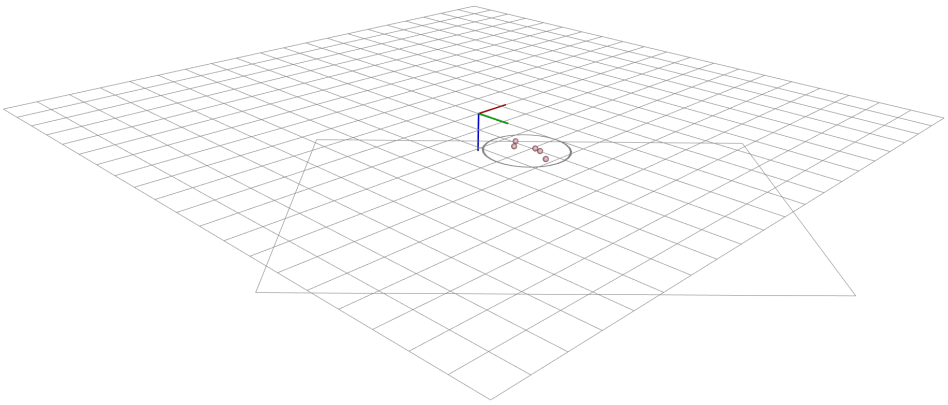


Figure 4.8: Rays projected on to the plane, centere of the circle representing the object

In Figure 4.8 we can see a type of uniform distribution, giving a somewhat of a peak in the center of where the object is expected to be in world coordinates.

4.4.2 Tiangulation

This section is testing the triangulation method described in section 4.3. The implementation of the method is based on two camera poses capturing the same keypoint. Projecting the point onto the focal plane by the the inverse of the calibration matrix. From these poses and points on the focal plane, there are constructed a set of 4 equations, that is solved with the singular value decomposition method.

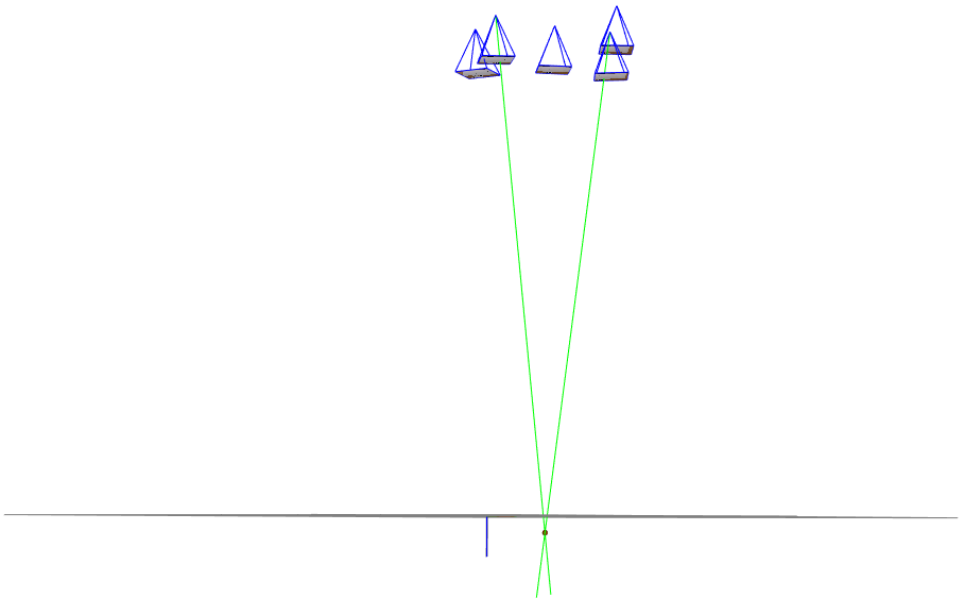


Figure 4.9: Rays triangulated using the SVD method, pose 2 and 4 used

In Figure 4.9 it can be seen that the method is giving out points that coincide with what is expected of such a method. The figure shows rays projected out of hand-picked poses that might have a high-quality estimate for their pose.

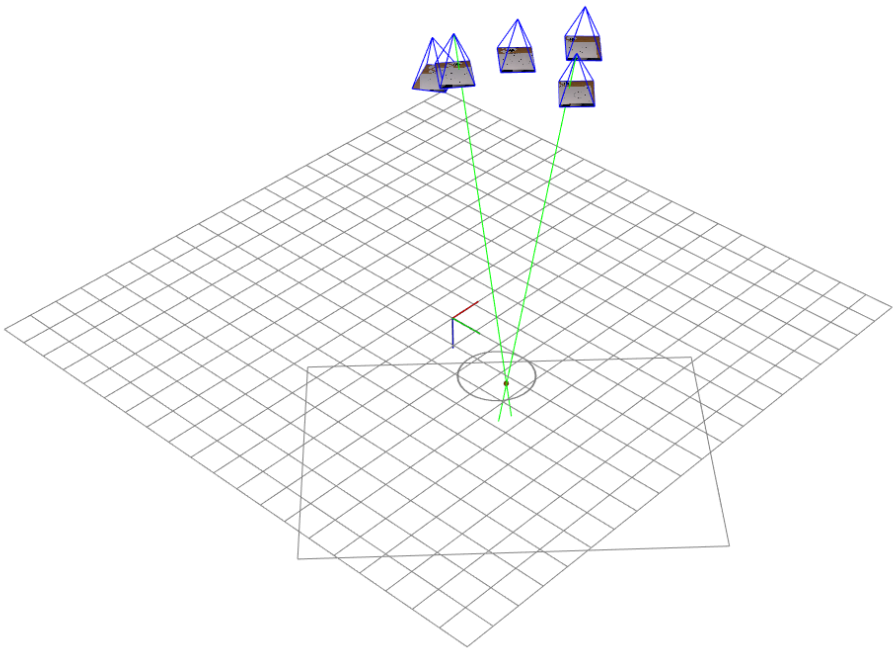


Figure 4.10: Rays from triangulated using the SVD method, pose 2 and 4 used

Taking the birds view of the scene into account (Fig. 4.10), it can be argued that the linear triangulation method is providing good estimates of the object observed. This is of course with a handpicked pair of poses, in Figure 4.11 it can be seen what can happen if one is not cautious having a good estimate on the poses, this error in the an estimate of the pose can propagate onto the estimate of the world coordinate outputted by the triangulation method.

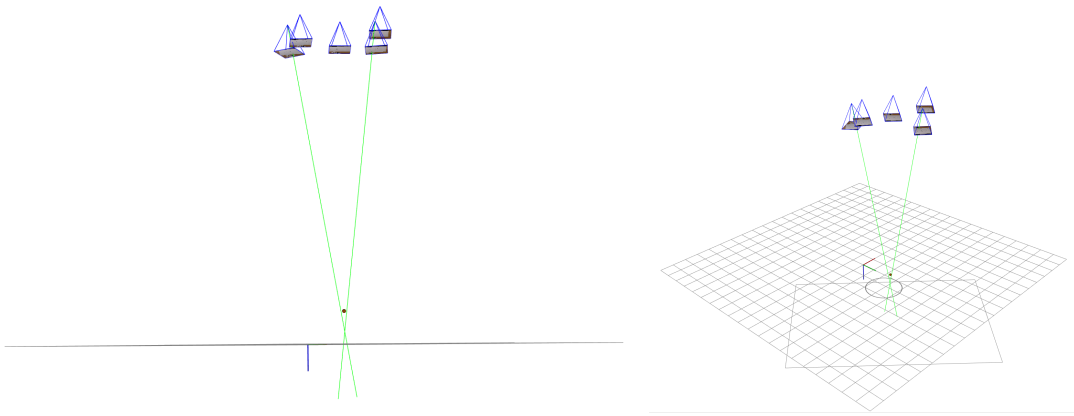


Figure 4.11: Rays projected down to the plane of the wold cordinate system, grid cell size 25mm

Chapter 5

Grid Mapping

Grid mapping is a common way of mapping in robotics, where the one is dividing the world into cells, each containing information of traversability. This, in turn, can be used by autonomous vehicles interpreting this map, for the purpose of traversing them in an optimal manner based on their goal.

For simplicity, the map was constructed as a rectangular matrix centered around the world frame with the properties defining the count of total cells and the size of each cell in meters.

The grid map can be seen at its own coordinate system. The need for converting observations made in the world frame to the grid map frame is present. The implementation method used is simply mapping one interval in the world frame in each cell in the grid map. This technique comes at the cost of a loss of accuracy when converting from a grid map frame to the world frame if the cell size is small. Figure 5.1 is displaying the grid map and its internal indexing

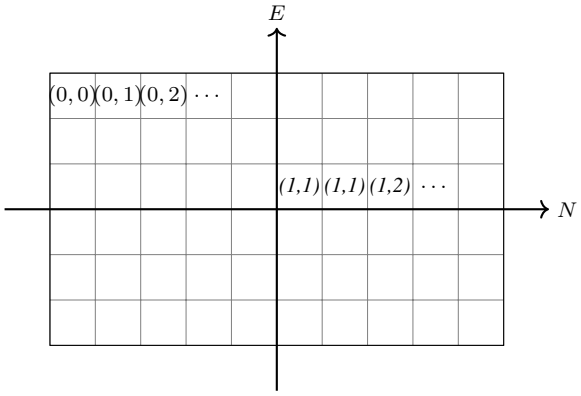


Figure 5.1: Gridmap defined with respect to NED

used. To the top left, it can be seen the traditional matrix indexing. And the inner right indexing can be described as the indexes of the grid map, making it easier to go from the grid mapping frame to NED.

Grid map data structure

There was a need to make the grid map its own class with its own data structure and utility functions. The map was decided to consist of three different layers as seen in Figure 5.2. The first grid was accumulating the raw landmark measurements of the object detection algorithm. The readings of the buoys were counted as negative values and the cages as positive. If enough measurements have come in, the measurements were possessed in two stages, first for the positive values, and finding local maximas, then for the negative values of the matrix. With this

operation, we have an estimate of where the structures lie. Then a series of utility functions can run to generate the safety line and in turn updating the map.

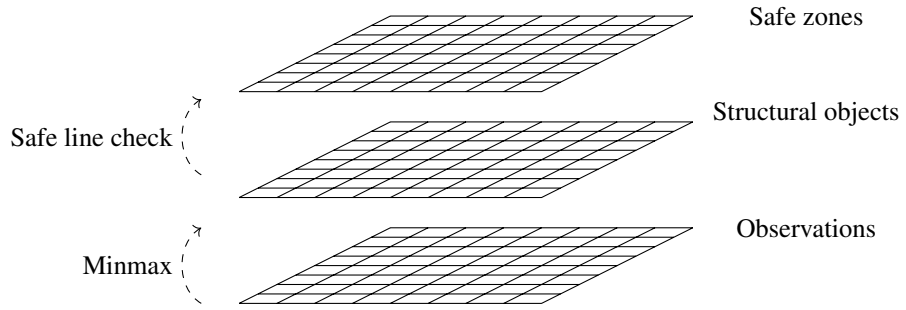


Figure 5.2: Grid map structure

Utilities and the gridmap update step

The software utility must be able to do the handling of map data by providing several helper functions. For example, a min-max based voting system, polygonal generative algorithms, and coordinate system transformation function to make for the interaction of the grid map in software.

Based on these utility functions the map can be able to update the map in the following manner:

1. Accumulate all observations
2. Find local maximas
3. Define outer points of buoys
4. Define outer points of circles
5. Generate safety line
6. Run ray casting algorithm

5.1 Safety line generation

For this problem, it can be introduced to a new idea. The idea of a safety line. The purpose of the line is to give some kind of encapsulation of the structures that can cause a collision. Ideally these lines would make up an collection of lines making up a polygon that fully captures the site seen in Figure 5.3. But for the real-time considerations the lines would rather have to make up a polyline, spanning from where the visual system initialized to the current region of operation.

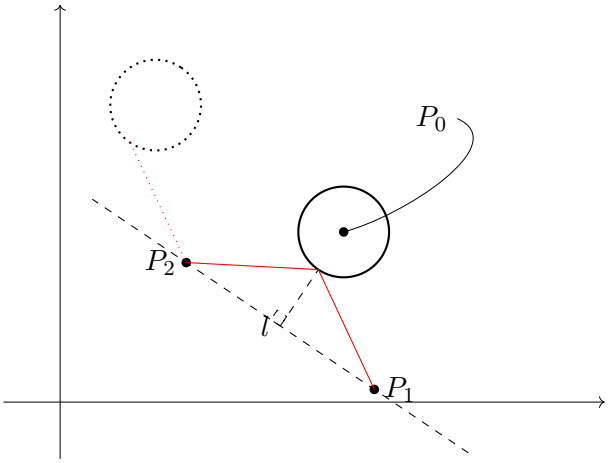


Figure 5.3: The geometric problem of defining a safety line

The construction of such a collection of lines there are some geometrical relations to consider. Constructing such an entity will be addressed in the following subsections.

5.1.1 Obtaining outerpoints

To generate the polyline the sequence of points (vertices of the line) needs to be found. For the buoys, the position of the vessel can be used. By Estimated its position by the GNSS signals being broadcasted. Defining the distance to the USV to the buoy one can determine by thresholding which side of the cage structure the buoy is located on.

5.1.2 Outer point of the circular shape

Knowing the location of the three points listed in 5.3 it can be investigated how to obtain the distance closest to the point P_0. The problem is depicted in Figure 5.4. As in the figure above the three points are described by P_0, P_1, P_2.

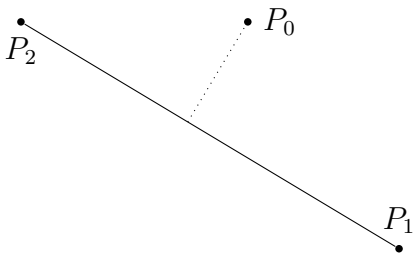


Figure 5.4: Distance from point to line

The general line passing through points is given as $y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$ is on the from $ax + by + c = 0$ where

$$a = -\frac{y_2 - y_1}{x_2 - x_1} \qquad b = 1 \qquad c = \frac{y_2 - y_1}{x_2 - x_1}x_1 - y_1$$

From literature, it is known that finding the distance from the point P_0 to a given line $ax + by + c = 0$ can be found by the equation (5.1) [25].

$$\mathrm{D}(ax + by + c = 0, (x_0, y_0)) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}. \tag{5.1}$$

Accounting for the fact that we are interested in finding the closest distance to the circle. We can write P_0 on the parametric from as

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} x_0 + r\ \cos t \\ y_0 + r\ \sin t \end{bmatrix} \tag{5.2}$$

Is possible to insert the parametric form of the shape into the Figure 5.2 and find the lowest function value of D.

$$\mathrm{D}(x_s, y_s) = \frac{|ax_s + by_s + c|}{\sqrt{a^2 + b^2}} \tag{5.3}$$

At first sight, this seems like there can be quite a simple derived analytical solution with some simple calculus, doing the derivation of $D(t)$ and setting the function to zero, and obtaining the minimal solution. This approach fast becomes out of hand, and then there can be implemented a simple minimization algorithm to find the minimum of the function with self-defined finite domain t.

$$\min_{t \in [0, 2\pi]} (D(t)) \tag{5.4}$$

,

Finally, the minimal solution t^* can be found and changed into cartesian coordinates x_s^*, y_s^*, thus making the following polyline possible to be drawn:

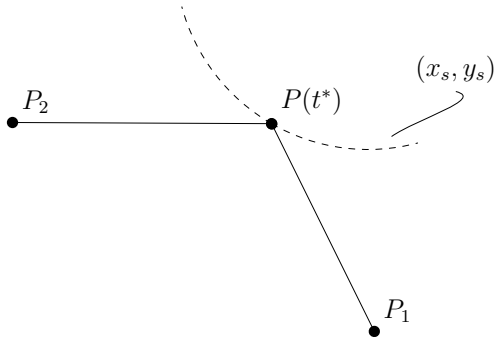


Figure 5.5: Segment of polyline connected with geometrical shape

5.2 Labeling grid cells

Assuming there exists a safety line (polyline) it can be investigated techniques of labeling the grid with respect to this line. The labeling of cells requires some concepts borrowed from computer science, these will be revised and converted to fit the problem of labeling.

