Master's thesis

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

Kristian Auestad

Depth Estimation and Object Detection using Stereo Vision for Autonomous Ferry

Master's thesis in Cybernetics and Robotics
Supervisor: Edmund Førland Brekke
Co-supervisor: Annette Stahl and Øystein Kaarstad Helgesen
May 2021

NTNU
Norwegian University of
Science and Technology

Kristian Auestad

Depth Estimation and Object Detection using Stereo Vision for Autonomous Ferry

Master’s thesis in Cybernetics and Robotics
Supervisor: Edmund Førland Brekke
Co-supervisor: Annette Stahl and Øystein Kaarstad Helgesen
May 2021

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

Abstract

In this project several methods for solving the correspondence problem and object detection in stereo vision are compared and evaluated. In regards to the correspondence problem, the local correspondence method *Sum of Absolute Differences* and the global method *Semi-Global Method* were tested on real data. *Stixel Tesselation* and *Euclidean Clustering* were tested in regards to object detection. This project was conducted as a part of the research project at NTNU called Autoferry, and provide improved knowledge on the use of stereo vision on the autonomous ferry, milliAmpere.

Autonomous vehicles that move in a space shared with other vehicles depend on accurate and reliable perception data. Detecting and estimating distance to objects far away is a difficult task. Lidar has been the "go-to" sensor for high frequency 3D-perception in short to medium ranges. However, these sensors can be expensive, have poor resolution and can be weather sensitive. Stereo cameras serve many of the same purposes with higher resolution, but often poorer accuracy.

To address this, I tested and evaluated different stereo vision algorithms on real data from milliAmpere. The results from these tests has been compared against each other as well as against lidar data to see which is best in terms of accuracy, detection and run time. The findings of this study show that stereo vision can provide valuable information to autonomous vehicles at greater distances than lidar, and that the accuracy is comparable to what the lidar provides. A stereo camera therefore seems like a good sensor for use on milliAmpere. However, the run time of the methods presented are not optimal, and future research should focus on reducing run time.

Sammendrag

I denne avhandlingen evaluerers flere methoder for løsing av korrespondanseproblemet og objektdeteksjon ved bruk av stereokamera. For løsing av korrespondanseproblemet ble den lokale metoden *Sum of Absolute Differences* og den globale metoden *Semi-Global Method* testet på virkelige data. Metoden *Stixel Tesselasjon* og *Euclidisk Gruppering* ble testet innen objektdeteksjon. Arbeidet ble gjort som en del av forskningsprojectet *Autoferry*, hvor dette prosjektet undersøker mulighetene for bruk av stereokamera på den autonome fergen, milliAmpere.

Autonome kjøretøy som beveger seg i et område delt med andre kjøretøy er avhengige av nøyaktig og robust oppfatning av omgivelsene. Deteksjon og estimering av distanse til objekter på lang avstand er utfordring å få til. Over lengre tid har lidar har vært den vanligste sensoren for logging av omgivelsedata i korte til medium lange avstander. Derimot, så kan lidar-sensorer være dyre, bli sterkt påvirket av værforhold og ha dårlig oppløsning på store avstander. Et alternativ til lidar er stereokamera. Et stereokamera dekker mange av de samme behovene som lidar, men med høyere oppløsning. Ulempen med stereokamera har tradisjonelt sett vært dårligere nøyaktighet og krevende med tanke på regnekraft.

I denne studien er de ulike metodene testet og evaluert gjennom eksperiementer gjort med den autonome fergen, milliAmpere i de faktisk omgivelsene milliAmpere skal operere i. Metodene har blitt sammenlignet med hverandre i tillegg til mot lidar-data. Funnene fra denne studien typer på at stereokamera kan tilby informasjon om omgivielsene på en større rekkevidde enn lidar, med en nøyaktighet som er sammenlignbar med lidar. Stereokamera kan derfor være en nyttig sensor ombord på milliAmpere. Derimot var kjøretiden til de forskjellige objektdeteksjons metodene ikke optimal, og videre arbeid bør fokusere på å redusere kjøretiden.

Preface

This thesis is the completion of a Master's programme in Cybernetics and Robotics at the Norwegian University of Science and Technology(NTNU). Throughout this project I have received valuable support and assistance, and I would like to express my gratitude to all of you who have contributed.

First and foremost, I would like to thank my supervisor, Edmund Førland Brekke for his valuable feedback and guidance. For sharing their knowledge with me, and providing technical input and valuable feedback on my work, I would like to thank my co-supervisors Annette Stahl and Øystein Kaarstad Helgesen. I also wish to thank my fellow students Martin Rånes Græsdal, Martin Eek Gerhardsen and Thomas Hellum. Your help during the experiments with milliAmpere was valuable and much appreciated.

Thank you, Egil Eide for organizing the use of milliAmpere and Håkon Hagen Helgesen for lending me the equipment for logging ground truth during the experiments. Glenn Angell at the ITK workshop at NTNU made two beautiful housings for the cameras, which ensured that the experiments could be conducted regardless of weather conditions.

Contents

Abstract . . . . **i**
Sammendrag . . . . **iii**
Preface . . . . **v**
Contents . . . . **vii**
Figures . . . . **ix**
Tables . . . . **xiii**
Symbols . . . . **xv**
1 Introduction **1**
1.1 Background 1
1.2 Thesis outline 2
2 Theory **3**
2.1 Camera as a Sensor 3
2.2 Epipolar Geometry 6
2.3 Stereo Calibration 7
2.4 Uncertainty in Stereo Vision Systems 11
3 The Correspondence Problem **19**
3.1 Disparity Map 20
3.2 Local Methods - Correlation Based 21
3.3 Local Methods - Feature Based 24
3.4 Global Methods 27
4 Evaluation of Matching Methods **31**
4.1 Local Methods - Feature Based 31
4.2 Local Methods - Correlation Based 32
4.3 Global Methods 32
4.4 SGM vs SAD 32
5 Object Detection using Stereo Vision **37**
5.1 The Ground Plane 38
5.2 Method 1 - Stixel Tesselation 38
5.3 Method 2 - Digital Elevation Map 42
5.4 Method 3 - Geometry-based Cluster 44
5.5 Method 4 - Direct Planar Hypothesis Testing 46
5.6 Method 5 - Euclidean Clustering 47
5.7 Evaluation of Detection Methods 49
6 System Description **51**
6.1 Hardware 51
6.2 Software 54
7 Experiments **59**
7.1 Ground truth 59
8 Results **65**

8.1 Calibration Results . . . 65
8.2 Distance Estimation . . . 68
8.3 Detection Rate . . . 76
8.4 Run time . . . 83
9 Discussion . . . 85
9.1 Stereo Calibration . . . 85
9.2 Stixel Tesselation . . . 85
9.3 Euclidean Cluster . . . 86
10 Conclusion and Future Work . . . 89
10.1 Conclusion . . . 89
10.2 Future Work . . . 90
Bibliography . . . 91
A Additional Material . . . 95
A.1 Maximum Likelihood Estimation - MLE . . . 95
A.2 Generalized Likelihood Ratio Test - GLRT . . . 95
B Scenario Overview . . . 97
C Point clouds - Euclidean Cluster - Stereo Vision . . . 101

Figures

2.1 Illustration of the principle behind the pinhole camera model 3
2.2 Relation between camera- and image coordinates 4
2.3 Illustration of coordinate systems 5
2.4 Epipolar plane spanned between camera origins and observed 3D point 6
11figure.caption.27
2.7 Baseline impact on overlapping field of view 12
2.8 Uncertainty regions with different camera angles 13
2.9 Uncertainty regions with different baselines 13
2.10 Example of how varying baseline affects estimation error. Fixed focal length (1230[px]) and disparity error (0.1[px]) 14
2.11 Illustration of reprojection error 15
2.12 Example of how reprojection error affects estimation error. Fixed focal length (1230[px]) and baseline (1.8[m]) 15
2.13 Parabola fit to disparity matches 16

3.2 Image frames before and after rectification 20
3.3 Illustration of patch sliding across scanline 21
3.4 Pyramid where each level has half the width and length as the previous level resulting in a quarter of the pixels 24
3.5 SIFT descriptor 25
3.6 SURF Box-Filtering 26
3.7 2D intra-scanline Source: [28] 27
3.8 Aggregation of cost 29

4.1 Keypoint detectors - 1 32
4.2 Keypoint detectors - 2 32
4.3 Comparison of disparity maps - 1 33
4.4 Comparison of disparity maps - 2 33
4.5 Input image - 1 34
4.6 Disparity map example - SAD - 1 34
4.7 Disparity map example - SGM 3WAY - 1 34
4.8 Disparity map example - SGM 8WAY - 1 35
4.9 Input image - 2 35
4.10 Disparity map example - SAD - 2 35
4.11 Disparity map example - SGM 3WAY - 2 35
4.12 Disparity map example - SGM 8WAY - 2 36

5.1 Variations of the occupancy grid 40
5.2 Illustation of freespace 41
5.3 Stixel distribution from scenario 7 42

5.4 Flow chart of DEM algorithm 43
5.5 Output of DEM algorithm 44
5.6 Double truncated cone 45
5.7 Segmentation process 3-D points classified as obstacle 46
5.8 Cones that visualize allowed deviations in local plane and obstacle 46

6.2 Point of intersection FoV 52
6.3 Sensors mounted on milliAmpere 53
6.4 Communication between units in the system 53
6.5 Flowchart for camera driver 55
6.6 Flowchart for *stereo_image_proc* 55
6.7 Image before and after resizing 56
6.8 Complete flow chart for stixel implementation 56
6.9 Complete flow chart for Euclidean cluster implementation 57
6.10 Tuning parameters in `rqt_reconfigure` 57

7.1 Havfruen and milliAmpere 59
7.3 Positional Error in Ground Truth 61
7.4 MilliAmpere during testing 62
62figure.caption.121
7.6 Flowchart of ROS-node for lidar 63

8.1 Overview of area covered during testing 65
8.2 Corners of checkerboard detected 66
8.3 Rectified image with horizontal epipolar lines 66
8.4 Checkerboard positions during calibration 66
8.5 Mean reprojection error for every image 67
8.6 Lower bound for depth error as a result of the calibration results 68
8.7 Illustration of scenario 1 68
8.8 Distance estimation for scenario 1 69
8.9 Distance error - Scenario 1 69
8.10 Illustration of scenario 2 70
8.11 Distance estimation for scenario 2 70
8.12 Distance error - Scenario 2 71
8.13 Illustration of scenario 3 71
8.14 Distance estimation for scenario 3 72
8.15 Distance error - Scenario 4 72
8.16 Illustration of scenario 4 73
8.17 Distance estimation for scenario 4 73
8.18 Distance error - Scenario 4 73
8.19 Illustration of scenario 5 74
8.20 Distance estimation for scenario 5 74
8.21 Distance error - Scenario 5 75
8.22 Detecton distribution for stixel method 76
8.23 Detecton distribution for Euclidean cluster method - Stereo Vision 77
8.24 Detecton distribution for Euclidean cluster method - Lidar 78
8.25 Examples of stixel detection in scenario 1 78
8.26 Examples of stixel detection in scenario 2 79
8.27 Examples of stixel detection in scenario 3 80
8.28 Examples of stixel detection in scenario 4 81
8.29 Examples of stixel detection in scenario 4 81

8.30 Examples of stixel detection in scenario 5 82

9.1 Rotation of left camera causes translation of the right camera 85
9.2 87

Tables

3.1 Explanation of elements in energy function . . . 29

4.1 Run times for SAD and SGM . . . 33
4.2 Summary of selected matching methods . . . 36

5.1 Summary of selected OD method . . . 50

6.1 Camera specifications . . . 52
6.2 Lens specifications . . . 52

7.1 Technical data for VPL 16 . . . 63

8.1 Extrinsic parameters from camera calibration . . . 67
8.2 Intrinsic parameters from camera calibration . . . 67
8.3 Duration of intervals without detection for different distances in total for all scenarios with the stixel method . . . 76
8.4 Duration of intervals without detection for different distances in total for all scenarios with the Euclidean cluster method - Stereo Vision . . . 77
8.5 Duration of intervals without detection for different distances in total for all scenarios with the Euclidean cluster method - Lidar . . . 77
8.6 Duration of intervals without detection for different distances for scenario 1, Stixel . 78
8.7 Duration of intervals without detection for different distances for scenario 1, Cluster stereo . . . 79
8.8 Duration of intervals without detection for different distances for scenario 1, Cluster lidar . . . 79
8.9 Duration of intervals without detection for different distances for scenario 2, Stixel . 80
8.10 Duration of intervals without detection for different distances for scenario 2, Cluster stereo . . . 80
8.11 Duration of intervals without detection for different distances for scenario 2, Cluster lidar . . . 80
8.12 Duration of intervals without detection for different distances for scenario 3, Stixel . 80
8.13 Duration of intervals without detection for different distances for scenario 3, Cluster stereo . . . 81
8.14 Duration of intervals without detection for different distances for scenario 3, Cluster lidar . . . 81
8.15 Duration of intervals without detection for different distances for scenario 4, Stixel . 81
8.16 Duration of intervals without detection for different distances for scenario 4, Cluster stereo . . . 82
8.17 Duration of intervals without detection for different distances for scenario 4, Cluster lidar . . . 82

8.18 Duration of intervals without detection for scenario 5, Stixel 82
8.19 Duration of intervals without detection for scenario 5, Cluster 82
8.20 Duration of intervals without detection for scenario 5, Cluster Lidar 82
8.21 Mean run times for each scenario, stixel 83
8.22 Mean run times for each scenario, stereo cluster 83
8.23 Mean run times for each scenario, lidar cluster 83

9.1 Mean run times for each scenario - Alternative filtering sequence - Stereo 87

Symbols

Coordinates

$\boldsymbol{X}_w$	World coordinates	$\boldsymbol{X}_c$	Camera coordinates
(x, y)	image coordinates	$\boldsymbol{p} = (u, v)$	Discretized image coordinates
$\sim$	Homogeneous coordinates	$\hat{}$	Estimate
d	disparity value	D	Disparity map

Camera Parameters

f_u	Horiztontal focal length [px]	u_0	Horiztontal principal point [px]
f_v	Vertical focal length [px]	v_0	Vertical principal point [px]
$\boldsymbol{B}$	Baseline	f	focal length [m]
$\boldsymbol{K}$	Intrinsic Matrix	$\boldsymbol{P}$	Projection Matrix
f_s	Skew parameter	λ	Scaling factor

Object Detection

v_B	Stixel base point	v_T	Stixel top point
S	Error funciton	θ	Maximum likelihood
$\mathcal{H}_f$	Null hypothesis	$\mathcal{H}_o$	Alternative hypothesis
$\boldsymbol{\mathcal{I}}$	Data vector	γ	Decision threshold
$\boldsymbol{\alpha}(\boldsymbol{p})$	Intensity bias at $\boldsymbol{p}$	$\boldsymbol{\eta}(\boldsymbol{p})$	Noise at $\boldsymbol{p}$
$\boldsymbol{G}$	Multivariate Gaussian	$\boldsymbol{\xi}$	Error
$\boldsymbol{\Gamma}$	Covariance matrix		

Misc.

$E_x(.)$	Energy function	$\boldsymbol{m}$	Stereo measurement
$\mathcal{E}$	Edge set	V	Vertex set
G	Graph	$\boldsymbol{A}$	Transformation matrix
σ	Scale	$\mathcal{L}$	Likelihood function
L	Scale space	Θ	Horizontal field of view
$\boldsymbol{e}$	Epipole	$\boldsymbol{F}$	Fundamental matrix
$\boldsymbol{E}$	Essential matrix	WD	Working distance
β	Inward rotation of camera	$\boldsymbol{H}$	Homography matix

Chapter 1

Introduction

1.1 Background

A crucial part of any autonomous vehicle is an accurate and reliable perception system. For an autonomous system to move safely, it depends on systems providing information about its surroundings, that enable the autonomous system to make correct decisions regarding its movement to avoid collisions. These systems also has to be able to detect objects of various size and range to plan trajectories early, and behave predictable for other vehicles in its vicinity. As robots are now taking their first steps out of factory floors and into the real world, these systems are becoming more important than ever.

In order for a robot to sense the surroundings it needs to be equipped with various sensors. Radar has the ability to detect objects at large distances, but its slow update frequency and poor accuracy makes it less ideal for close encounters [1]. Lidar is accurate in perceiving depth at closer distances, but it has poor resolution at long ranges and can be heavily affected by weather conditions, especially rain. Stereo cameras have a denser resolution than lidar, but traditionally has lacked the same depth accuracy. However, in recent years stereo vision has grown in popularity due to new and improved algorithms and its affordability compared to the lidar. This thesis explores opportunities to implement stereo vision on a robot in the maritime environment.

An Unmanned Surface Vehicle (USV) is a vehicle operating on the sea-surface without on-board personnel. A sub-category of these vehicles is the Autonomous Surface Vehicle (ASV), which is a vehicle able to operate without human interaction. There has been conducted some research on ASVs and their applications [2] in recent years, but as autonomy is becoming more and more common in the automotive industry, it is still a scarce feature in the maritime sector. Nonetheless, a suitable area for ASVs can be in human transport at sea, namely ferries. According to [3] human error was a factor in more than 70% of the accidents in the study. This makes room for a large leap in safety if the human error can be eliminated with the use of autonomy.

This thesis is a part of the autoferry research project at NTNU. The main goal of the research project is to "*develop groundbreaking new concepts and methods which will enable the development of autonomous passenger ferries for transport of people in urban water channels*" [4]. The ferry, milliAmpere is going to transport passengers between Ravnkloa and Vestre kanalhavn in the Trondheim canal. The passengers shall be able to call the ferry by the push of a button, and the rest is to be fully autonomous.

Earlier work done regarding stereo vision on the autoferry project includes Lina Theimann and Trine Ødegårds master thesis [5] where they used a correlation based local method to obtain a disparity map and a Convolutional Neural Network (CNN) for object detection. A specialization project done by the same authors compared different local methods for solving the correspondence problem [6] In my specialization project in TTK4551 during the autumn of 2020 [7] I tried to take this further by comparing their finds with the more recent method called the Semi-Global Method (SGM) [8]. This project showed that SGM could operate in real time and resulted in denser point cloud than local methods, but at roughly the same accuracy regarding depth.

The aim of this thesis is to evaluate different stereo vision systems regarding object detection and depth estimation in a way that contribute to a better understanding of how the different methods work and what their strengths and weaknesses are. In addition to this the most suitable methods will be chosen for use on milliAmpere. As there are many different methods and algorithms to choose for the different stages in the process, this thesis tries to combine the finds by Olsen and Theimann in [6] and my specialization project [7], and present a more complete system. This is done by testing out systems in the working environment to help determine the performance of the system in real life scenarios. The thesis addresses the following tasks:

- General design of a stereo vision system.
- Evaluation of matching methods.
- Evaluation of object detection methods.
- Implementation of a stereo vision system on milliAmpere.
- Evaluation of performance.

1.2 Thesis outline

The thesis is structured such that in chapter 2 relevant background theory will be presented. Then the backbone of stereo vision, namely solving the correspondence problem will be presented in chapter 3, and evaluated in chapter 4. In chapter 5 different methods for generic object detection based on stereo vision will be explained, followed by an evaluation of the methods with regards to their use in this thesis. Chapter 6 explains the stereo system used in this thesis and the design choices that were made. In chapter 7 and chapter 8 the experiments conducted and the following results, respectively. Lastly a thorough discussion and conclusion is made in chapter 9 and chapter 10.

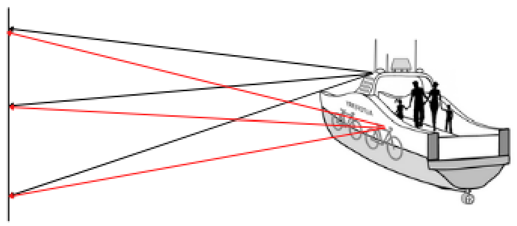
Chapter 2

Theory

In this chapter relevant theory for understanding a stereo vision system will be presented. How to use a monocular camera as a sensor is fundamental and it will therefore be explained first. Epipolar geometry is key in understanding why stereo vision systems differs from a setup with two unrelated cameras, and has therefore a central part in this chapter. Finally, the theory behind stereo camera calibration and accuracy in stereo vision systems will be described.

2.1 Camera as a Sensor

In general the camera conducts a mapping of the 3D world onto a 2D image [9, p. 153]. There is a range of different methods for this type of mapping, but the simplest and most popular is the pinhole camera model. The pinhole model is based on the observation that when a plane with a small aperture is placed in front of a scene, the scene will be projected as an reversed image of the reality on the other side of the plane. This phenomena can be observed in a dark room with a keyhole as the aperture. An illustration of the can be seen in fig.2.1b.


(a) No barrier causes blurred image

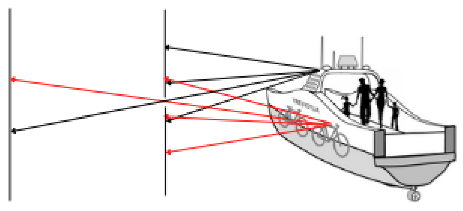

(b) Barrier with aperture

Figure 2.1: Illustration of the principle behind the pinhole camera model

In practice this is done by defining a plane with $Z_c = f$, parallel to the camera coordinate system (where f is the focal length of the camera, fig. 2.2). Points in the 3D world are observed by the camera as a line between origin of the camera coordinate system and the 3D point. This line will intersect the image plane, and the relationship between image- and world points can be described by the means of similar triangles (fig. 2.2).

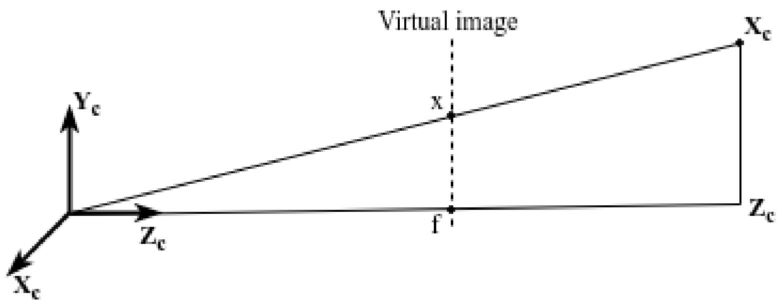


Figure 2.2: Relation between camera- and image coordinates

$$(X_c, Y_c, Z_c)^T \rightarrow \left(f\frac{x}{z}, f\frac{y}{z}, f\right)^T \tag{2.1}$$

Homogeneous Coordinates

Homogeneous coordinates is a coordinate system for projective spaces. They allow us to express the relation between 2D and 3D coordinates very elegantly by simplifying mathematical derivations [9, p. 27]. For instance can several transformations such as translation, rotation, scale, affine, etc. be expressed by matrices. Transformation between Euclidean and homogeneous space can be done as described in eq. 2.2 and eq. 2.3 where $\boldsymbol{p}$ is Euclidean and $\tilde{\boldsymbol{p}}$ is homogeneous.

$$\boldsymbol{p} = \begin{bmatrix} u \\ v \end{bmatrix} \longrightarrow \tilde{\boldsymbol{p}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{2.2}$$

$$\tilde{\boldsymbol{p}} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \longrightarrow \boldsymbol{p} = \begin{bmatrix} u/w \\ v/w \end{bmatrix} \tag{2.3}$$

Homogeneous coordinates are only defined up to scale.

$$\tilde{\boldsymbol{p}} = \lambda \boldsymbol{p}, \qquad \lambda \neq 0 \tag{2.4}$$

Another important property is that points infinitely far away in the 3D world can be expressed with finite coordinates.

$$\tilde{\boldsymbol{p}}_{infinite} = \begin{bmatrix} u \\ v \\ 0 \end{bmatrix} \tag{2.5}$$

By utilizing these properties a series of complex transformations can be described by a single transformation matrix, created by a series of matrix multiplications.

$$\tilde{\boldsymbol{p}}' = \boldsymbol{A}\tilde{\boldsymbol{p}}, \qquad \text{where } \mathbf{A} \text{ is a transformation matrix} \tag{2.6}$$

From World to Image

To transform the data given by the camera system into usable information describing the real world a transformation between coordinate systems is required. In a camera system there is often four different coordinate systems being used:

1. $\boldsymbol{X}_w$ - The world coordinate system describing 3D points with origin in an arbitrary location.
2. $\boldsymbol{X}_c$ - The camera coordinate system describing 3D points with origin in the camera center.
3. (x, y) - The image plane coordinate system describing 2D points on the image plane with origin in the center of the image plane.
4. $\boldsymbol{p} = (u, v)$ - The discretized image coordinate system describing 2D points in pixel units with origin in the upper left corner of the image.

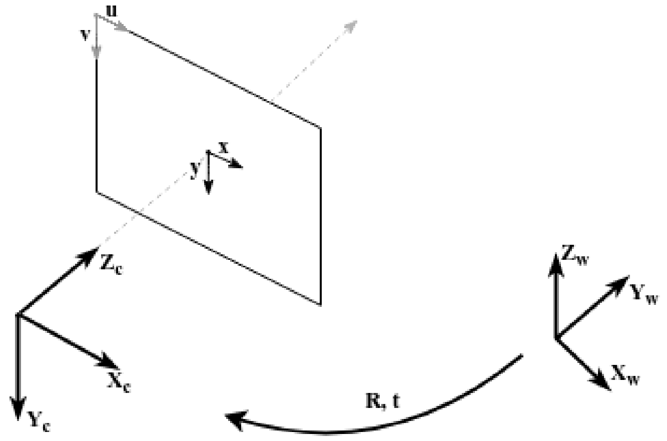


Figure 2.3: Illustration of coordinate systems

In most cases it is the transformation from world coordinates ($\boldsymbol{X}_w$) to discretized image coordinates, $\boldsymbol{p}$, that is desirable, while the camera coordinate system and image coordinate system are just intermediate steps required to complete the transformation. The first step in this process is to transform a world point, $\boldsymbol{X}_w$ into a camera point, $\boldsymbol{X}_c$. This is done by a rigid body transformation. In practice this means rotating and translating $\boldsymbol{X}_w$ to coincide with $\boldsymbol{X}_c$.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.7}$$

After transforming $\boldsymbol{X}_w$ to $\boldsymbol{X}_c$ the point can be projected to the image plane. As the image plane has coinciding X -and Y-axis with $\boldsymbol{X}_c$ and is only shifted along the Z-axis with the same value as the focal length (f) this projection can be described as in eq. 2.8, where Z is a scaling factor also represented as $Z = \frac{1}{\lambda}$.

$$Z\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{2.8}$$

The transformation from image coordinates to discretized image coordinates is done by multiplying the coordinates with the cameras intrinsic matrix ($\boldsymbol{K}$) which transforms the point into pixel units and shifts the origin to the top left corner. This matrix is obtained by camera calibration and is described in detail in section 2.3. The complete transformation can be seen in eq.2.9 where $\boldsymbol{\Pi}$ is known as the projective matrix.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_u & f_s & u_0 \\ 0 & f_v & v_o \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{K}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\boldsymbol{\Pi}} \underbrace{\begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ 0 & 1 \end{bmatrix}}_{\boldsymbol{R}|\boldsymbol{T}} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{2.9}$$

$$\lambda \tilde{\boldsymbol{p}} = \boldsymbol{K}\boldsymbol{\Pi}[\boldsymbol{R}|\boldsymbol{T}]\tilde{X}_w \tag{2.10}$$

2.2 Epipolar Geometry

Epipolar geometry is the geometry between two views [9]. It describes the relative pose between the cameras regardless of what scene is observed. Knowing this relationship between the cameras is essential in stereo vision systems as it is this knowledge that allows for depth estimation in a single stereo image. If a point $\boldsymbol{X_w}$ is viewed from two different views $\boldsymbol{p}_l$ and $\boldsymbol{p}_r$ an *epipolar plane* $\boldsymbol{\pi}$ is the plane spanned between these three points. The straight line between the origin of the camera coordinate systems is the *baseline*.

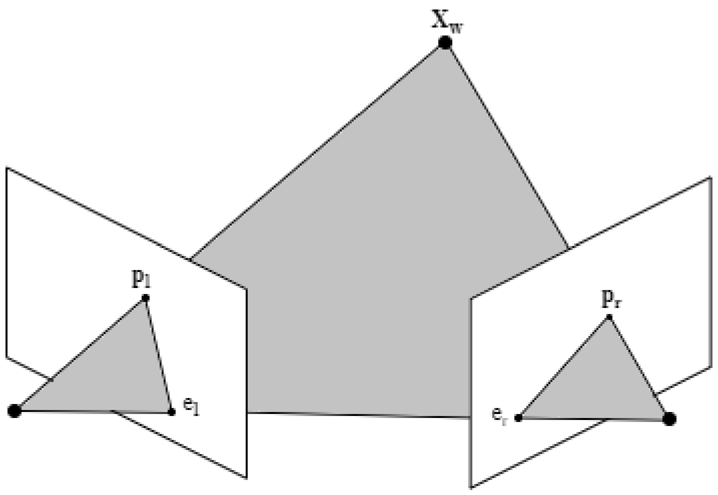
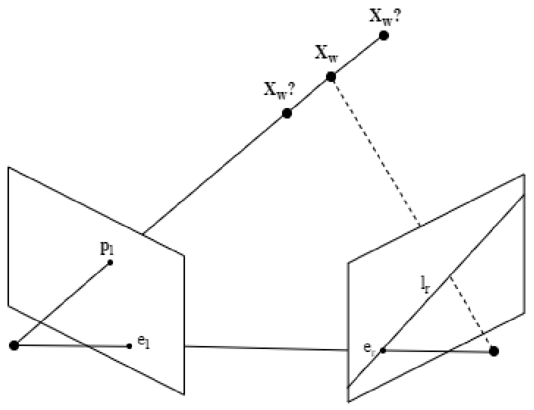
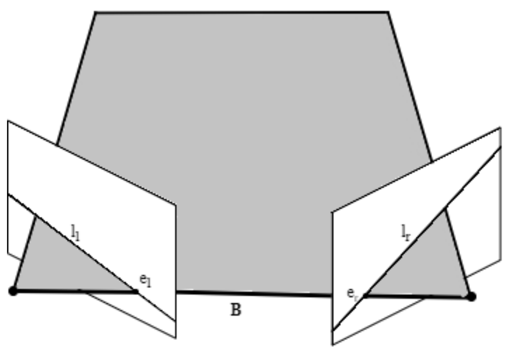


Figure 2.4: Epipolar plane spanned between camera origins and observed 3D point
Source: Based on fig. 9.1 in [9]

The point of intersection between the baseline and the image plane is known as the *epipole*. Perhaps the most important feature in epipolar geometry is the *epipolar line*. This is line is the intersection of the epipolar plane and the image plane. In other words; if $\boldsymbol{X_w}$ is observed in the left image view, the same point has to be located along the epipolar line in the right image view, and vice versa.



(a) Epipolar line
Source: Based on fig. 9.1 in [9]



(b) Epipolar plane
Source: Based on fig. 9.2 in [9]

The algebraic representation of epipolar geometry is known as the *Fundamental Matrix* [9]. This matrix has some useful properties listed below.

$$p_r^T \boldsymbol{F} p_l = 0 \quad \text{for corresponding points } p_r, p_l \tag{2.11}$$

$$l_r = \boldsymbol{F} p_l, \quad l_l = \boldsymbol{F^T} p_r \quad \text{where l is an epipolar line} \tag{2.12}$$

$$\boldsymbol{F} e_l = 0, \quad \boldsymbol{F^T} e_r = 0 \quad \text{where e is an epipole} \tag{2.13}$$

The *essential matrix* is a specialised version of the fundamental matrix [9, p. 257] that can be obtained if the camera intrinsics, $\boldsymbol{K}$, is known.

$$\hat{\boldsymbol{p}}_l = \boldsymbol{K}^{-1} \boldsymbol{p}_l \tag{2.14}$$

$$= [\boldsymbol{R}|\boldsymbol{t}]\boldsymbol{X} \tag{2.15}$$

In 2.2 is now expressed in normalized coordinates

$$\boldsymbol{E} = [\boldsymbol{t}]_x \boldsymbol{R} = \boldsymbol{R}[\boldsymbol{R^T} \boldsymbol{t}]_x \tag{2.16}$$

$$\hat{p}_r^T \boldsymbol{E} \hat{p}_l = 0 \tag{2.17}$$

$$\boldsymbol{E} = \boldsymbol{K}_r^T \boldsymbol{F} \boldsymbol{K}_l \tag{2.18}$$

2.3 Stereo Calibration

In this section the theory and process of stereo camera calibration will be presented. A proper calibration is needed to minimize the uncertaintly in the stereo measurements of a stereo system. As the stereo calibration consists of estimating two different sets of parameters, namely the intrinsic -and the relative extrinsic parameters they will be presented accordingly.

Calibration Introduction

Many of the available tools such as Matlabs Camera Calibration App, and OpenCV's calibration functions are based on the same method, namely *Zhangs Method*. This is a method first presented by Zhang in 2000 [10]. In this method a set of pictures of planar object with a known geometrical pattern is captured. The most common pattern is a checkerboard pattern, where all the squares have known dimensions. Either the camera or the checkerboard is then locked in place while the other is changing position, and images are captured of the pattern from various distances and angles. A world coordinate system with origin in the upper left corner of the checkerboard with the axis X, Y and Z pointing right, down and inward, respectively. By doing this all the corners in the checkerboard has known world coordinates. The same corners can easily be detected in an image and given a image coordinate. As origin of the world coordinate system is on the checkerboard Z=0 for all the points on the plane, and the relationship between image points and world coordinates can be described as in eq. 2.19.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} \tag{2.19}$$

Rewriting $\boldsymbol{R}$ in eq. 2.19 using r_i to represent every column of the matrix to define a new matrix, $\boldsymbol{H} \in \mathcal{R}^{3\times3}$ known as the *Homography matrix*. 2D homography is a linear transformation of 2D points from one plane to another [9, p.87]. The matrix **H** is defined up to a scale factor (λ) and has eight degrees of freedom [11].