Ray casting algorithm and line segment intersection

In computer graphics literature we find the similar problems of the one at hand. Where one is concerned about labeling a specific point in a region to be inside or outside a polyline [26]. The ray casting algorithm is a popular choice and have seen use in graphical software like .svg and postscript.

Suppose the$(P_0, P_1, \ldots, P_n)$ make up a polygon. Considering a polygon with vertices $\mathbf{v}_0 = P_1 - P_0, \mathbf{v}_1 = P_2 - P_1, ..., \mathbf{v}_1 = P_0 - P_n$. Take P_iP_{i+1} the inward egde is defined as positive and negative as $\times\mathbf{v}_i$ and $- \times \mathbf{v}_i$ respectively.

It can be done a test whenever a point in the plane interior or exterior of a polygon. This is done by casting a ray from the point at hand in any direction. By finding all intersections of the ray with the polygon edges, and classify each as either entering (if the ray vector and the outward edge has a positive inner product) or leaving (negative inner product). The difference between the leaving and entering intersections is called a winding number of polygon about the point.

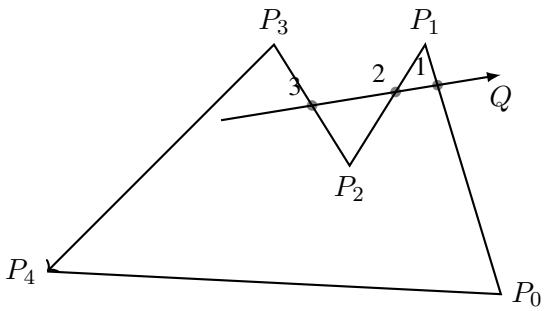


Figure 5.6: Illustion of intersection ray casting

In Figure 5.6 Q has a total of three intersections. With two inner products being negative and one positive. Leaving the sum to be a negative one, means that the ray is leaving the polygon. This is, of course, a good method if one is given a polygon, but in the real-time operation, we have a polyline (the safety line). Thus the technique requires some tweaking.

There exists a technique for deciding whether a region is said to be inside or outside a region of line segments. If we consider the situation of the operation many of the same principles can be applied. The safety line does not make up its own polygon, making it impossible to cast rays at random. However, if we consider the rays to be cast in the direction of the USV one can expect some results. Also, taking lines of no intersection to be defined as being traversable. With this the visual system can get a good estimate of traversability for the USV, assuming the UAV is operating far from the USV, thus sitting on enough a priori information. Figure 5.7 is showing this concept of casting rays in the direction of the USV.

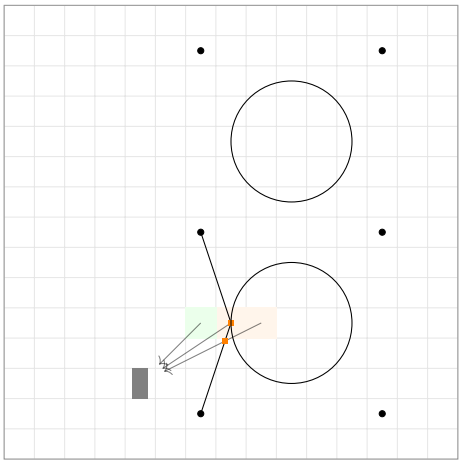


Figure 5.7: Example of showing line intersection algorithm in in practice

Although, the line intersection algorithm is a good starting point, it is missing the ability to make conservative judgments of what is safe (in the mapped regions). These shortcomings is seen in the Figure 5.8. It is clear that some of the cells are marked as safe when not being so. Also for this particular implementation, the intersection is undefined for a point starting on the line.

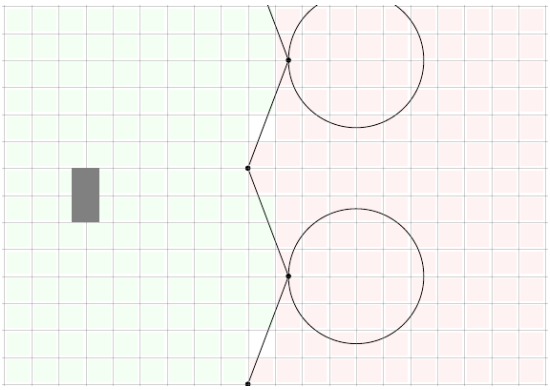


Figure 5.8: Example showing the lack of comprehensive filling of unsafe zones when applying the line intersection algorithm

In the following section, we look to increase the systems conservative when it comes to marking cells in the grid as safe.

Bresenham's algorithm

Bresenham algorithm is defined as "An efficient algorithm to render a line with pixels. The long dimension is incremented for each pixel, and the fractional slope is accumulated." by The National Institute of Standards and Technology [27]. It was named after J. E. Bresenham who developed it in 1963 and was first published in 1965 [28]. Originally, the algorithm was designed to draw lines on plotters.

The algorithm fills pixels from point p_1 to p_2 in the 2-d space, choosing the pixels that most closely fits the real line. This process is called rasterization [29].

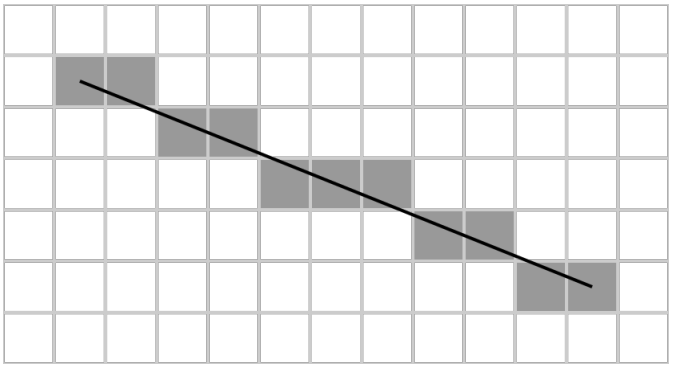


Figure 5.9: Bresenham's algorithm

In a nutshell, the algorithm is going along the x-axis, incrementing in each step by one and looking at an error then decides to within what way to move the y component.

In recent years there has been a modification to the algorithm, making it possible to specify the thickness of the line drawn [30]. The algorithm can be seen with a thickness equal to 1 in Figure 5.10

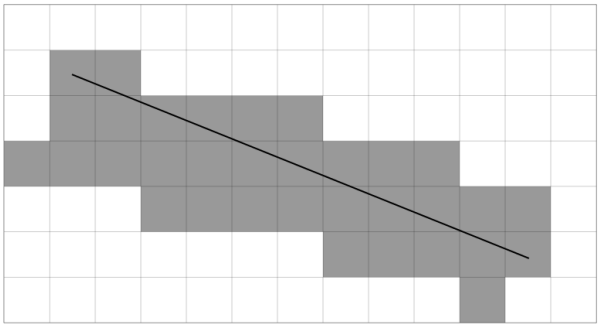


Figure 5.10: Murphy's modification to Bresenham's algorithm

Combining these two techniques, we first draw the safety line from the geometrical calculation and apply the intersection check. Then to get the more conservative guess we apply the line drawing step. By this sequence of operations, we now how a better and more conservative estimate of where the safe zones of operation are for the USV.

Other vehicles

There are not only surface vehicles operating the site. As mentioned in the introduction, this study is pointed towards mapping the safe zone for the USV. However, the static map of the objects, found in the inner layer in the of the grid map structure (shown in Fig. 5.2). It can be used to extrapolate this into accounting for the underwater vehicles as well.

5.3 Testing grid map utility

This test was done with the same aruco markers used for the 3D recovering methods tested in the prevoius chapter about computer vision geometry. However, the method used for obtaining the 3D landmark is not of interest in this section. The focus is on proving the concept of the grid mapping utility. To start off, we can consider the raw measurements made by the computer vision module. These are represented as pink spheres in Figure 5.11.

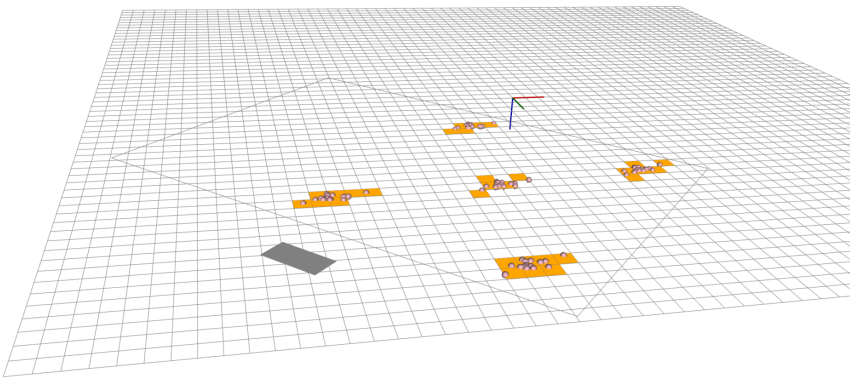


Figure 5.11: Accumulated landmark measurements

These measurements are accumulated into "buckets" corresponding to the cells of the grid map. This is the first layer of the grid map data-structure. These buckets accumulate all the measurements within its area.

Later, when a satisfactory number of measurements are accumulated. The min-max step can take place. This determines where the structures most likely have appeared (assuming Gaussian distributed measurements). This step is illustrated in Figure 5.12

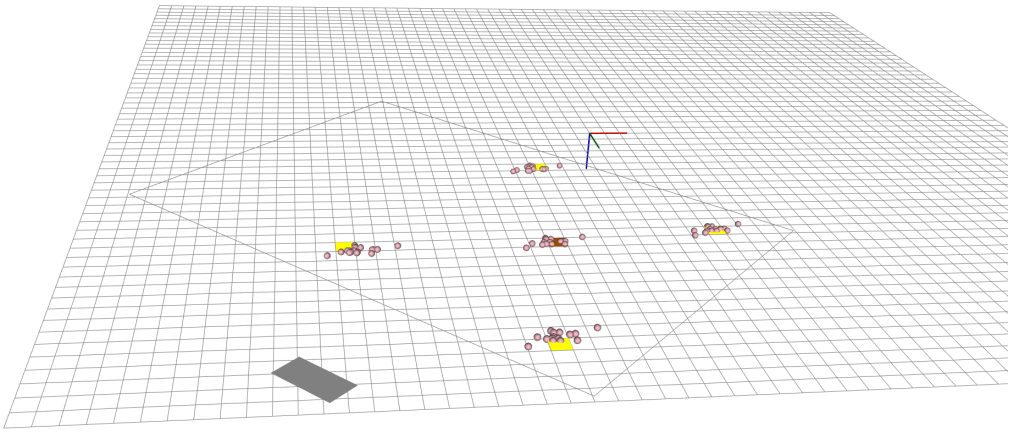


Figure 5.12: Local min-max

Lastly, the third and last layer of the grid map is filled. This was done by iterating through the grid casting rays in the direction of the USV. Figure 5.13 displays the results of this intersection step, where the white is representing traversability and red not.

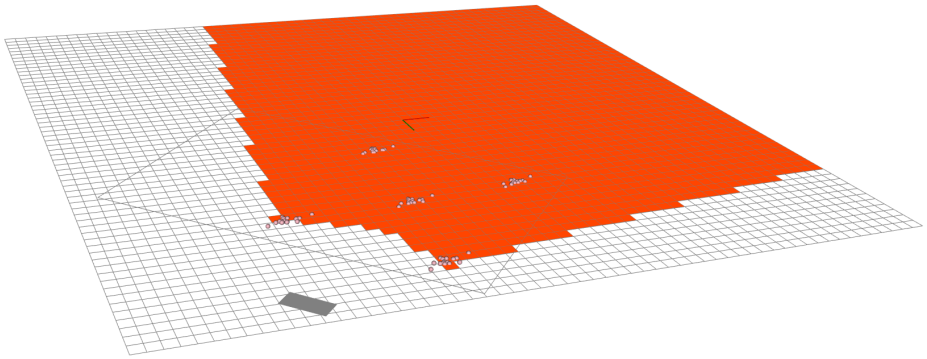


Figure 5.13: Ray intersection step

The method had some good characteristics, although there are some problems with how conservative the labeling of the grid zones is made to be. The labeling of the safe zones was done by the ray intersection algorithm in this test, as the modified version of the modified Bresenham's algorithm was not ready for implementation.

Chapter 6

Results

This chapter strictly contains results from the data gathered at Korsneset. Using methods described in previous chapters. First, the performance of each image processing algorithm will be tested on a sampled dataset from the video recording. Later on, some experimental results will be shown mapping the scene from Korsneset, using the projection method.

6.1 Detecting objects of interest

This section is looking at and comparing the algorithms reviewed for detecting the objects of interest. The objects being the cage and the buoy. There was used a series of 100 images in order to get a good sample size. The results of the image detection algorithms can be found, in this section, as plots or in the appendix as images.

Labeled set

For comparing each algorithm described in the previous theory, the ground truth need to be labeled. For the whole video section spanning over the four farms that would be many images to label, therefore the video were subsampled to an image sequence of 100 images. Each image is 1 second apart from each other. These images were then marked by the location of the buoy and cages with pixel values. Each image was hand labeled with the center of each object denoted in a .csv file. The circle was considered to exist in the image if 50 percent or more was visible. The buoys were defined to be in the image depending on if the whole structure was visible.

Preformance metrics

To quantify the performance of the object detection algorithm some metrics are needed. These metrics need to provide information about how well each algorithm is performing. Enabling a comparison to be done. From the literature of statistics have the confusion matrix metric, also known as the error matrix (Fig. 6.1).

There need to be some conventions in the field of image processing how to interpret this matrix. In [31] we find one definition of the confusion matrix:

- TP: true positives, i.e., number of correct matches;
- FN: false negatives, matches that were not correctly detected;
- FP: false positives, proposed matches that are incorrect;
- TN: true negatives, non-matches that were correctly rejected

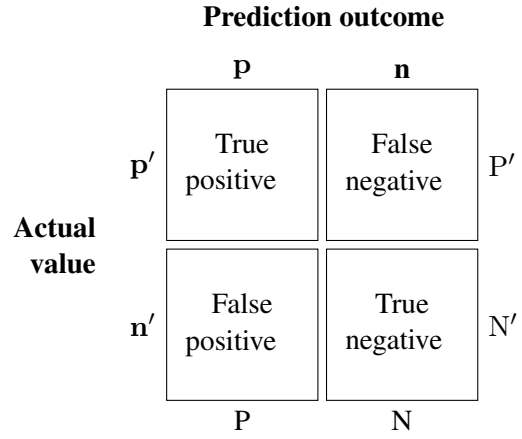


Figure 6.1: Confution matrix

Making some clarification for the application of the detection of objects in this section we have: TP (True Positive) accounting for the algorithm guessing that there is an object and there is. FP (False Positive), when the algorithm has no outputs in the location of the object. FN (False Negative), when there is an object that is not detected. TN (True Negative), when there exists no object and there are no outputs that there exists one. However, the true negative will not be considered in this result section. As it does not apply to the image processing algorithms developed.

The correct guess is defined by its Euclidean distance to the center of where the object is labeled to exist. By setting a distance threshold around the labeled point. This gives the metric some slack, and it does not need to output the exact pixel where the object is defined to exist. Guesses falling outside this Euclidean distance of the center are considered as being wrong.

Also, some useful derivations from the confusion matrix have been done. Where the concepts false positive rate (FNR) and the false discovery rate (FDR) especially useful metrics for the application at hand. These are defined as

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} \tag{6.1}$$

and

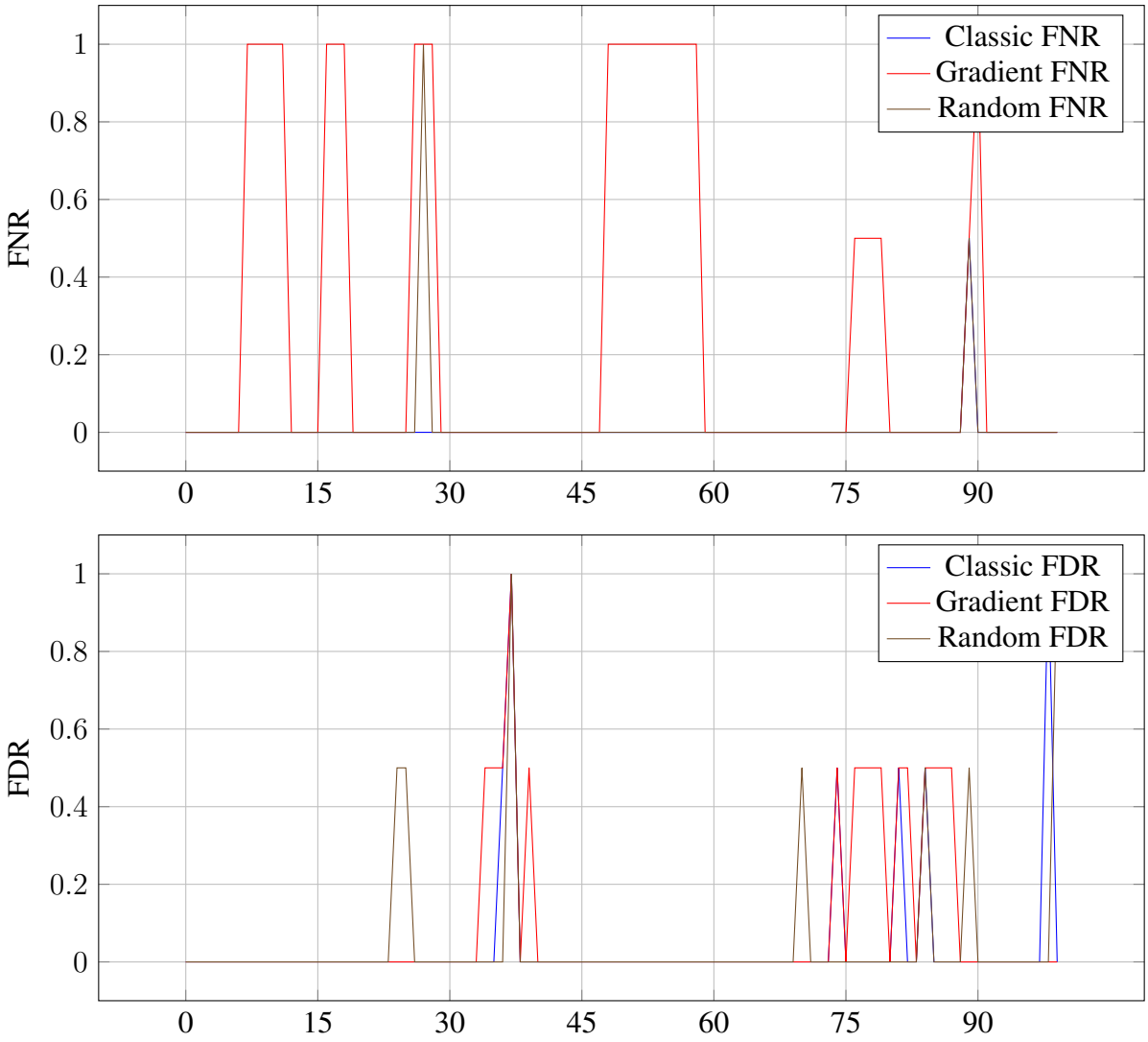
$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}} \tag{6.2}$$

respectively. These metrics are aimed at quantifying the performance of a given prediction in the interval of zero to one. Where the predictive performance of the algorithm is high these values are close to zero and close to one when it's low. Therefore, these metrics are defined to be zero when all components of the equation are zero, effectively yielding zero when there are no false predictions or false discoveries being made.

6.1.1 Circle detection

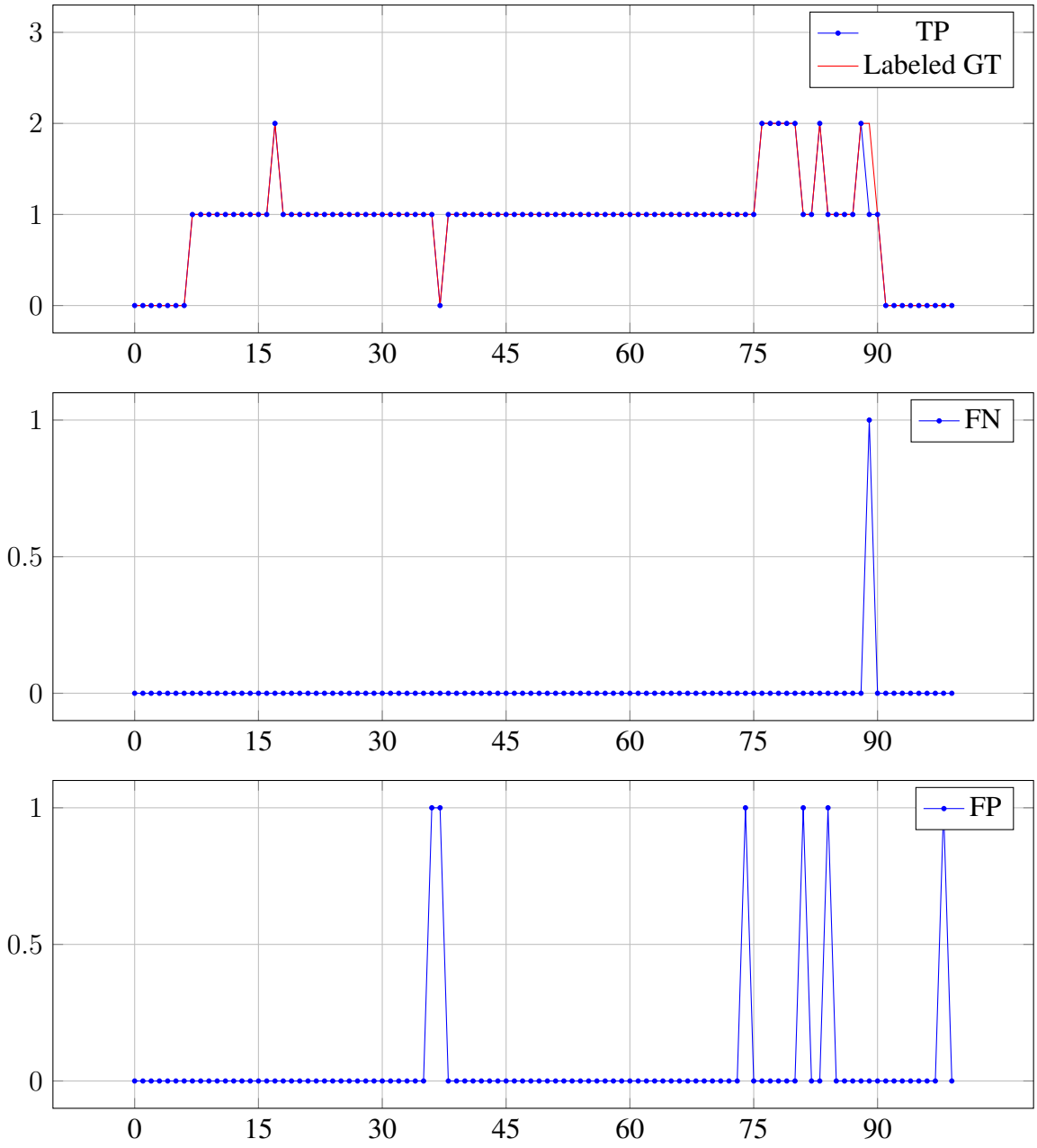
This section is looking at the results of circle detection. The same error metric as described above is used. The circle is defined to exist in the image if more or half of the circle is present. The same Non-maximal suppression operation is used for all the methods below. That is if one of the methods is outputting the same circle in a considerably close distance to each other this will be tackled by the same Non-maximal suppression operation for all the circle finding methods. The parameters are set with some in thought behind, but truly not done by finetuning. The fact that the parameters are not tuned in a tireless process, can be a good indicator of the robustness of the algorithms.

The following are the results of the three methods used to detect cages in the edged image series found in Appendix A. False positive rate and false discovery rate respectively:



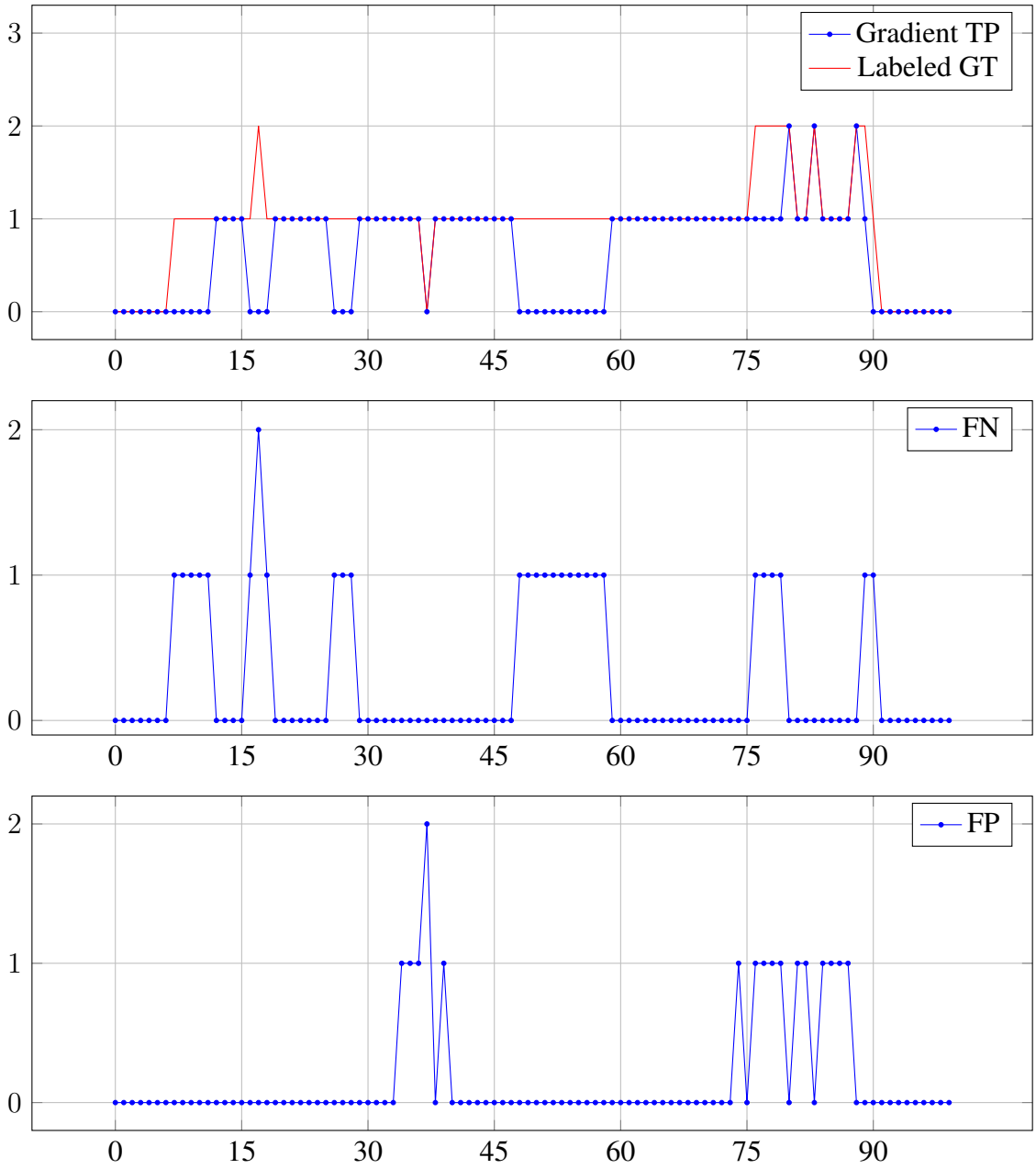
Classical Hough circle detection

As expected the classical approach is performing extremely well. As this method is analyzing the edged image in the most thorough way of these methods, iterating a large number of times. In total there is only 1 circle missed, with a corresponding reasonably low FP rate.