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{K} \begin{bmatrix} \boldsymbol{r_1} & \boldsymbol{r_2} & \boldsymbol{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \boldsymbol{H} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \tag{2.20}$$

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} \tag{2.21}$$

The homogeneous set of equations in eq. 2.21 can be turned into inhomogenous equations by dividing by the scalar term ($\lambda = h_{31}X + h_{32}Y + h_{33}$).

$$u = \frac{h_{11}X_w + h_{12}Y_w + h_{13}}{h_{31}X_w + h_{32}Y_w + h_{33}} \tag{2.22}$$

$$v = \frac{h_{21}X_w + h_{22}Y_w + h_{23}}{h_{31}X_w + h_{32}Y_w + h_{33}} \tag{2.23}$$

Rewriting eq. 2.22 and eq. 2.23 into vector form we get:

$$\boldsymbol{a}_u^T \boldsymbol{h} = 0 \tag{2.24}$$

$$\boldsymbol{a}_v^T \boldsymbol{h} = 0 \tag{2.25}$$

Combining this set of equations for every image pair and assembling them into one matrix. $\boldsymbol{A}$, we get

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_{u1}^T \\ \boldsymbol{a}_{v1}^T \\ \vdots \\ \boldsymbol{a}_{un}^T \\ \boldsymbol{a}_{vn}^T \end{bmatrix}, \qquad 4 \leq n \tag{2.26}$$

$$\boldsymbol{A} = \begin{bmatrix} -X_w & -Y_w & -1 & 0 & 0 & 0 & u_1X_w & u_1Y_w & u_1 \\ 0 & 0 & 0 & -X_w & -Y_w & -1 & v_1X_w & v_1Y_w & v_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -X_w & -Y_w & -1 & 0 & 0 & 0 & u_nX_w & u_nY_w & u_n \\ 0 & 0 & 0 & -X_w & -Y_w & -1 & v_nX_w & v_nY_w & v_n \end{bmatrix}, \qquad 4 \leq n \tag{2.27}$$

$$\boldsymbol{A}\boldsymbol{h} = \boldsymbol{0} \tag{2.28}$$

As $\boldsymbol{H}$ has eight degrees of freedom at least four correspondences is necessary to solve the set of linear equations. Using Singular Value Decomposition (SVD) the values of $\boldsymbol{H}$ can be extracted from the last column of $\boldsymbol{V}$ in eq. 2.29.

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{2.29}$$

Intrinsic Parameters

From eq. 2.20 we know that $\boldsymbol{H}$ contains the information about both the intrinsics ($\boldsymbol{K}$) and the extrinsics ($\boldsymbol{R}, \boldsymbol{t}$), this can also be seen in eq. 2.30. The next step is therefore to extract $\boldsymbol{K}$ from $\boldsymbol{H}$. Only the resulting equations will be presented here. For details the reader is reffered to the original paper by Zhang [10].

$$\boldsymbol{H} = \frac{\boldsymbol{K}}{\lambda_H}\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \tag{2.30}$$

A new matrix $\boldsymbol{B}$ is defined up to a scale factor λ_B.

$$\boldsymbol{B} = \lambda_B(\boldsymbol{K}\boldsymbol{K}^T)^{-1} \tag{2.31}$$

$$v_0 = \frac{B_{12}B_{13} - B_{11}B_{13}}{B_{11}B_{22} - B_{12}^2} \tag{2.32}$$

$$\lambda_B = B_{33} - \frac{B_{12}^2 - v_0(B_{12}B_{13} - B_{11}B_{23})}{B_{11}} \tag{2.33}$$

$$f_u = \sqrt{\frac{\lambda_B}{B_{11}}} \tag{2.34}$$

$$f_v = \sqrt{\frac{\lambda_B B_{11}}{B_{11}B_{22} - B_{12}^2}} \tag{2.35}$$

$$f_\Theta = \frac{B_{12}\alpha_x^2\alpha_y}{\lambda_B} \tag{2.36}$$

$$u_0 = \frac{sy_0}{\alpha_y} - \frac{B_{13}\alpha_x^2}{\lambda_B} \tag{2.37}$$

Extrinsic Parameters

As the intrinsic matrix, $\boldsymbol{K}$, is constant for all the homographies in the calibration this can be used to extract the extrinsic components $\boldsymbol{R}$ and $\boldsymbol{t}$, which is the rotation and translation between the two cameras. The first step is to rewrite eq. 2.31:

$$\lambda \boldsymbol{K}^{-1}\begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \tag{2.38}$$

From eq. 2.38 we get three equations:

$$r_1 = \lambda \boldsymbol{K}^{-1} h_1 \tag{2.39}$$

$$r_2 = \lambda \boldsymbol{K}^{-1} h_2 \tag{2.40}$$

$$r_3 = r_1 \times r_2 \tag{2.41}$$

$$t = \lambda \boldsymbol{K}^{-1} h_3 \tag{2.42}$$

$$\tag{2.43}$$

where

$$\lambda = \frac{1}{||\boldsymbol{K}^{-1}h_1||} = \frac{1}{||\boldsymbol{K}^{-1}h_2||} \tag{2.44}$$

Resulting in the extrinsic matrix being constructed like:

$$\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \tag{2.45}$$

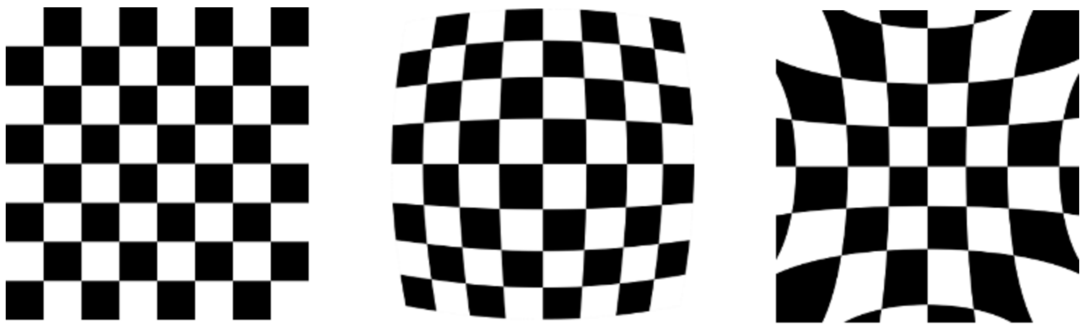
However, due to noise in the data this procedure results in with a $\boldsymbol{R}$ that does not satisfy the properties of a rotation matrix [10]. To solve the reprojection error can be minimized using a solver for nonlinear optimization problems, such as the Levenberg-Marquardt algorithm. More details about this process can be found in Zhangs original paper [10].

Distortion Coefficient Estimation

Lens distortion is when the shape of the lens or inaccuracies in the placement of the lens cause errors in the image. These errors cause deformation of objects making the image not a real representation of the real world[12, p. 58]. These deformation makes pixels in the image appear at the incorrect place to what the pinhole camera model expects and therefore needs to be corrected.

The most prominent distortion is known as the radial distortion and cause straight lines to appear as curved in the image. The distortion can be expressed as a non linear function. In eq.2.46 x_d and y_d are the distorted image coordinates, r is the distance from the principal point (u_0, v_0) to x, y, and k_n for $n \in \mathcal{N}$ are constants for each specific case.

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \begin{bmatrix} x - u_0 \\ y - v_0 \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \tag{2.46}$$



(a) No radial distortion **(b)** Positive radial distortion **(c)** Negative radial distortion

Figure 2.6: Radial distortion.
Source: OpenCV documentation[1]

The second form of distortion is the tangential distortion. This has a much smaller impact on the image than the radial distortion, and is often ignored all together. This distortion happens when the lens and the photo sensor are not completely parallel. This results in a compressing effect on the image [13, p. 686].

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} 2k_4 (x - u_0)(y - v_0) + k_5 \left(r^2 + 2(x - u_0)^2\right) \\ k_4 \left(r^2 + 2(y - v_0)^2 + 2k_5 (x - u_0)(y - v_0)\right. \end{bmatrix} \tag{2.47}$$

2.4 Uncertainty in Stereo Vision Systems

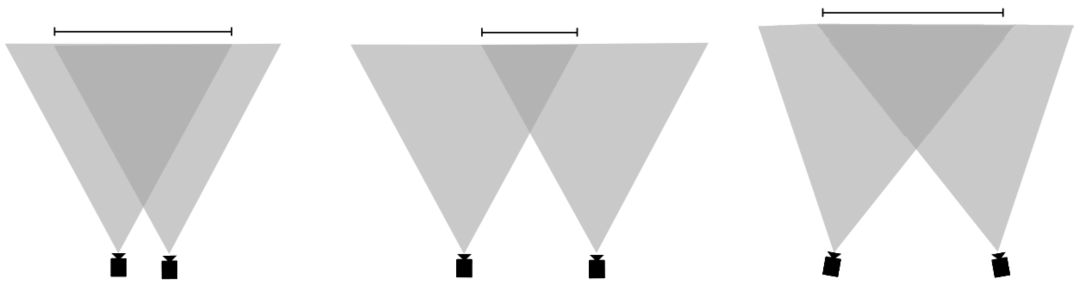
As the main drawback of using stereo vision compared to lidar, has been poor depth estimation it is important to evaluate the accuracy in the estimations. The accuracy of depth estimation in stereo vision systems depends on the geometric structure of the system [14]:

1. Interpixel distance, focal length, baseline, camera angle
2. The accuracy of these parameters

The first point show that in order to achieve the best possible results the stereo vision setup regarding cameras, lenses and stereo parameters has to be suitable for the desired operating range. The accuracy of these parameters is ensured by a calibration of the stereo camera, which is done to estimate intrinsic- and extrinsic parameters. The error in depth is described in eq. 2.48 which is derived in [13]. The depth error ΔZ depends on the depth Z, the mean of the reprojection error Δp_x, focal length f_u and baseline B.

$$\Delta Z_c = \frac{Z_c^2 \Delta p_x}{fB} \tag{2.48}$$

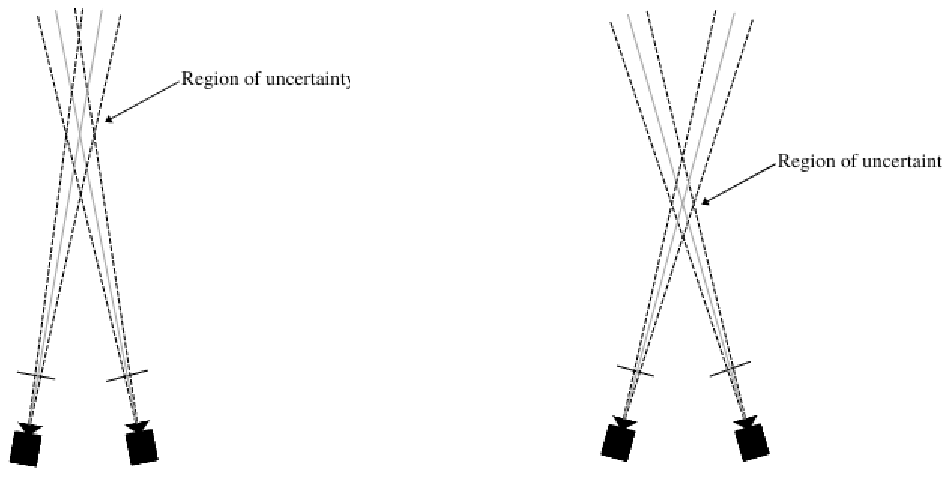
From eq.2.48 and in fig. 2.9 one can see that the error will be reduced by increasing the baseline. However, increasing the baseline will reduce the amount of overlap, (fig.2.7) and therefore increasing the minimum distance in front of the cameras where matching can occur.



(a) Small baseline **(b)** Large baseline **(c)** Large baseline + angled cameras

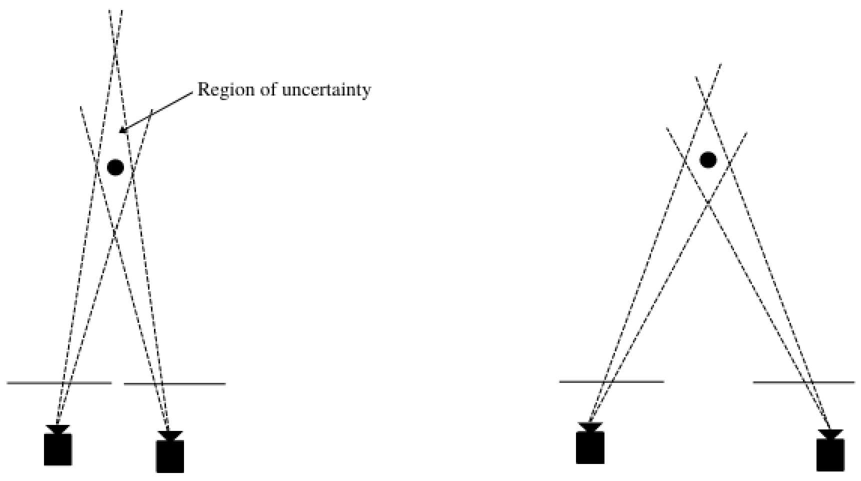
Figure 2.7: Baseline impact on overlapping field of view

Angling the cameras inward will increase the common field of view between the cameras and also lower the uncertainty in the measurements, but at the cost of a lower angular resolution making it harder to detect details in small objects. Fig. 2.9 and fig. 2.8 illustrates the changes in uncertainty. This means that the baseline has to be chosen based on a compromise between the two which is explained in chapter 6.



(a) Small angle **(b)** Large angle

Figure 2.8: Uncertainty regions with different camera angles



(a) Small baseline **(b)** Large baseline

Figure 2.9: Uncertainty regions with different baselines

In fig 2.10 the error in depth estimations are shown with a varying baseline, and where all other parameters in eq. 2.48 are fixed. Here it is clear that the error increases exponentially, and a large baseline will have a significant impact on the accuracy at long ranges.

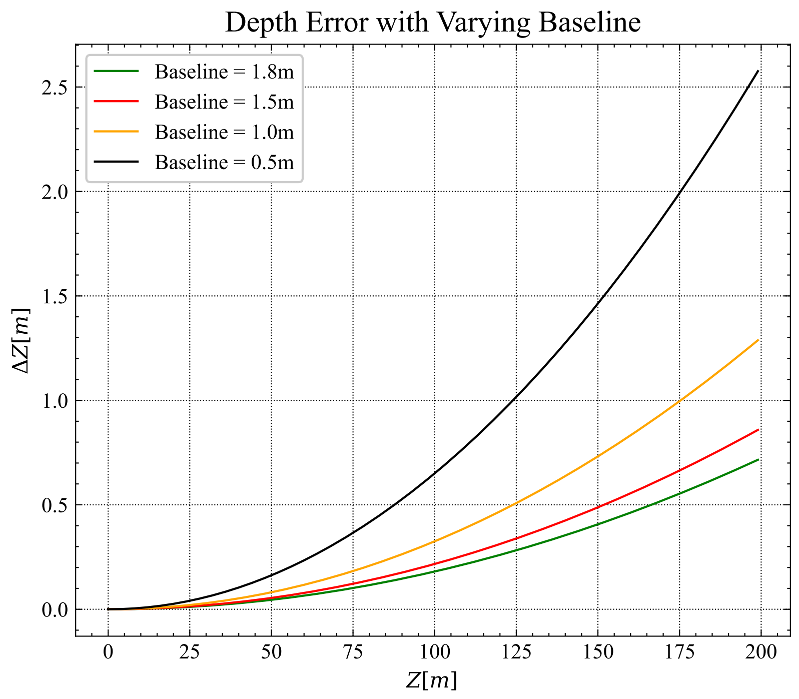


Figure 2.10: Example of how varying baseline affects estimation error. Fixed focal length (1230[px]) and disparity error (0.1[px])

The other physical parameter is the focal length, but as this is less practical to change and does not have the same range to vary within it will not be discussed further.

The second point regarding the accuracy of the previous parameters is achieved via calibration. Proper calibration is required as this sets the lower boundary of uncertainty in the measurements. From the results in [14] it is clear that yaw and non parallel photo sensor and lens are the two most important contributors to error in depth estimations. Then pitch, roll and radial distortion. This means that a proper calibration of both intrinsic- and extrinsic parameters is vital. The reprojection error is a measure of how well the calibration results are. This is a measure of the error between a measured point and the reprojected point onto the same image plane given the extrinsic parameters from the calibration. A large reprojection error indicates that the parameters provided by the calibration does not fit well with the real world. The equation for reprojection error (eq. 2.49) can be split into three steps.

- $||\boldsymbol{p}'_{il} - \boldsymbol{P}_l(\boldsymbol{p}_i)||^2$ - The square of the absolute difference between a measured -and projected point in the left image plane.
- $||\boldsymbol{p}'_{ir} - \boldsymbol{P}_r(\boldsymbol{p}_i, \boldsymbol{R}, \boldsymbol{t})||^2$ - The square of the absolute difference between a measured -and projected point in the right image plane given a rotation and translation.
- $\sum_{i=1}^{n}$ - Sum the error over all the points in an image pair.

$$\epsilon = \sum_{i=1}^{n} ||\boldsymbol{p}'_{il} - \boldsymbol{P}_l(\boldsymbol{p}_i)||^2 + ||\boldsymbol{p}'_{ir} - \boldsymbol{P}_r(\boldsymbol{p}_i, \boldsymbol{R}, \boldsymbol{t})||^2 \tag{2.49}$$

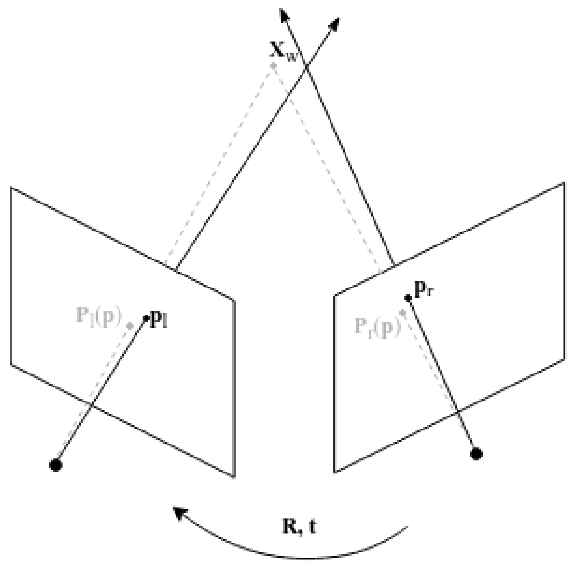


Figure 2.11: Illustration of reprojection error

As the reprojection error is not constant for all the points in an image it is common to calculate the mean, and use this value for the entire image. In fig. 2.12 the error in depth based on eq. 2.48 vary with rather small changes in mean reprojection error while the other parameters are fixed. The examples are all within the accuracy of a single pixel so-called *sub-pixel accuracy*, but at longer distances the error becomes significant.

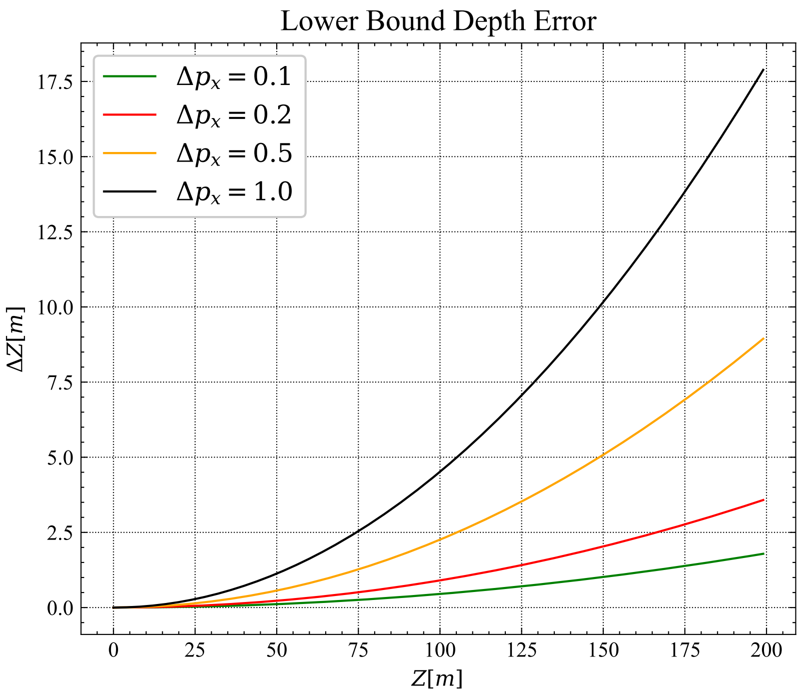


Figure 2.12: Example of how reprojection error affects estimation error. Fixed focal length (1230[px]) and baseline (1.8[m])

Sub-pixel Estimation

A problem regarding estimation in stereo vision systems is the disparity in pixel units has a physical limitation in resolution. The minimum resolution of one pixel severely affects the accuracy of depth estimations, especially for objects at large distances. The solution to overcome this limitation is *sub-pixel estimation*. One of the methods presented in [15] is by fitting a parabola over the best match and the nearest neighbors.

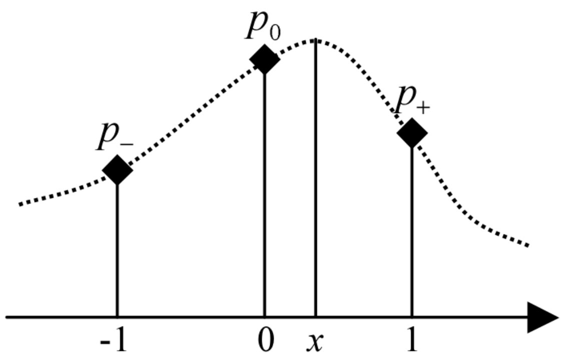


Figure 2.13: Parabola fit to disparity matches
Source:[15]

The parabola is defined as in eq. 2.50, and its local extrema by setting its differentiation equal to zero as in eq. 2.51

$$y = ax^2 + bx + c \tag{2.50}$$

$$\frac{dy}{dx} = 2x + b = 0 \tag{2.51}$$

$$x = \frac{-b}{2a} \tag{2.52}$$

The pixel values p_0, p_- and p_+ in fig. 2.13 correspond to to the matching score of the best match, the score at d-1 and at d+1 respectively. Applying the pixel positions to 2.50 results in:

$$\begin{aligned} y(-1) &= p_- = a - b + c \\ y(0) &= p_0 = c \\ y(1) &= p_+ = a + b + c \end{aligned}$$

Rewriting 2.51 we can solve for the local maximum using the pixel values.

$$x = \frac{p_- - p_+}{2(p_- + p_+) - 4p_0} \tag{2.53}$$

Estimation of corresponding points with sub-pixel accuracy allows for a more precise disparity value to be calculated. The new disparity estimate can now be expressed as in eq. 2.54.

$$d_{est} = d + x \tag{2.54}$$

Chapter 3

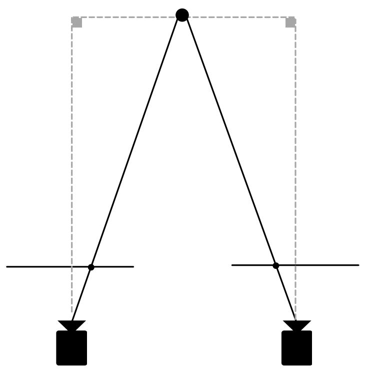
The Correspondence Problem

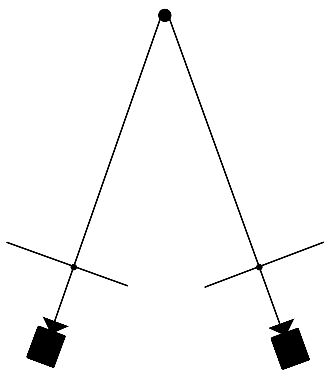
As described in section 2.2 a point in the world coordinate system can be estimated if seen from two different image views with known relative geometry. Recognizing the same point in both image views is known as the correspondence problem and is a key part of any stereo vision system. This problem can be formalized with two constraints [16]:

1. *Uniqueness* - Each pixel only have one corresponding match
2. *Continuity* - A scene is composed by piecewise continuous surfaces

A stereo system can be divided into two cases: *the simplified case* and *the general case*. In the simplified case the two cameras are aligned and identical, creating a simple representation of a system where the epipolar lines are horizontal. This simplifies the search for correspondences as the matching pixel is bound to be on the same image row in both images. However, in real life there is always some differences between the cameras in the system. Either wanted by design such as angled cameras or unwanted as there will always be some inaccuracy in the mechanical setup. This classifies as *the general case*.

A system in the general case can be transformed into the simplified case by rectification. This is a process where the rigid body transformation between the cameras, acquired by stereo calibration is applied to the images and remapping them onto a common image plane. Details about this process are described in section 2.3.


(a) Simplified stereo setup


(b) General stereo setup

Most stereo matching algorithms either require or benefit from having rectified images as input as this lowers the computational complexity of the problem.

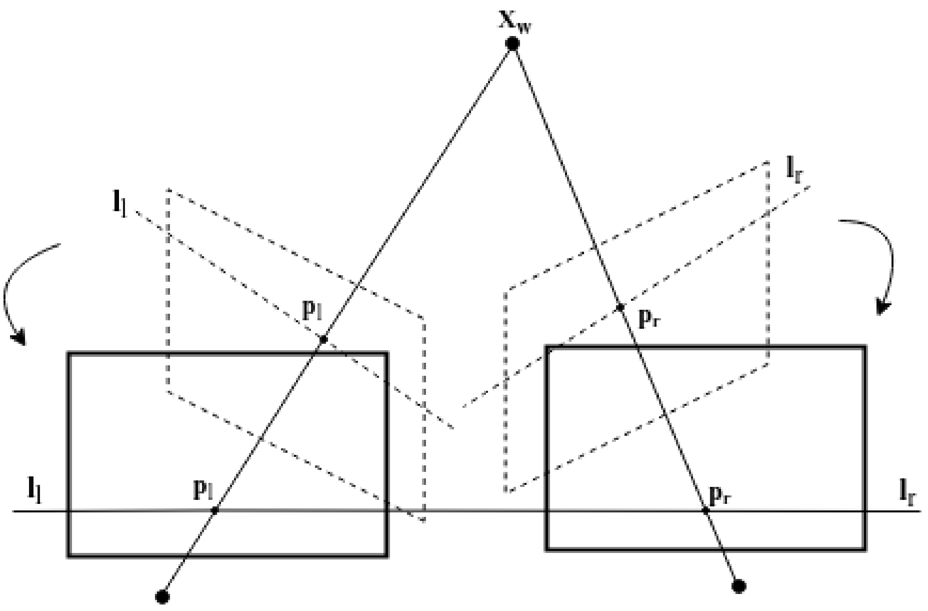


Figure 3.2: Image frames before and after rectification

3.1 Disparity Map

As a result of solving the correspondence problem the shift in pixel position between the two images is obtained. This shift in pixel position is directly proportional to the distance to the objects in the image. Features in the scene that are closer to the camera will change more than those that are further away. By mapping the pixels and giving them a *disparity value* based on how much the pixel changes between the two images a *disparity map* can be created. For two arbitrary pixels in a rectified image pair, matched by solving the correspondence problem $\tilde{\boldsymbol{p}}_l = (u_l, v_l, 1)^T$ and $\tilde{\boldsymbol{p}}_r = (u_r, v_r, 1)^T$ the disparity $(d)$is expressed in eq. 3.1.

$$u_l - u_r = d \tag{3.1}$$

By doing this for all the pixels in an image, a disparity map can be created. This is a new image where all the pixels are described by $\boldsymbol{p} = (u, v, d)^T$. After obtaining the disparity map each pixel can be transformed into actual depths by solving eq. 3.2, where f is focal length and B is the baseline.

$$Z_c = \frac{f_U * B}{d} \tag{3.2}$$

In order to establish a 3D point with origin in the camera center for every pixel in the disparity map eq. 3.3 can be solved. In this case u_0 and v_0 are the principal points of the camera, and T_x is the translation of the optical center between the left and right camera in the X-axis.

$$\boldsymbol{Q}\begin{bmatrix} u \\ v \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ W_c \end{bmatrix}, \quad where \quad \boldsymbol{Q} = \begin{bmatrix} 1 & 0 & 0 & -u_0 \\ 0 & 1 & 0 & -v_0 \\ 0 & 0 & 0 & f_u \\ 0 & 0 & -\frac{1}{T_x} & \frac{u_0 - u_0'}{T_x} \end{bmatrix} \tag{3.3}$$

Over the years, as computing power has increased, more and more complex methods for solving the correspondence problem has been presented. However, the methods are traditionally divided into two groups; local and global[17]. As indicated by their names, the local methods only consider the local area around a pixel, while the global approach takes the entire image into account when calculating disparity for a pixel.

3.2 Local Methods - Correlation Based

Correlation based matching methods use an area or matrix around a given pixel to describe that particular part of the image. This part of the image can then be searched for in in the matching image in order to establish a match. Often the images are rectified and epipolar lines are used as horizontal scan lines where one can slide the patch from the origin image along the scanline to search for a matching patch. Once a matching area is found the disparity in pixel position can be extracted.

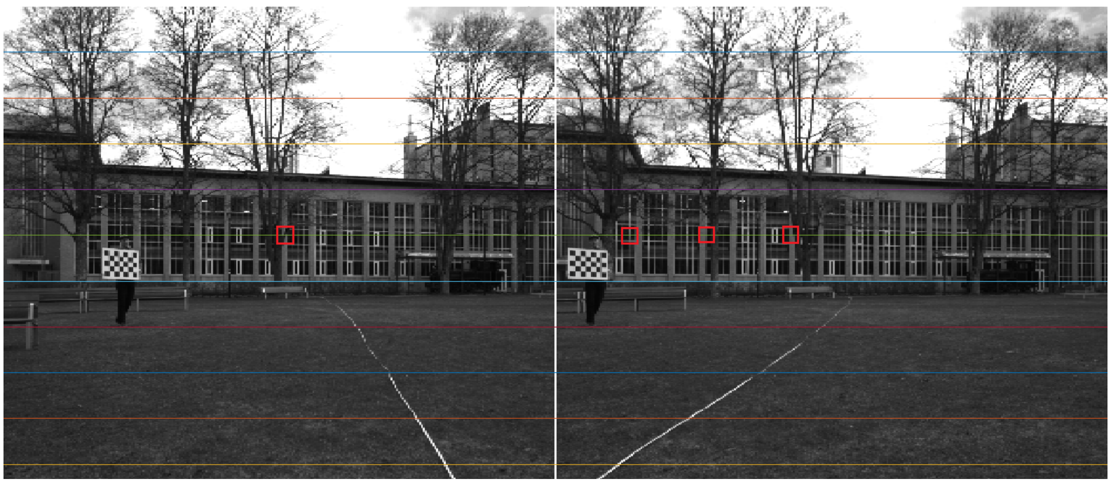


Figure 3.3: Illustration of patch sliding across scanline

These methods result in a dense disparity maps and are generally computationally fast, but yield poor results in certain settings. Listed below are common terms used in the different mathematical equations describing the methods.

$$\begin{aligned} I_L &\text{—Left stereo image} \\ I_R &\text{—Right stereo image} \\ W_m &\text{—Matching window} \\ d &\text{—disparity} \\ I(u,v) &\text{—Pixel intensity} \\ C(d) &\text{—Cost of matching with disparity} d \end{aligned}$$

Sum of Absolute Differences - SAD

The Sum of Absolute Differences approach is the most common stereo matching algorithm [18]. After locating an area of interest in the origin image, a patch around that area is saved and moved along a search line on the matching image. For each pixel along the search line the sum of absolute differences is calculated.

$$C_{SAD}(d) = \sum_{(u,v)\in W_m} |I_L(u,v) - I_R(u-d,v)| \tag{3.4}$$

In eq. 3.4 the absolute difference between two blocks is calculated. The disparity, d can be extracted where the function output is lowest. There is a compromise between accuracy and execution time in varying the size of the matching block. Increasing the size of the block makes the algorithm less prone to errors, but increases execution time.

Sum of Squared Differences - SSD

The SSD matching method works by subtracting the values from the extracted patch pixel by pixel along a search line. In theory the matching patch is found when the SSD returns zero, but in practice the lowest return value is accepted [19].

$$C_{SSD}(d) = \sum_{(u,v)\in W_m} [I_L(u,v) - I_R(u-d,v)]^2 \tag{3.5}$$

This simple principle yields a fast algorithm, but in turn, it fails at depth discontinuities as the whole patch is considered to have the same disparity. For this reason the SSD-method is not suited for images containing many or small objects.

Normalized Cross Correlation - NCC

NCC is both more complex and robust than SSD. As the values in NCC are normalized, the method handles change in luminosity between cameras well [19]. In eq.3.6 $\bar{I}$ is the average pixel intensity of the patch and is only calculated once for each matching operation.

$$C_{NCC}(d) = \sum_{(u,v)\in W_m} \hat{I}_L(u,v)\hat{I}_R(u-d,v) \tag{3.6}$$
$$where, \quad \hat{I}(u,v) = \frac{I(u,v)-\bar{I}}{||I-\bar{I}||_{W_m}}$$

Because the NCC is more complex than the SSD, it is difficult to achieve run times that are low enough for ream time operations. Further, the NCC is also prone to error at discontinuities as the other methods.

Rank Transform

Rank Transform is a non-parametric algorithm, meaning it does not consider the value of the pixel intensity itself, rather its relative value. By evaluating the pixels in a patch/block around the center pixel the rank of that patch can be determined by simply the number of pixels with a lower value [20] (Non parametric).

$$R(P) = ||P' \in N(P)|I(P') < I(P)|| \tag{3.7}$$

In practice this rank calculation can be described as in eq. 3.8

$$Rank(u,v) = \sum_{(i,j)(u,v)} L(i,j), \qquad L(i,j) = \begin{cases} 0 & : I(i,j) < I(u,v) \\ 1 & : otherwise \end{cases} \tag{3.8}$$

After the image has been rank transformed one of the methods presented earlier such as SAD can be used on the transformed images.

$$C_{RT}(d) = \sum_{(u,v)\in W_m} |Rank_L(u,v) - Rank_R(u-d,v)| \tag{3.9}$$

Doing this achieves a lower run time as the rank transform lowers the complexity of the problem [21].

Census Transform

Census Transform is also a non-parametric measure of local intensity [20] In this case the values surrounding a pixel is set to 0 or 1 based on whether they are higher (0) or lower (1) than the value of the center pixel. These values are then mapped into a bit string. This bit string can then be compared with a bit string originating from the second stereo image using the Hamming distance [20]. Minimizing the hamming distance is done to establish the correct match. This method is robust when there is a high change in intensity in a few pixels.

$$Census(u,v) = Bitstring_{(i,j)\in W_m}(I(i,j) \geq I(u,v)) \tag{3.10}$$
$$C_{CT}(d) = \sum_{(u,v)\in W_m} Hamming(Census_L(u,v) - Census_R(u-d,v)) \tag{3.11}$$

3.3 Local Methods - Feature Based

A feature, or key point is an area in an image with a certain uniqueness that can be described and stored. This uniqueness can be corners, edges, patches and other areas that is able to stand out in the image.

A feature detector is the method that searches for these areas and stores the most prominent ones as *feature descriptors*. There are several types of feature descriptors. Common for all of them is that they are designed to be quite robust, meaning that during searching for a matching feature, a false positive should seldom occur. Feature detection and matching is a key element in almost every computer vision application and is often used in areas such as object recognition and structure from motion.