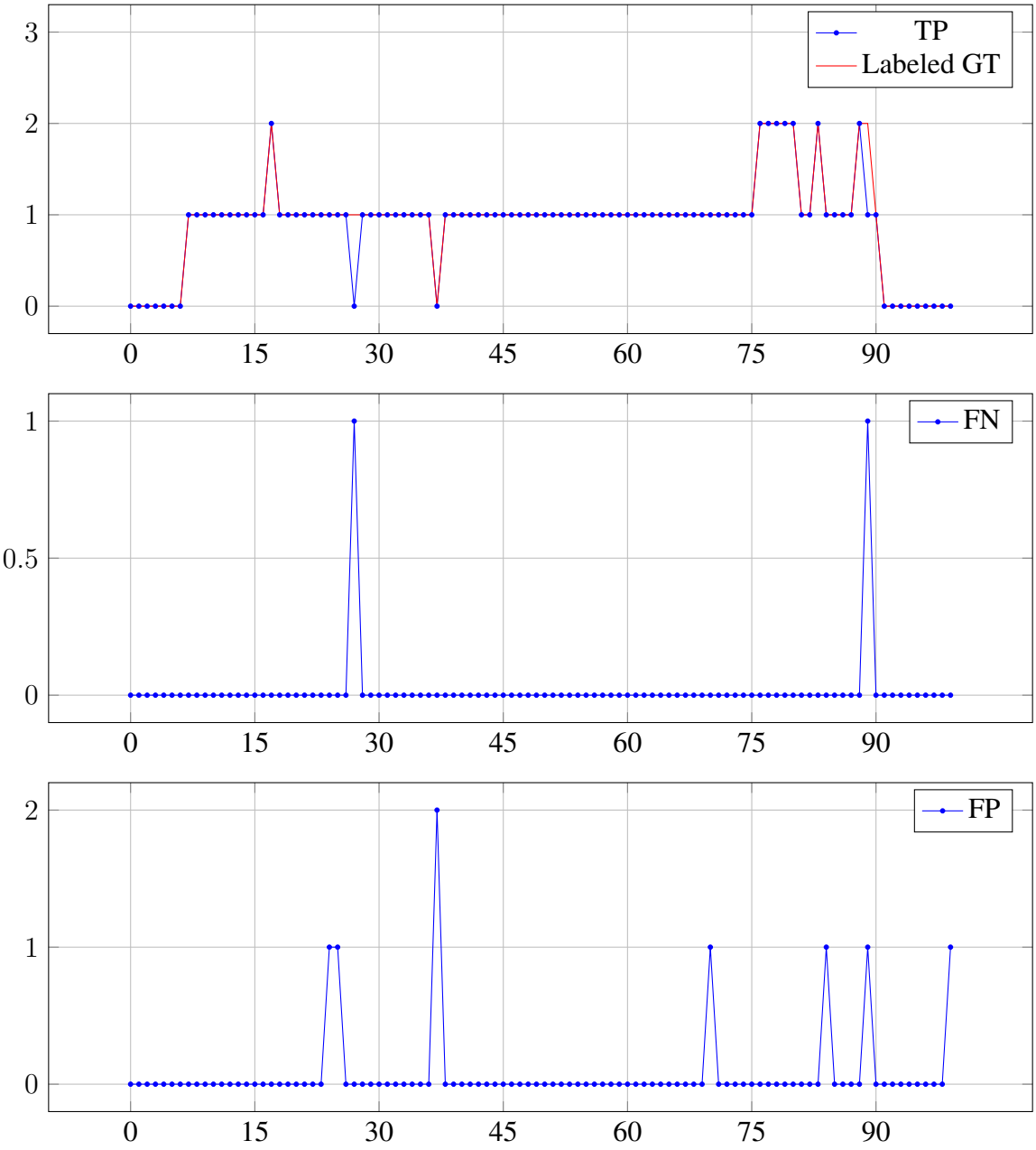
Gradient Hough circle detection

The gradient method showed considerable worse performance. This might have accrued for a number of reasons. One reason might have been dependent on how the ground truth was defined. The method depends on lines that intersect in the middle of the circle. Recalling that the circle is defined to exist in the image if 50 percent or more of the circle is visible. Thus, making the algorithm prone to the noise of the circle is deformed in a way the accumulated center point might have ended outside the image, making for a circle that is not detected. Secondly, the gradient line was accumulated across the whole image making for some maxima might that has had contributions from several circles.



Randomized circle detection

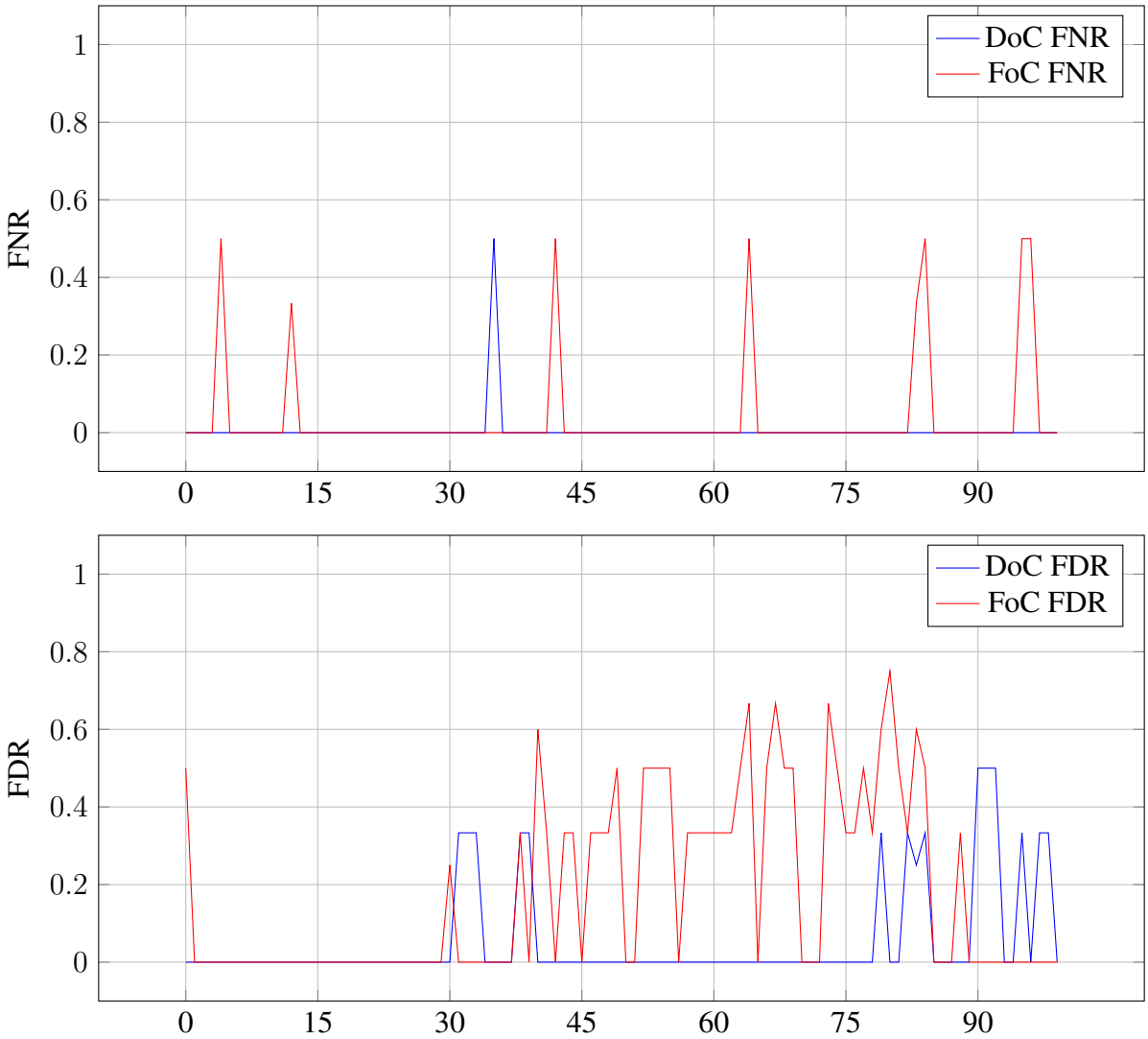
This method was the most impressive performer taking runtime into account. However, in the long run, the randomized approach will have some extremes where there it is proposed a circle where there absolutely is not a circle present, this is in the nature of nondeterministic approaches. Also, worth noting. This is not an exact metric of how the algorithm is performing, acknowledging the randomized circle detection is not a deterministic one. Although this is a quite large dataset, the overall performance will vary. Thus, one needs to be careful interpreting the results.



6.1.2 Buoy detection

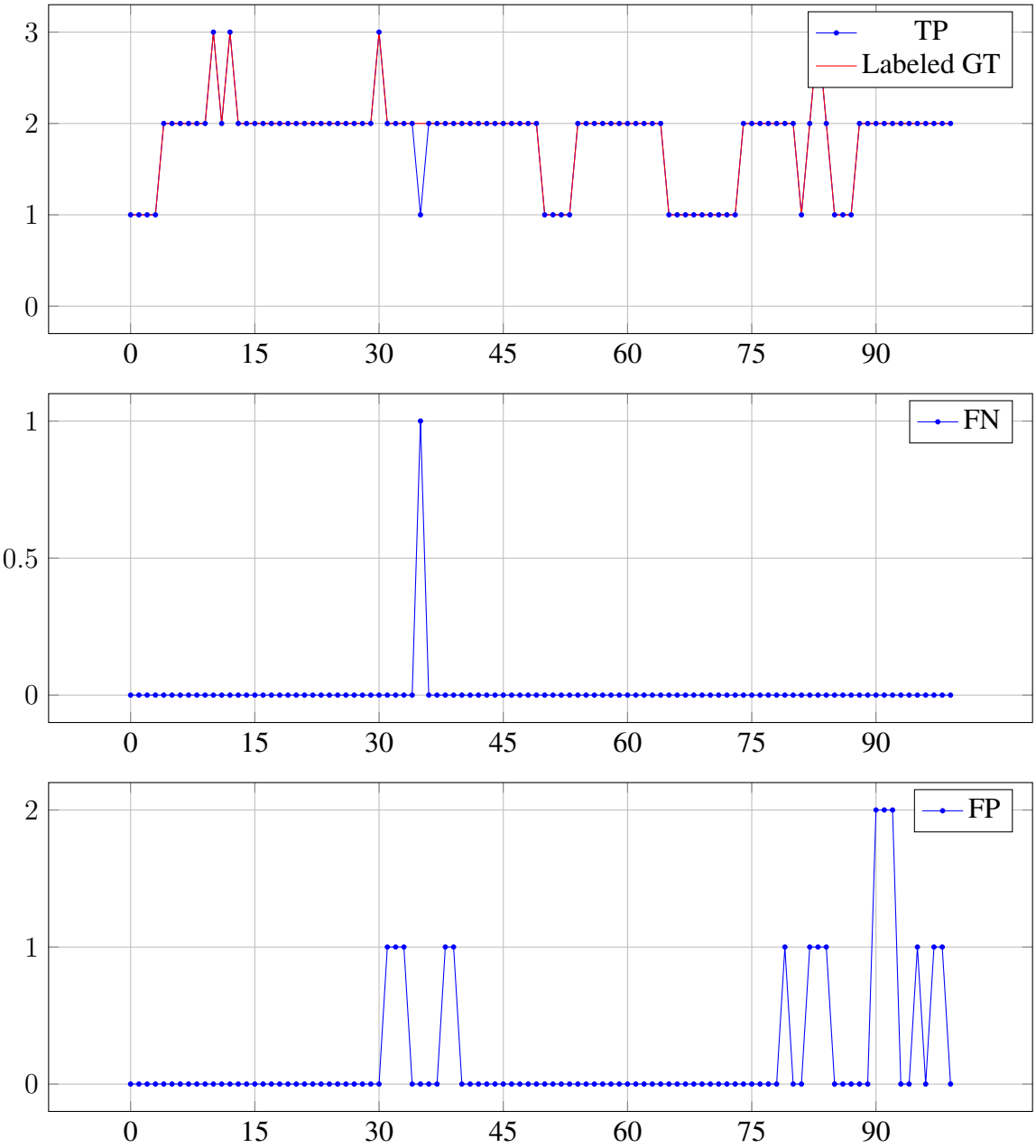
Displays the results of the algorithms used in detecting the buoys. Described as high-intensity regions in the previous section about object detection. Contrary to the previous section, these results were obtained with the use of software libraries. The parameters used were inspired by the hand-coded tests.

The following are the results of the three methods used for buoy detection, obtained by processing the image series of Appendix A. False positive rate and false discovery rate respectively:



Difference of hessian

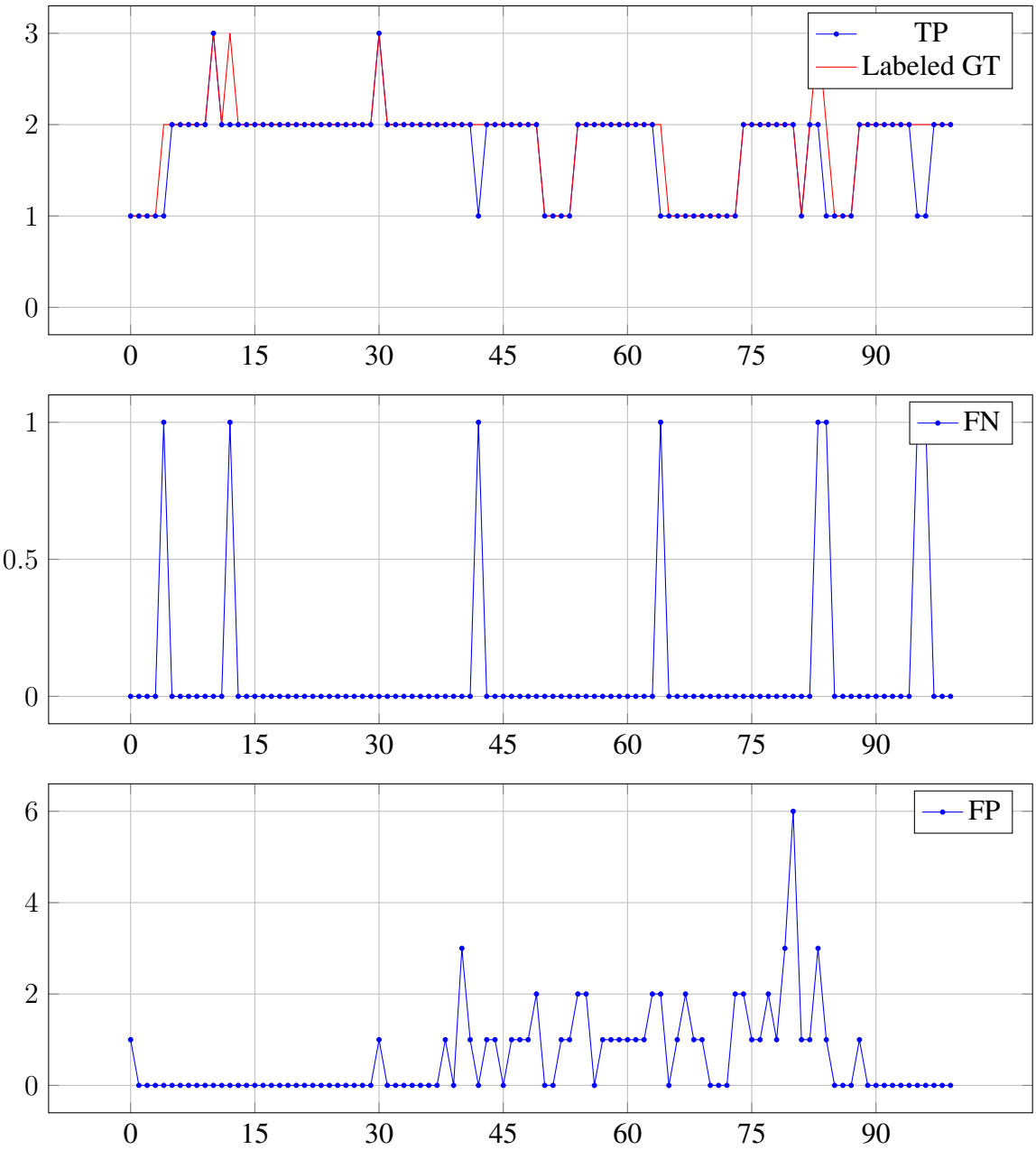
The sigma of the Gaussian filter was set to start at 3 and end at 7. The scale space was set to be incremented in a linear fashion at an interval of 10. This approach is impressive considering the constant splashes inside the fish cages. Also, the result is possible to have suffered from the way the ground truth was defined. In the images, it can be observed that the method is very aggressive towards the regions that are exerting strong illumination at the edges of the image.