Scale Invariant Feature Transform - SIFT

Because good feature might be located at different scales in an image it is beneficial to search through a range of scales when identifying a feature. This is the idea behind the method, SIFT, first presented by David G. Lowe in [22] and has since then been the benchmark method regarding accuracy and robustness in visual features in computer vision. The method consists of four stages:

Scale-space Extrema Detection

Detecting keypoints in an image that is invariant to scale is done by applying Gaussian blur to the image with varying magnitude, then "stacking" the images on top of each other and search for extreme points. This procedure is done for every level in a scale pyramid. This pyramid is created by consecutively lowering the image width and length. Subsampling and smoothing the image pixels is done to achieve a lower resolution.

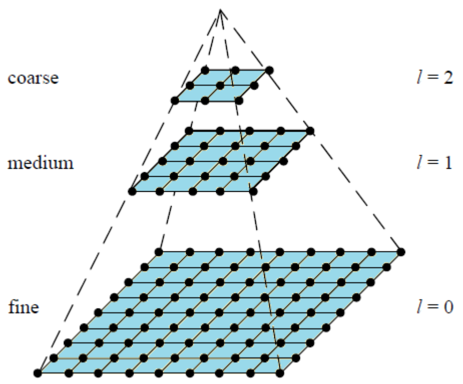


Figure 3.4: Pyramid where each level has half the width and length as the previous level resulting in a quarter of the pixels.
Source:[23, p. 132]

The scale-space of an image is defined in eq. 3.12. Where $L(u,v,\sigma)$ is the scale-space, I(u,v) is the input image, and $G(u,v,\sigma)$ is the Gaussian blur, and * is the convolution operator.

$$L(u,v,\sigma) = G(u,v,\sigma) * I(u,v) \tag{3.12}$$

The difference-of-Gaussians (DoG), $D(u,v,\sigma)$, is calculated by subtracting a scales-space image with another where a constant, k, is separating them in scale.

$$D(u,v,\sigma) = L(u,v,k\sigma) - L(u,v,\sigma) = (G(u,v,k\sigma) - G(u,v,\sigma)) * I(u,v) \tag{3.13}$$

Keypoints are found using DoG, then blurred with Gaussian blur of different magnitudes. Blurred images are subtracted from each other and stacked on top of each other to enhance for extreme points.

Accurate Keypoint Localization

At this time in the process both stable and unstable keypoints are detected. A quadratic model is fit to the nearby points depending on location and scale. A second order Taylor expansion is used to get a more accurate location and scale of the keypoints. SIFT then chooses the most stable keypoints and discards the others.

Orientation Assignment

Each keypoint is assigned an orientation based on local image properties. Gradient direction and magnitude are calculated for every keypoint. By giving the keypoint a fixed orientation the keypoints becomes invariant to rotation which is needed as matching needs to occur even with rotated images.

Keypoint Descriptor

The descriptor is what describes a unique keypoint and it is therefore important that it is as unique as possible. In SIFT an area of 16×16 pixels around the keypoint is divided into 16 subsections. In each subsection eight directions are calculated yielding $16 \times 8 = 128$ elements in each descriptor vector. The vector is then normalized to unit length to be more robust against illumination changes.

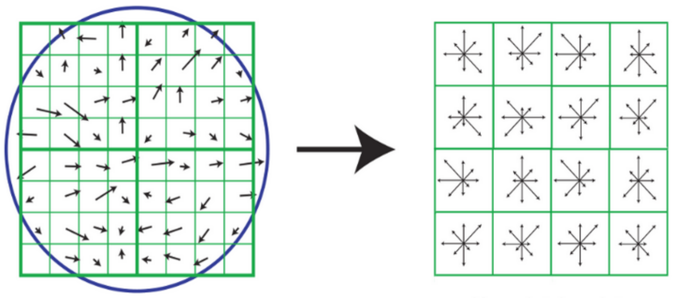


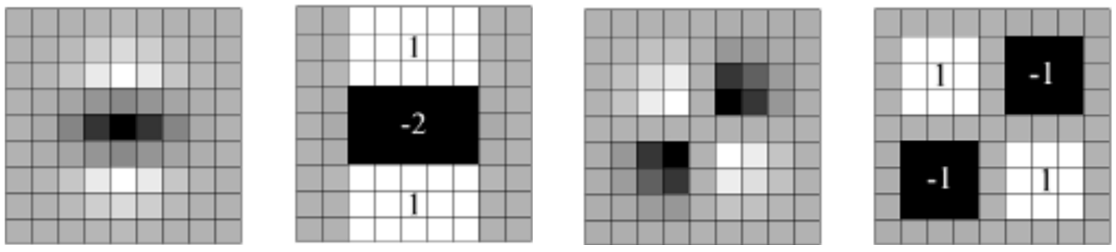
Figure 3.5: SIFT descriptor
Source: [22]

Speeded Up Robust Features - SURF

SURF is a feature extractor and feature descriptor created as a faster alternative to SIFT. Although SURF is based on the same steps as SIFT the authors claim that it is several times faster, and more robust than SIFT during several image transformations [24].

SURF uses box-filters to approximate the Laplacian of Gaussians. It is in this step SURF gains its computational advantage against SIFT as filtering the image using a square and an integral image is much faster than how SIFT does its filtering. Selecting point ($\boldsymbol{p}$) and scale (σ), the determinant of the Hessian is calculated. The Hessian is defined in eq. 3.14, where L_{uu}, L_{uv} and L_{vv} are convolutions of the Gaussian second order derivative.

$$H(u,v,\sigma) = \begin{bmatrix} L_{uu}(u,v,\sigma) & L_{uv}(u,v,\sigma) \\ L_{uv}(u,v,\sigma) & L_{vv}(u,v,\sigma) \end{bmatrix} \tag{3.14}$$



(a) Prior to filtering **(b)** After L_{uv} filtering **(c)** Prior to filtering **(d)** After L_{vv} filtering

Figure 3.6: SURF Box-Filtering
Source: [24]

The determinant of the Hessian (eq. 3.15) is used as a measure of changes in the local area around a point. The parameter w is a balancing weight making the approximated Gaussian kernels equal to the Gaussian kernels. The point is then chosen where the determinant is maximum.

$$det(\mathcal{H}_{approx} = D_{uu}D_{vv} - (wD_{uv})^2) \tag{3.15}$$

The SURF descriptor is based on the same properties as the SIFT descriptor but scaled down to be less complex. Creating the descriptor consists of two steps, where firstly the orientation is determined calculating the Haar-wavelet in both the u and v direction. This is required to make the descriptor invariant to rotation. Once the orientation is determined, a square region aligned with the orientation is created, and the descriptor is extracted from this region. Further information and details about SURF is available in the original paper [24].

Oriented FAST and Rotated BRIEF - ORB

The ORB feature detector was developed by OpenCV as an unpatented alternative to the previous mentioned SIFT and SURF [25]. As the name suggests, it is built using the FAST key point detector and BRIEF descriptor. For more details about the underlying key point detector and descriptor the reader is referred to their original papers, [26] and [27] respectively. Firstly ORB detectes FAST key points in the image, then selects the best points by employing a Harris corner measure. Because FAST do not calculate orientation the authors added a step where an intensity weighted centroid of a corner is calculated, and a vector from the center of the corner to the centroid is calculated to determine orientation. The moments of the image patch is calculated in eq. 3.16 and the centroid in eq. 3.17.

$$m_{lr} = \sum_{u,v} u_l v_r \boldsymbol{I}(u,v) \tag{3.16}$$

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{3.17}$$

The orientation of the vector from the center to the centroid is now a matter of trigonometry.

$$\theta = \text{atan2}(m_{01}, m_{10}) \tag{3.18}$$

To handle variance in scale a pyramid scheme is used where FAST features filtered by Harris corners is employed on all levels of the pyramid. ORB is popular in real-time and low power applications as it is considered to be resistant to noise and it is more efficient than SIFT.

3.4 Global Methods

The global methods are based on optimization. Many methods are formulated using an energy-minimization framework where the goal is to find a disparity map (D) that minimizes a global energy function (eq.3.19) [23]. This means that the global methods tries to establish a disparity map for the entire image all in once, wheas the local methods only considered small patches.

$$E(D) = E_D(D) + \lambda E_s(D) \tag{3.19}$$

In eq. 3.19 $E_D(D)$ is a measure of how well the disparity correspond with the input images. $E_s(D)$ is a smoothness term which in practise penalizes disparity changes in neighbouring pixels. However, as 2D optimization for eq. 3.19 can be NP-hard in some cases [23, p. 485] other approaches needs to be considered to solve the problem within reasonable time.

Dynamic Programming - DP

In 1985 Otha and Kanade presented a method where a pair of rectified images could be matched by first doing a series of 1D line optimizations called *intra-scanline search* [28]. The idea is that the intensities along the scanline at the two images are mapped onto a 2D plane. From this map dynamic programming can be utilized to find the shortest path from start to finish on both images. A node (i,j) on the path represent a match between the images.

Figure 3.7: 2D intra-scanline
Source: [28]

Restructuring the optimization problem from 2D to $n \cdot 1D$ problems reduced the run time from exponential to linear, but now each scanline is independent of the others. If there is a vertical edge across multiple scanlines the correspondences in one scanline should be strongly dependent on the neighboring scanlines. To account for this dependency another scanline called the *inter-scanline* has the task to localize vertical edges crossing multiple intra-scanlines. This is done by identifying the minimum cost path between 3D nodes in a stack of 2D planes. A 3D node is formed when an edge is on the same location across multiple 2D planes. This 3D optimal-path-finding method is conducted using dynamic programming in the same way as for the 2D problem.

Graph Cuts - GC

GC is a method of solving the stereo matching problem that does not rely on epipolar constraints. Rather it formalizes the problem as an optimization problem that can be solved using graph cuts. $G =< V, E >$ is a weighted graph with two end vertices called terminals. V is the set of vertices and $\mathcal{E}$ is the edge set. A cut, C, is a set of edges that separates the terminals, where the cost of the cut, $|C|$ is the sum of the edge weights. The correspondence problem can be designed as a minimum cut problem, meaning the task of finding the set of cuts with lowest cost.

The goal of solving the correspondence problem is to label every pixel in an image with a disparity value. A cost function can be assigned to the labeling process, and the goal is to minimize this cost function. There are several different algorithms solving the problem in different ways, but common in them all is that by designing the graph and edge weights carefully the minimization of the cost function can be expressed as a min-cut problem [29].

The disadvantage of the GC approach is that it is slow. The average run time is roughly $O(n^{1.2}d^{1.3})$ where n is the number of pixels in the image and d is the disparity range. This makes it unsuitable for real time applications such as milliAmpere and will therefore not be explained in more detail. For more information the reader is referred to Boykovs paper explaining two of the fist GC-algorithms [30].

Semi-Global Method - SGM

The Semi-Global Method is trying to maintain the accuracy from global methods while still maintaining the speed of the local methods. It performs pixelwise matching based on Mutual Information (MI) and an approximation of a global smoothness constraint, and was first introduced by Hirschmuller in 2005 [8]. The algorithm uses line optimization from multiple directions, and calculates the aggregated cost to each pixel $\boldsymbol{p}$ with disparity d.

Pixelwise Cost Calculation

The matching cost is calculated for a pixel $\boldsymbol{p}$ in the left image $I_{l\boldsymbol{p}}$ and its correspondence in the right, matching image, $I_{r\boldsymbol{p}}$ based on the intensity of the first pixel. The matching cost calculation is based on MI, which is defined by entropy (H). This is used as a measure of how much information the two images have in common. The entropy is calculated from the probability distributions of intensities of the right image [8].

$$MI_{I_l,I_r} = H_{I_l} + H_{I_r} - H_{I_l,I_r} \tag{3.20}$$

Calculation of the joined entropy H_{I_l,I_r} can be expressed by a sum of data terms (h_{I_l} and h_{I_r}) using the Taylor expansion [31]. The probability distribution must only be calculated for corresponding areas in order to ignore occlusions. The probability function is defined by eq.3.21 where T is an operator returning 1 if the arguments is true and 0 otherwise.

$$P_{I_l,I_r}(u,v) = \frac{1}{n}\sum_{\boldsymbol{p}} T[(u,v) = (I_{l\boldsymbol{p}}, I_{r\boldsymbol{p}})] \tag{3.21}$$

$$MI_{I_l,I_r} = \sum_{\boldsymbol{p}} mi_{I_l,I_r}(\boldsymbol{I}_{lp}, \boldsymbol{I}_{rp}) \tag{3.22}$$

$$mi_{I_l,I_r}(i,k) = h_{I_l}(i) + h_{I_r}(k) - h_{I_l,I_r}(i,k) \tag{3.23}$$

Pixel wise cost calculation using MI determines the absolute minimum difference of intensity at the pixel in the query image, $\boldsymbol{p_r}$, and at the pixel in the origin image, $\boldsymbol{p_l}$.

$$C_{MI}(\boldsymbol{p_l}, d) = -mi_{I_1,I_2}(I_{1p_l}, I_{2p_r}) \quad with \quad \boldsymbol{p_r} = \boldsymbol{p_l} + d \tag{3.24}$$

Aggregation of Cost

With pixel wise cost calculation incorrect matches commonly occur as a result of noise in the image. This is due to the noise causing a lower cost than the correct match. To overcome this challenge additional constraints regarding the area surrounding the pixel in question is added. These constraints penalize changes is disparity in the neighborhood surrounding the pixel. Adding these cost terms yields an energy function (eq. 3.25) that can be minimized to find the best disparity image $D_{\boldsymbol{p}}$.

$$E(D) = \sum_{\boldsymbol{p}_l} C(\boldsymbol{p}_l, D_{p_l}) + \sum_{p_r \in N_{p_l}} P_1 T[|D_{p_l} - D_{p_r}| = 1] + \sum_{p_r \in N_{p_l}} P_2 T[|D_{p_l} - D_{p_r}| > 1] \tag{3.25}$$

Equation (3.25) can be broken down into three parts explained better in table 5.1.

$\sum_{\boldsymbol{p}_l} C(\boldsymbol{p}_l, D_{p_l})$	Pixel matching cost for the disparities D
$\sum_{p_r \in N_{p_l}} P_1 T[\lvert D_{p_l} - D_{p_r}\rvert = 1]$	Constant penalty for small changes in disparity
$\sum_{p_r \in N_{p_l}} P_2 T[\lvert D_{p_l} - D_{p_r}\rvert > 1]$	Larger constant penalty for larger changes in disparity

Table 3.1: Explanation of elements in energy function
Source: [7]

However, minimizing eq. 3.25 can be NP-hard [30], and thus too complex for a real time system. To work around this SGM uses line optimization from multiple different directions (fig. 3.8) to approximate a global smoothness constraint [8]. This leads to a new cost function (eq. 3.26) which is the sum of all the paths to the pixel in question.

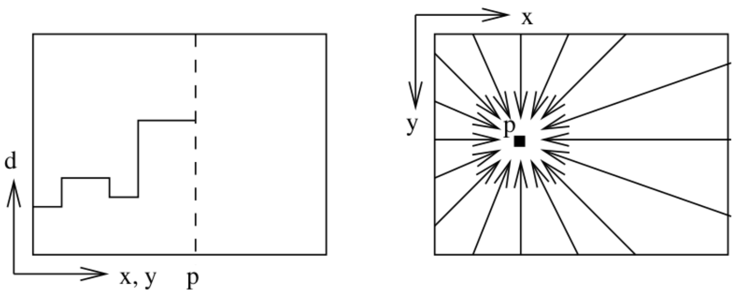


Figure 3.8: Aggregation of cost
Source: [8]

$$S(\boldsymbol{p}_l, d) = \sum_{r} L_r(\boldsymbol{p}_l, d) \tag{3.26}$$

The amount of paths to the target pixel can be varied. Having many paths, for instance 16 will create a more accurate disparity map than using eight. This increase in accuracy comes at the cost of an increased computational complexity and should therefore be evaluated for each specific use.

Chapter 4

Evaluation of Matching Methods

One of the goals of the thesis is to evaluate matching methods. The evaluation that was conducted as a combination of a literature study and physical experiments where the most promising methods according to the literature were tested. A comparison study of local algorithms done in the fall of 2019 on the same equipment ([6]) will be the main source of evaluating local methods. The extensive survey done in [32] and [29] are also central in this evaluation.

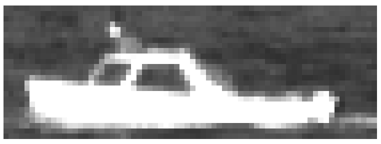
To evaluate what methods are better than others certain criteria for the given applications needs to be established. These criteria is what I consider to be important for a stereo vision system on milliAmpere, and will differ from what is considered important on other systems.

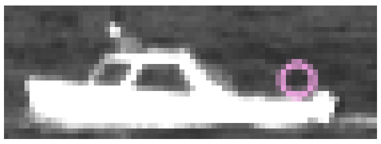
- *Dense disparity map* - A dense disparity map provides a disparity value for all pixels in an image. This naturally provides more information than a sparse disparity map where large areas of the image has an undefined disparity value. A dense representation will give a more realistic view of the world.
- *Real time performance* - As the system is to be used in applications such as collision avoidance and tracking the system has to be able perform in real-time.

4.1 Local Methods - Feature Based

As the feature based methods only consider a small set of independent points in the image these methods result in a sparse disparity map [33]. According to the results in [6] ORB was considered a viable option regarding accuracy, but as the tests was conducted indoors at a distance of <10 meters the testing conditions were quite different than what milliAmpere is operating in. For a quick comparison of the three most promising methods a small experiment was conducted to see if the methods were able to find any features on a boat in the range 40-60 meters from milliAmpere.

From fig.4.1 and fig.4.2 it can be seen that using the ORB detector in this scenario would result in the boat being undetected by the system. The SURF key point would likely be considered as noise if it was matched between images at all. SIFT on the other hand was able to identify more key points.


(a) ORB - 0 key points


(b) SURF - 1 key point

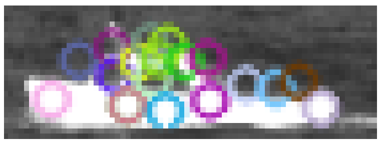
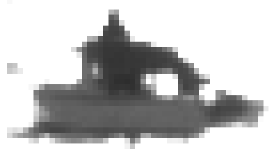
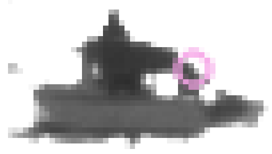

(c) SIFT - 19 key points

Figure 4.1: Keypoint detectors - 1


(a) ORB - 0 key points


(b) SURF - 1 key point

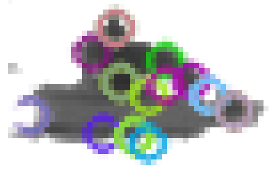

(c) SIFT - 19 key points

Figure 4.2: Keypoint detectors - 2

4.2 Local Methods - Correlation Based

Multiple of the correlation based matching methods could probably have been used. However, NCC is eliminated from the process due to its computational complexity. SAD is known for being fast, fairly accurate and easy to implement I chose to use this for further analysis. In an earlier comparison study of local methods [5] SAD was also chosen to be the best option.

4.3 Global Methods

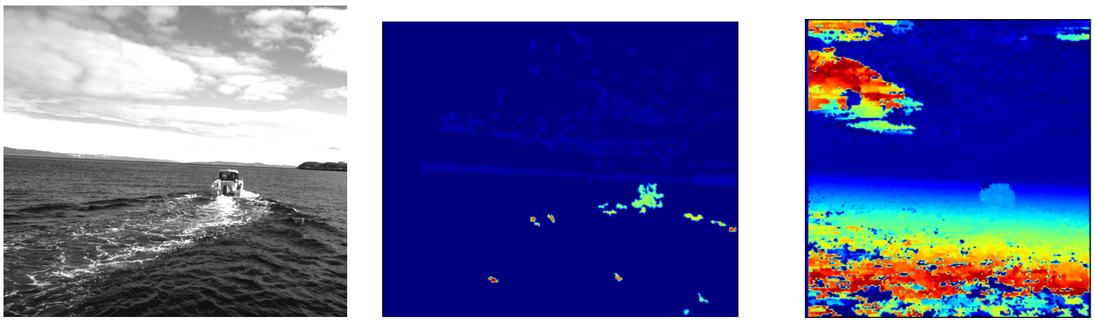
As the traditional global methods are known for having large computational complexity making them unsuitable for real time applications even on small images. Especially the GC-methods, but also algorithms based on DP. This leaves us with only the SGM alternative as a viable option from this category. The SGM is also the newest of the lot, built specifically to be faster than the earlier global methods.

4.4 SGM vs SAD

After comparing the different methods of solving the correspondence problem the correlation based method, Sum of Absolute Differences and the global method, Semi-Global Method is chosen to be evaluated in detail. There are two factor that will be considered in choosing the final matching method:

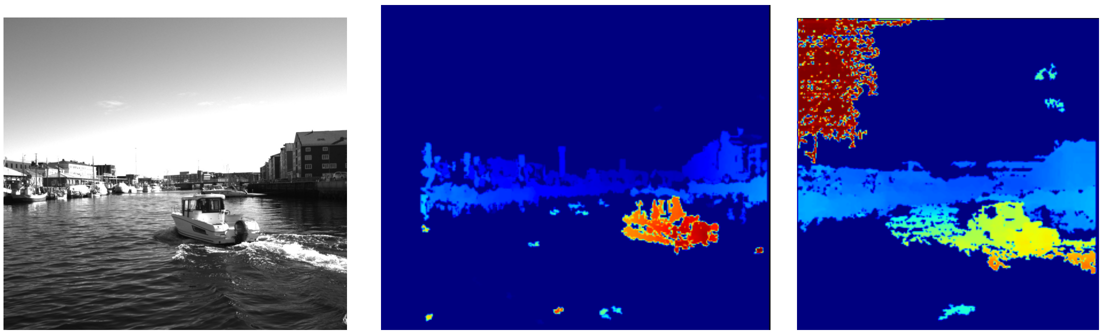
- Disparity map - The quality of the disparity map regarding level of detail and overall information
- Run time - As this is going to be used in an real time application the run time needs to be low enough

Prior to generating the disparity maps, a series of stereo images was used to tune the parameters for each method. This tuning process was done using a disparity tuner, which will be explained further in chapter 6. The disparity maps presented here were generated using the parameters yielding the best results in this tuning process.



(a) Input image **(b)** Disparity map - SAD **(c)** Disparity map - SGM

Figure 4.3: Comparison of disparity maps - 1



(a) Input image **(b)** Disparity map - SAD **(c)** Disparity map - SGM

Figure 4.4: Comparison of disparity maps - 2

The run times for each disparity map was also timed. As the run time of the SGM method heavily relies on how many directions is used two different variations of the SGM was tested. One with eight directions and one with three.

SAD [ms]	SGM 3-WAY [ms]	SGM 8-WAY [ms]
54.18	125.55	443.16

Table 4.1: Run times for SAD and SGM

In table 4.1 it is clear that SAD is more than two times faster than *SGM 3-WAY* and eight times faster than *SGM 8-WAY*. Knowing how huge the differences in run time is between the methods, a trade off between accuracy and run time has to be made with regards to the specifications of the system. Comparing the disparity maps presented below, there is an obvious difference in detail between the SAD and SGM, and close to no difference between the two SGM variations.

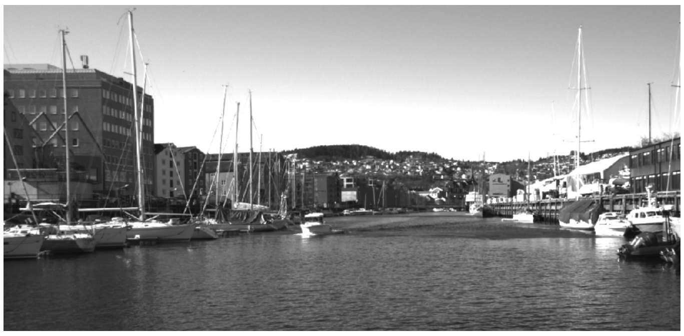


Figure 4.5: Input image - 1

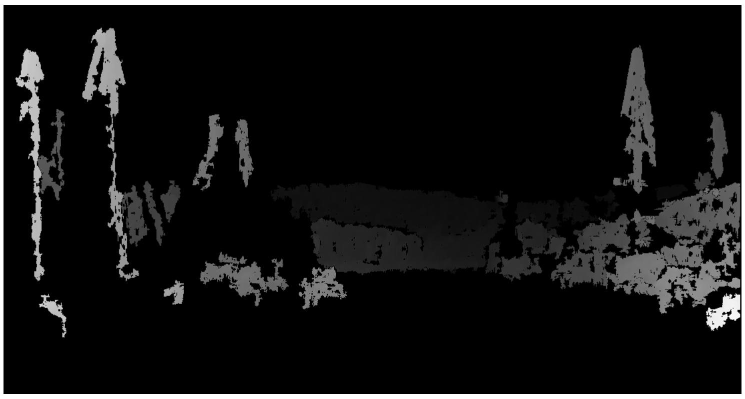


Figure 4.6: Disparity map example - SAD - 1

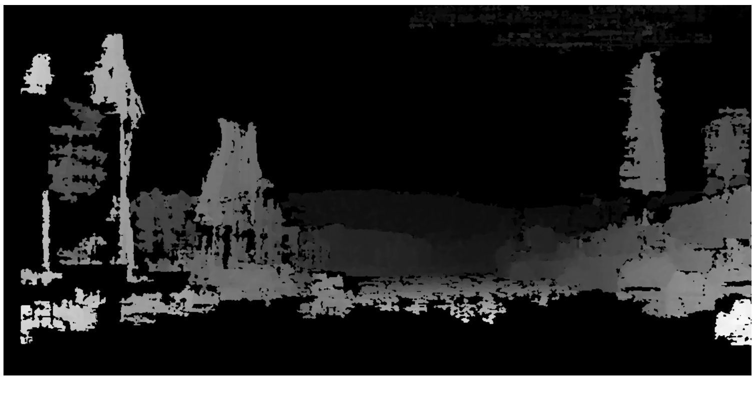


Figure 4.7: Disparity map example - SGM 3WAY - 1

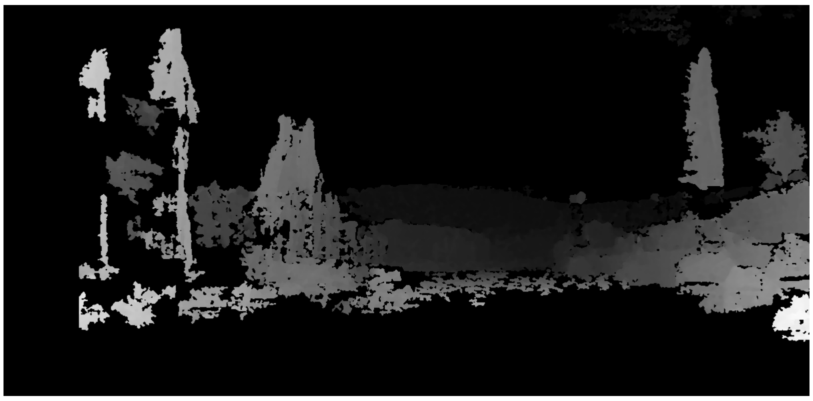


Figure 4.8: Disparity map example - SGM 8WAY - 1

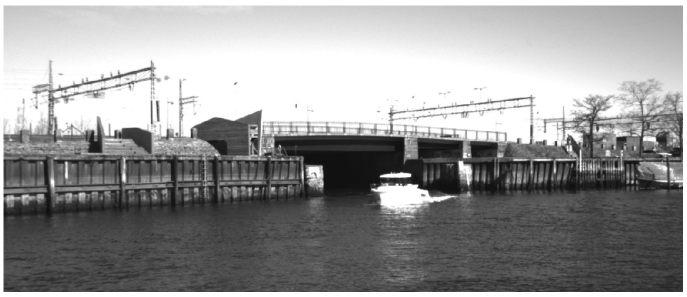


Figure 4.9: Input image - 2

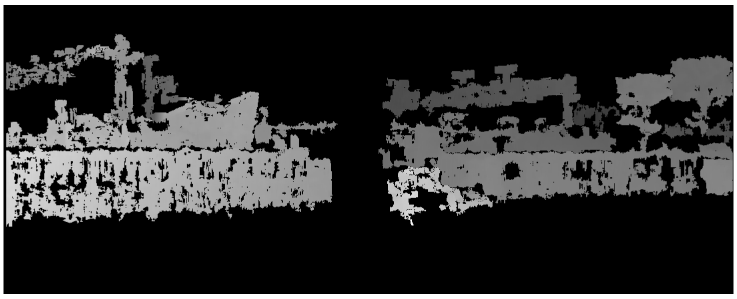


Figure 4.10: Disparity map example - SAD - 2

Figure 4.11: Disparity map example - SGM 3WAY - 2

Figure 4.12: Disparity map example - SGM 8WAY - 2

After analyzing both run times and details in the disparity map, the Semi-Global Method with three directions further. The increase in in run time is worth the increase in detail in the disparity map. In table 4.2 is a summary of the matching methods.

Method	Local/global	Feat./Corr.	Disp.map
SAD	Local	Corr	Dense
SSD	Local	Corr	Dense
NCC	Local	Corr	Dense
RT	Local	Corr	Dense
CT	Local	Corr	Dense
ORB	Local	Feat.	Sparse
SURF	Local	Feat.	Sparse
SIFT	Local	Feat.	Sparse
OthaKanade	Global	-	Dense
GC	Global	-	Dense
SGM	Global	-	Dense

Table 4.2: Summary of selected matching methods

Chapter 5

Object Detection using Stereo Vision

Object detection is an important component in all autonomy systems [34]. Autonomous vehicles are often designed to traverse in dynamic environments, sharing a common space with other vehicles, people and stationary objects. The vehicle therefore is dependent on reliable and accurate object detection and depth estimation to plan safe and efficient trajectories and avoid collisions. For this reason an object detection algorithm has to meet these four criteria [35]:

1. *Compact* - It needs to reduce the data volume to achieve real time execution.
2. *Complete* - The information of interest needs to be preserved.
3. *Stable* - Small changes in the input data needs to handled without large changes in the output data.
4. *Robust* - Noise and outliers in the input data must have minimal impact on the output.

The traffic in the Trondheim canal consists of an array of surface vehicles, for instance the Munkholmen ferry, leisure boats, sailing boats, kayaks, SUP-paddlers and so on. Appearance based methods with classification needs to be trained on huge data sets containing multiple variations of all possible objects to be detected. The size and variations of the data set needed to train such an algorithm to be accurate and reliable in the case of milliAmpere is to big to consider in this thesis. The lack of a high quality data set might result in the detector missing or wrongly classifying objects which could lead to a possible dangerous situation. For this reason only generic object detection algorithms will be considered in this thesis.

In this chapter five of the most recent advances in object detection using stereo vision will be presented; *Stixel Tesselation*, *Digital Elevation Map - DEM*, *Geometry Based Cluster*, and *Direct Planar Hypothesis Testing - DPHT*, and *Euclidean Clustering*. The first three are presented in a study by Berning et. al [36]. This is a survey showing the most recent algorithms regarding stereo vision object detection able to operate in real time. Further, the more recently developed method, DPHT, presented by Pinggera et. al [37] will also be described. This method differs from the traditional methods by not being dependent on a disparity map, but rather directly using the image information. The last is a combination of filtering, clustering and accumulation used by Olsen and Theimann in their master's thesis [6].

5.1 The Ground Plane

Many obstacle detection approaches depend on a ground plane, a single planar surface that defines the area the vehicle can move, and obstacles are characterized by being above this surface. Some of the early methods include [38] and [39]. The main drawback of these methods is that the performance of the obstacle detection method heavily relies on the accuracy of this plane. When traversing off-road terrain or other areas where such a plane does not fit well with the real world more sophisticated ground profile models needs to be used, or a completely different approach all together.

As most of the research in obstacle detection field has been focused on the automotive industry alternative methods to the ground plane has been a huge focus, and has been the main reason for methods such as [34] and [37]. However, as the operating environment for milliAmpere is the Trondheim canal, the flat world assumption is much more viable.

5.2 Method 1 - Stixel Tesselation

One major contribution within object detection is the Stixel tesselation, first introduced by Badino et. al in [35]. This algorithm was developed as a result of the Semi-Global Method that was described in section 3.4, enabling accurate and dense disparity maps to be generated in real time. This algorithm separates objects from the rest of the image and represents them as columns or *stixels*. Each stixel is defined by its 3D position relative to the camera and stands vertically on the surface plane.

Probabalistic Occupancy Map

The stixel method is built upon the concept of a *Probabalistic Occupancy Map*. A definition of an occupancy grid by Badino et. at [40] is:

"*An occupancy grid M is a two-dimensional array or grid which models occupancy evidence of the environment. The 3D world is orthographically projected on a plane, parallel to the surface*".

This concept was first introduced in [41] where the authors used a sonar mounted on a mobile robot to map the surroundings. This resulted in a a two-dimensional grid map labeling the grid cells either free, occupied or empty based on the probability of there being an object at the location. Later the concept was expanded to multiple dimensions enabling each cell to contain information such as texture and color.

A simplified version of the problem is presented in eq. 5.1 where q is value of the grid cell, and $\boldsymbol{m_k}$ is the stereo measurements.

$$p(q|m_1, ..., m_k) \tag{5.1}$$

To convert the stereo measurements into occupancy likelihoods we can follow the method presented in [40]. A stereo measurement is defined as $\boldsymbol{m_k} = (u_l, v_l, d)^T$, where u_l and v_l is the left image coordinate and d is the disparity in pixels. $\boldsymbol{m_k}$ is then a projection of the real world point $\boldsymbol{X_w}$ onto the image plane. The notation f_u and f_v represents the focal length of the camera, u_0 and v_0 is the principal point, and B is the baseline of the stereo system.

$$\boldsymbol{m_k} = \boldsymbol{P}(\boldsymbol{X_w}) = \frac{1}{z}\begin{bmatrix} f_u x \\ f_v y \\ f_u B \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \\ 0 \end{bmatrix} \tag{5.2}$$

Naturally the real world point is unknown and the inverse of eq. 5.2 is the triangulation equation providing the real world information

$$\boldsymbol{X_w} = \boldsymbol{P}^{-1}(\boldsymbol{m}_k) = \frac{B}{d}\begin{bmatrix} (u-u_0) \\ (v-v_0)\frac{f_u}{f_v} \\ f_u \end{bmatrix} \tag{5.3}$$

The measurement $\boldsymbol{m}_k$ contains the real and unknown $\bar{\boldsymbol{m}}_k$ and a noise term $\bar{\xi}_k$ which is assumed to be a zero mean random process with probability density function $\boldsymbol{G}_{\bar{m}_k}$. Eq. 5.4 models the likelihood of obtaining an error ξ_k given the fact that the real state is $\bar{\boldsymbol{m}}_k$ [40]. In eq. 5.4 $\bar{\boldsymbol{\Gamma}}_k$ is the covariance matrix of the measurement $\bar{\boldsymbol{m}}_k$.

$$\boldsymbol{G}_{\overline{\boldsymbol{m}}_k}(\xi_k) = \frac{1}{(2\pi)^{3/2}|\bar{\boldsymbol{\Gamma}}_k|} exp(-\frac{1}{2}\xi_k^T \bar{\boldsymbol{\Gamma}}_k^{-1} \xi_k) \tag{5.4}$$

The goal is to estimate the occupancy likelihood of each cell in the grid which can be described by the function $\mathcal{L}_{ij}(\boldsymbol{m}_k)$. This function describes the likelihood of occupancy on grid cell (i,j) given measurement $\boldsymbol{m}_k$. Over time multiple measurements are given to the system and the likelihoods of these measurements are added to the current likelihood.

$$D(i,j) = \sum_{k=1}^{m} \mathcal{L}_{ij}(\boldsymbol{m}_k) \tag{5.5}$$

There are three different representations of the occupancy grid map, and the calculation of L_{ij} differs in the three.

1. **Cartesian Grid**
   In the Cartesian grid the world is mapped linearly to a grid of fixed dimensions giving a intuitive representation.

   $$\mathcal{L}_{ij}(\boldsymbol{m}_k) = \boldsymbol{G}_{\boldsymbol{m}k}(\boldsymbol{P}(\boldsymbol{p}_{ij} - \boldsymbol{m}_k)) \tag{5.6}$$

   in eq. 5.6 the likelihood is the multivariate Gaussian distribution dependent on the difference between measurement and projection. This representation is however computationally expensive as each measurement has a different likelihood. Updating every cell for every new measurement can be avoided by only updating cells that are are changed significantly.

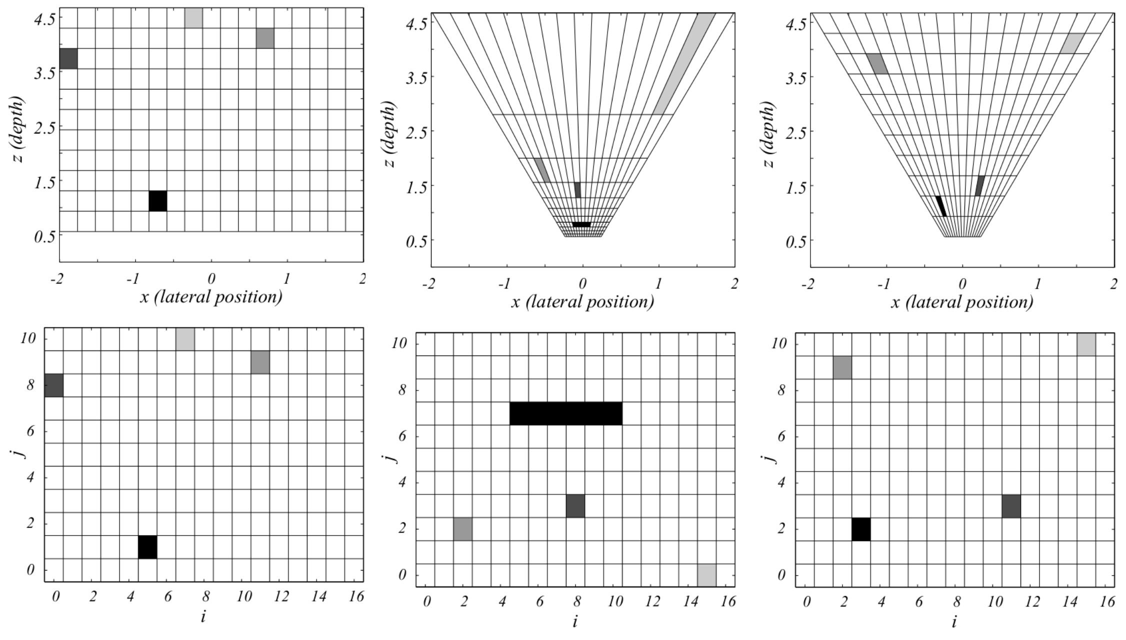
2. **Column/Disparity Map**
   In this variation the height component (v) in $\mathbf{m_k} = (u, v, d)^T$ is ignored and each cell in the

grid correspond to discretized values of (u,d). Each cell in the grid is therefore defined as $(i,j) = (u_{ij}, d_{ij})$. The likelihood function for cell (i,j) is shown in eq. 5.7.

$$\mathcal{L}_{ij}(\boldsymbol{m}_k) = \boldsymbol{G}_{\boldsymbol{m}k}((u_{ij} - u, 0, d_{ij} - d)^T) \tag{5.7}$$

3. **Polar Occupancy Grid**
   In the polar occupancy grid, the grid is made up of discretized values of (u, Z_w) where u is the image column and Z_w is the depth in 3D world coordinates. By doing this the grid maintains resolution to distant objects compared to a Column/Disparity map. This is due depth varying inversely proportional to with disparity in stereo triangulation [40]. A cell in the polar occupancy grid is defined as $(i,j) = (u_{ij}, z_{ij})$ yielding the likelihood function eq.5.8.

$$\mathcal{L}_{ij}(\boldsymbol{m}_k) = \boldsymbol{G}_{\boldsymbol{m}k}((u_{ij} - u, 0, d'_{ij} - d)^T), \quad \text{where} \quad d'_{ij} = \frac{f_u B}{z_{ij}} \tag{5.8}$$



(a) Cartesian **(b)** Column/Disparity **(c)** Polar

Figure 5.1: Variations of the occupancy grid
Source: [40]

Free Space and Height Segmentation

The Stixel Tesselation uses the disparity map generated by SGM to generate a stochastic polar occupancy grid which is described in section 5.2. In this map the image column is used to represent the angular part, and the disparity as the range part of the polar coordinate [35].

In order to detect obstacles they need to be separated from the area around the vehicle. This area is called *Free Space*, and is defined as the area around a vehicle where movement without any collision is guaranteed [40]. In other words, free space describes the area until the first obstacle in every direction is met. This can be seen in the polar occupancy grid by starting at the bottom and moving upwards until an occupied cell is encountered. Doing this for every column will create a line across the occupancy grid defining the free space. This line can also be found using Dynamic Programming (DP) [40]. The DP-algorithm tries to find the optimal path across the occupancy grid from left to right creating the same line as if every column was evaluated separately. The output of the DP-algorithm is a set of vector coordinates $(u, \hat{d}_u)$ representing the column of the image and the disparity respectively. These coordinates can be triangulated into 2D world coordinates (X_u, Z_u). These 2D points along with the origin of the camera defines the free space in front of the camera system.

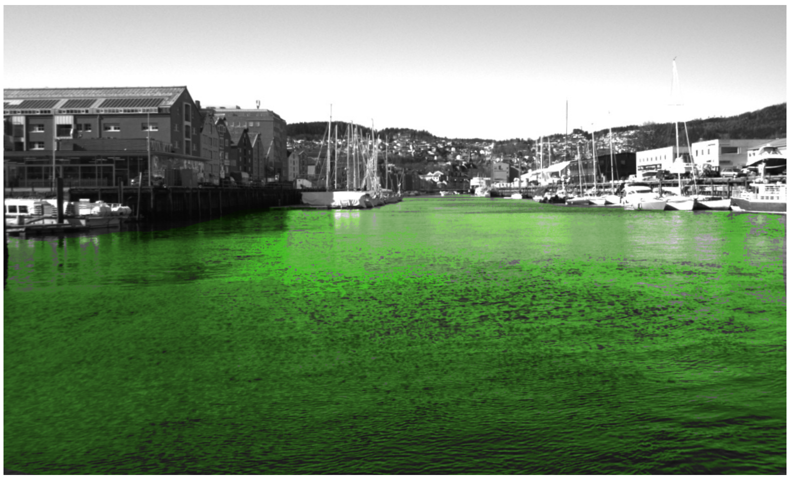


Figure 5.2: Illustation of freespace

Now as the distance and direction to the objects have been established in 2D, height segmentation can be conducted. As we now have the vector coordinates $(u, \hat{d}_u)$, and the 2D position (X_u, Z_u) of an object the task is to find out the upper boundary of the object located at (X_u, Z_u). To determine if a pixel is a part of the foreground or the background its disparity is compared to the expected disparity if there was no objects in the scene [35]. This is done by approximating a boolean membership of either foreground or not not foreground with eq.5.9.

$$M_{u,v}(d) = 2^{(1-(\frac{d-\hat{d}_u}{\Delta D_u})^2)} - 1 \tag{5.9}$$

In eq. 5.10 ΔD_u is a parameter, $\hat{d}_u$ is the disparity obtained from the free space calculation and $f_d(z)$ is the expected disparity based on the baseline (B) and focal length (f_u).

$$\Delta D_u = \hat{d}_u - f_d(Z_u - \Delta Z_u), \quad \text{where} \quad f_d(z) = \frac{B * f_u}{z} \tag{5.10}$$

A cost function based on the sum of the membership values is then created (eq.5.11). Because of the large difference in disparity in the transition between the top of an object and the background, the cost function yields a clear difference in membership in these areas.

$$C(u,v) = \sum_{i=0}^{i=v-1} M_{u,v}(d(u,i)) - \sum_{i=v_f}^{i=v} M_{u,v}(d(u,i)) \tag{5.11}$$

Having calculated both the base and top on all the objects the extraction of stixels is simple. The parameters for base point (v_B), and top point (v_T) along with a predefined width of each stixel span a frame for each stixel. As the occupancy grid is build of discrete values it will have a finite resolution which will affect the accuracy in depth. The disparities within the stixel frame produces by the SGM earlier is stored in a histogram. By using this information more accurate depth information can be obtained while simultaneously filter outliers and reduce noise [35]. Calculating depth for each stixel is done by first calculating Z_c as in eq.5.12 . Then calculating X_c as in eq.5.13, before calculating the euclidean distance to the stixel. As the average disparity is calculated for the whole stixel, the distance is only calculated in the cameras X/Z-plane.

$$Z_c = \frac{f_u B}{d} \tag{5.12}$$

$$X_c = \frac{x}{f_u} \cdot Z_c \tag{5.13}$$

$$distance = \sqrt{X_c^2 + Z_c^2} \tag{5.14}$$

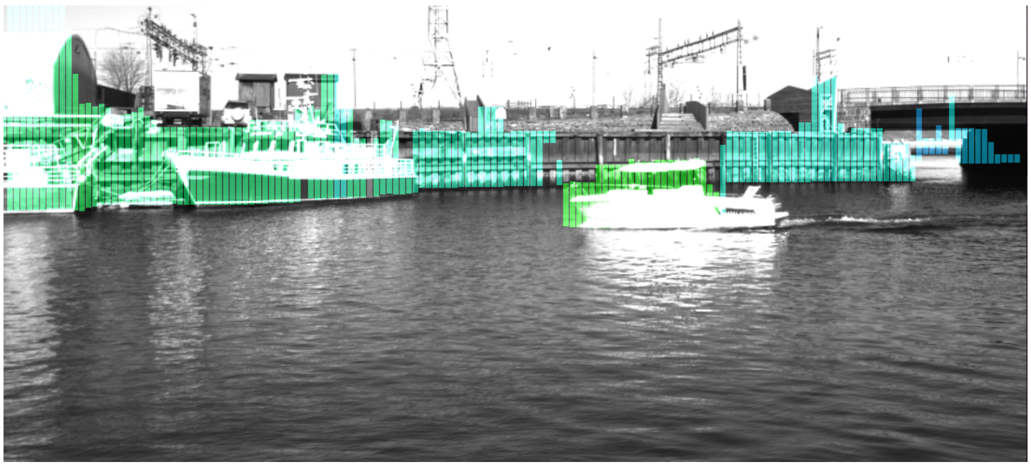


Figure 5.3: Stixel distribution from scenario 7

5.3 Method 2 - Digital Elevation Map

A digital elevation map is a method of representing the 3D world using either 2D or 3D sensors [36]. The map is then divided into DEM-cells resembling a Cartesian occupancy grid. This method is usually used to map terrain, but Oniga and Nedevschi [42] presented a method of separating a planar surface and detecting objects. By fitting a quadratic surface model using RANSAC the authors could use the density in the 3D points to classify DEM-cells as either road, traffic isle or obstacle.

The first step in this process is then to obtain the surface model. When using a coordinate system where (x,y,z) = (lateral, height, depth) an algebraic representation of the model can be shown as:

$$Y = -a \cdot X - a' \cdot x^2 - b \cdot Z - b' \cdot Z^2 - c \tag{5.15}$$

Fitting the model to n 3D-points involves minimizing an error function S. The function is a sum of squared errors where Y_i is the height of the 3D point and $\bar{Y}_i$ is the height of the surface model at point (X_i, Z_i). It is possible to minimize over the surface instead of just the Y-axis, but an increase is accuracy will only be noticeable in very curved terrain, but the computational complexity increases drastically [42].

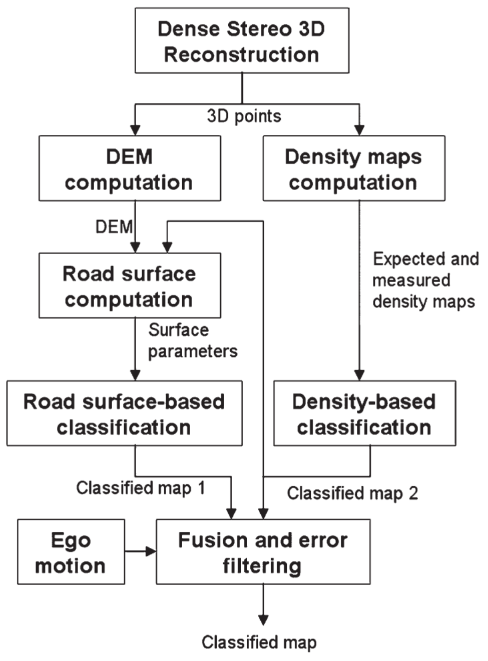

Dense Stereo 3D Reconstruction
3D points
DEM computation
Density maps computation
DEM
Road surface computation
Expected and measured density maps
Surface parameters
Road surface-based classification
Density-based classification
Classified map 1
Classified map 2
Ego motion
Fusion and error filtering
Classified map

Figure 5.4: Flow chart of DEM algorithm
Source: [42]

$$S = \sum_{i=1}^{n} (Y_i - \bar{Y}_i) \tag{5.16}$$

Inserting eq. 5.15 into eq. 5.16 results in eq. 5.17. Minimizing this function can be done by taking the derivative with respect to the unknowns (a, a', b, b' and c). This results in a set of five linear equations that can be solved with algebra.

$$S = \sum_{i=1}^{n} (Y_i + a \cdot X_i + a' \cdot X_i^2 + b \cdot Z_i + b' \cdot Z_i^2 + c)^2 \tag{5.17}$$

The DEM consists of a grid similar to the occupancy grid map with a predefined size for both the the size of the map and size of each cell in the map. The 3D points calculated from the disparity map will be located above a DEM-cell and only the highest point within each cell is stored. In addition to this, two density maps are calculated for each cell, *expected surface density map* and *average measured density map*. The expected surface density map is a map describing the 3D density of a cell if no obstacles are present. The number of actual 3D points provided by the disparity map is counted and stored for each cell. As the density of obstacles vertical to the surface provides more 3D points than the surface a comparison between the two maps is done to quickly distinguish between surface and non-surface features.

As the density in both objects and surface decays with distance two criteria are used to classify a cell as an object:

1. If the density is T_H times higher than the estimated surface density where T_H is a constant

based on maximum allowed slop in surface.

2. If a cell is adjacent to a cell classified as obstacle and has density higher than $\frac{T_H}{2}$ times the estimated surface density.

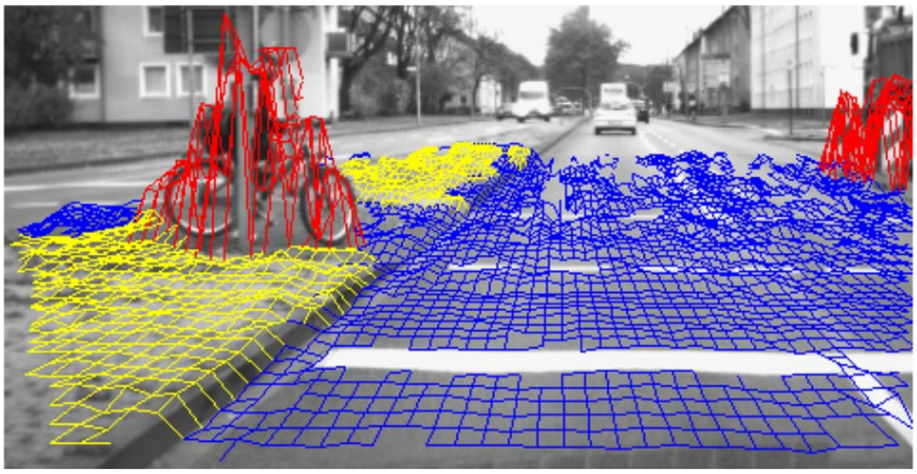


Figure 5.5: Output of DEM algorithm
Source: [42]

The density of points in the expected surface density map decrease with increased distance from the vehicle. From the experiments done in the original paper [42], the density was approximately $0.1\frac{pts}{cell}$ at 35m. Without knowledge about the ground surface, it is impossible to detect objects at a longer distance. Fig. 5.5 illustrates how the different areas in the image are classified. It is also worth noticing that the cars ahead in the image are not detected as they are out of the predefined range. The detection range can be increased, but it is obvious in fig.5.5 that the accuracy of the surface model decreases heavily at the end of the range.

5.4 Method 3 - Geometry-based Cluster

This is a rather generic category with multiple algorithms, but if a real time constraint is considered one of the methods presented in [34] is a good option. In this method a set of conditions is applied on a point pair ($\boldsymbol{p_1}$ and $\boldsymbol{p_2}$), and if these conditions are satisfied, both points are considered *compatible* and are labeled as *obstacle points*.

1. $H_\gamma < |v_2 - v_1| < H_{max}$ (The difference in height is between a given interval)
2. $\frac{|v_2 - v_1|}{|\boldsymbol{p_2} - \boldsymbol{p_1}|} > \cos(\theta_\gamma)$ (The line joining the points creates an angle in the horizontal plane.)

To determine all obstacle points you can use a brute force approach which is testing all possible point pairs, resulting in N^2 tests for N points. A more efficient method is possible when realizing the two conditions create a double truncated cone with p_1 in the centre.

The double truncated cone can then be projected onto the image plane as a double truncated triangle centered in pixel $\boldsymbol{p}$. A simple detection algorithm is then presented as:

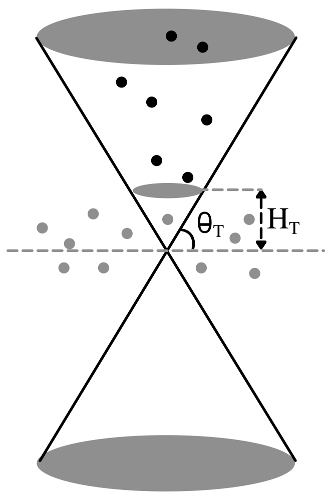


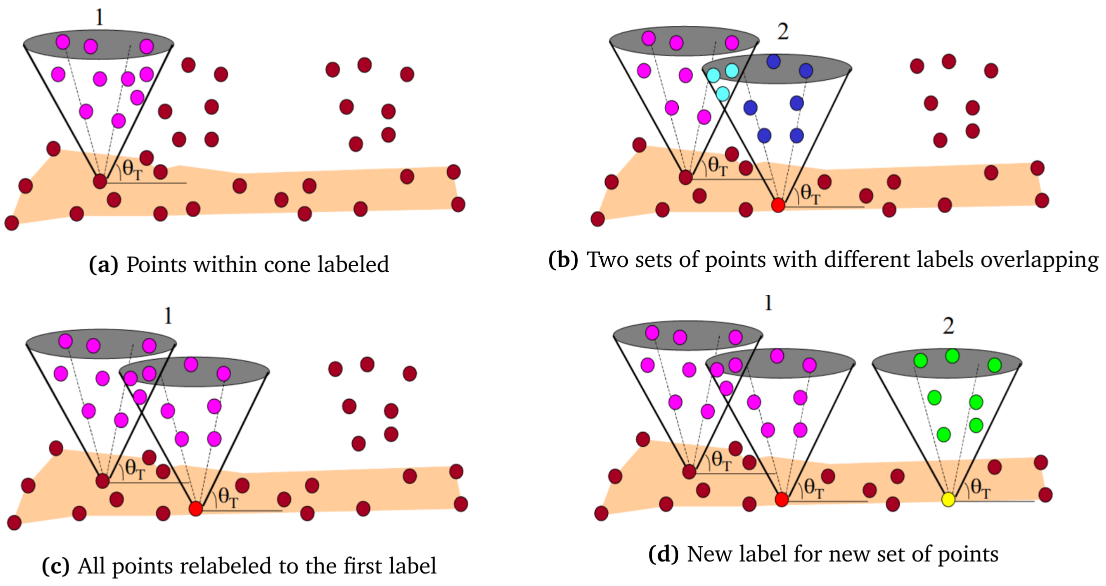
Figure 5.6: Double truncated cone

1. For every pixel $\boldsymbol{p}$ determine a set I_p containing all the pixels in the truncated triangles.
2. Scan the set for pixels *compatible* with $\boldsymbol{p}$. If such a pixel is found, classify $\boldsymbol{p}$ as obstacle. If not, $\boldsymbol{p}$ is not an obstacle.

Further optimization can be done to avoid testing the same point pairs twice, but the concept remains the same. The next task is to segment the different obstacle points into different objects.

In order to segment different clusters of points into objects we can express the points as an undirected graph. This graph is constructed by applying a property saying if $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ belong to the same object, and $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ belong to the same object, all three points belong to the same object. This means that an object can be defined as the maximal connected subgraph of the point graph [34].

To distinguish between objects the points classified as obstacle within a cone is labelled. If two or more cones are sharing points, all points within the these cones are given the same label.



(a) Points within cone labeled

(b) Two sets of points with different labels overlapping

(c) All points relabeled to the first label

(d) New label for new set of points

Figure 5.7: Segmentation process 3-D points classified as obstacle
Source: [34]

5.5 Method 4 - Direct Planar Hypothesis Testing

The method presented in [37] is a joint detection and localization algorithm. The method differs from the ones presented above by not assuming a global free space plane. Instead it performs a patch-wise binary classification of either free-space or obstacle. This is formulated as a statistical hypothesis testing problem where free-space is represented by the null-hypothesis $\mathcal{H}_f$ and obstacles correspond to $\mathcal{H}_o$. Each local plane is defined by a vector $\boldsymbol{\theta} = (n_x, n_Y, n_Z, d)^T$ where $\boldsymbol{n}$ is the normal vector of the plane and d is the distance from origin. Each local plane is allowed to vary within certain parameters ϕ_f and ϕ_o which is illustrated in fig.5.8. The axis in fig.5.8 are parallel with the camera coordinate system. By tuning these constrains the method can be optimized for the expected terrain and obstacles to be detected.

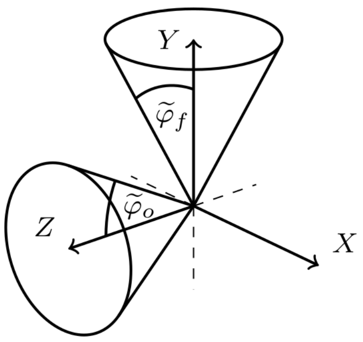


Figure 5.8: Cones that visualize allowed deviations in local plane and obstacle
Source: [37]

The authors argue that a adding processing steps in the algorithm will cause loss in performance. More specifically, they are aiming at the calculation of a disparity map. Where most other algorithms needs to have an accurate and often dense disparity map as an input to the algorithm this method operates directly on the image data.

To evaluate if an image patch is free-space ($\mathcal{H}_f$), or obstacle ($\mathcal{H}_o$) a Generalized Likelihood Ratio Test is performed. This test replaces the unknown parameters $\boldsymbol{\theta}_i$ in every hypothesis $\mathcal{H}_i = \mathcal{H}_{f,o}$ with their Maximum Likelihood Estimates $\hat{\boldsymbol{\theta}}_i$. This test is performed instead of the likelihood ratio test as it requires knowledge about the full probability function of the data model in the hypothesis.

$$\mathcal{L}(\mathcal{I}) = \frac{p(\mathcal{I};\hat{\boldsymbol{\theta}}_o,\mathcal{H}_o)}{p(\mathcal{I};\hat{\boldsymbol{\theta}}_f,\mathcal{H}_f)} > \gamma \tag{5.18}$$

In eq. 5.5 $\mathcal{I}$ is a data vector from the stereo image pair, γ is the decision threshold and $\mathcal{L}(\mathcal{I})$ is the likelihood ratio. The left and right image patch values, $I_l(\boldsymbol{p})$ and $I_r(\boldsymbol{p})$ are considered as noisy samples of the continuous image signal g at position $\boldsymbol{p}$.

$$I_l(\boldsymbol{p}) = g(\boldsymbol{p}) + \alpha_l(\boldsymbol{p}) + \eta(\boldsymbol{p}) \tag{5.19}$$

$$I_r(W(\boldsymbol{p},\boldsymbol{\theta})) = g(\boldsymbol{p}) + \alpha_l(\boldsymbol{p}) + \eta(\boldsymbol{p}) \tag{5.20}$$

The terms $\alpha(\boldsymbol{p})$ and $\eta(\boldsymbol{p})$ in eq.5.19 and eq.5.5 represents the local intensity bias and noise respectively. $W(\boldsymbol{p},\boldsymbol{\theta})$ is the warp of coordinates from the left to the right image. This transformation is described in detail in section 2.2.

$$\boldsymbol{H} = \boldsymbol{K}(\boldsymbol{R} - \frac{1}{d}\boldsymbol{t}\boldsymbol{n}^T)\boldsymbol{K}^{-1} \tag{5.21}$$

$$ln(p(\mathcal{I};\hat{\boldsymbol{\theta}}_i,\mathcal{H}_i)) = \sum_{\boldsymbol{p}\in\Omega} C_1 - C_2 \cdot \rho(I_r(W(\boldsymbol{p},\boldsymbol{\theta})) - g(\boldsymbol{p})) \tag{5.22}$$

$$\hat{\boldsymbol{\theta}}_i = \arg\min_{\boldsymbol{\theta}_i}(-ln(p(\mathcal{I};\hat{\boldsymbol{\theta}}_i,\mathcal{H}_i))) \qquad s.t \quad |\phi_i| \leq \tilde{\phi}_i \tag{5.23}$$

5.6 Method 5 - Euclidean Clustering

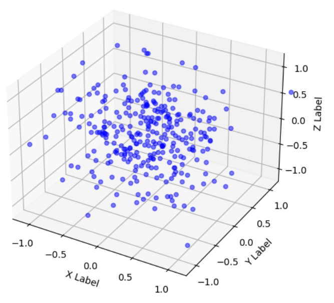
Euclidean clustering is not a object detection method by it self, but in the combination of a series of filtering, accumulation and clustering relevant objects can be extracted from noisy point clouds. The method was tested on milliAmpere by Olsen and Theimann in their master's thesis [5].

Cut-off Filtering

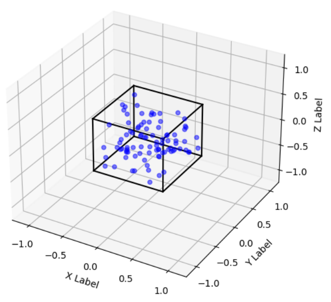
The first step is to reduce the amount of information to process by simply setting minimum and maximum ranges in 3D space for points to be considered. This is done by applying a pass-through or cut-off filter. These ranges can be set to exclude points that are obvious noise by being out of the field of view of the cameras, or for instance at such heights and depths that are not relevant for the application.

Statistical Outlier Removal

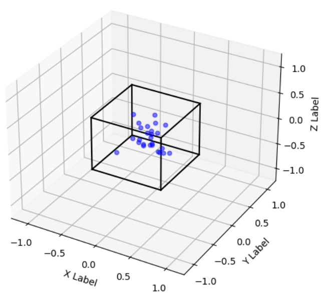
The next filter is based on the fact that areas of interest in a point cloud has a higher density than areas with only noise. Assuming this density to be Gaussian the mean distance from one point to its neighbors can be calculated. A limit of maximum mean distance for a point to its neighbors can then be set, and all points with a higher mean distance will be removed. By doing this dense areas in a point cloud will remain dense, and sparse areas will become even sparser.



(a) Original point cloud



(b) Point cloud after cut-off filtering



(c) Point cloud after SOR-filtering

Voxel Grid

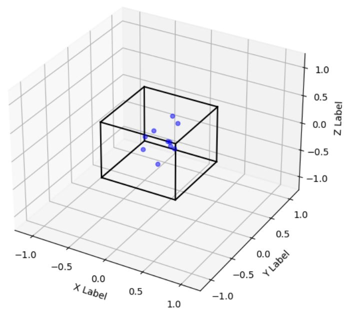
The point cloud can now be down sampled to reduce computational complexity. This is done by applying a Voxel Grid filter. This filter acts as the 3D space is filled with rectangular boxes and all points within the box is approximated to their centroid point.

Accumulation

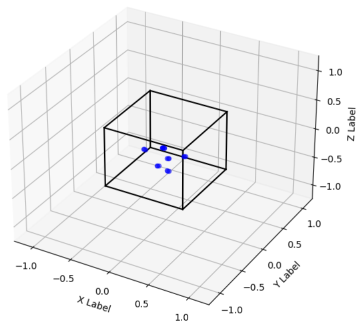
Accumulation is done to increase robustness. By combining point clouds from a number of consecutive point clouds the result is a denser point cloud is areas of interest while the remaining noise stays sparse. This also has the benefit of maintaining a good point cloud even if the input cloud should be poor for a couple of instances.

Euclidean Clustering

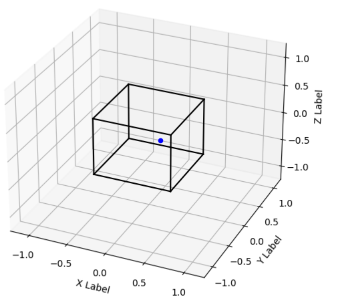
In the final step the now filtered and accumulated point cloud can be clustered into single points that are easier to track.



(a) Point cloud after Voxel Grid down sampling



(b) Point cloud after accumulation



(c) Point cloud after after clustering

5.7 Evaluation of Detection Methods

As one of goals of this thesis is to determine viable options regarding object detection using stereo vision on milliAmpere it is important to evaluate the the methods presented against how well they will fit for the intended use, namely medium to long range object detection and depth estimation in an urban marine environment, and the criteria set in the beginning of this chapter.

The stixel method presented in section 5.2 is one of the methods that depend on the calculation of free-space. As the sea level within the Trondheim canal is relatively flat, the Stixel method is well suited for use on milliAmpere. However, how waves caused by other surface vehicles affect the stability and robustness in the free space calculation is unknown. The height segmentation used in the stixel method, presented in section 5.2, only allows for one object per stixel. This means that if a taller vehicle is present behind a lower one they will not be detected as two separate objects. There are specializations of the stixel method that allows for this type of separation, that could be investigated if the stixel method is to be used further in the development of milliAmpere, but will not presented in this thesis as it is considered more of an optimization of a working method than proving a that a method can work.

The DEM method presented in section 5.3, does not rely on a planar surface model, but the quadratic model is more computationally expensive to obtain, and probably not necessary at flat sea level. As only the highest point in each DEM-cell is stored, the accuracy of this method is determined on the size of the DEM-cell. This technique will make the algorithm compact, but how complete it is depends on the size of the cells. Making the DEM-cells too small will preserve more information, but result in a considerable increase in computational time. Another disadvantage for the method is that as its performance quickly decays with distance. As an important aspect of milliAmperes perception system is to detect other vehicles at medium to long distances as quickly as possible, the DEM-method is not considered a good alternative for milliAmpere.

The geometry based cluster method presented in section 5.4, is the oldest of the methods considered in this study. The algorithm is quite simple but suffers from the same segmentation problem as the stixel method where objects too close to each other will be considered as the same object. Possible range is hard to evaluate, but no apparent restrictions can be seen. The computational complexity is low, but the results presented in [34] shows some weaknesses regarding missed objects. In the case of milliAmpere robust detection is crucial, and the geometry based cluster will there for not be investigated any further.

Direct Planar Hypothesis Testing is the newest, and perhaps the method that differs the most from the previous ones. It does not rely on any assumptions of a ground plane nor a disparity map. The results presented by Pinggera et. al [37] are impressive, with good detection and depth estimation for long distances. However, few other resources than the original paper is available, and therefore little information about how well the algorithm performs in other scenarios.

Euclidean clustering has been proven to work [5], but it has the disadvantage of losing accuracy and being incomplete, as the many steps in the filtering and clustering process results in a single point that can be located anywhere in the initial cluster of points generated by the target. This point may also move within the target from frame to frame making it less consistent than other methods. This method, as many of the other will perform poorer at longer ranges as at longer ranges the initial point cloud will contain fewer points and therefore be more likely to be considered as noise by the filtering process. Nonetheless, it has excellent properties regarding noise filtering. Combining this with the accumulation results in what should be a very robust method.

Method	Range	Real time	FG.ass
Stixel	GOOD	YES	PLANAR
DEM	POOR	YES	QUADRATIC
Geometry based	GOOD	YES	PLANAR
PHT	EXCELLENT	YES	NONE
Euclid.Cluster	GOOD	YES	NONE

Table 5.1: Summary of selected OD method

After evaluating the five methods in regards to how well it translates to milliAmpere and fulfills the four criteria, the stixel method, Euclidean cluster and DPHT appears to be a good alternatives. With more available resources, the DPHT method could have been a viable option. But as only the original paper is available on the matter, and the source code is not publicly available as it was developed in Daimler Research Lab, this method is not investigated any further. The stixel method is well documented through a number of well documented experiments done by others. It offers real time computation with a satisfying accuracy for distance estimation and detection. These properties makes it suitable for use on milliAmpere, and will therefore be used further in the thesis.

The Euclidean cluster also seems as a good alternative due to its proved results on point clouds generated with the SAD-method [6], and using SGM should only improve on these results. Its robustness also is an appealing factor as detection of surrounding vehicles is very important. This method was therefore chosen alongside the stixel method to be tested on real data from milliAmpere.

Chapter 6

System Description

In this chapter the stereo system used in this thesis will be presented. This includes the different hardware components such as cameras and lenses, choices made regarding mechanical set up, software solutions and processing pipeline. The system is created for use on the autonomous ferry, milliAmpere in the Trondheim canal for detection and estimation of objects in front of the ferry.