Filtering of controus

The contours were set to filter at levels seen at similar levels at the previous section about contours. The contours where fileted on hierarchy, area, circularity and intensity.

There were done some iterations regarding the parameter setting. The problems with the splashes from the fish cage seem to be present. The illumination levels are similar to the buoys considering the whole set of images. One consideration could be to have an adaptive threshold set within the processing of each image, because of the changing lighting situation.



6.2 Experimental results from korsneset

In this section, there are some experimental results projecting the keypoints found by object detection. As one will see, these results had not satisfactory accuracy, making the problem of creating the grid map tough. The accuracy grid mapping resolution needed for the min-max step was too large, making it hard to estimate where the buoy was located on the grid map.

The error seen in the following results may have originated from a few sources. However, the main problem could be that the internal measurements made by the drone were not coupled with the gimbal. There was no data on the rotation of the gimbal. This might have caused the problems. The best estimate for the actual rotation of the gimbal was the internal drone measurements. Having a poor estimation of the camera rotation at an altitude of 100 meters could be the reason for the large drift that accrued.

In the following figures, the rotation of the camera (gimbal) was estimated using the internal measurements of the drone. The image processing module was set to do conservative object detection, making for some of the smaller buoys (often confused with splashes in the fish farm) not being projected and displayed.

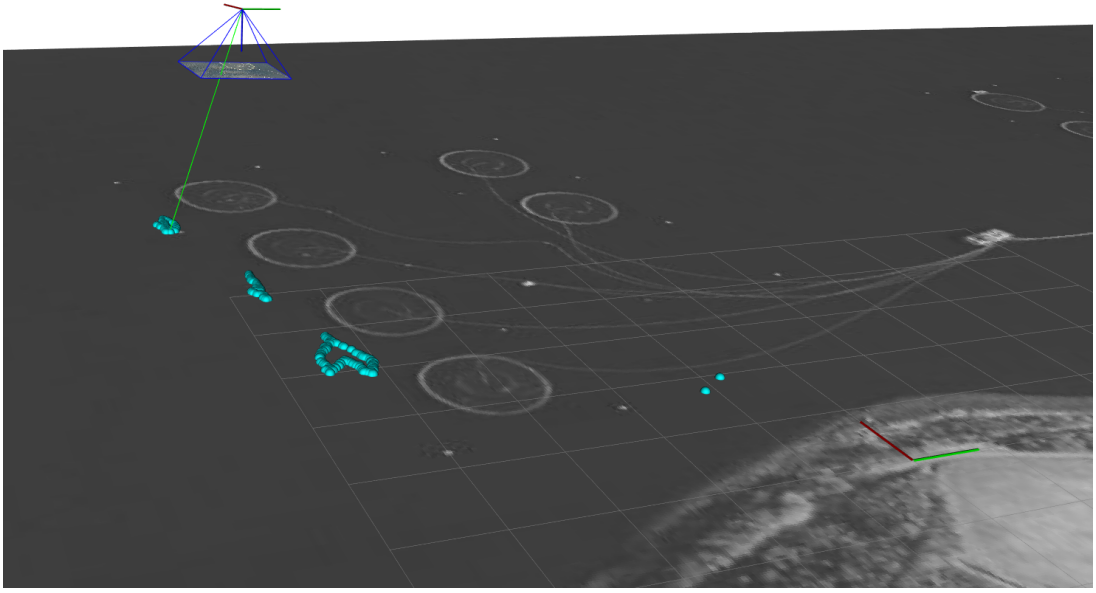


Figure 6.2: UAV projecting visual measurements

Figure 6.2 is depicting the UAV projecting the visual measurements down on the sea surface. As one can see, the divination of the measurements is quite large.

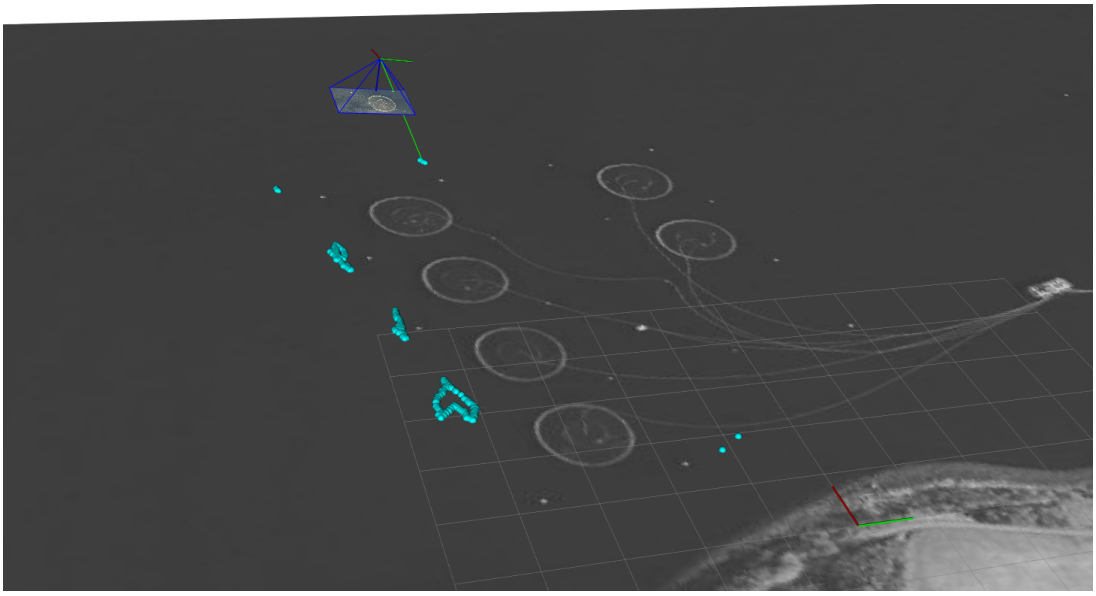


Figure 6.3: UAV projecting visual measurements

In this method, there is a certain consistency of the measurements that were obtained. Figure 6.3 shows that the visual system is able to track the buoys and represent the geometrical structure of the changing system with some clarity.

Another approach

If one considers that the gimbal truly was set to be planar with the sea surface at the start of the flight. Also, the control system of the gimbal was fast enough. This could give indications that one could approximate the change in pitch and roll to be zero. This was experimented with. Figure 6.4 shows that the system is able to get a better estimate early on in the flight. In fact, both the first and third buoys are more consistently perceived.

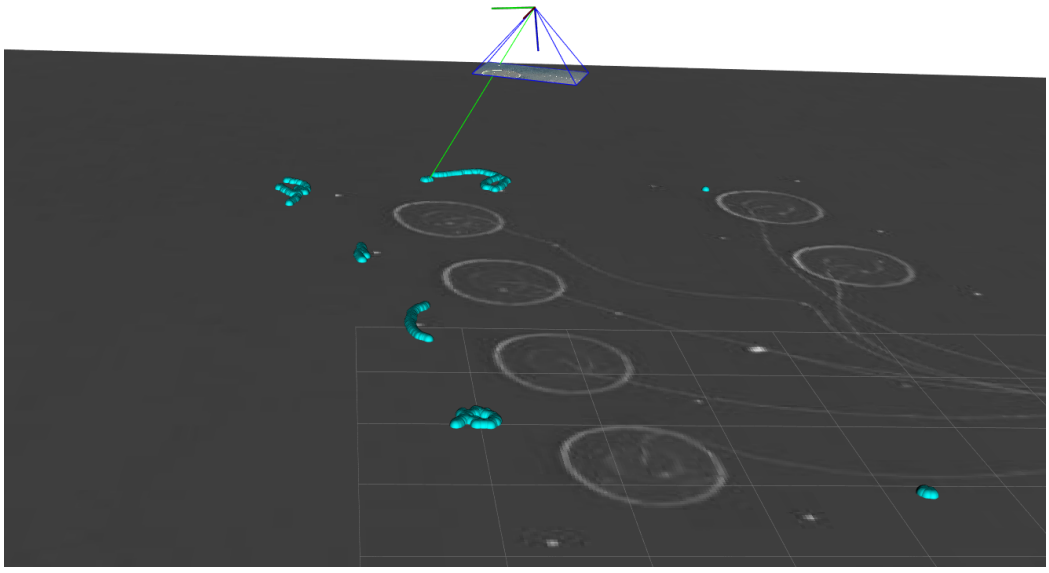


Figure 6.4: UAV projecting visual measurements

However, in Fig. 6.5 is it possible to see that this assumption does not hold. Later on in the flight, the UAV was carrying out rapid maneuvers, where it can be thought that the gimbal control system fails to track. Thus making the assumption of no pitch and roll a weak one.

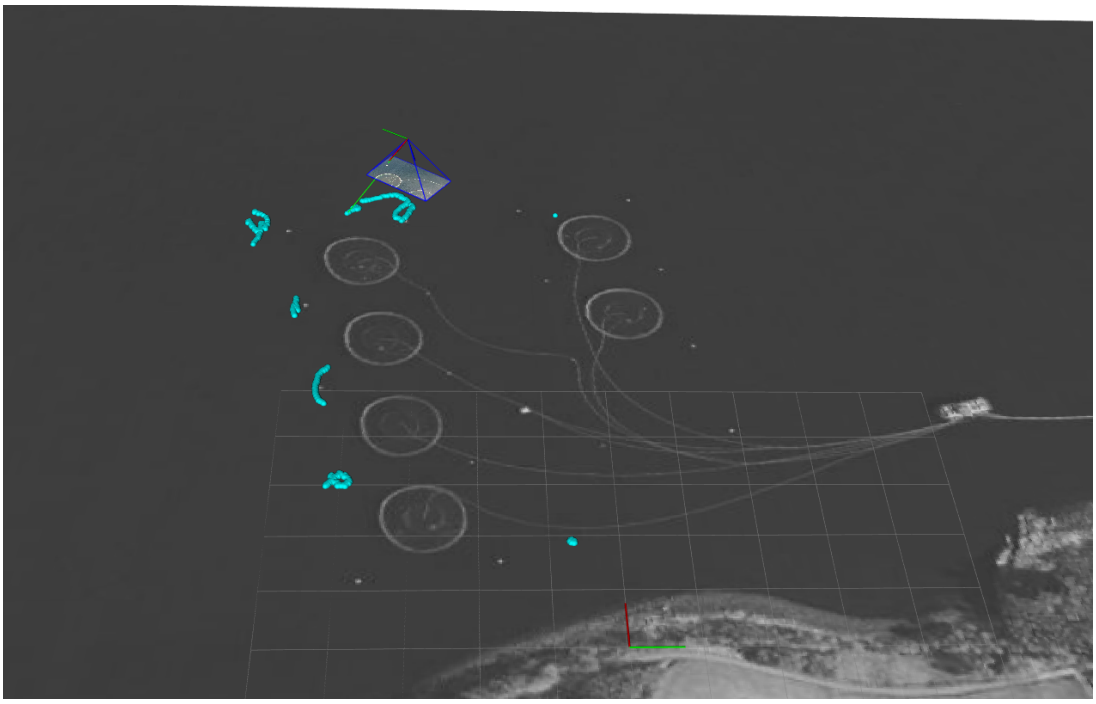


Figure 6.5: UAV projecting visual measurements

In both the tests, there were seen drift associated with the location of the structures. These structures are expected to have a static placement with small divinations, because of their anchor and the good weather considerations at the time of flight. Dift of this magnitue is unlikely to have originated from the currents in the ocean. The possible error sources are discussed in more detail in the proceeding chapter.

Chapter 7

Discussion

This study can be been seen as a step in the direction of mapping the safe zones of operation at aquacultural growing facilities. Methods of detecting objects of interest based on simplistic models were reviewed and tested. Furthermore, methods of retrieving the 3D landmarks have been studied, the methods tested on two independent data sets. Techniques of extracting the structures based on the 3D landmark measurements have been proposed. Based on these structures, a way of carrying out mapping the safe zones has also been proposed.

The techniques used for object detection has been largely satisfactory. There has been a quite high accuracy for namely for the classical Hough circle transform to detect cages and the difference of hessian technique for detection of buoys. Although they both showed good results in detecting nearly all objects when present, there is an argument to be made that the parameters of the algorithm should be tuned to make somewhat more conservative guesses while carrying out object detection in real-time. In the sense that one is trading lower frequency of visual measurements obtained in the return of having fewer faulty readings.

In addition to this, two ways of retrieving the 3D landmark were tested. The technique based on the assumption that the sea surface can be thought of as falt seemed good in the sense of low complexity. Although, one should be aware that there exist highly sophisticated methods of building systems that take more views into account when estimating the 3D landmark. These techniques were not tested and may have yielded good results.

One way of building the grid map infrastructure was proposed. However, the implementation was not done in time for the deadline of the assignment. The grid map as of now has a class like structure where it is incorporating the three levels of the grid structure. Where the first one is storing raw 3D landmark measurements, the second one is storing the local maxima giving an absolute estimate of where the objects were detected. However the last one, the safe zone map, still needs implementation. As of now, the grid gap has the utilities for filling the two first levels. However, the utilities for creating the safe zone map is not complete. It still needs more utilities such as checking the intersection and more.

Reflection and Future work

The data from the flights of Korsneset was at times problematic. It is not an intuitive task synchronizing the video taken with the drone internal data. Also, there was no real way of exactly knowing the position of the gimbal, at high altitudes, this becomes especially problematic. This may indicate that the task was very challenging tackling in the timespan of one semester.