6.1 Hardware

The setup consist of two identical cameras and lenses. The cameras, Blackfly S GigE are well suited as they have a global shutter sensor. This means that all pixels are activated simultaneously as opposed to the more common rolling shutter sensor where pixels are activated sequentially. A global shutter sensor is important in this application as information in an instant is to be compared to another image in the same instant. With a a rolling shutter moving objects can be distorted or generally not representing the real world in that given instant. Each camera is fitted with a fixed focal length lens from Edmund Optics.

(a) Blackfly S GigE camera
Source: Blackfly S gigE documentation [a]

[a]https://www.flir.com/products/blackfly-s-gige/

(b) Edmund Optics lens
Source: Lens documentation [a]

[a]https://www.edmundoptics.com/p/85mm-c-series-fixed-focal-length-lens/14947/

Blackfly S GigE specifications	
Frame rate	24
Resolution	2448x2048
Pixel size	$3.45m$
Readout method	Global shutter
Communication	GigE
Operating Temp.	$0°C - 50°C$

Table 6.1: Camera specifications

Edmund Optics C-series lens	
Focal length	8.5mm
Type	Fixed focal length

Table 6.2: Lens specifications

Camera Setup

As mentioned in section 2.4 the physical setup regarding baseline and camera angle affects the accuracy, how close objects can be seen, and how wide the field of view of the complete system is. To keep the complexity of the system as low as possible the two cameras are mounted on a aluminium bar only shifted in the x-direction. To utilize the benefits from having a large baseline it was set to 1.75 meters, as this is the largest practically possible on milliAmpere. However, having a large baseline generates an area in front of the cameras where the images do not overlap. This distance can be calculated using eq. 6.1 where **B** is the baseline, and θ is the horizontal field of view.

$$distance = \tan\left(\frac{180 - \Theta}{2}\right) * \frac{B}{2} \tag{6.1}$$

According the the specifications $\Theta = 59.1°$ and with $\mathbf{B} = 1.75$m the distance to where the images start overlapping is 1.54m.

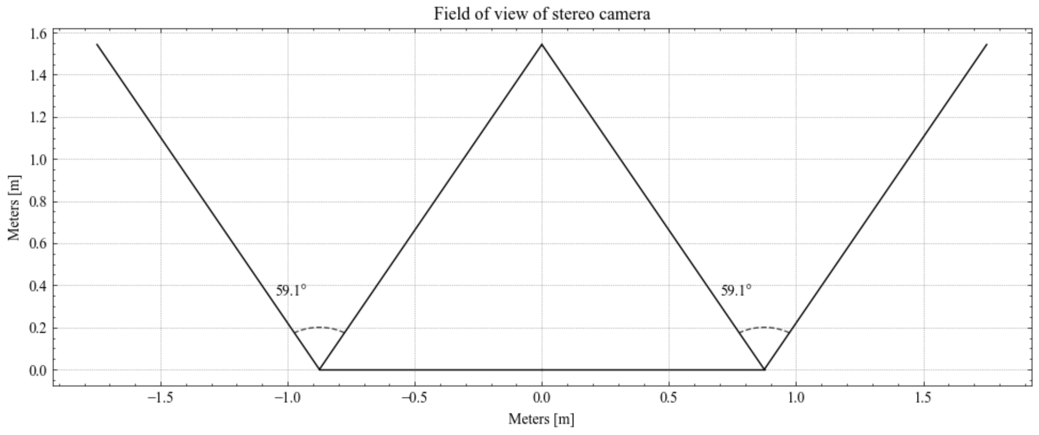


Figure 6.2: Point of intersection FoV

To increase the overlapping field of view and increase accuracy the cameras where angled slightly inwards. This also has the benefit of creating a fixation point. This is the point where the optical axis' cross, and can help to separate objects at this distance. This fixation point can be calculated as in eq. 6.2 where WD is the working distance, B is baseline, and β is inward rotation around the Y-axis.

$$WD = \frac{B}{2} * \tan(90^\circ - \beta) \tag{6.2}$$

Using the same baseline of 1.75m and rotation of 1° inward each, the fixation point is 51.56 meters ahead of the cameras which is a suitable and desirable operating distance for the system.

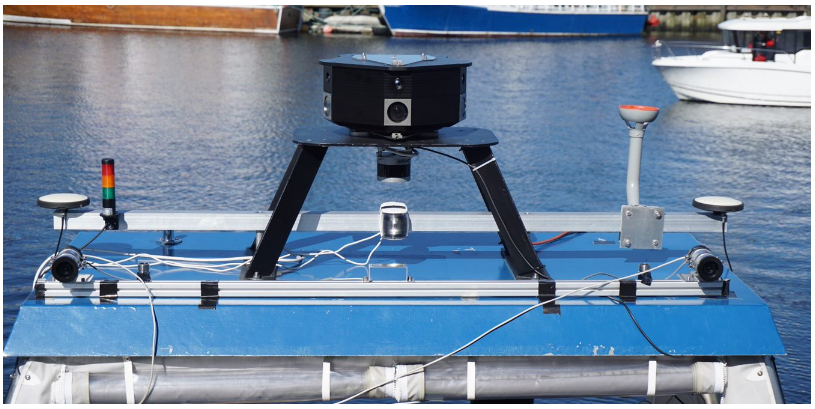


Figure 6.3: Sensors mounted on milliAmpere

Communication

The cameras are connected to a *Power over Ethernet* (PoE) switch. Another ethernet cable is then connected to a computer controlling the cameras. In stereo vision it is cruical that the images are taken simultaneously such that the two images are looking at the scene at the exact same time. Because of this the cameras are arranged in a master/slave-setup, where the left camera is the master and the right camera is hardware triggered by the left camera with a synchronization cable between them.

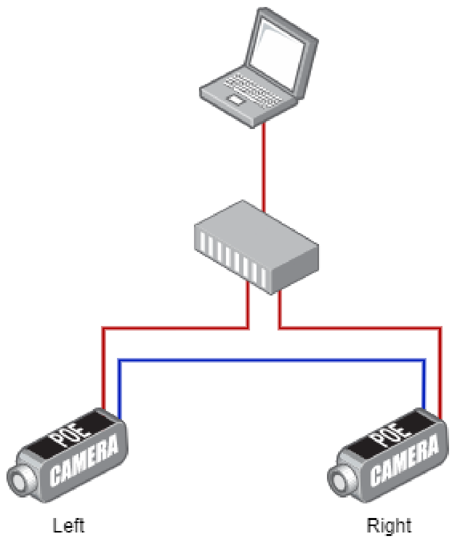


Figure 6.4: Communication between units in the system

6.2 Software

The main components of the software being used will be explained in this section to get a better understanding of how it all is tied together. The milliAmpere ferry is run on the Ubuntu 16.04 LTS operating system using Robot Operating System (ROS) Kinetic as its main control system.

ROS - Robot Operating System

ROS is an open-source framework with tools and libraries for developing software for robotics. It was developed as an attempt to standardize and modularize robotic software as it quickly becomes complex [43]. It has five philosophical goals

- Peer-to-peer
- Tool-based
- Multi-lingual
- Thin
- Free and Open-Source

A ROS system consists of many separate processes doing their designated tasks. These process are connected in a peer-to-peer topology with processes dependent on each other connected directly. This is done to avoid unnecessary traffic flow to and from central servers. Messages sent between nodes in a ROS-network is of standardized small text files. This means that a number of programming languages are supported, and languages can therefore be chosen based on personal preferences or task suitability.

As software development in robotics quickly can become complex, code has a tendency to become entangled in projects making reusability hard. ROS encourages development of drivers and algorithms to be in standalone libraries that are independent of ROS. By placing all the complexity in libraries outside of ROS makes it easier to follow the *thin*-ideology.

In this project the code for the specific methods are written as ROS-nodes. These nodes subscribes and publishes ROS-messages to *topics*. By connecting the nodes together results in a flow of information being processed on its way. This flow of information can be seen in the explanation of some of the most central nodes in this project below.

Open Source Computer Vision Library - OpenCV

OpenCV is an open source software library focusing on computer vision and machine learning. The library contains over 2500 algorithms covering both traditional and state-of-the-art computer vision algorithms. It has interfaces for C++, Python, Jaca and MATLAB and supports Windows, Linux, Android and MacOS. In this project OpenCV is used in numerous instances such as resizing, disparity map generation, image visualization and camera calibration.

Point Cloud Library - PCL

PCL is a C++ library containing algorithms for manipulation of point clouds. This includes, filtering, feature estimation, surface reconstruction, segmentation and more [44]. In this project it is heavily used in the Euclidean cluster method explained in section 5.6.

Camera Driver

The images are acquired using a camera driver that supports the our cameras provided in ROS [1]. Using this driver one can customize the image acquisition process by determining frame rate, colors, shutter time etc. The driver then publishes two topics for each camera `image_raw`, and `camera_info`, where all the calibration parameters are stored. Configuration for multiple camera setup in possible making in possible to start multiple cameras at once.

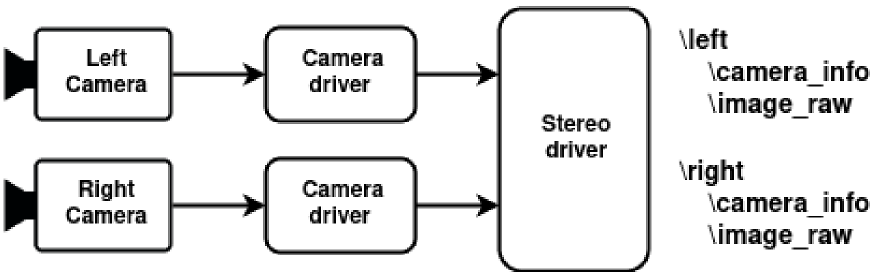


Figure 6.5: Flowchart for camera driver

ee

Stereo Package

The next step in the image pipeline is to transform the raw images provided by the camera driver to useful information. To do this the ROS-package, stereo_image_proc [2] is used. This package uses the topics provided by the camera driver and publishes a new set of topics including rectified images, disparity maps and point clouds. Functionality for generating disparity maps using both the SAD and SGM method is supported. The point clouds are generated by applying eq. 3.3 to every pixel in the disparity image.

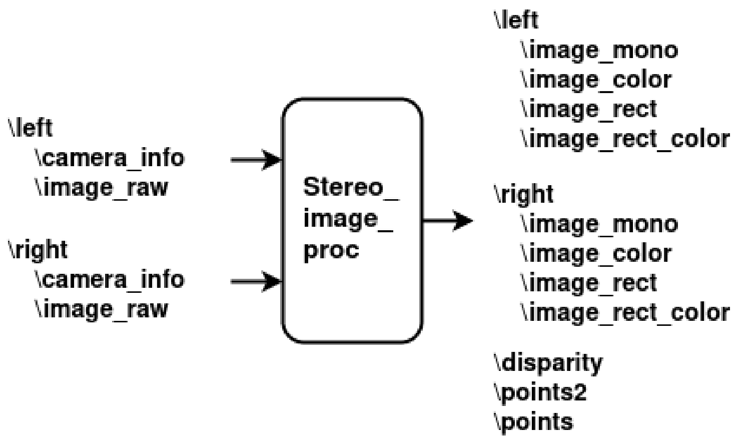


Figure 6.6: Flowchart for *stereo_image_proc*

[1]http://wiki.ros.org/spinnaker_sdk_camera_driver
[2]http://wiki.ros.org/stereo_image_proc

Stixel Package

As it is a goal to have the system being able to run in real time, and be integrated with milliAmpere, it is built with ROS integration in every step of the process. The stixel package is a ROS package written in c++, specifically for this project. Base code for the stixel implementation [3] was used, and reworked to be ROS compatible, and suitable for this project.

Using topics from the `stereo_image_proc` package as input a separate package for resizing the image topics to suitable dimensions was written. This works as an intermediate step before using them in the stixel world package. In the resizing step a lot of the sky is removed from the images as this does not provide any information but adds to the computational complexity. The images are resized from $(1224, 1024)px$ →(1224,600)px.

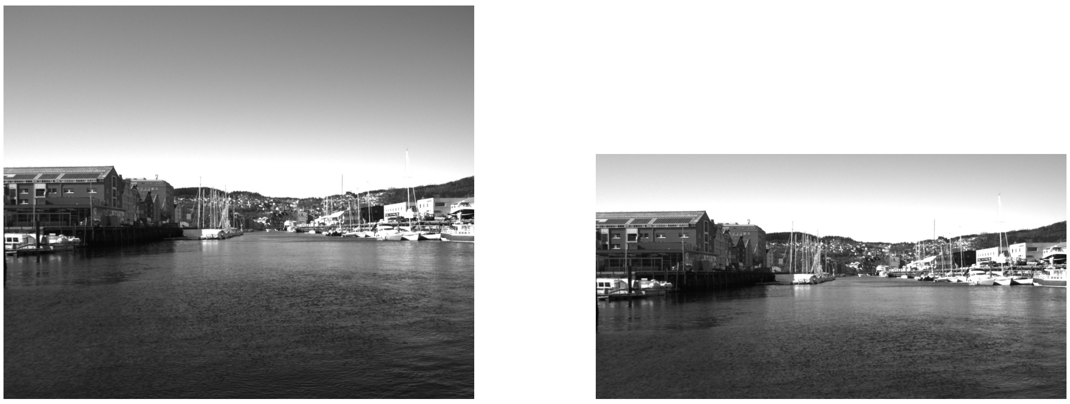


Figure 6.7: Image before and after resizing

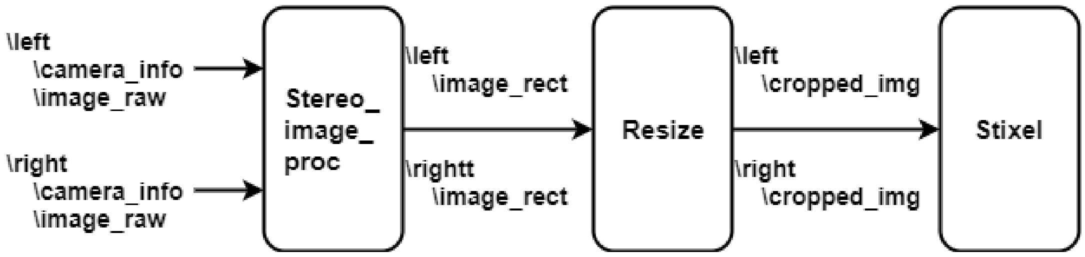


Figure 6.8: Complete flow chart for stixel implementation

Clustering Package

The clustering package is also a ROS package written for this project. it that uses the `points2` topic from the `stereo_image_proc` package as input. It transforms the point cloud from a ROS message to a datatype recognized by the PCL library mentioned in section 6.2. The package then handles all the filtering, accumulation and clustering explained in section 5.6. It publishes point clouds from all the steps in the process for easier evaluation of each filter.

[3]https://github.com/gishi523/stixel-world

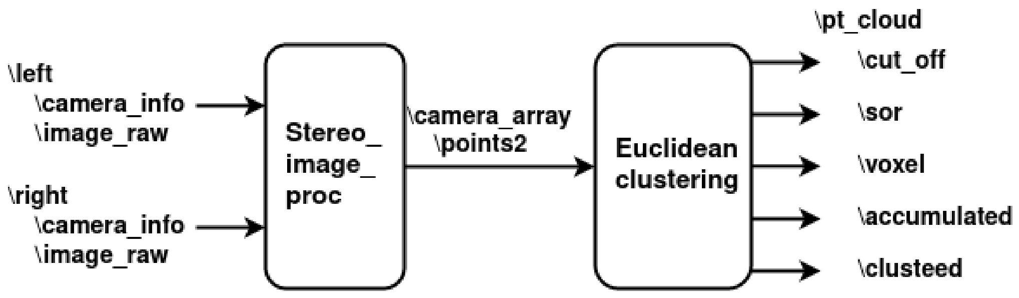
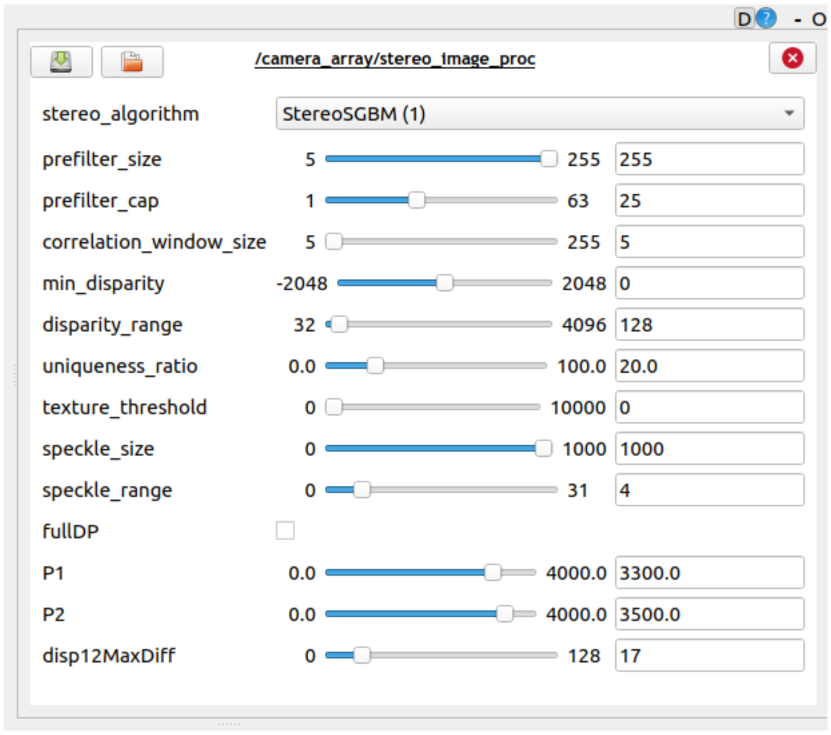


Figure 6.9: Complete flow chart for Euclidean cluster implementation

Disparity Tuning

When creating the disparity map there are several parameters that needs to be tuned in order to produce the best result. These factors can be the penalty terms P1 and P2 described in section 3.4 about SGM, the number of disparities, the size of the search window etc. To do this in real time while the algorithms were running a ROS tool called `rqt_reconfigure` was used. When running the ROS stereo package, `stereo_image_proc`, `rqt_reconfigure` has access to these variables. In this environment all the parameters can be tuned and saved in .yaml files for use later.


/camera_array/stereo_image_proc

stereo_algorithm	StereoSGBM (1)		
prefilter_size	5	255	255
prefilter_cap	1	63	25
correlation_window_size	5	255	5
min_disparity	-2048	2048	0
disparity_range	32	4096	128
uniqueness_ratio	0.0	100.0	20.0
texture_threshold	0	10000	0
speckle_size	0	1000	1000
speckle_range	0	31	4
fullDP	☐		
P1	0.0	4000.0	3300.0
P2	0.0	4000.0	3500.0
disp12MaxDiff	0	128	17

Figure 6.10: Tuning parameters in `rqt_reconfigure`

Chapter 7

Experiments

In order to test the system on the hardware in the operating environment a series of experiments were conducted. Different scenarios targeted to test different aspects with the system were designed. An overview of the trajectories for all five scenarios can be seen in appendix B. The main goal of the experiments was:

1. Evaluate the robustness of the object detection algorithm. Determining how well the algorithm detects vehicles and other objects and how well the detection is kept in a dynamic environment
2. Evaluate the accuracy of the depth estimations. Comparing depth estimations with ground truth data
3. Evaluate run time of the system

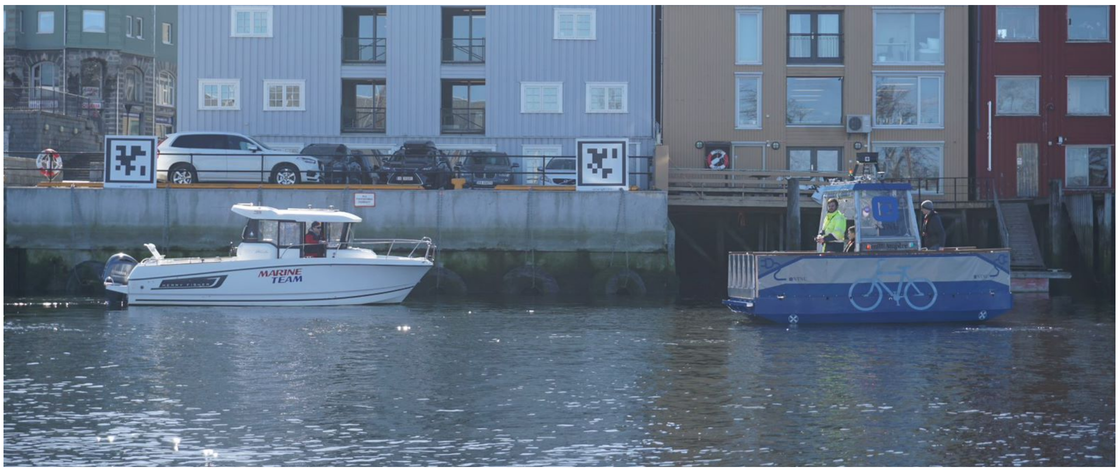


Figure 7.1: Havfruen and milliAmpere

7.1 Ground truth

A target boat *Havfruen* was used as a target for the system. It is a 7 meter long leisure boat representing the average boat in the area. The experiments were designed to figure out how far away the system is able to detect a boat of this size, and how well it performs in regards of maintaining detection at different ranges, angles and speeds.

The target boat was equipped with a *Spatial Dual* positioning system from Advanded Navigation.[1] This is a GPS aided inertial navigation system that through fusing of a inertial measurement unit (IMU), pressure sensor, and a dual antenna RTK GNSS receiver can provide accurate estimates regarding position, velocity, acceleration and orientation. Accurate positional estimates of the target boat can be used as ground truth by comparing the distance between the target boat and milliAmpere to the depth estimates provided by the stereo vision system.

(a) Dual antennas mounted on the roof of Havfruen

(b) Spatial Dual control unit
Source: Spatian Dual Documentation[1]

The distance between milliAmpere and Havfruen was calculated using this Haversine formula. This formula calculated the shortest distance between two coordinates on earths surface modelling the earth as a great circle. In the equations below, (eq.7.1) ϕ is latitude, ρ is longitude, and R is earths radius.

$$\begin{aligned} a =& \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos\phi_1 \cdot cos\phi_2 \cdot \sin^2\left(\frac{\rho\phi}{2}\right) \\ c =& 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{(1-a)}) \\ distance =& R \cdot c \end{aligned} \tag{7.1}$$

(7.2)

According to the documentation the Spatial Dual unit as a positional accuracy of 1.2m in the configuration used during these experiments. However, an average lower positional error of 0.23m was recorded during the experiments.

[1]https://www.advancednavigation.com/products/spatial-dual

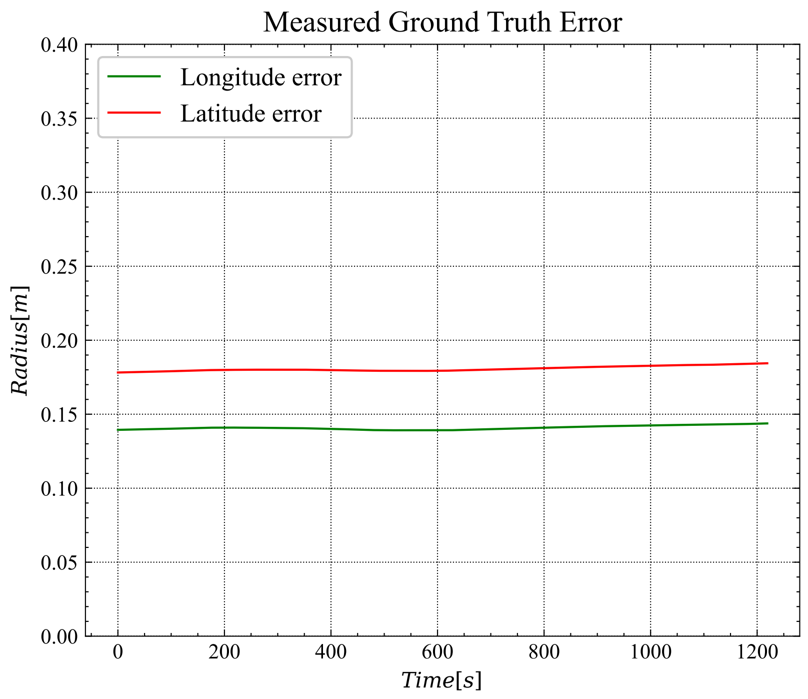


Figure 7.3: Positional Error in Ground Truth

As the shape of the boat is not spherical, the distance from the GNSS antenna to the sides of the boat is not equal for all sides of the boat. The stereo vision camera calculates the distance to the surface of an object, so this offset is taken into account for each scenario as a constant as the boat is mostly seen from the same angle through out the scenario.

As the cameras and lidar detect the surface of the boat, and the antennas were placed in the middle of the boat there will be a a constant offset between ground truth. This offset will vary regarding if the boat is seen from the side, frond and rear. The data will be adjusted for each scenario accordingly.

MilliAmperes on board computer (OBC) measures its position via a GNSS-antenna which has its own reference frame on milliAampere. As the stereo camera is temporarily mounted on milliAmpere it has its own local reference frame located at the center for the left camera. In order to avoid a constant error between the ground truth measurements and stereo measurements the measurements from the stereo vision system needs to be translated to the GPS reference frame. The translation vector, $\boldsymbol{t}_c^{GPS} = [1.86, 0, -0.29]$ translates the coordinates systems to a common origin point. In this thesis only distance is estimated and not orientation or position in a static world coordinate system. Therefore rotating the camera frame to a NED-configuration is unnecessary.

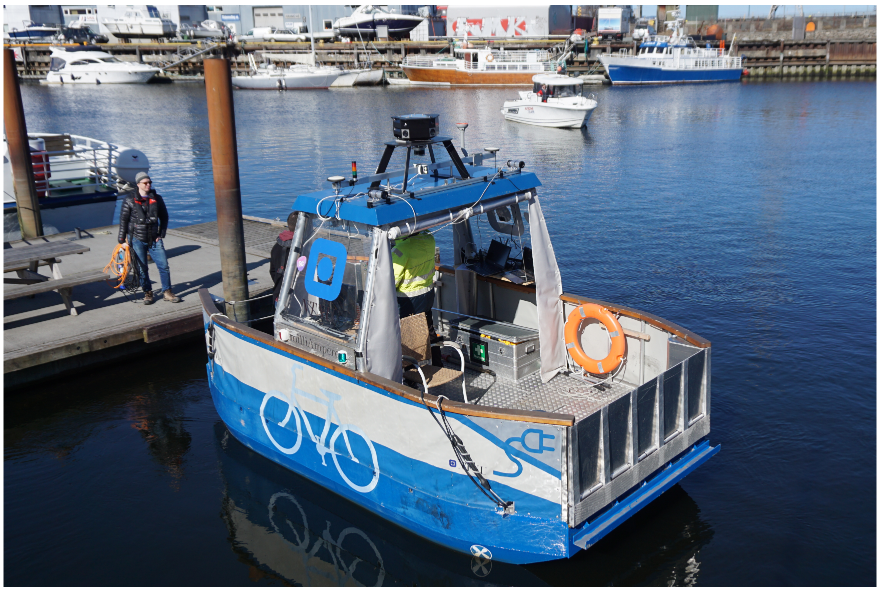


Figure 7.4: MilliAmpere during testing

Light Detection and Ranging - Lidar

As the main alternative sensor to a stereo system is a lidar, measurements from a lidar sensor has also been included for comparison. The Lidar is an active sensor as it emits laser pulses and measures the time it takes from the pulse being sent until it returns to the sensor. Using this principle for distance measurements is done in several different sensors such as ultra sound and randar and is called *Time of Flight*-sensors (TOF). For light emitting sensors the distance can be expresses as in eq. 7.3, where c is the speed of light and Δt is the time of flight.

$$distance = \frac{c \cdot \Delta t}{2} \tag{7.3}$$

The lidar mounted on milliAmpere is a VLP 16, and its most important specifications can be found in table 7.1.

Figure 7.5: Velodyne VP16 Lidar
Source: VP16 documentation [2]

Considering the center of the lidar to be the origin of a local coordinate system points can be described in spherical coordinates where r is the radial distance, ψ is the azimuth, and η is the elevation

Range	0-100m
Horizontal FoV	360°
Vertical FoV	30°
Accuracy	$\pm 3cm$
RPM	300-1200
Resolution	0.1°- 0.4°

Table 7.1: Technical data for VPL 16

angle. These spherical coordinates can be transformed to Euclidean space by applying eq. 7.4.

$$\boldsymbol{P}_{lidar} = \begin{bmatrix} X_{lidar} \\ Y_{lidar} \\ Z_{lidar} \end{bmatrix} = \begin{bmatrix} r\cos(\eta)\sin(\psi) \\ r\cos(\eta)\cos(\psi) \\ r\sin(\eta) \end{bmatrix} \tag{7.4}$$

Having obtained P_lidar, the distance to the point can be calculated using the euclidean distance, as shown in eq.7.5.

$$Distance_lidar = \sqrt{X_{lidar}^2 + Y_{lidar}^2 + Z_{lidar}^2} \tag{7.5}$$

The same Euclidean clustering method as described in section 5.6 is applied to the point cloud produced by the lidar. A separate ROS-node was crated for the lidar making it easy to run lidar and stereo in parallel to compare results.

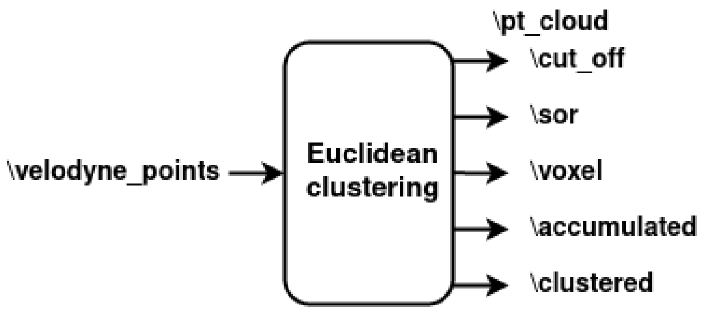


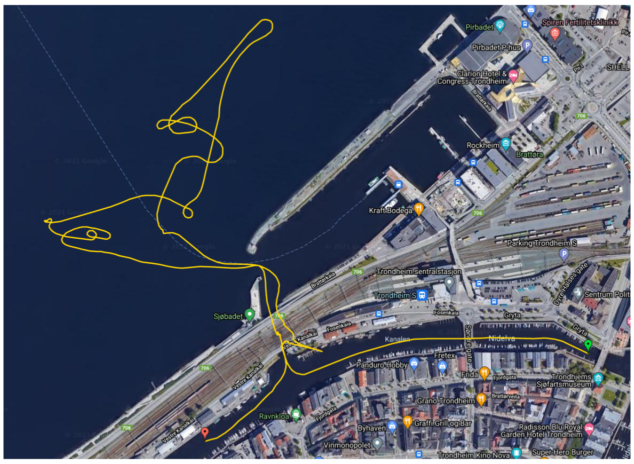
Figure 7.6: Flowchart of ROS-node for lidar

As for the stereo cameras a translation vector transforming the lidar reference frame into the GPS reference frame is applied to the points before distance calculations are done. The vector $\boldsymbol{t}_{lidar}^{GPS} = [0, 0.93, 0]$ ensures that the two reference frames have coinciding origins.

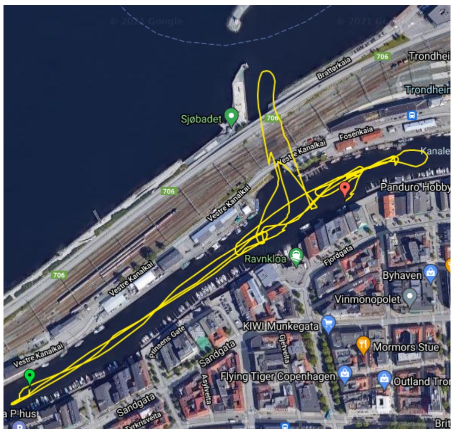
Chapter 8

Results

Data collection was conducted both within the Trondheim canal and in the open sea just north of the Trondheim canal. The results are based on five different scenarios representing different normal encounters with other boats for milliAmpere. An overview of these scenarios can be seen in appendix-B. In fig. 8.1 the ground truth of the target boat is shown. This ground truth includes five different scenarios testing different aspects of the system. In the scenarios the target boat will range from 10 meter to several hundred meters.



(a) Ground truth data, open sea



(b) Ground truth data, harbour

Figure 8.1: Overview of area covered during testing

8.1 Calibration Results

Prior to the data collection a stereo camera calibration was performed. Unfortunately when analyzing the calibration results after the data collection it became obvious that the results were far from optimal with a mean reprojection error ($\Delta p_x > 2$). Although it is not optimal, it was considered necessary to perform a new calibration after the equipment was transferred back to campus. This is not optimal as calibration should generally be done prior to use, and at the cite where the system is operating. Possible sources of error is that there might have been small changes in the camera setup during transport. However, the camera rig was transported as one unit, and the lighting conditions were similar to when the data collection was conducted.

Using a checkerboard and the `MATLAB Stereo Camera Calibrator App` multiple images were taken at various ranges and angles. In fig. 8.2 and fig. 8.3 the detection of checkerboard and images after rectification is shown.

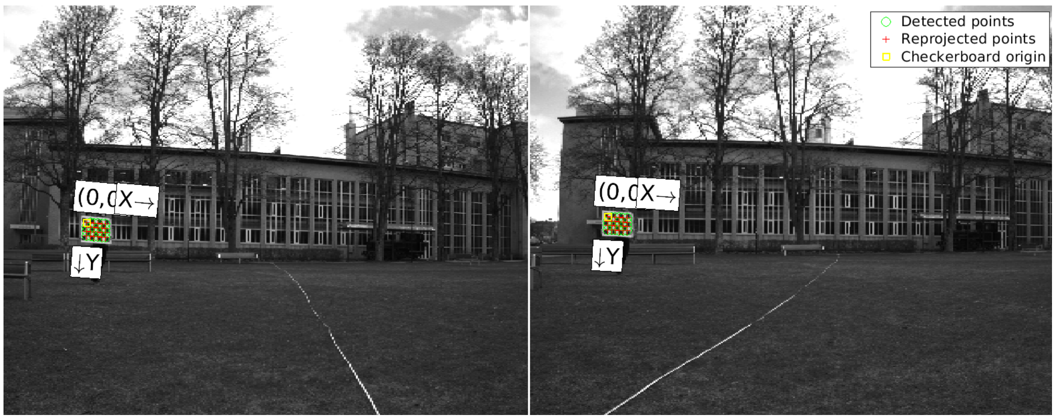


Figure 8.2: Corners of checkerboard detected

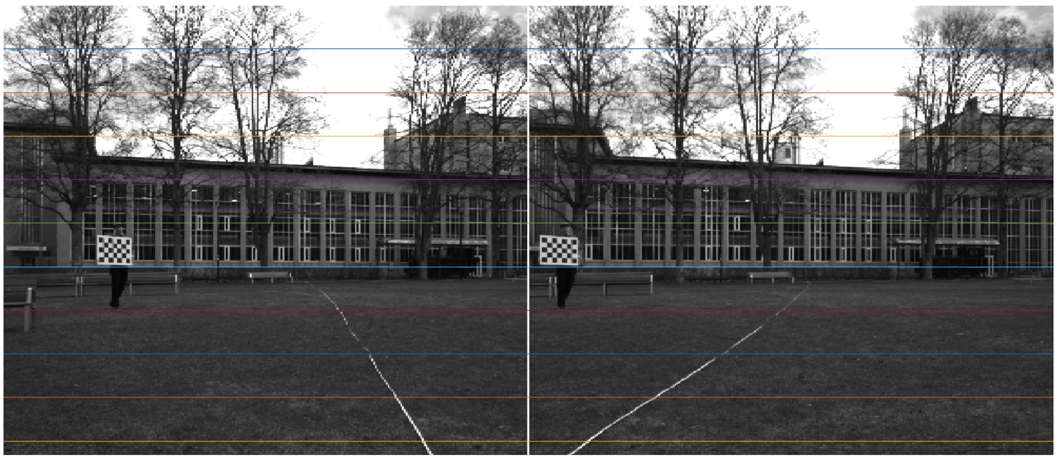


Figure 8.3: Rectified image with horizontal epipolar lines

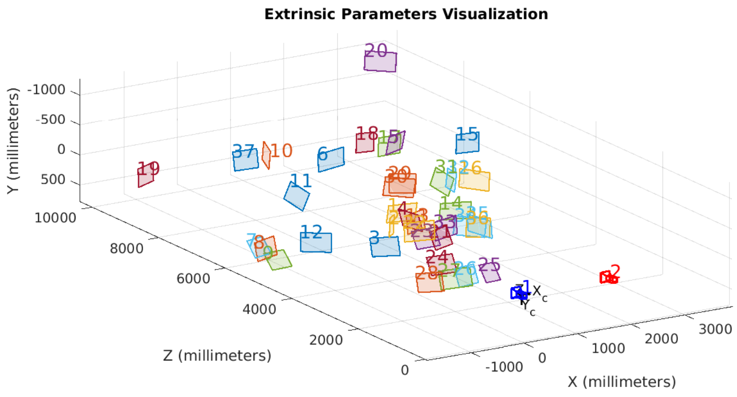


Figure 8.4: Checkerboard positions during calibration

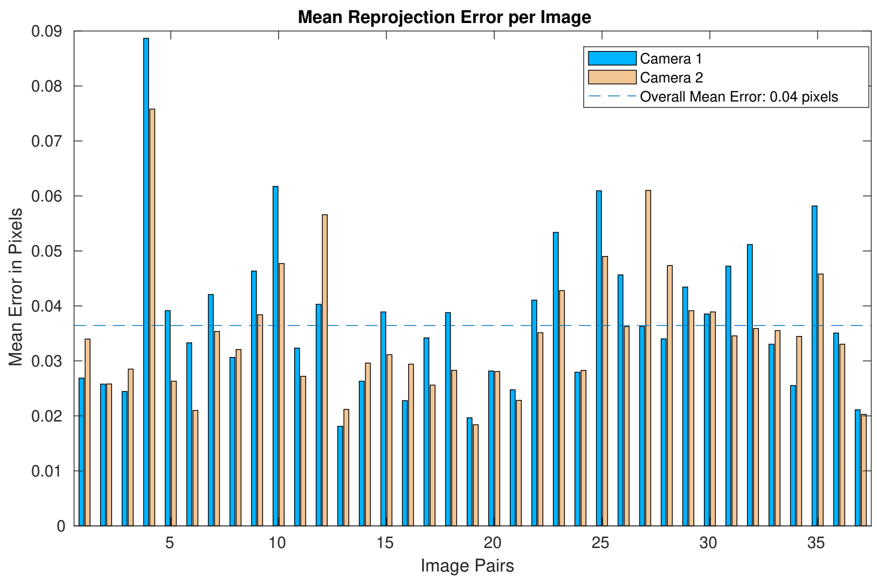


Figure 8.5: Mean reprojection error for every image

Extrinsic Parameters					
Translation [mm]			Rotation [deg]		
X	Y	Z	X	Y	Z
-1740.235 ± 0.114	-9.591 ± 0.028	87.338 ± 0.597	0.688 ± 0.001	5.712 ± 0.001	0.733 ± 0.001

Table 8.1: Extrinsic parameters from camera calibration

Intrinsic Parameters						
	Left Camera			Right Camera		
f_u	1236.1239 ± 0.2641			1236.7399 ± 0.2676		
f_v	1235.4177 ± 0.2636			1236.9865 ± 0.2662		
u_0	620.1205 ± 0.6002			642.7828 ± 0.5872		
v_0	534.8395 ± 0.1968			530.3827 ± 0.1881		
Rad.dist.	-0.398	0.234	-0.113	-0.400	0.243	-0.131
Tan.dist.	-0.0002		-0.0005	-0.0003		0.0005

Table 8.2: Intrinsic parameters from camera calibration

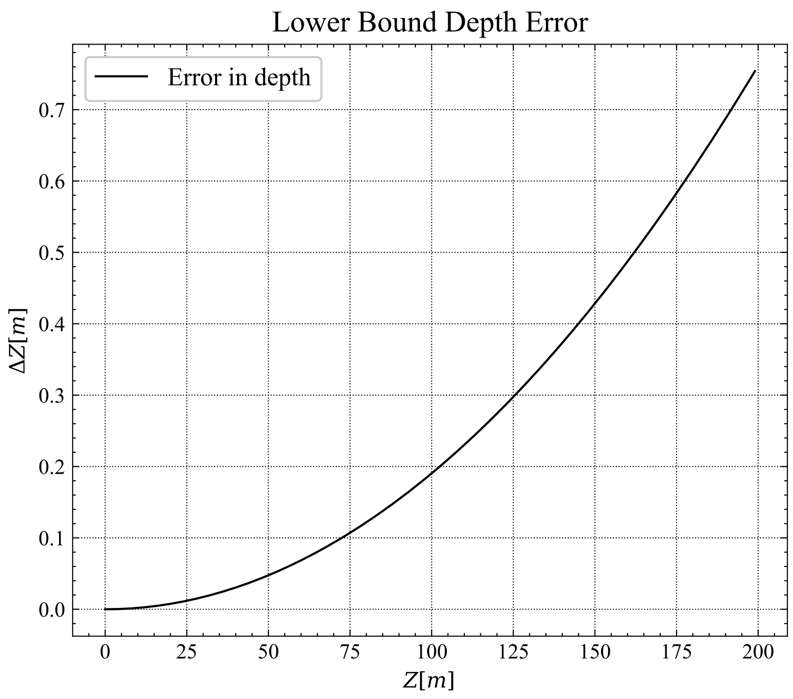


Figure 8.6: Lower bound for depth error as a result of the calibration results

8.2 Distance Estimation

The main evaluation criteria of the system performance used in this thesis is distance estimation. Meaning how well do the different methods estimate the distance to surrounding objects in the image frame when encountered in different scenarios.

Scenario 1

In scenario 1 Havfruen starts close to milliAmpere, then drives away. The purpose of this scenario was to get an understanding of how the system performs with little surrounding objects, how far the methods are able to detect Havfruen, and how reliable the detection is at the different distances.

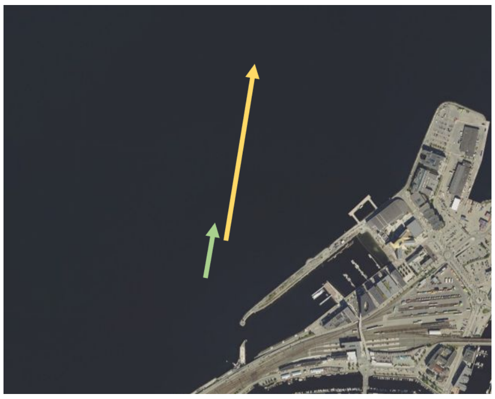


Figure 8.7: Illustration of scenario 1

FPS	20
Image resolution	1224 × 1024
Color/grayscale	grayscale

In fig. 8.8 the two stereo methods, the lidar method, and the ground truth is plotted. It shows a linear increase in distance between milliAmpere and Havfruen. in fig.8.9 the error of the three methods are shown, both in regards to the distance between milliAmpere and Havfuren, but also in regards to the distance between them.

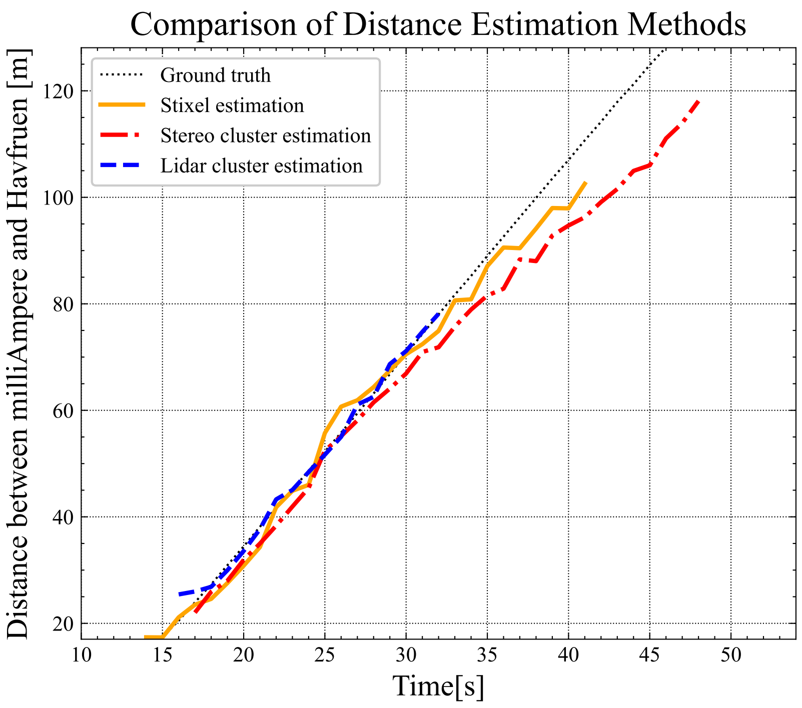
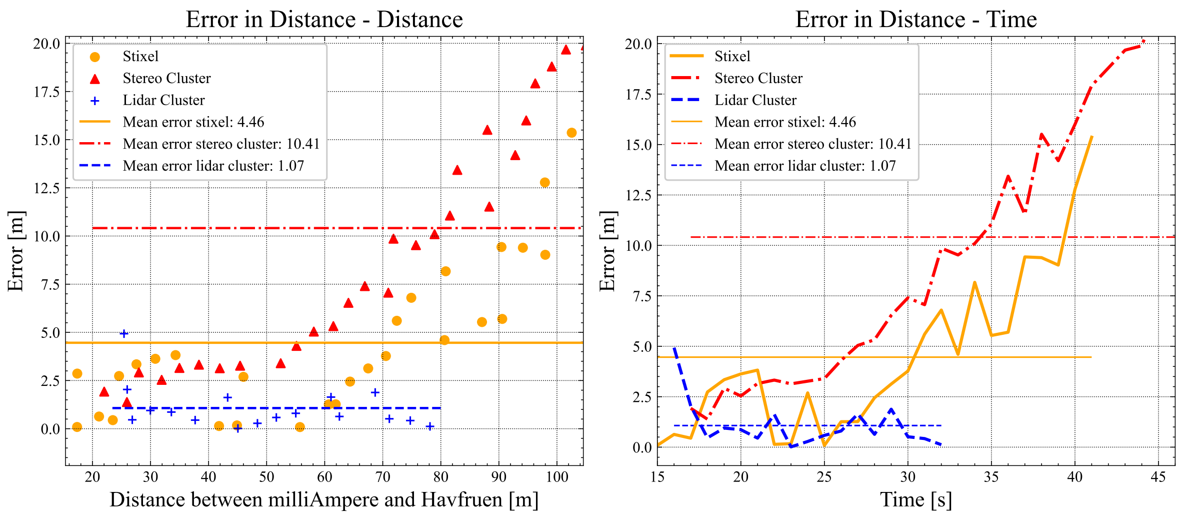


Figure 8.8: Distance estimation for scenario 1



(a) Error in regards to distance **(b)** Error in regards to time

Figure 8.9: Distance error - Scenario 1

Scenario 2

In scenario 2 Havfruen is approaching milliAmpere from ahead. As the scenario is taken within the canal there are a lot of surrounding buildings and boats that clutter the image compared to scenario 1. The purpose of this scenario is to see how well the different methods are in separating objects in a cluttered scene and also estimate the distance to an approaching object.

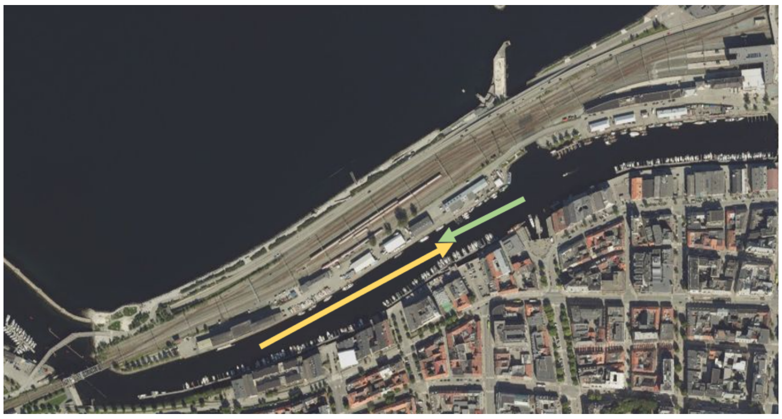


Figure 8.10: Illustration of scenario 2

FPS	20
Image resolution	1224 × 1024
Color/grayscale	grayscale

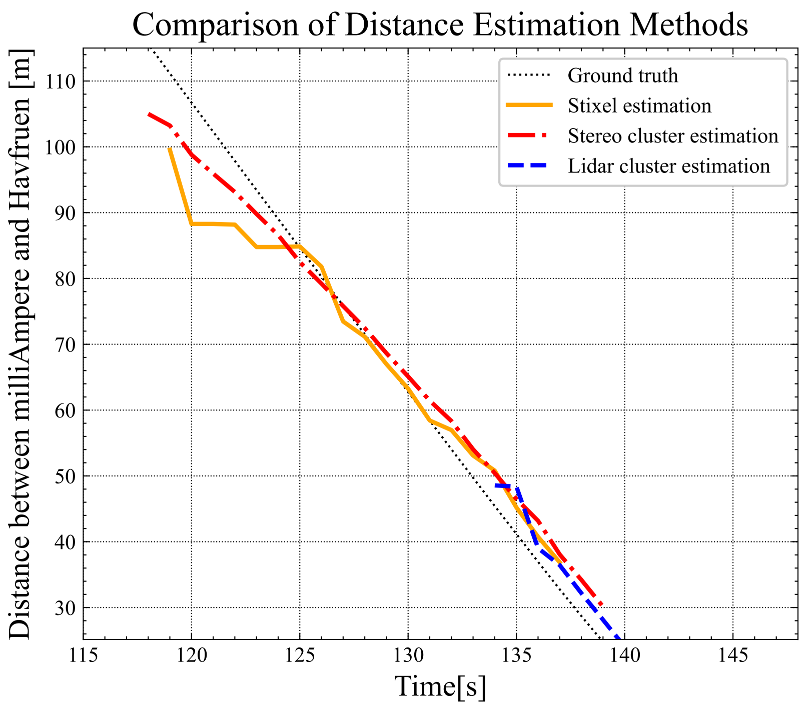
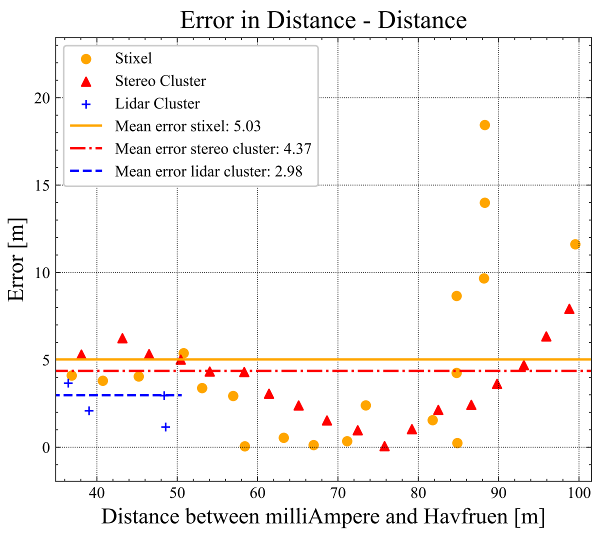
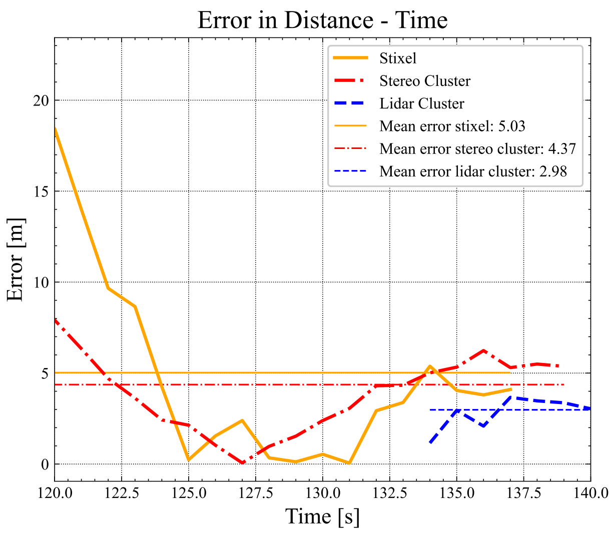


Figure 8.11: Distance estimation for scenario 2



(a) Error in regards to distance



(b) Error in regards to time

Figure 8.12: Distance error - Scenario 2

Scenario 3

In scenario 3 milliAmpere is stationary and facing the ship tunnel. Havfruen exits the shiptunnel before turning starboard and crossing infront of milliAmpere. The purpose of this scenario is to see how well the methods detect and estimate the distance to an object emerging from a dark spot in the images. In fig. 8.14 there is a large error in ground truth in the beginning. This is due to the ground truth being obtained by GNSS, and the signal is lost while under the bridge. The error estimation in fig. 8.15 also suffers from this error.

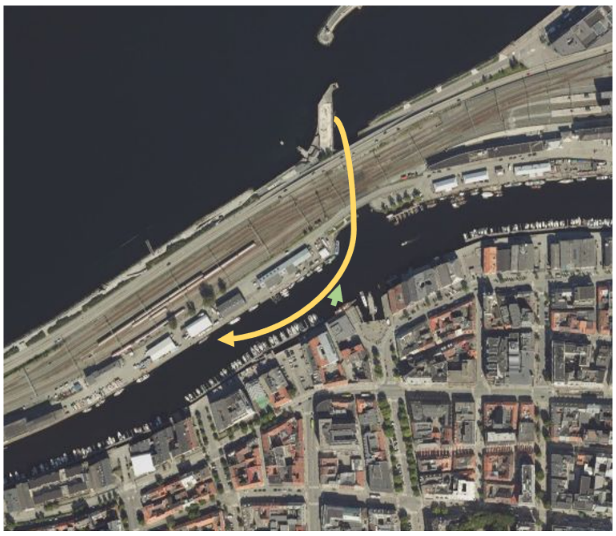


Figure 8.13: Illustration of scenario 3

FPS	20
Image resolution	1224 × 1024
Color/grayscale	grayscale

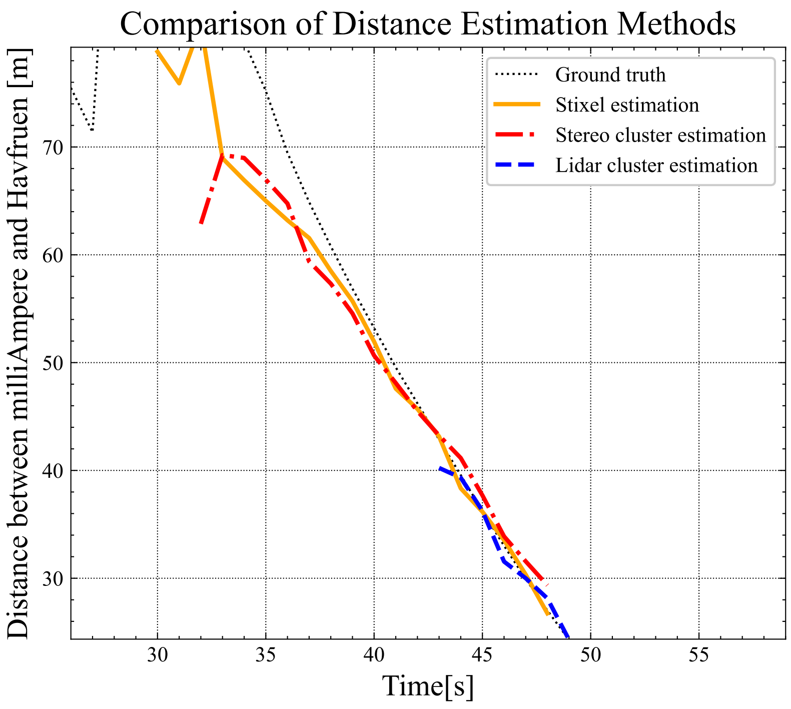


Figure 8.14: Distance estimation for scenario 3

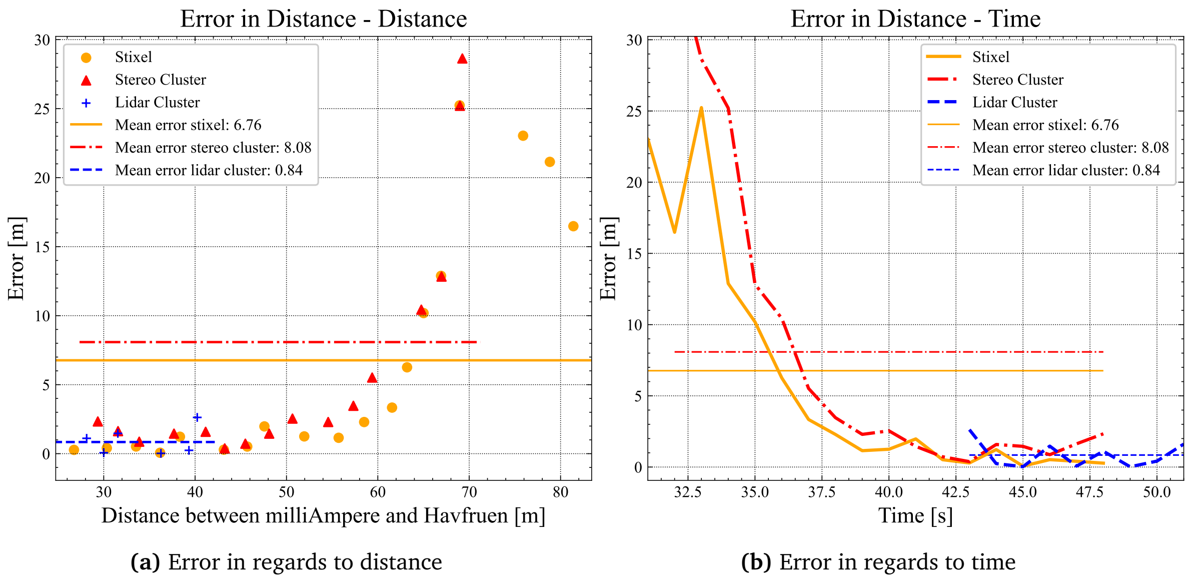


Figure 8.15: Distance error - Scenario 4

Scenario 4

In scenario 4 Havfruen overtakes milliAmpere on its starboard side. It continues in a straight line before turning and heading straight towards milliAmpere. In this scenario the system is tested on how it handles change in direction of a target.

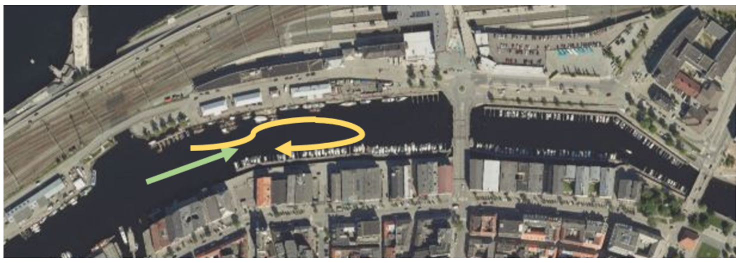


Figure 8.16: Illustration of scenario 4

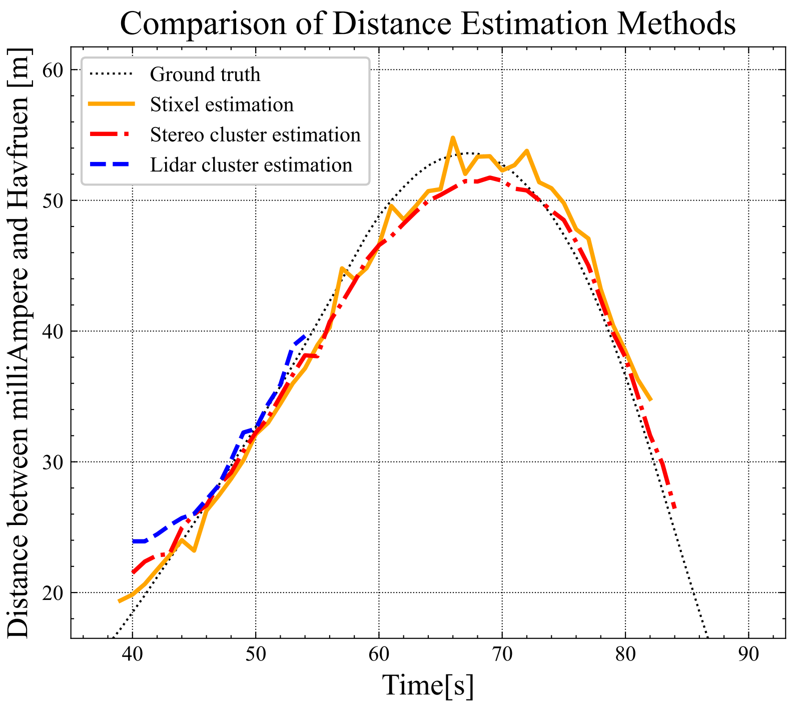
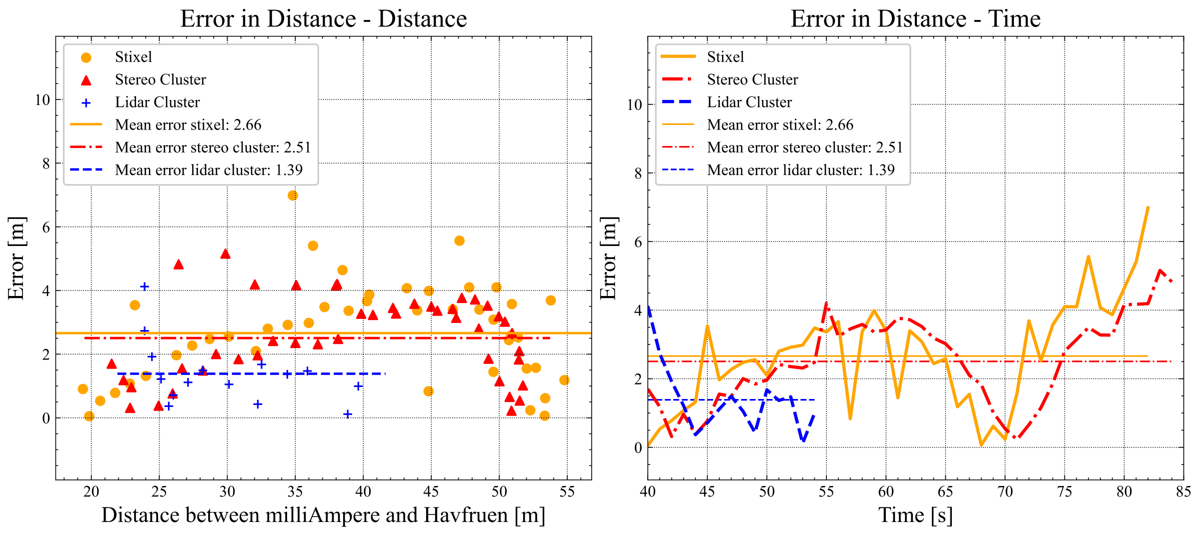


Figure 8.17: Distance estimation for scenario 4



(a) Error in regards to distance **(b)** Error in regards to time

Figure 8.18: Distance error - Scenario 4

Scenario 5

In scenario 5 Havfruen crosses in front of milliAmpere from right to left at a constant speed while milliAmpere is stationary at Ravnkloa facing Vestre Kanalhavn.

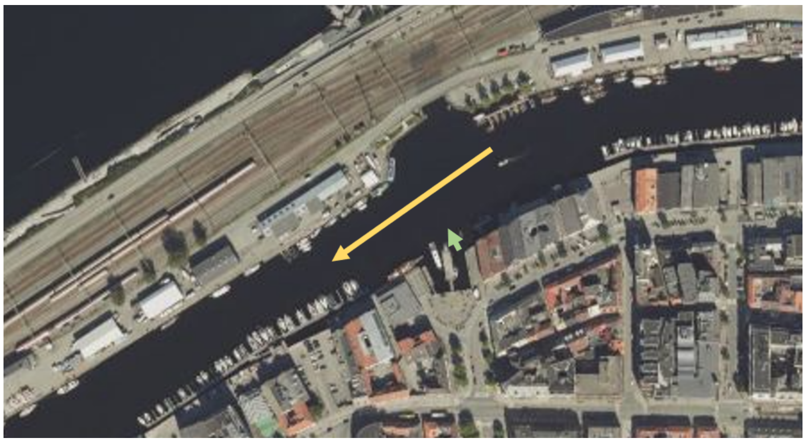


Figure 8.19: Illustration of scenario 5

FPS	20
Image resolution	1224 × 1024
Color/grayscale	grayscale

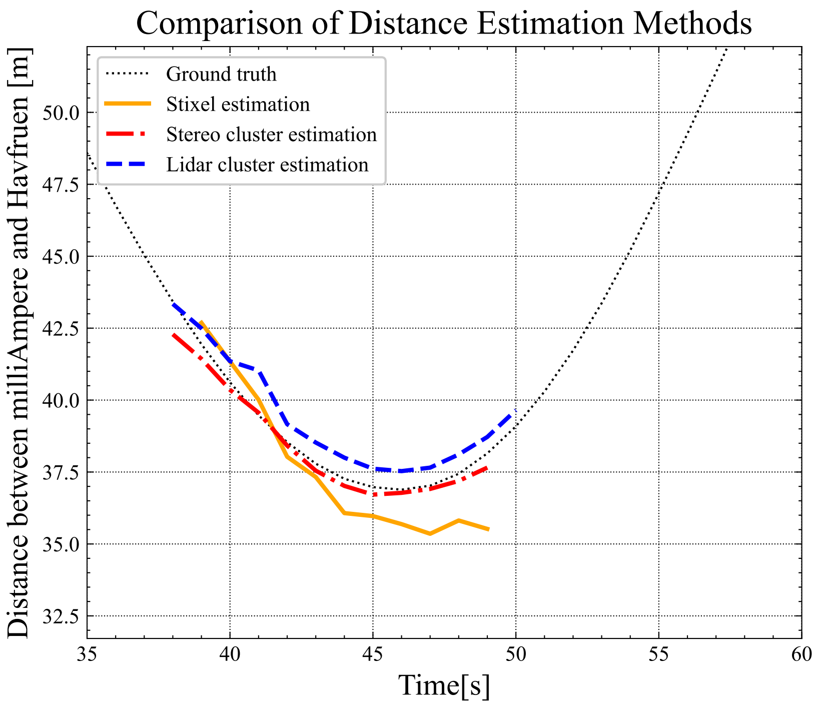
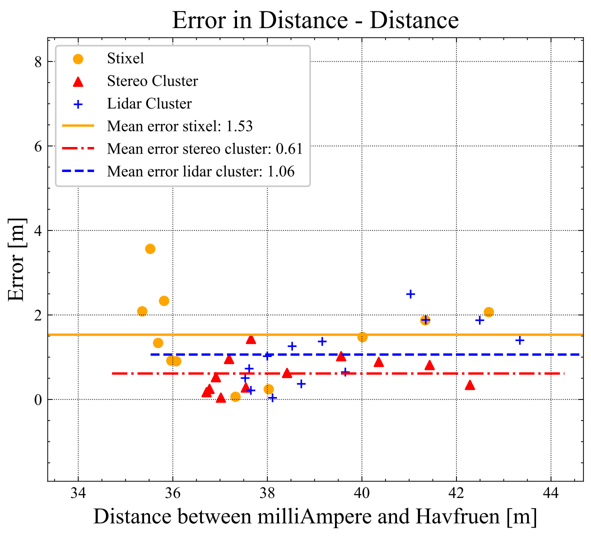
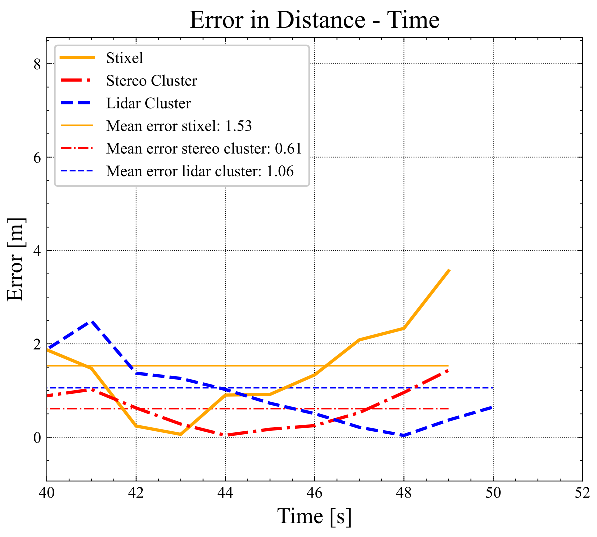


Figure 8.20: Distance estimation for scenario 5



(a) Error in regards to distance



(b) Error in regards to time

Figure 8.21: Distance error - Scenario 5

8.3 Detection Rate

In this section the results regarding detection will be presented. The results for all the scenarios combined for each method will be presented first, then each scenario by it self. Comparison between cluster methods and stixel will not represent the truth 100% as the accumulation step in the clustering methods use three consecutive inputs to determine a centroid point in a cluster this will lead to a higher detection rate than without this step. The stixel method has no accumulation of knowledge about the state in previous iterations.