There have been efforts in recovering the pose of the camera to get a better understanding of how the gimbal behaved during flight, but these techniques have yielded no immediate results and were not included. Efforts such as deploying a visual odometry software [32] were tested,

but the tracking was not robust enough, losing track every few minutes. These results were not included.

With this in mind, it may be better to directly mount the camera on the bottom of the UAV when carrying out future experiments. This is highly likely to yield better results. This kind of mounting option would suggest one can use the internal drone measurements directly in calculating the pose of the camera with respect to the world coordinate.

One can also consider in future work, in regard to helping pose estimation. Techniques of directly applying the visual measurements of the UAV to enhance the cameras pose estimate. These days the VO (visual pose estimation) is a hot topic and a truly wide one. The overall performance of the system could have been positively impacted by a visual module helping out the internal measurements. Examples of successful implementations relating to this can be found [33] . Helping out the pose estimation can be useful for further improvements in mapping the scene given the data gathered from Korsneset in 2018.

Also, while considering further improvements to the system. One topic that is left out and could be of interest in future work, is the topic of deep learning. It is one of today's highly researched topics and its application in the field of computer vision is wide and diverse. The techniques have shown to be highly effective compared to classical methods. This way of recognizing objects, one would not need to include or specify a model of the objects of interest, with back propagation the model is able to obtain this model of how to classify itself. There are of course many things to specify in regards to what type of architecture the network will have. Opposite some of the more classical methods, the system does not require an exact model of the world.

Chapter 8

Conclusion

In this thesis, objects making up the structural geometry of the fish farm where reviewed. The subset accounting for the structural geometry of the cage system where considered and defined. Assumitons made upon these objects, enabling some simple models to be dervied. The cage was modeled as a circular shape and the buoy as a high-intensity region with circular properties.

Various methods of detecting these objects based on their assumed attributes where studied. A total of 3 different methods was reviewed for detecting the modeled circle, methods heavily or fully reliant by the Hough transform. Two methods of detecting the high-intensity region, one based on computing properties of the regions in the binarized image the other inspired by critical points in scale-space. The capabilities of the methods were tested on an image set acquired from Korsneset aquacultural production site. The results were measured against each other and analyzed.

To gain insight in recovering 3D coordinates, theory relating to the recovering of 3D landmarks where investigated. These methods where tested and the techniques validated. Method that showed the best result where picked. From these tests, it was concluded that the projection of image points was most promising to be carried out in the mapping of the structural geometry. This method was dependent on modeling the seafloor and projecting down the detected image points to the plane. This teqniuqe was used testing the grid mapping utility and for acquiring experimental results from the dataset of Korsneset.

Methods and functionalities of the grid mapping utility where explored in some detail. Some of the functionalities where tested, such as accumulating the 3D landmark measurements in the grid map coordinate frame and determining the static location of the objects perceived. Also, based on this static digital map a simple method for obtaining an estimate for the safe zone of traveral where tested.

The accumulated experience was at last tested on the dataset from Korsneset. This ended up in some experimental results, recognizing that the drift seen is highly likely to have accured form the decoupled dynamics of the UAV and the gimbal.

Bibliography

[1] G. V. Nyboe, "Estimation of structural geometry: Sensors and software to generate real-time maps of structural geometry in fish farms," Jun. 2019.

[2] Norgeskart. [Online]. Available: https://www.norgeskart.no/

[3] F. Cardia and A. Lovatelli, *Aquaculture operations in floating HDPE cages. A field handbook.*

[4] DJI matrice 600 pro - DJI. [Online]. Available: https://www.dji.com/no/matrice600-pro

[5] D-RTK GNSS - specs - DJI. [Online]. Available: https://www.dji.com/no/d-rtk/info

[6] "Simple library for converting coordinates to/from several geodetic frames (lat/lon, ECEF, ENU, NED, etc.): ethz-asl/geodetic_utils," original-date: 2015-09-24T07:17:50Z. [Online]. Available: https://github.com/ethz-asl/geodetic_utils

[7] An invitation to 3-d vision - from images to geometric models | yi ma | springer. [Online]. Available: https://www.springer.com/gp/book/9780387008936

[8] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press.

[9] J. Prewitt, "Object enhancement and extraction, picture processing and psychopictorics," vol. 59, pp. 75–149.

[10] L. S. Davis, "A survey of edge detection techniques," vol. 4, no. 3, pp. 248–270. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0146664X7590012X

[11] J. Canny, "A computational approach to edge detection," vol. PAMI-8, no. 6, pp. 679–698.

[12] P. V. C. Hough, "METHOD AND MEANS FOR RECOGNIZING COMPLEX PATTERNS." [Online]. Available: https://www.osti.gov/doepatents/biblio/4746348-method-means-recognizing-complex-patterns

[13] ——, "Machine analysis of bubble chamber pictures," vol. C590914, pp. 554–558.

[14] M. Nixon and A. S. Aguado, *Feature Extraction & Image Processing for Computer Vision, Third Edition*, 3rd ed. Academic Press, Inc.

[15] C. Kimme, D. Ballard, and J. Sklansky, “Finding circles by an array of accumulators,” vol. 18, no. 2, pp. 120–122. [Online]. Available: http://doi.acm.org/10.1145/360666.360677

[16] L. Xu, E. Oja, and P. Kultanen, “A new curve detection method: Randomized hough transform (RHT),” vol. 11, no. 5, pp. 331–338. [Online]. Available: http://www.sciencedirect.com/science/article/pii/016786559090042Z

[17] T.-C. Chen and K.-L. Chung, “An efficient randomized algorithm for detecting circles,” vol. 83, pp. 172–191.

[18] S. Suzuki and K. be, “Topological structural analysis of digitized binary images by border following,” vol. 30, no. 1, pp. 32–46. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0734189X85900167

[19] An iterative image registration technique with an application to stereo vision. [Online]. Available: https://dl.acm.org/citation.cfm?id=1623280

[20] T. Lindeberg, “Feature detection with automatic scale selection,” vol. 30, no. 2, pp. 79–116. [Online]. Available: https://link.springer.com/article/10.1023/A:1008045108935

[21] C. Harris and M. Stephens, “A combined corner and edge detector,” in *In Proc. of Fourth Alvey Vision Conference*, pp. 147–151.

[22] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and R. Medina-Carnicer, “Generation of fiducial marker dictionaries using mixed integer linear programming,” vol. 51.

[23] F. Romero Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, “Speeded up detection of squared fiducial markers,” vol. 76.

[24] OpenCV: cv::viz namespace reference. [Online]. Available: https://docs.opencv.org/3.4/d9/d62/namespacecv_1_1viz.html

[25] J. P. Ballantine and A. R. Jerbert, “Distance from a line, or plane, to a point*,” vol. 59, no. 4, pp. 242–243. [Online]. Available: https://www.jstor.org/stable/2306514

[26] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice (2Nd Ed.)*. Addison-Wesley Longman Publishing Co., Inc.

[27] Bresenham’s algorithm. [Online]. Available: https://xlinux.nist.gov/dads/HTML/bresenham.html

[28] J. E. Bresenham, “Algorithm for computer control of a digital plotter,” vol. 4, no. 1, pp. 25–30.

[29] A. LaMothe, *Tricks of the Windows Game Programming Gurus*, 2nd ed. Sams.

[30] Murphy’s modified bresenham line algorithm. [Online]. Available: http://homepages.enterprise.net/murphy/thickline/index.html

[31] T. Fawcett, “An introduction to ROC analysis,” vol. 27, no. 8, pp. 861–874. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016786550500303X

[32] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," vol. 31, no. 5, pp. 1147–1163. [Online]. Available: http://arxiv.org/abs/1502.00956

[33] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2502–2509. [Online]. Available: https://ieeexplore.ieee.org/document/8460664/

Appendix

APPENDIX A.1

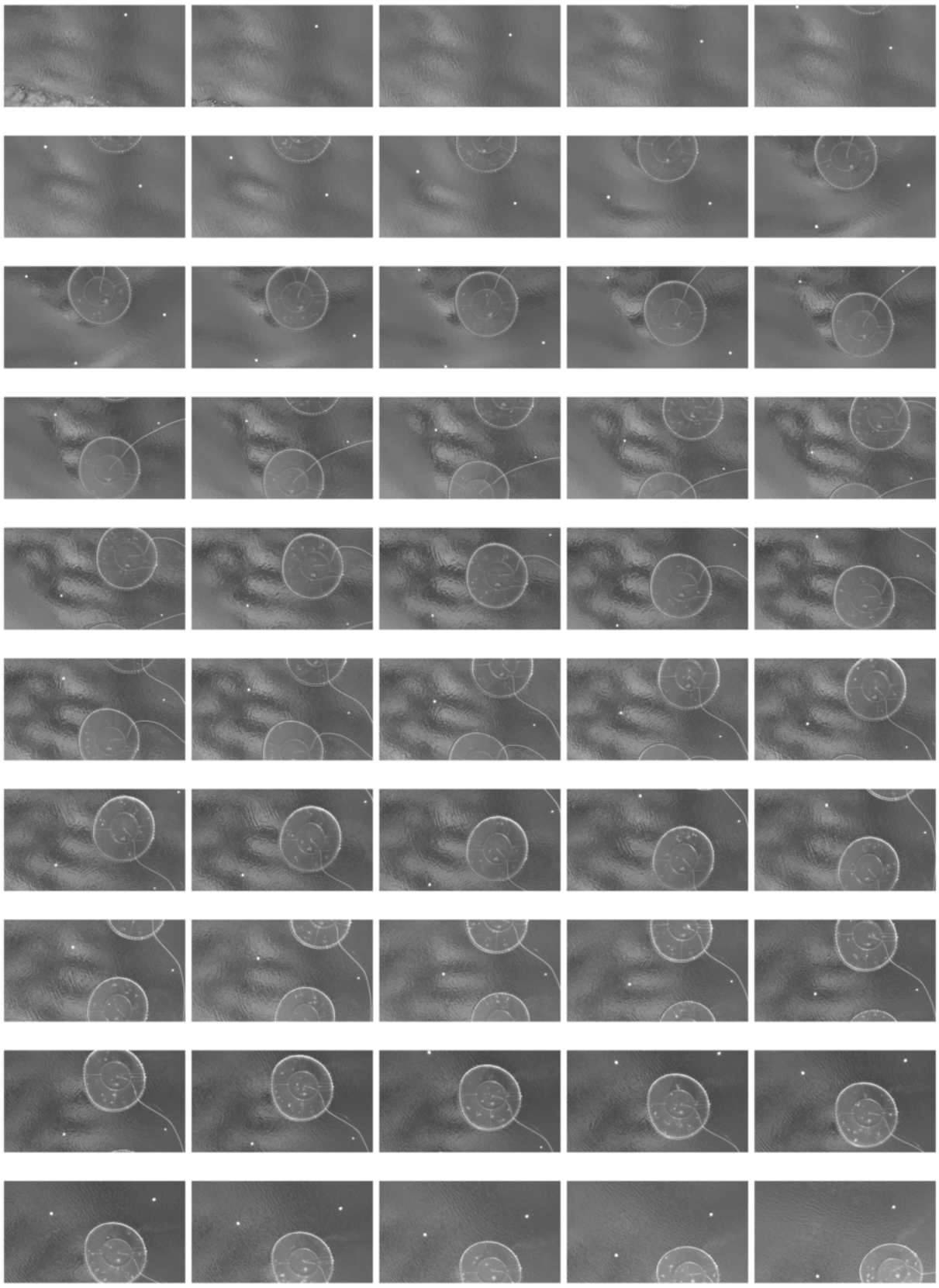


Figure 8.1: Images 1-50 graysacle originals

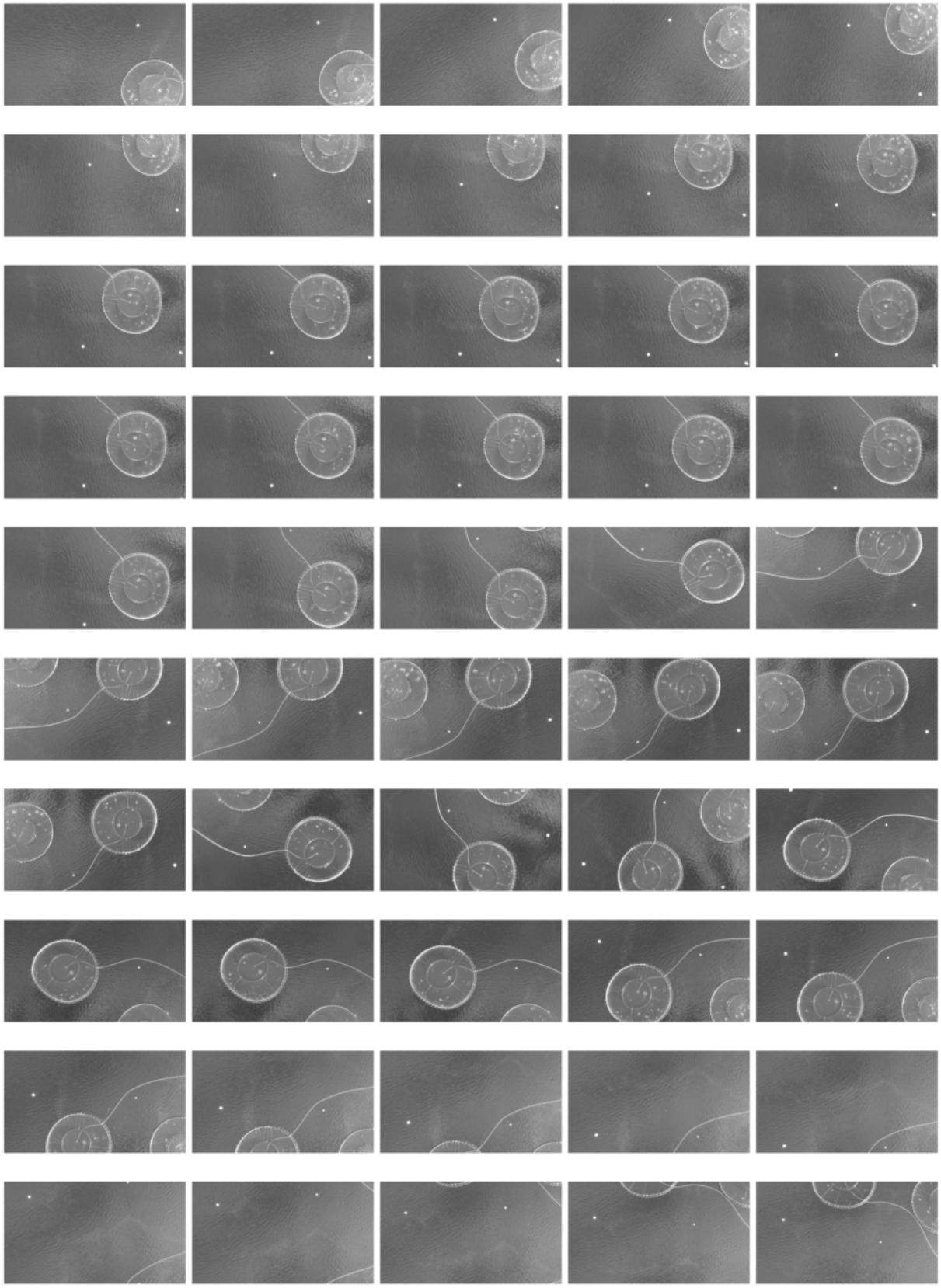


Figure 8.2: Images 50-100 graysacle originals

APPENDIX A.2

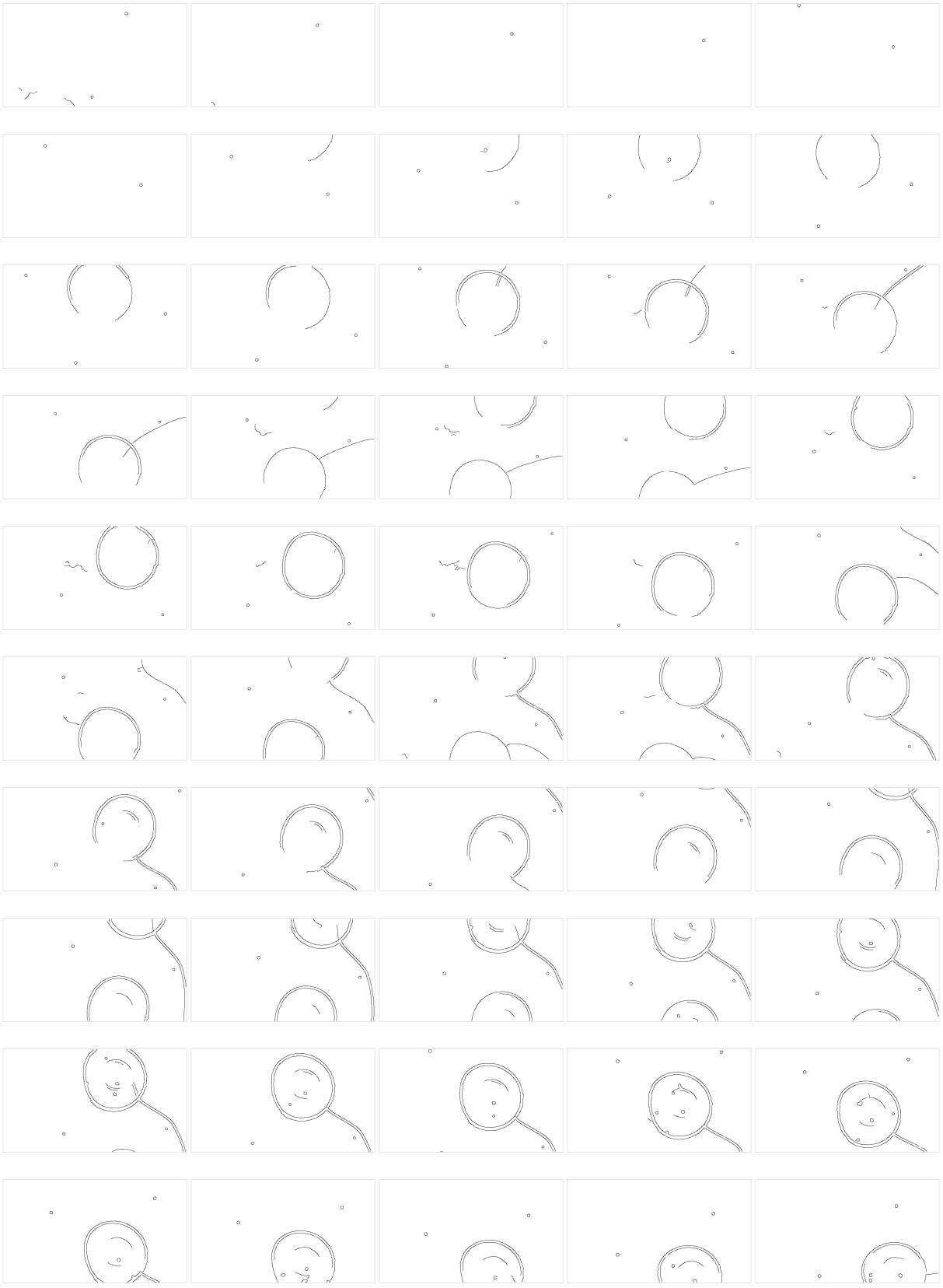


Figure 8.3: Images 1-50 canny edge filtered

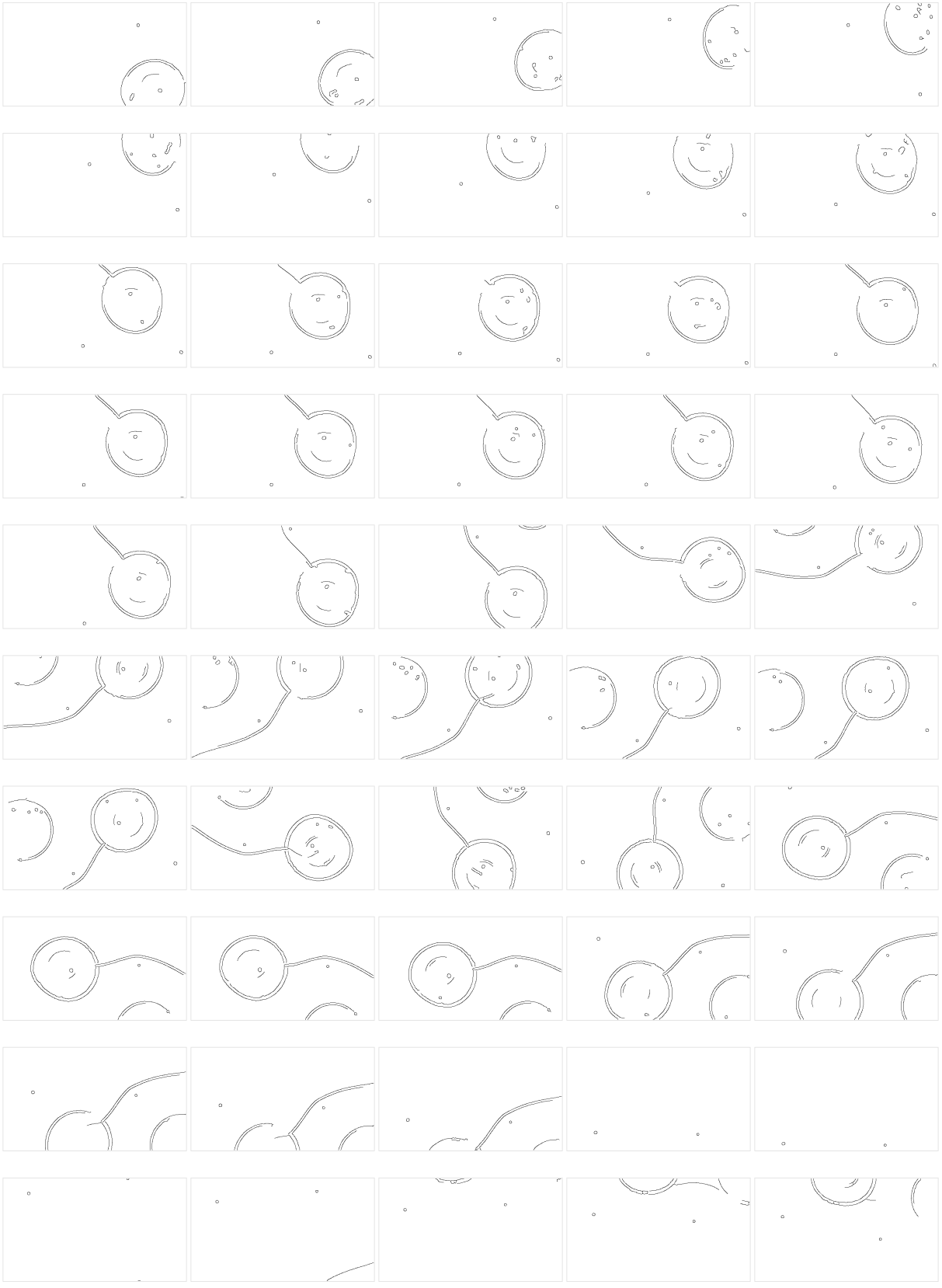


Figure 8.4: Images 50-100 canny edge filtered

APPENDIX B

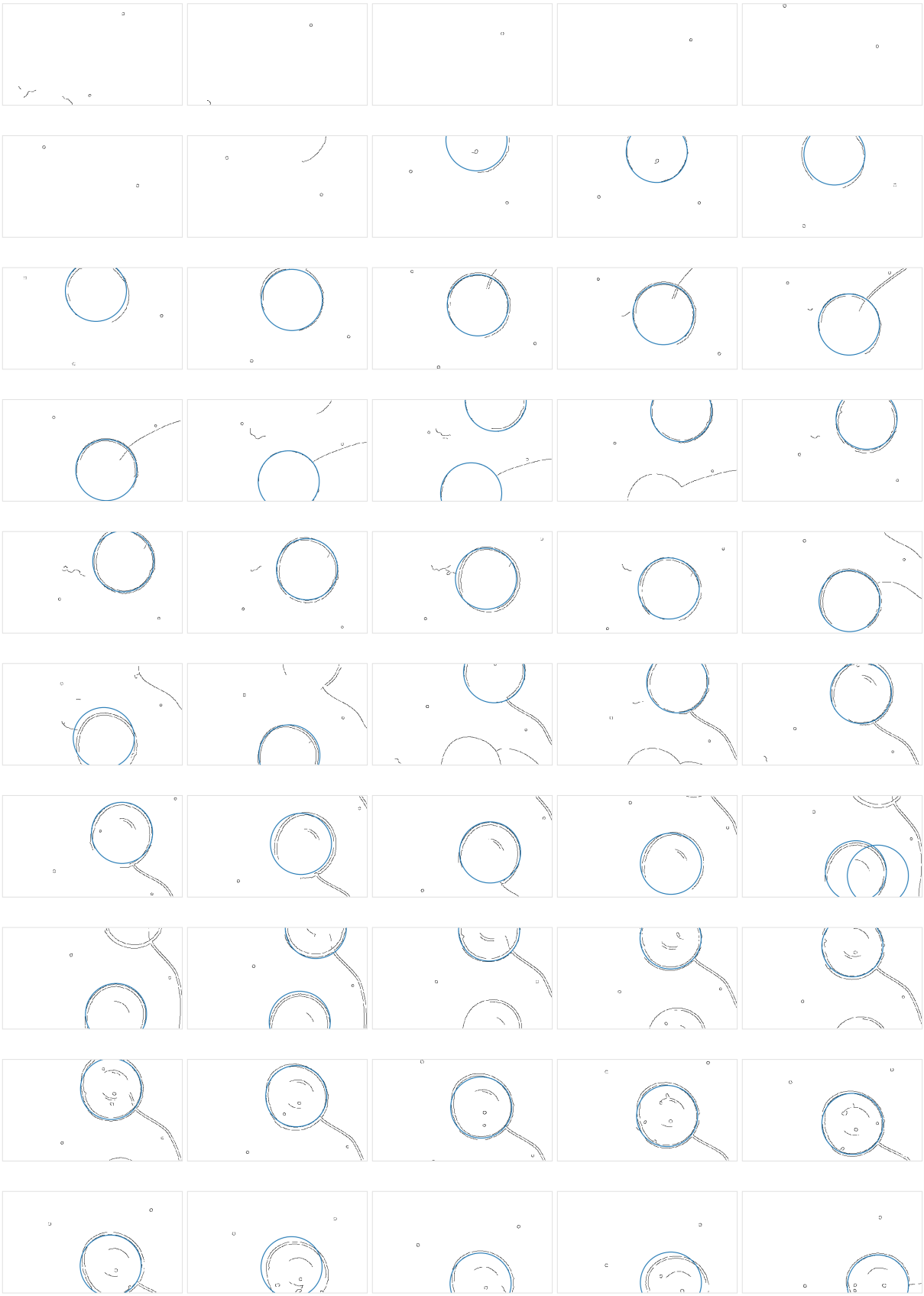


Figure 8.5: Images 1-50 circle detection | classical method

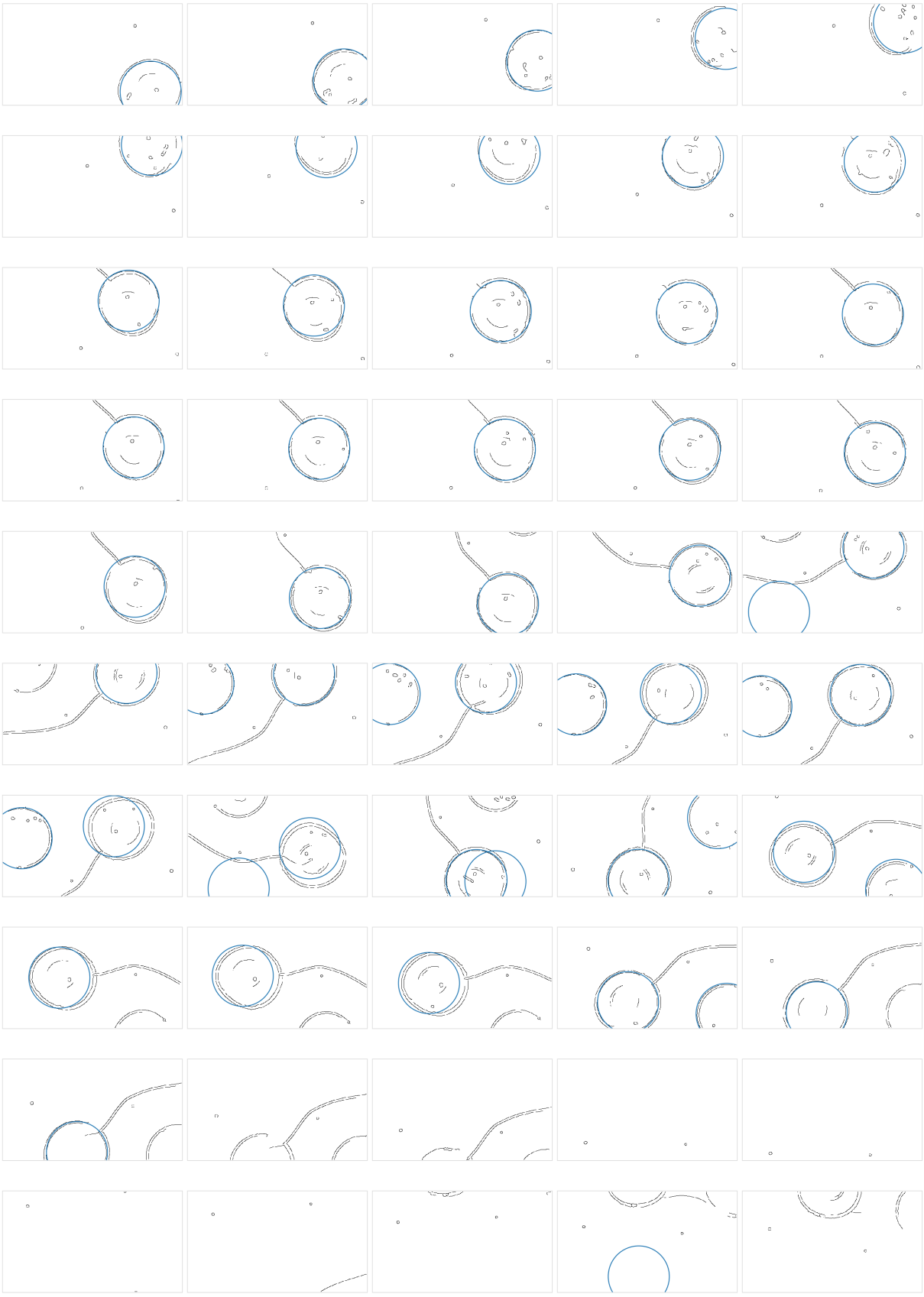


Figure 8.6: Images 50-100 circle detection | classical method

Figure 8.7: Images 1-50 circle detection | gradient method

Figure 8.8: Images 50-100 circle detection | gradient method

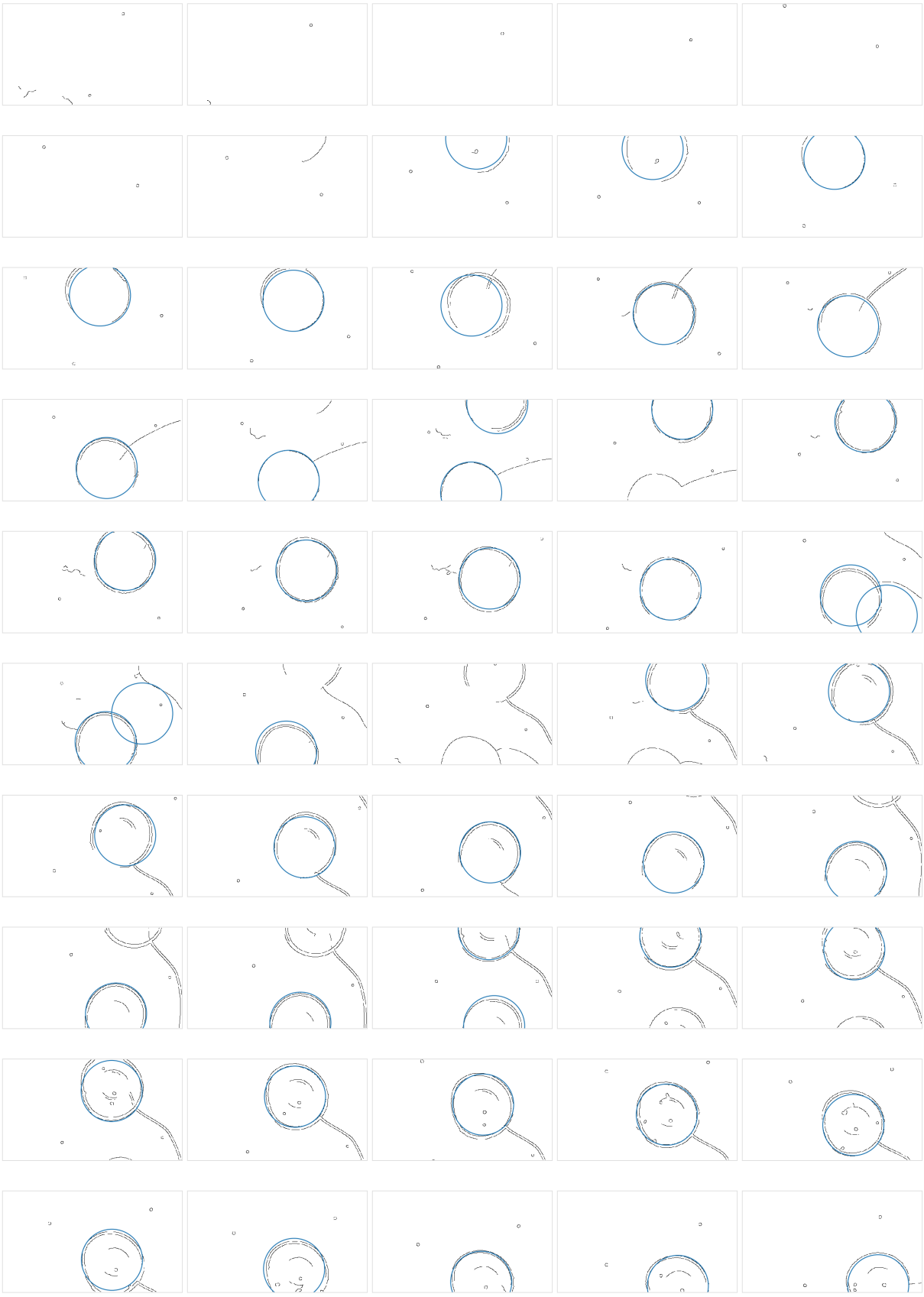