The detection tables show the distance measurements in the leftmost column. To the right is the number of iterations of output with failed detection, and the right most column shows the longest sequence of consecutive missed detections. The plots show one column for each scenario with the ratio of detections and missed detections.

Stixel Tesselation

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
0-20	9	0	1	1	1	0	0	0	0	4
20-40	223	22	6	0	0	2	1	1	0	7
40-60	292	37	15	7	5	1	0	0	0	5
60-80	98	12	5	2	1	1	1	2	0	7
80-100	46	7	4	6	3	2	0	1	1	12
>100	4	1	0	1	0	2	0	0	1	11

Table 8.3: Duration of intervals without detection for different distances in total for all scenarios with the stixel method

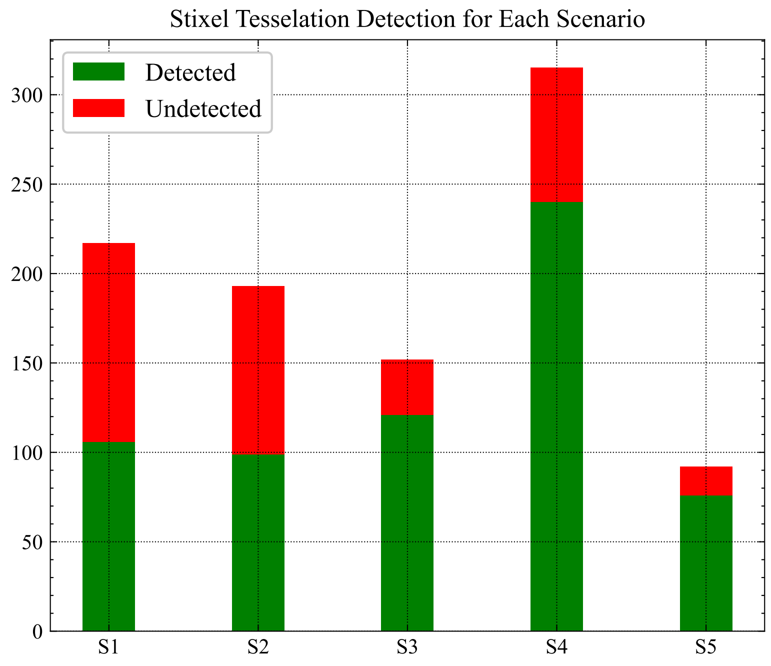


Figure 8.22: Detecton distribution for stixel method

Euclidean Cluster

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
0-20	6	0	1	1	0	0	1	0	0	5
20-40	194	3	1	0	0	2	1	0	0	6
40-60	271	3	1	0	1	0	0	0	1	10
60-80	124	0	0	0	0	0	0	0	0	0
80-100	90	0	1	0	0	0	0	0	0	2
>100	7	1	0	0	1	0	0	0	0	4

Table 8.4: Duration of intervals without detection for different distances in total for all scenarios with the Euclidean cluster method - Stereo Vision

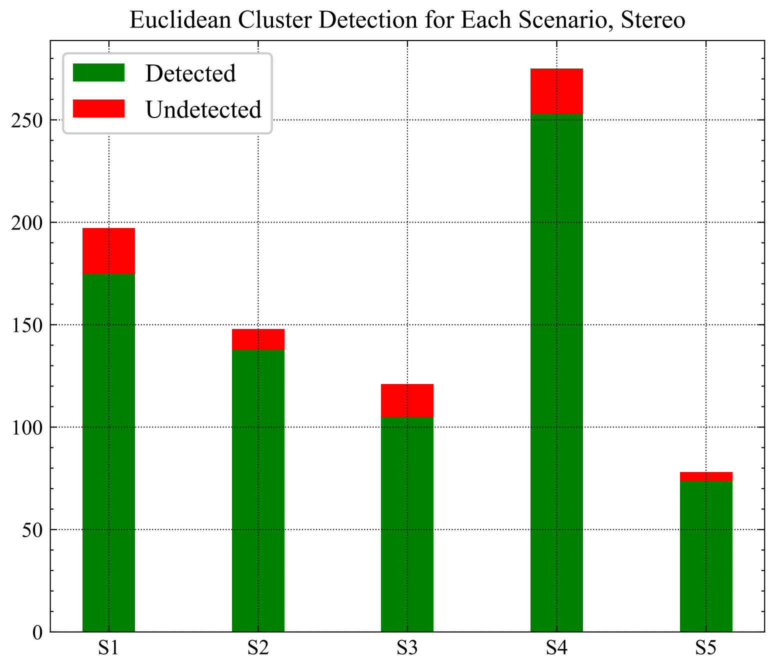


Figure 8.23: Detecton distribution for Euclidean cluster method - Stereo Vision

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
0-20	0	0	0	0	0	0	0	0	1	14
20-40	404	2	2	1	0	2	1	6	5	85
40-60	106	1	0	1	0	2	2	0	9	100
>60	59	0	0	0	0	1	2	0	5	162

Table 8.5: Duration of intervals without detection for different distances in total for all scenarios with the Euclidean cluster method - Lidar

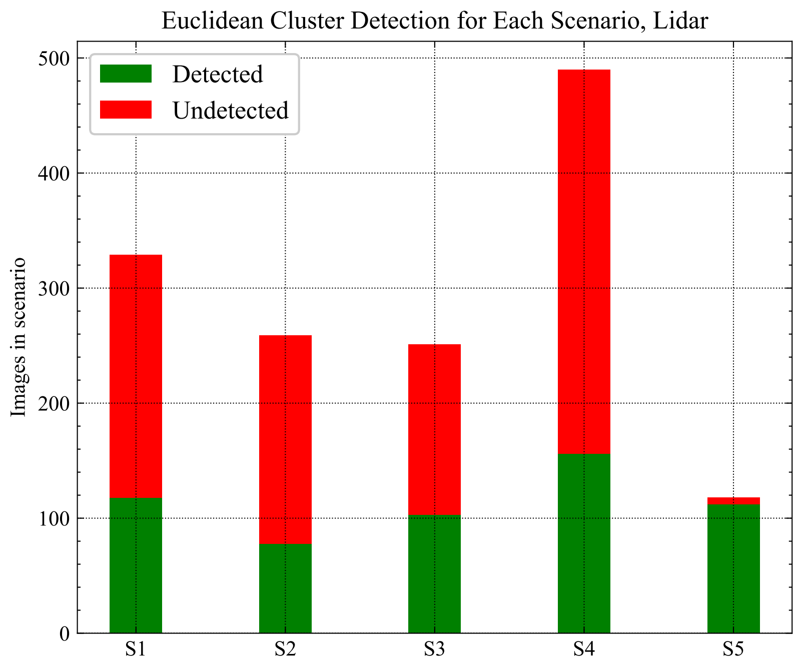
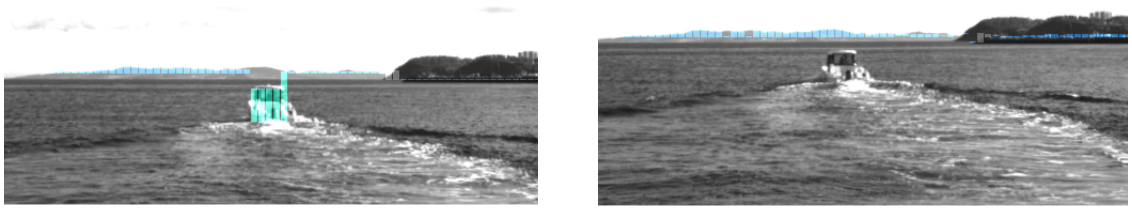


Figure 8.24: Detecton distribution for Euclidean cluster method - Lidar

Scenario 1



(a) Successful detection **(b)** Failed detection

Figure 8.25: Examples of stixel detection in scenario 1

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
0-20	9	0	1	1	1	0	0	0	0	4
20-40	44	4	3	0	0	0	0	1	0	7
40-60	14	2	1	1	1	0	1	0	0	6
60-80	34	5	4	1	1	1	1	1	0	7
80-100	18	3	1	2	2	2	0	1	3	12
>100	4	1	0	1	0	2	0	0	1	11

Table 8.6: Duration of intervals without detection for different distances for scenario 1, Stixel

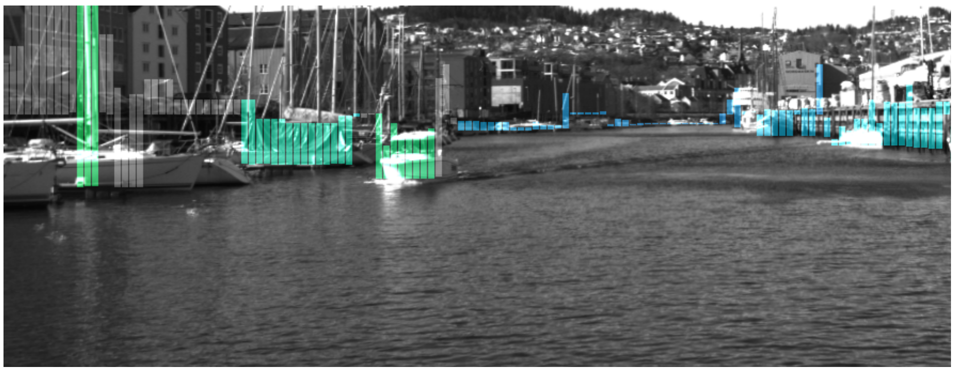
Distance [m]	0	1	2	3	4	5	6	7	>7	Max
0-20	6	0	1	1	0	1	0	0	0	5
20-40	51	2	0	0	0	0	0	0	0	1
40-60	15	3	0	0	0	0	0	0	0	1
60-80	57	0	0	0	0	0	0	0	0	0
80-100	45	0	1	0	0	0	0	0	0	2
>100	7	1	0	0	1	0	0	0	0	4

Table 8.7: Duration of intervals without detection for different distances for scenario 1, Cluster stereo

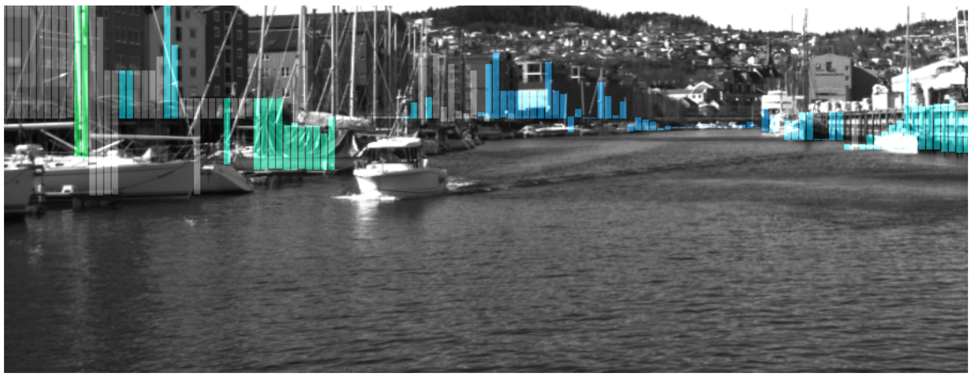
Distance [m]	0	1	2	3	4	5	6	7	>7	Max
0-20	0	0	0	0	0	0	0	0	1	14
20-40	54	2	0	0	0	0	0	0	1	12
40-60	23	0	0	0	0	1	2	0	3	16
60-80	14	0	0	0	0	1	2	0	2	10
>80	45	0	0	0	0	0	0	0	1	162

Table 8.8: Duration of intervals without detection for different distances for scenario 1, Cluster lidar

Scenario 2



(a) Successful detection



(b) Failed detection

Figure 8.26: Examples of stixel detection in scenario 2

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	2	1	1	0	0	1	0	0	0	5
40-60	34	4	3	1	0	0	0	0	0	3
60-80	28	5	0	1	0	0	0	1	0	7
80-100	38	4	4	4	1	0	0	0	1	10

Table 8.9: Duration of intervals without detection for different distances for scenario 2, Stixel

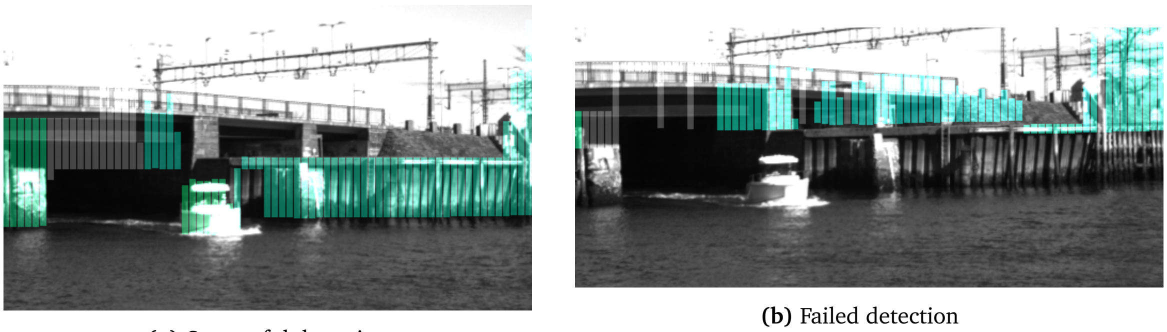
Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	8	0	0	0	0	2	0	0	0	5
40-60	41	0	0	0	0	0	0	0	0	0
60-80	44	0	0	0	0	0	0	0	0	0
80-100	45	0	0	0	0	0	0	0	0	0

Table 8.10: Duration of intervals without detection for different distances for scenario 2, Cluster stereo

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	65	0	0	0	0	0	1	0	0	6
40-60	12	0	0	0	0	0	0	0	1	15
>60	0	0	0	0	0	0	0	0	1	160

Table 8.11: Duration of intervals without detection for different distances for scenario 2, Cluster lidar

Scenario 3


(a) Successful detection

(b) Failed detection

Figure 8.27: Examples of stixel detection in scenario 3

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	39	3	1	0	0	0	1	0	0	6
40-60	46	3	0	1	1	0	0	0	0	4
60-80	36	2	1	0	0	0	0	0	0	2

Table 8.12: Duration of intervals without detection for different distances for scenario 3, Stixel

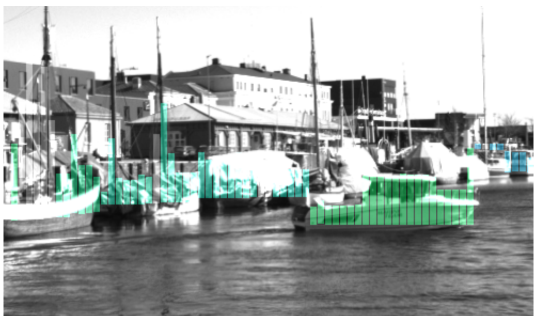
Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	39	0	0	0	0	0	1	0	0	6
40-60	45	0	0	0	0	0	0	0	1	10
60-80	23	0	0	0	0	0	0	0	0	0

Table 8.13: Duration of intervals without detection for different distances for scenario 3, Cluster stereo

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	102	0	1	1	0	0	0	0	1	9
40-60	0	0	0	0	0	0	0	0	1	100
60-80	0	0	0	0	0	0	0	0	1	34

Table 8.14: Duration of intervals without detection for different distances for scenario 3, Cluster lidar

Scenario 4


(a) Successful detection

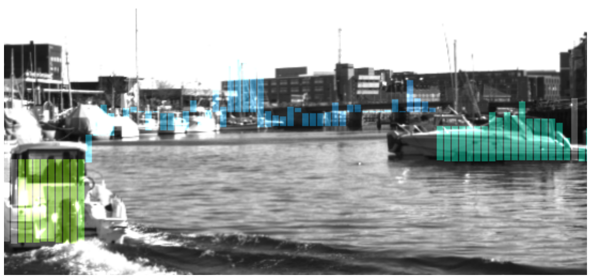
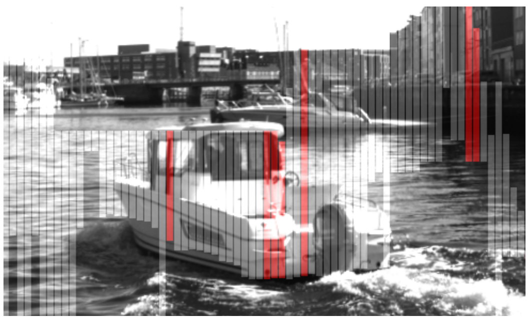

(b) Successful detection

Figure 8.28: Examples of stixel detection in scenario 4


(a) Failed detection

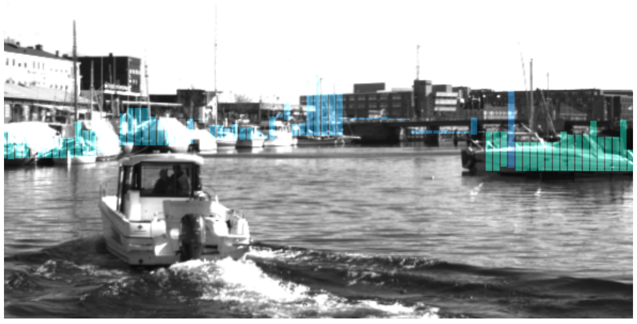

(b) Failed detection

Figure 8.29: Examples of stixel detection in scenario 4

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	124	12	1	0	0	1	0	0	0	5
40-60	116	13	9	3	3	1	0	0	0	5

Table 8.15: Duration of intervals without detection for different distances for scenario 4, Stixel

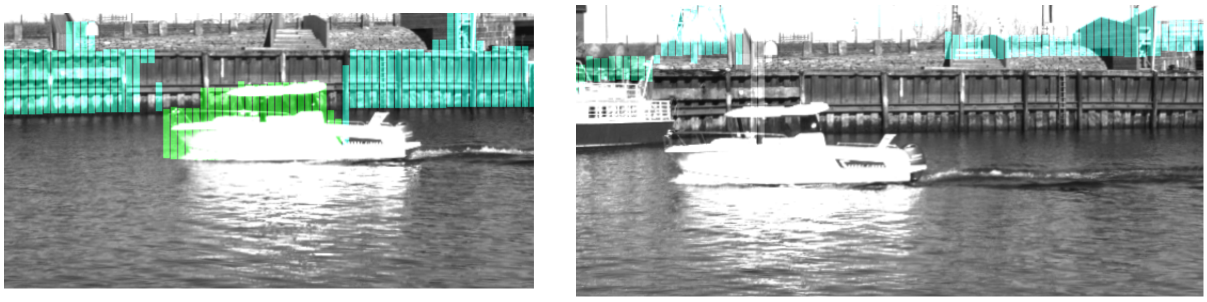
Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	119	1	1	0	0	0	1	0	1	11
40-60	112	0	1	0	0	0	0	0	0	2

Table 8.16: Duration of intervals without detection for different distances for scenario 4, Cluster stereo

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	105	0	0	0	0	1	0	0	3	85
40-60	31	0	0	1	0	0	0	0	4	84

Table 8.17: Duration of intervals without detection for different distances for scenario 4, Cluster lidar

Scenario 5



(a) Successful detection **(b)** Failed detection

Figure 8.30: Examples of stixel detection in scenario 5

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	14	2	0	0	0	0	0	0	0	0
40-60	82	15	2	1	0	0	0	0	0	3

Table 8.18: Duration of intervals without detection for scenario 5, Stixel

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	16	0	0	0	0	0	0	0	0	0
40-60	58	0	0	0	1	0	0	0	0	4

Table 8.19: Duration of intervals without detection for scenario 5, Cluster

Distance [m]	0	1	2	3	4	5	6	7	>7	Max
20-40	78	0	0	0	0	1	0	0	0	5
40-60	40	1	0	0	0	0	0	0	0	1

Table 8.20: Duration of intervals without detection for scenario 5, Cluster Lidar

8.4 Run time

In this section the run times for the algorithms will be presented. Each table consists of the mean total run time for each scenario and the the sub task within the algorithm that is the largest contributers to the run time.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
Disp. calc.[ms]	91.6	96	89	86.8	93.2
Surface mod.[ms]	145.8	141	149.4	159.8	155.4
Total.[ms]	**273.2**	**274.8**	**277.8**	**284**	**282.4**

Table 8.21: Mean run times for each scenario, stixel

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
ROS to PCL. [ms]	105.8	112.6	116.6	115.6	114.2
Cut off [ms]	10.0	12.6	11.4	12.6	10.8
SOR [ms]	<0.1	310.2	707.7	298.2	525.8
Total. [ms]	**116.4**	**453.6**	**875.0**	**451.6**	**661.2**

Table 8.22: Mean run times for each scenario, stereo cluster

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
Clustering [ms]	<0.1	8.9	8.6	5.7	2.5
SOR [ms]	<0.1	7.9	8.2	4.46	1.5
Total. [ms]	**0.25**	**20.1**	**20.0**	**11.1**	**5.4**

Table 8.23: Mean run times for each scenario, lidar cluster

Chapter 9

Discussion

9.1 Stereo Calibration

The stereo calibration, as seen in fig. 8.5, resulted in a low reprojection error of 0.04px, which in turn leads to a low distance error. Given perfect matching the distance error can be seen in fig. 8.6. In table 8.1 one can see that the rotation around the Y-axis is 5.7° and not 2.0° which was intended by design. A possible explanation for this deviation is poor accuracy when mounting the cameras on the stereo rig. This is also the reason for the large translation in the Z axis seen in table 8.1. As the transformation is with respect to the reference frame of the left camera, a rotation of that camera will cause the illusion of the right camera being translated in the Z-direction, this is illustrated in fig. 9.1.

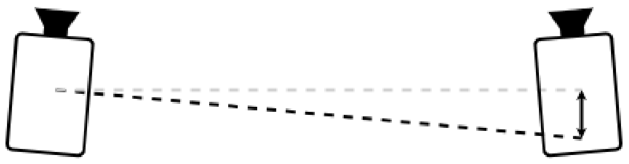


Figure 9.1: Rotation of left camera causes translation of the right camera

9.2 Stixel Tesselation

The stixel tesselation method was less robust than anticipated because of a loss of detection seemingly at random. However, in most where cases with loss of detection occured the algorithm also fails in estimating the free space surrounding the target. As free space calculation is the foundation of the algorithm, this calculation should be both robust and accurate. This type of error can be seen in fig. 8.27 and fig.8.30 a), but it cannot account for all the detection misses such as the one in fig.8.30 b). This error might be caused by changes in the pitch of milliAmpere during calculation, but it also occurred during still water. Running the algorithm on other data sets such as the *Daimler Stixel Grouth Truth Dataset* [1] produced a much more stable result with no loss of detection. This indicates that an improved result on milliAmpere data could be obtained with more tuning of both disparity map calculation and free space calculation.

[1]http://www.6d-vision.com/ground-truth-stixel-dataset

The depth estimation is comparable to the other methods, having an error of $\pm 1m$ compared to the stereo cluster method in most of the scenarios. Calculating the mean disparity of a stixel was expected to improve on accuracy compared to evaluating pixel by pixel, but over all it was slightly worse than the Euclidean Cluster method. The depth estimations might have suffered from the irregular shape of the target and at times difficulties with detection.

One improvement to the stixel method can be done in the depth calculations. In the implemented method the distance to each stixel was calculated in the X_c/Z_c-plane. This was done because of stixels having a constant disparity across the entire stixel, and most object being in the same height as milliampere. However, this simplification can give unnecessary errors when calculating the distance to lower targets close to milliAmpere such as kayaks or smaller motor boats. To improve the distance estimation to these objects, the mid point in the vertical direction for each stixel could be extracted as the top and bottom points are known. This mid point could then be used to estimate distance in the 3D rather than 2D.

Comparing the stixel method to the lidar implementation it is obvious that the lidar is more accurate in terms of distance estimation error, but in terms of operating range the stixel method is superior. For instance, in scenario 2 the stixel algorithm was able to detect Havfruen at a distance of approximately 100m, while the lidar was first able to detect it at 50 meters (fig 8.11). A similar example can be seen in scenario 3 where the stixel method established detection as soon as Havfruen exited the ship tunnel, while the lidar detected it 13 seconds later at half the distance.

The run time results presented in section 8.4 shows how long it took for the algorithm to run on an ordinary desktop computer. In table 8.21 we can see that the run time for the stixel method is consistent for all scenarios. A run time in the range 270-300ms results in a operating frequency of <4Hz. Whether this is a fast enough run time to be considered real time depend on the constraints determined by the designer of milliAmpere perception system as a whole. However, with an average run time of **278.46**ms for all scenarios it is clear that the system is slower than what was hoped for. In table 8.21 one can see that computing the surface model is taking more than half the run time. In the Daimler data set free space computation consisted for a little under half the run time, but with a total run time of $\sim\boldsymbol{10ms}$ for images of size $(1024, 333)px$. I believe this leaves room for more improvement with tuning of the algorithm. Another way of reducing the run time is to further reduce the size of the input images by altering the ROS resizing node presented in section 6.2.

9.3 Euclidean Cluster

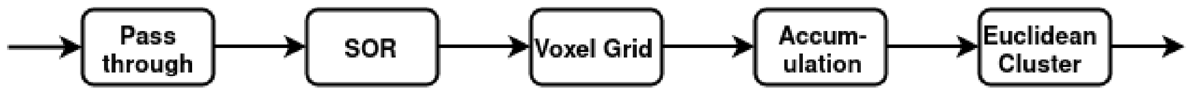
The Euclidean Cluster method was very robust and able to detect Havfruen far more often than the stixel algorithm. In most scenarios the Euclidean Clustering method and the Stixel algorithm was able to detect Havfruen at similar distances, but in general the detection was maintained both at longer distances, such as in scenario 1 (fig. 8.8), but also at closer distances such as in scenario 4 (fig. 8.17). The comparison is however, somewhat unfair as the accumulation step in the Euclidean Clustering method builds one point cloud of the three previous time steps. If a similar feature was added to the stixel algorithm the detection rate would most likely be improved.

In spite of the good detection rate, the Euclidean Cluster apporach is not that precise. The centroid point of the point cloud often drifted within the target point cloud. If the filtering failed to eliminate noise such as the vortexes produced by the propeller the centroid point would occasionally be located behind the boat it self. In this evaluation these instances was still considered a positive detection, but with poor precision.

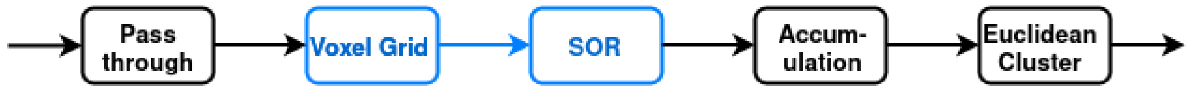
Another aspect to consider is that this method reduces all object down to points. A kayak and sailing boat would both be represented as single points. Depending on what the information provided by the perception system is going to be used for this reduction in information seems wasteful.

The run time while using the Euclidean Clustering method was heavily dependent on the amount of objects in the scene. In table 8.22 the run time spans from 116.4ms in the scenario in open water to 875ms in the most cluttered scenario from within the canal. For use in the canal this algorithm can run with a frequency of 2Hz, which is slow. Especially if the system is going to be used for close encounters.

To reduce the run time for the euclidean clustering method some measures can be done. If the Voxel Grid -and Statistical Outlier filter changed places in the sequence of filters, as shown in fig. 9.2 the run times where reduced to the ones showed in table 9.1. However, this affected the detection rate of the system. With more tuning the performance may have been on the same level as the original one, but due to lack of time this was not investigated further.



(a) Original filtering sequence



(b) Alternative filtering sequence

Figure 9.2

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
ROS to PCL. [ms]	114.6	127.0	123.8	131.0	130.2
Cut off [ms]	11.6	14.0	20.6	16.2	21.4
SOR [ms]	0.24	26.2	30.0	28.6	34.6
Total. [ms]	**127.8**	**209.4**	**239**	**236.8**	**253.8**

Table 9.1: Mean run times for each scenario - Alternative filtering sequence - Stereo

The run time of the Euclidean Clustering method on the lidar data is shown in table 8.23. Also in this case the run times are dependent on the complexity of the point cloud, but with the run times varying from 0.25ms in scenario 1 to 20ms in scenario 2 and 3, resulting in an frequency of >50Hz, it is more than fast enough for real time applications.

Chapter 10

Conclusion and Future Work

10.1 Conclusion

Several methods for solving the correspondence problem has been evaluated. The feature based algorithms simply does not provide a dense enough point cloud for this application. There are probably multiple of the correlation based methods that could provide decent results. However, the choice of using the SGM method has yielded in over all good results.

Five different object detection algorithms were also evaluated whereof two where tested on real life scenarios. The differences in the algorithms also showed in the results where the Euclidean Cluster method scored best regarding detection rate, but it has its drawbacks with presenting any object as a single point. The accuracy and operating range were good, but the run time in the given configuration was not low enough to be considered applicable for a real time application.

The stixel method had a poorer detection rate, but good accuracy and operating range. The run time was better than Euclidean Cluster, but still higher than ideal. The stixel method has the advantage of providing basic information about the shape of the objects detected in the form of stixel height. The width of objects can also be determined by counting the number of stixels with the same depth. Having this additional information primitive information about the shape of the objects can be extracted.

In comparison to the lidar the stereo vision system performed as expected being slightly less accurate than the lidar, but able to operate at a much larger distance than the lidar currently mounted on milliAmpere. However, the lidar has a much lower run time making it able to operate at a much higher frequency than the stereo vision alternatives.

Both of the presented stereo vision systems has complementary features to the lidar. With further improvements the implementations of some of the suggested future work, the stereo camera can be a valuable sensor on board milliAmpere, either alone, or in combination with the lidar.

10.2 Future Work

- As mentioned in section 5.2 there exists several variations and improvements of the stixel algorithm presented in this thesis. This includes multi layer variations where objects at different depths in the same direction are detected. There are segmentation variations where groups of stixels are grouped into objects.

- Permanent mounting on milliAmpere needs to be done. This increases the accuracy as the transformation between reference frames is constant.

- Examine if use of GPU based processing can reduce run time in this scenario.

- Online calibration - Implementing an online calibration scheme that can recalibrate the system when needed will improve the over all accuracy.

- Fusing stereo vision and lidar data to utilize complementary aspects.

Bibliography

[1] B.-S. Shin, X. Mou, W. Mou and H. Wang, 'Vision-based navigation of an unmanned surface vehicle with object detection and tracking abilities,' *Machine Vision And Applications*, vol. 29, no. 1, pp. 95–112, 2018.

[2] Z. Liu, Y. Zhang, X. Yu and C. Yuan, 'Unmanned surface vehicles: An overview of developments and challenges,' *Annual Reviews in Control*, vol. 41, pp. 71–93, 2016, ISSN: 1367-5788. DOI: `https://doi.org/10.1016/j.arcontrol.2016.04.018`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S1367578816300219`.

[3] A. S. Golden and R. E. Weisbrod, 'Trends, Causal Analysis, and Recommendations from 14 Years of Ferry Accidents. Journal of Public Transportation,' *Journal of Public Transportation*, vol. 19, no. 1, pp. 17–27, 2016.

[4] *Autonomous all-electric passenger ferries for urban water transportation (Autoferry)*. [Online]. Available: `https://www.ntnu.edu/autoferry`.

[5] T. Ø. Olsen and L. C. K. Theimann, 'Stereo vision for autonomous ferry,' Ph.D. dissertation, 2020.

[6] T. Ø. Olsen, 'Stereo vision using local methods for autonomous ferry,' *Stereo Vision for autonomous ferry*, no. January, 2020.

[7] K. Auestad, 'A comparison of local- and Semi-GlobalMatching methods,' 2020.

[8] H. Hirschmuller, 'Accurate and efficient stereo processing by semi-global matching and mutual information,' in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, IEEE, 2005, 807–814 vol. 2, ISBN: 0769523722.

[9] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004, pp. 302–309, ISBN: 978-0-511-18618-9.

[10] Z. Zhang, 'A flexible new technique for camera calibration,' *IEEE transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000, ISSN: 0162-8828.

[11] S. Se and N. Pears, 'Passive 3D imaging,' in *3D imaging, Analysis and Applications*, Springer, 2012, pp. 35–94.

[12] M. Yi and S. Stefano, *An Invitation To 3D Vision*. Springer Science, 2004, ISBN: 9781441918468. [Online]. Available: `http://download.springer.com/static/pdf/195/bok%253A978-0-387-21779-6.pdf?originUrl=http%3A%2F%2Flink.springer.com%2Fbook%2F10.1007%2F978-0-387-21779-6&token2=exp=1458745767~acl=%2Fstatic%2Fpdf%2F195%2Fbok%25253A978-0-387-21779-6.pdf%3ForiginUrl%3Dhttp%25`.

[13] W. Förstner and B. P. Wrobel, *Photogrammetric Computer Vision: Statistics, Geometry, Orientation and Reconstruction*, ser. Geometry and Computing. Cham: Springer International Publishing AG, 2016, vol. 11, ISBN: 3319115499.

[14] W. Zhao and N. Nandhakumar, 'Effects of camera alignment errors on stereoscopic depth estimates,' *Pattern Recognition*, vol. 29, no. 12, pp. 2115–2126, 1996, ISSN: 0031-3203. DOI: `https://doi.org/10.1016/S0031-3203(96)00051-9`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0031320396000519`.

[15] D. G. Bailey, 'Sub-pixel estimation of local extrema,' in *Proceeding of Image and Vision Computing New Zealand*, 2003, pp. 414–419.

[16] D. Marr and T. Poggio, 'A computational theory of human stereo vision,' *Proceedings of the Royal Society of London - Biological Sciences*, vol. 204, no. 1156, pp. 301–328, 1979, ISSN: 09628452. DOI: `10.1098/rspb.1979.0029`.

[17] N. Lazaros, G. C. Sirakoulis and A. Gasteratos, 'Review of Stereo Vision Algorithms: From Software to Hardware,' *International Journal of Optomechatronics*, vol. 2, no. 4, pp. 435–462, 2008. DOI: `10.1080/15599610802438680`. [Online]. Available: `https://doi.org/10.1080/15599610802438680`.

[18] R. H. Thaher and Z. K. Hussein, 'Stereo Vision Distance Estimation Employing SAD with Canny Edge Detector,' *International Journal of Computer Applications*, vol. 107, no. 3, 2014.

[19] M. B. Hisham, S. N. Yaakob, R. A. A. Raof, A. B. A. Nazren and N. M. Wafi, 'Template Matching using Sum of Squared Difference and Normalized Cross Correlation,' in *2015 IEEE Student Conference on Research and Development (SCOReD)*, 2015, pp. 100–104. DOI: `10.1109/SCORED.2015.7449303`.