Figure 8.9: Images 1-50 circle detection | randomized method

Figure 8.10: Images 50-100 circle detection | randomized method

APPENDIX C

Figure 8.11: Images 1-50 buoy detection | filtering contours method

Figure 8.12: Images 50-100 buoy detection | filtering contours method

Figure 8.13: Images 1-50 buoy detection | DoH method

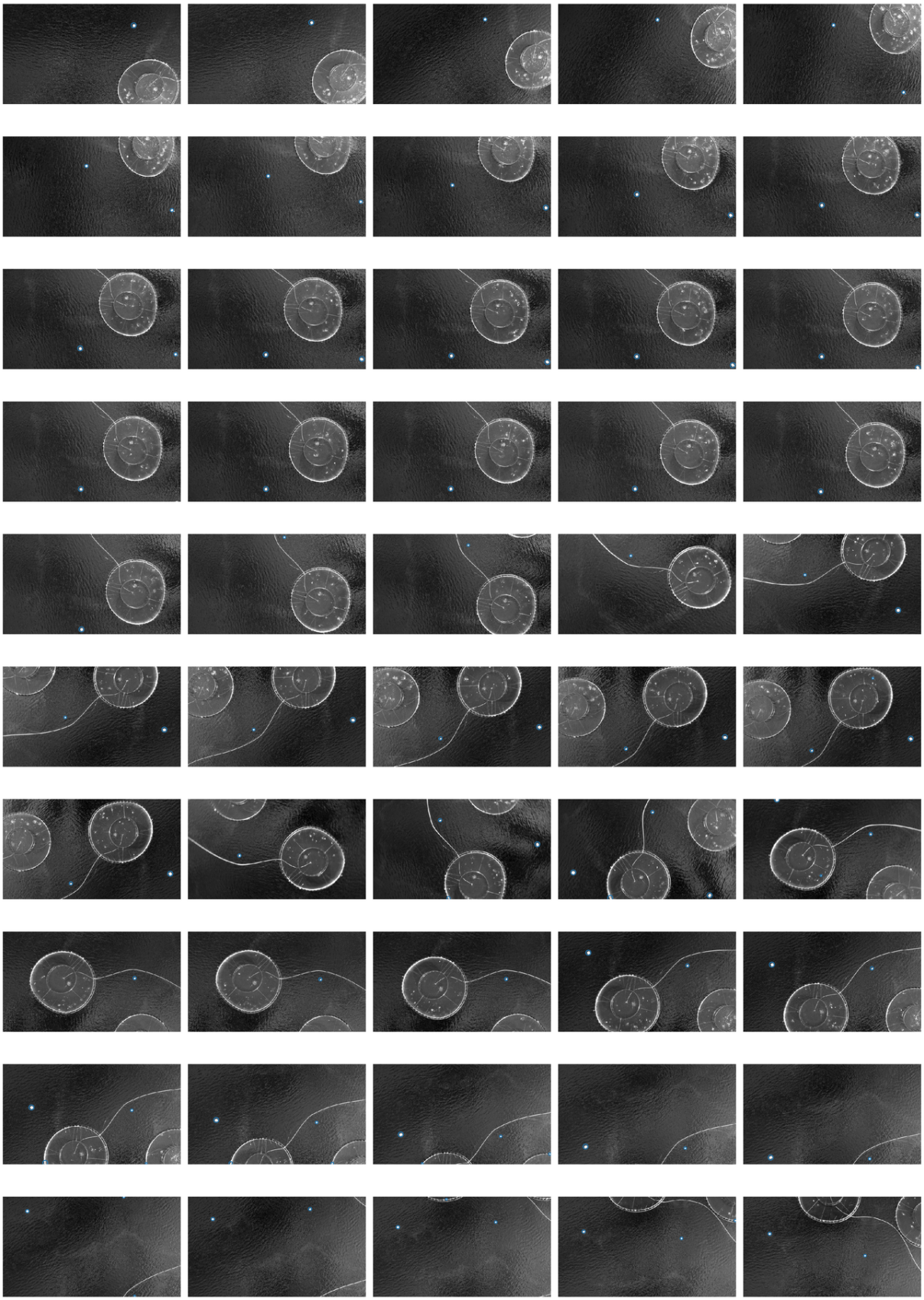


Figure 8.14: Images 50-100 buoy detection | DoH method

APPENDIX D - Cordinate systems

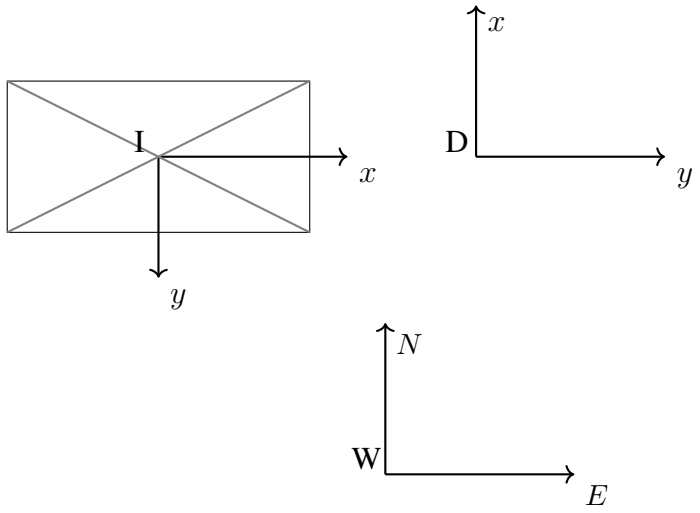


Figure 8.15: Birds view showing the different frames with respect to each other (z pointing down in all cases)

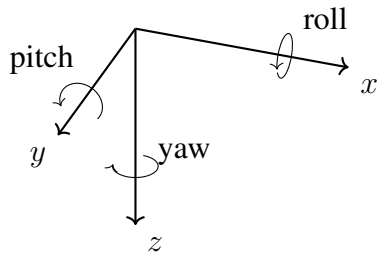
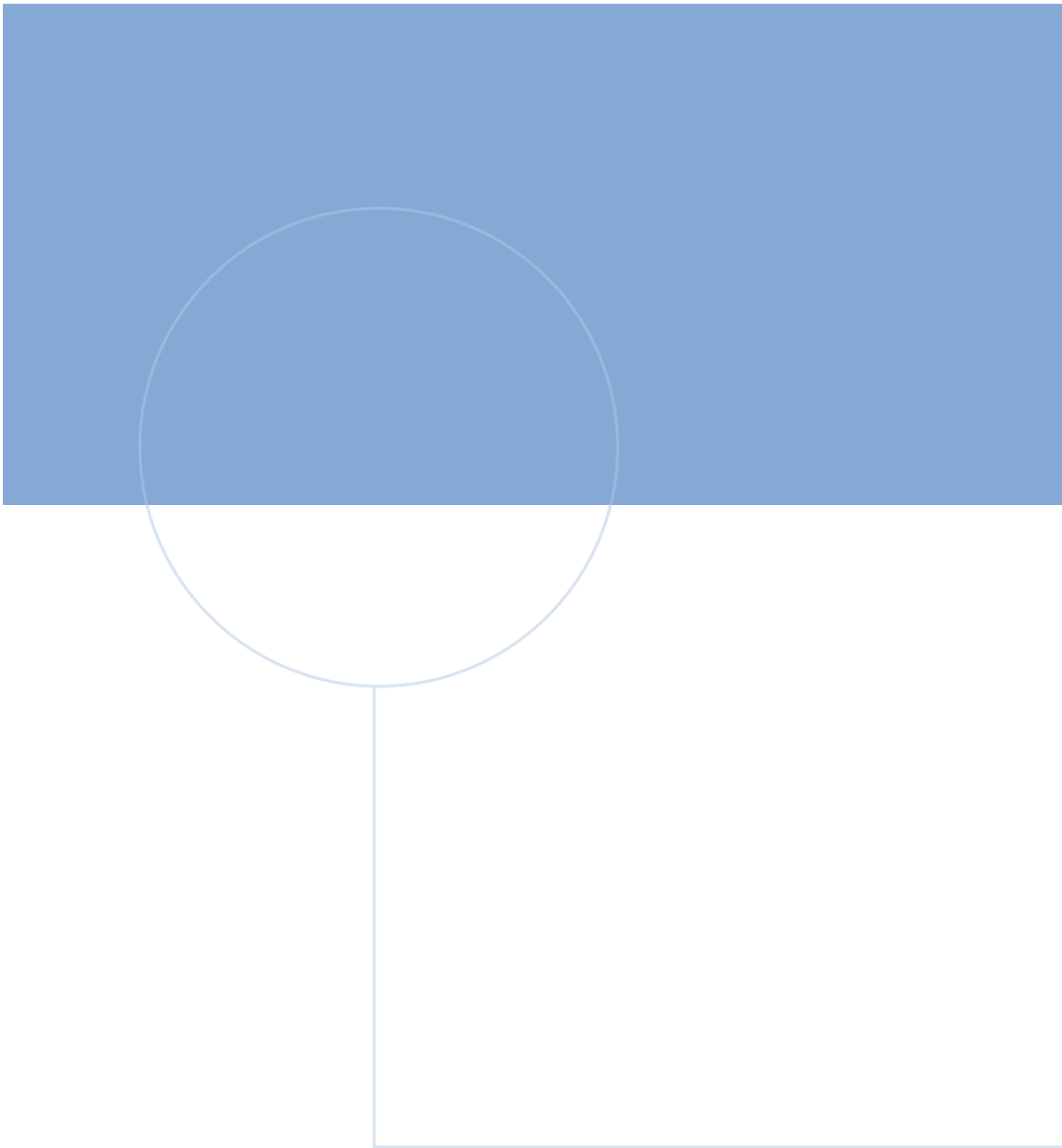


Figure 8.16: Motion of the UAV in 6 degrees of freedom



NTNU
Norwegian University of
Science and Technology