[20] R. Zabih and J. Woodfill, 'Non-parametric local transforms for computing visual correspondence,' in *Computer Vision — ECCV '94*, J.-O. Eklundh, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 151–158, ISBN: 978-3-540-48400-4.

[21] J. Banks and P. Corke, 'Quantitative Evaluation of Matching Methods and Validity Measures for Stereo Vision,' *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 512–532, 2001. DOI: `10.1177/02783640122067525`. [Online]. Available: `https://doi.org/10.1177/02783640122067525`.

[22] D. G. Lowe, 'Distinctive image features from scale-invariant keypoints,' *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004, ISSN: 09205691. DOI: `10.1023/B:VISI.0000029664.99615.94`.

[23] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science, 2010.

[24] H. Bay, T. Tuytelaars and L. Van Gool, 'SURF: Speeded Up Robust Features,' in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof and A. Pinz, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 404–417, ISBN: 978-3-540-33833-8.

[25] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, 'ORB: An efficient alternative to SIFT or SURF,' in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571. DOI: `10.1109/ICCV.2011.6126544`.

[26] E. Rosten and T. Drummond, 'Machine Learning for High-Speed Corner Detection,' in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof and A. Pinz, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 430–443, ISBN: 978-3-540-33833-8.

[27] M. Calonder, V. Lepetit, C. Strecha and P. Fua, 'BRIEF: Binary Robust Independent Elementary Features,' in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos and N. Paragios, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 778–792, ISBN: 978-3-642-15561-1.

[28] Y. Ohta and T. Kanade, 'Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming,' *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, no. 2, pp. 139–154, 1985, ISSN: 1939-3539. DOI: `10.1109/TPAMI.1985.4767639`.

[29] Y. Liu and J. K. Aggarwal, *Local and Global Stereo Methods*, Second Edi. Elsevier Inc., 2005, pp. 297–308, ISBN: 9780121197926. DOI: 10.1016/B978-012119792-6/50081-4. [Online]. Available: http://dx.doi.org/10.1016/B978-0-12-119792-6.50081-4.

[30] Y. Boykov, O. Veksler and R. Zabih, 'Fast approximate energy minimization via graph cuts,' *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001, ISSN: 0162-8828.

[31] J. Kim, V. Kolmogorov and R. Zabih, 'Energy Minimization and Mutual Information,' *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pp. 1033–1040, 2003.

[32] R. A. Hamzah and H. Ibrahim, 'Literature survey on stereo vision disparity map algorithms,' *Journal of Sensors*, vol. 2016, 2016, ISSN: 16877268. DOI: 10.1155/2016/8742920.

[33] T. V. Haavardsholm, *A handbook in Visual SLAM*. Unpublished - Part of curriculum in TTK21 at NTNU, 2010.

[34] A. Talukder, R. Manduchi, A. Rankin and L. Matthies, 'Fast and reliable obstacle detection and segmentation for cross-country navigation,' in *Intelligent Vehicle Symposium, 2002. IEEE*, vol. 2, 2002, pp. 610–618. DOI: 10.1109/IVS.2002.1188019.

[35] H. Badino, U. Franke and D. Pfeiffer, 'The stixel world-a compact medium level representation of the 3d-world,' in *Joint Pattern Recoginition Symposium*, Springer, 2009, pp. 51–60.

[36] N. Bernini, M. Bertozzi, L. Castangia, M. Patander and M. Sabbatelli, 'Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey,' in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 873–878. DOI: 10.1109/ITSC.2014.6957799.

[37] P. Pinggera, U. Franke and R. Mester, 'High-performance long range obstacle detection using stereo vision,' in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 1308–1313. DOI: 10.1109/IROS.2015.7353537.

[38] Zhongei Zhang, Weiss and Hanson, 'Qualitative obstacle detection,' in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 554–559. DOI: 10.1109/CVPR.1994.323881.

[39] T. Williamson and C. Thorpe, 'A specialized multibaseline stereo technique for obstacle detection,' in *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*, 1998, pp. 238–244. DOI: 10.1109/CVPR.1998.698615.

[40] H. Bandino, U. Franke and R. Mester, *Free space computation using stochastic occupancy grids and dynamic programming*. 2007.

[41] A. Elfes, 'Sonar-based real-world mapping and navigation,' *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987, ISSN: 2374-8710. DOI: 10.1109/JRA.1987.1087096.

[42] F. Oniga and S. Nedevschi, 'Processing Dense Stereo Data Using Elevation Maps: Road Surface, Traffic Isle, and Obstacle Detection,' *IEEE Transactions on Vehicular Technology*, vol. 59, no. 3, pp. 1172–1182, 2010, ISSN: 1939-9359. DOI: 10.1109/TVT.2009.2039718.

[43] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng, 'ROS: an open-source Robot Operating System,' vol. 3, no. 3.2, 2009.

[44] R. B. Rusu and S. Cousins, '3D is here: Point Cloud Library (PCL),' in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[45] E. Brekke, *Fundamentals of Sensor Fusion*, 2nd ed. 2020.

[46] S. M. Kay, *Fundamentals of statistical signal processing : Vol. 2 : Detection theory*, Upper Saddle River, N.J, 1998.

Appendix A

Additional Material

A.1 Maximum Likelihood Estimation - MLE

The MLE is a well known maximizing estimator that estimates the parameters of a probability distribution by maximizing the likelihood function. This is done so that a distribution can be fitted to the data as best as possible [45].

$$\hat{x} = \arg\max_{x} p(z|x) \tag{A.1}$$

A.2 Generalized Likelihood Ratio Test - GLRT

The GLRT is a test that decides which of two hypothesis that is most likely to be true in a given scenario. The more common *Likelihood Ratio Test* can be used when the probability density functions are known. In the cases where the pdf is unknown, the GLRT is used [46]. In eq. A.2 and eq. A.3 two generic hypothesis are presented.

$$\mathcal{H}_0 : X \sim p_0 \in p_0(x|\theta_0), \quad \theta_0 \in \Theta_0 \tag{A.2}$$
$$\mathcal{H}_1 : X \sim p_1 \in p_1(x|\theta_1), \quad \theta_1 \in \Theta_1 \tag{A.3}$$

The GLRT replaces unknowns ($\boldsymbol{\theta}_i$) by their MLE's ($\hat{\boldsymbol{\theta}}_i$), as described in A.1. Then a threshold can be set with γ to decide whether the null hypothesis ($\mathcal{H}_0$) is to be kept or rejected.

$$L_G(\boldsymbol{x}) = \frac{p(\boldsymbol{x}; \hat{\boldsymbol{\theta}}_1, \mathcal{H}_1)}{p(\boldsymbol{x}; \hat{\boldsymbol{\theta}}_0, \mathcal{H}_0)} > \gamma \tag{A.4}$$

Appendix B

Scenario Overview

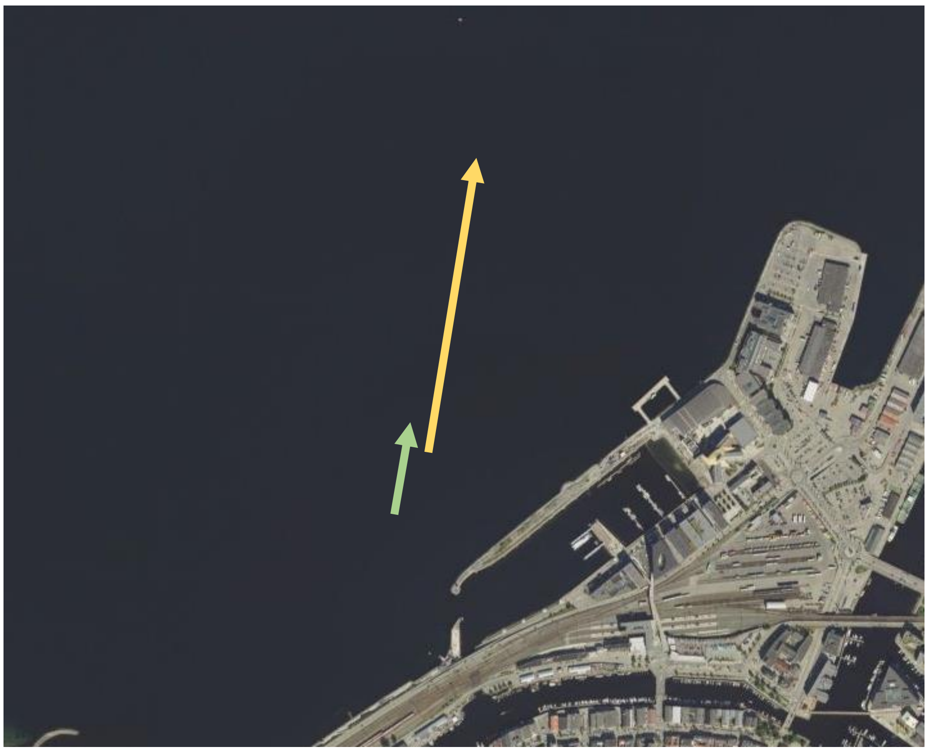
Experiment Overview

Overview of the different scenarios tested in the experiments.

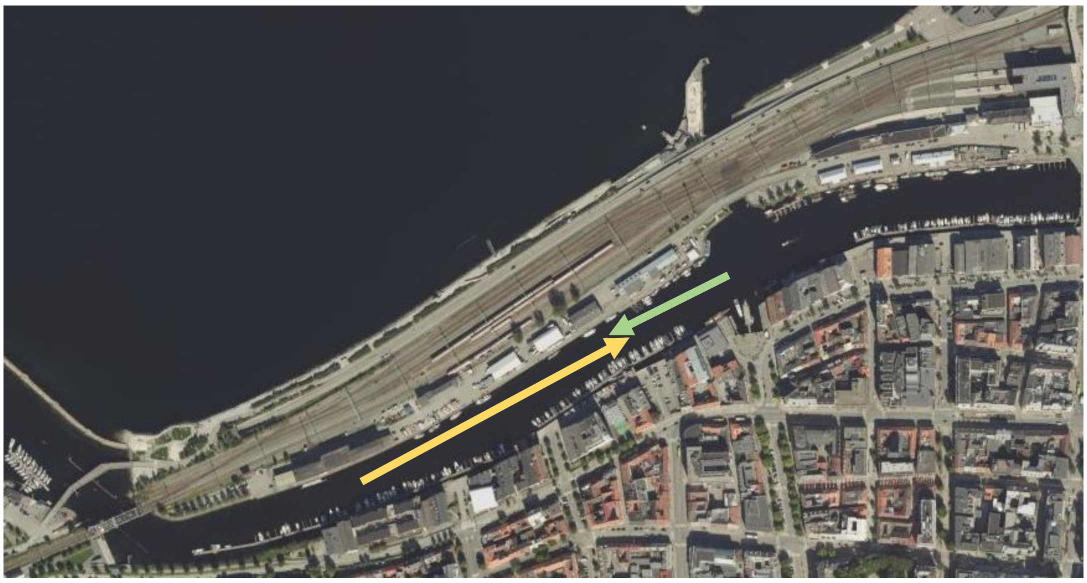
Yellow arrow = Havfruen

Green arrow = milliAmpere

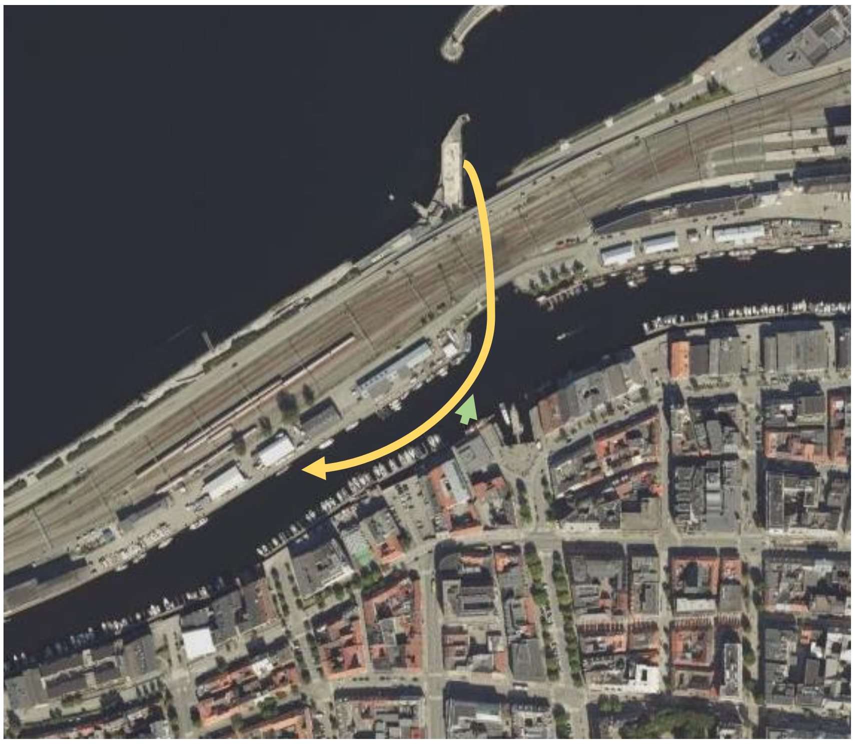
Scenario 1



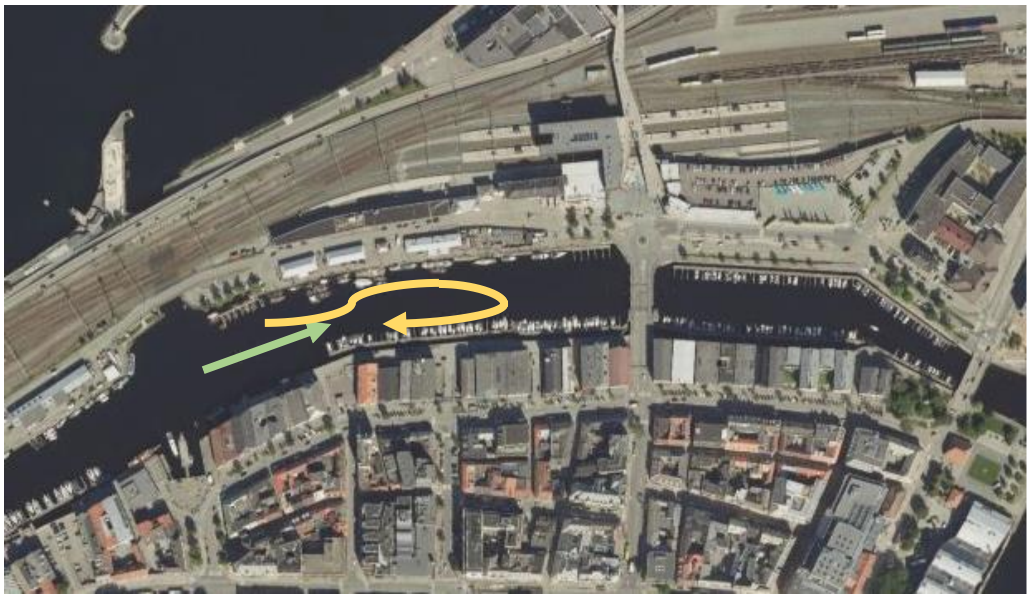
Scenario 2



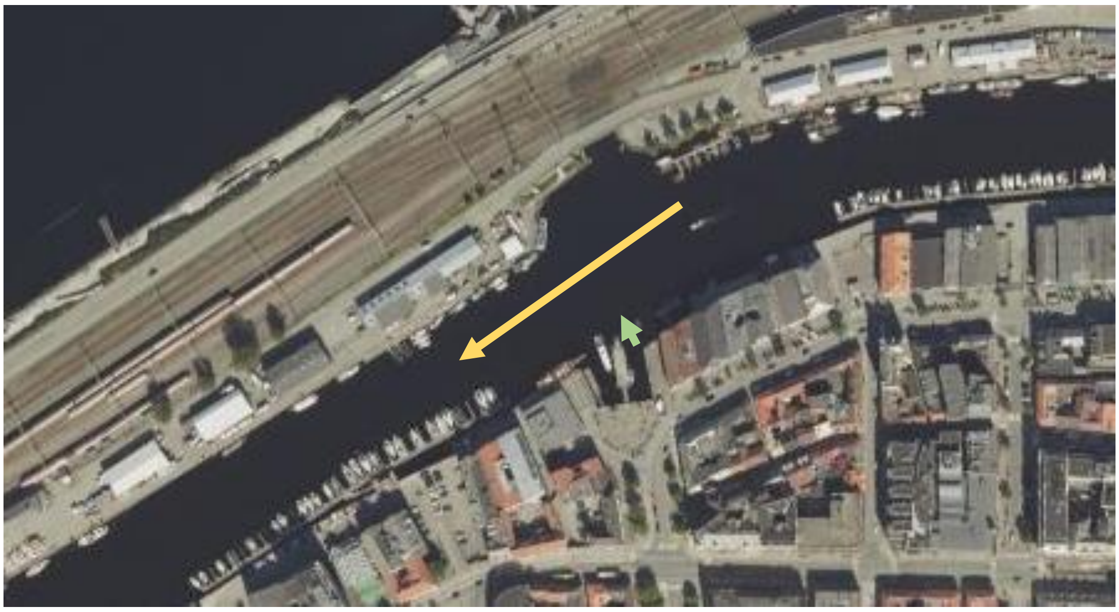
Scenario 3



Scenario 4



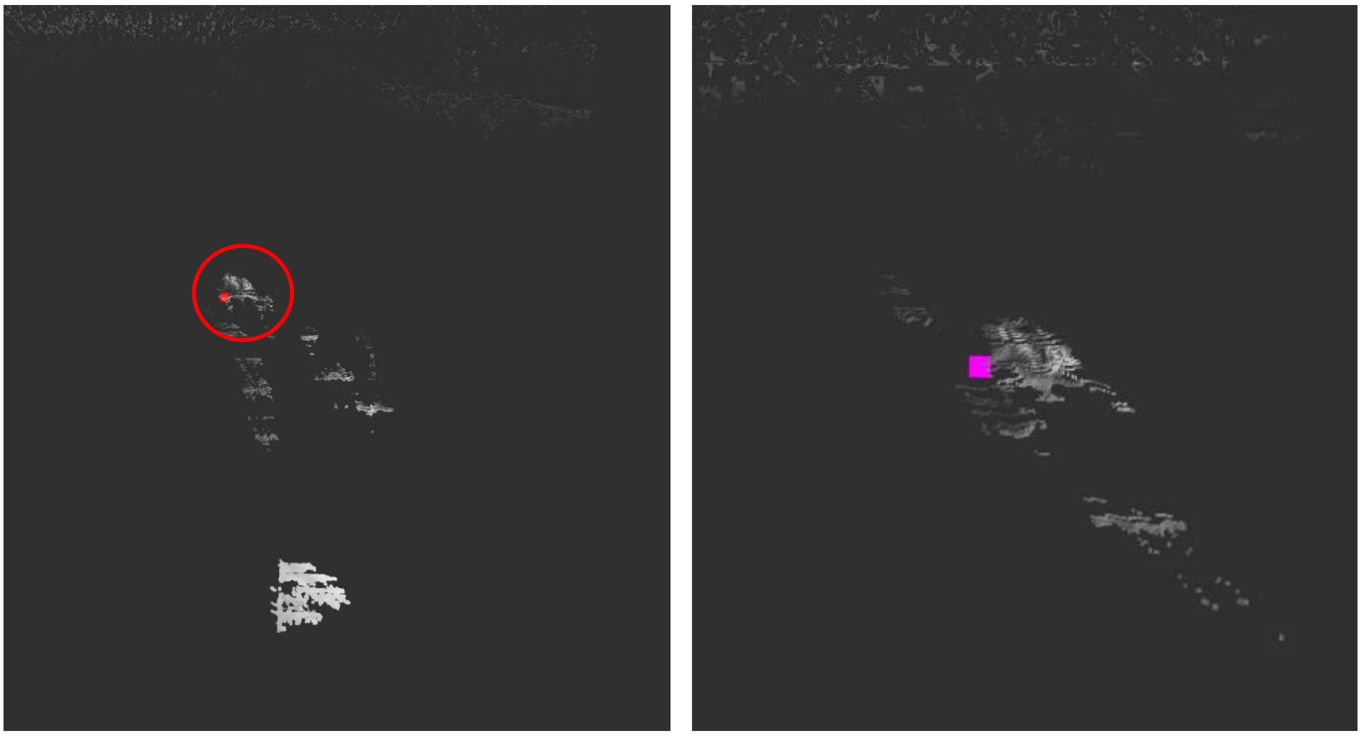
Scenario 5



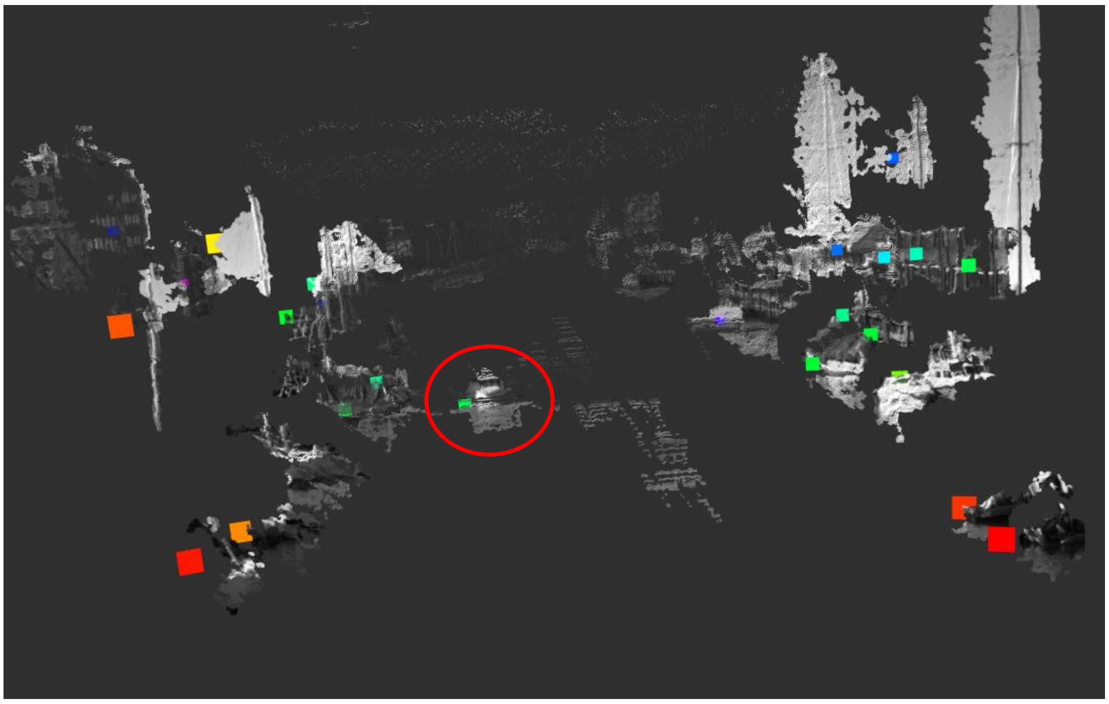
Appendix C

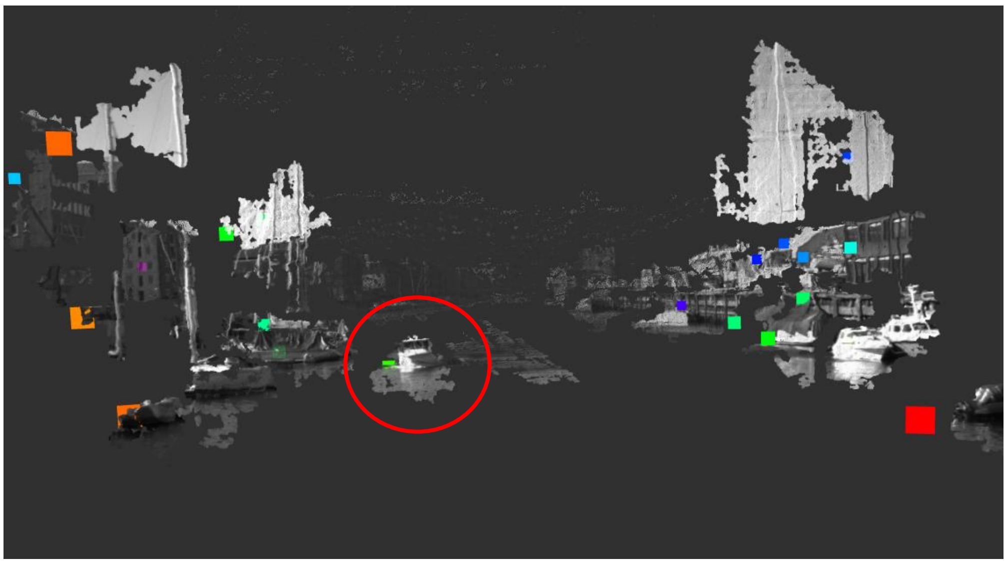
Point clouds - Euclidean Cluster - Stereo Vision

Scenario 1

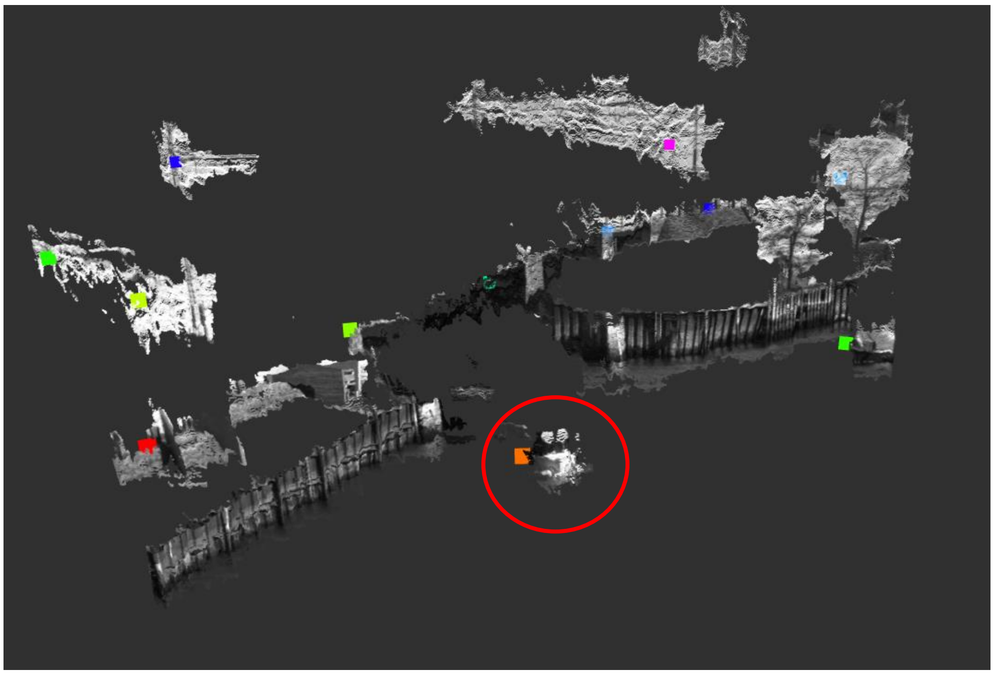


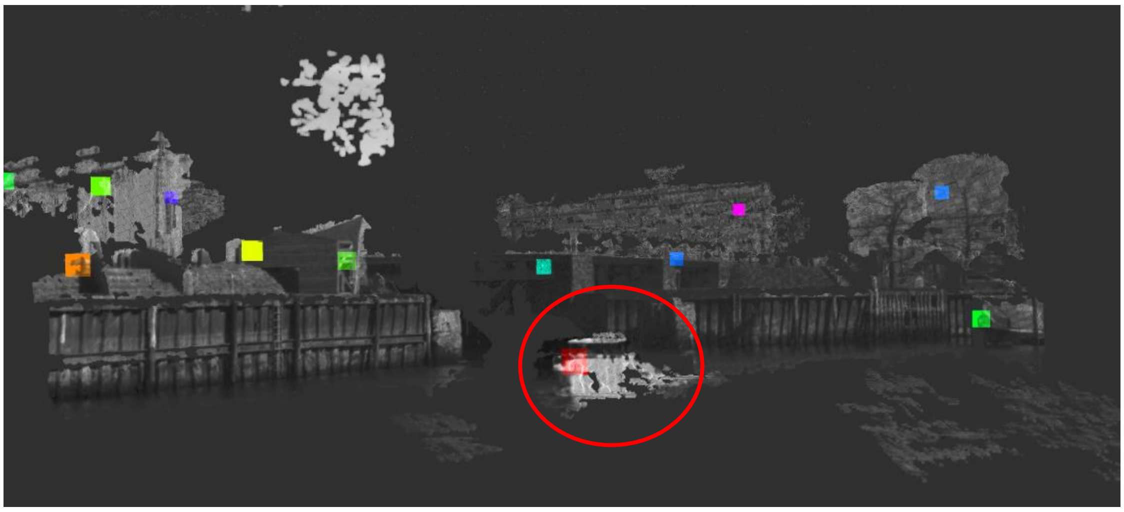
Scenario 2



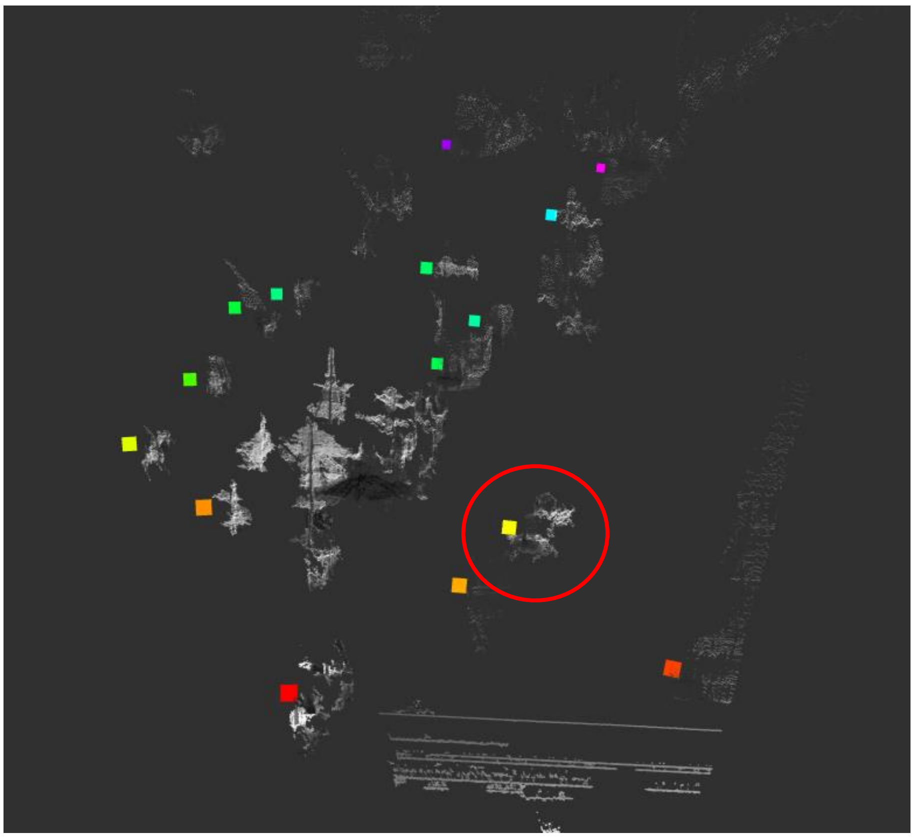


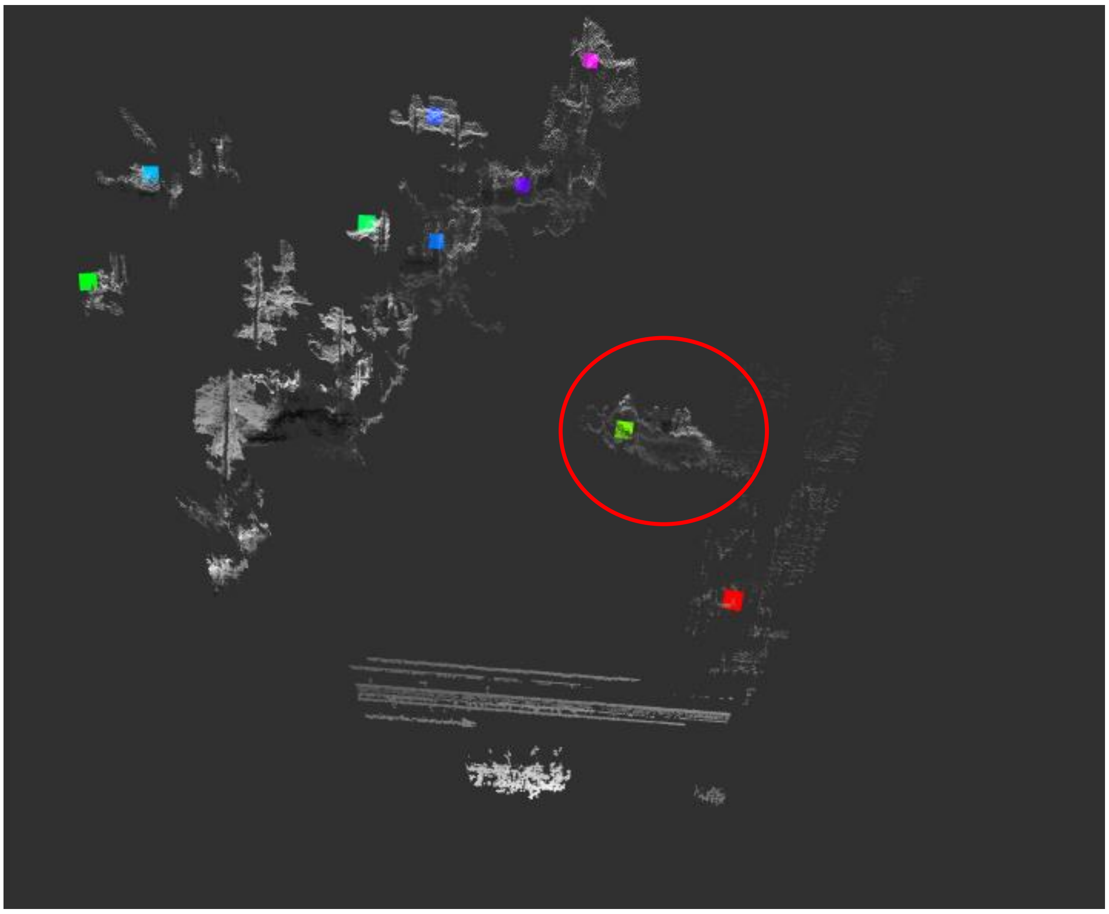
Scenario 3



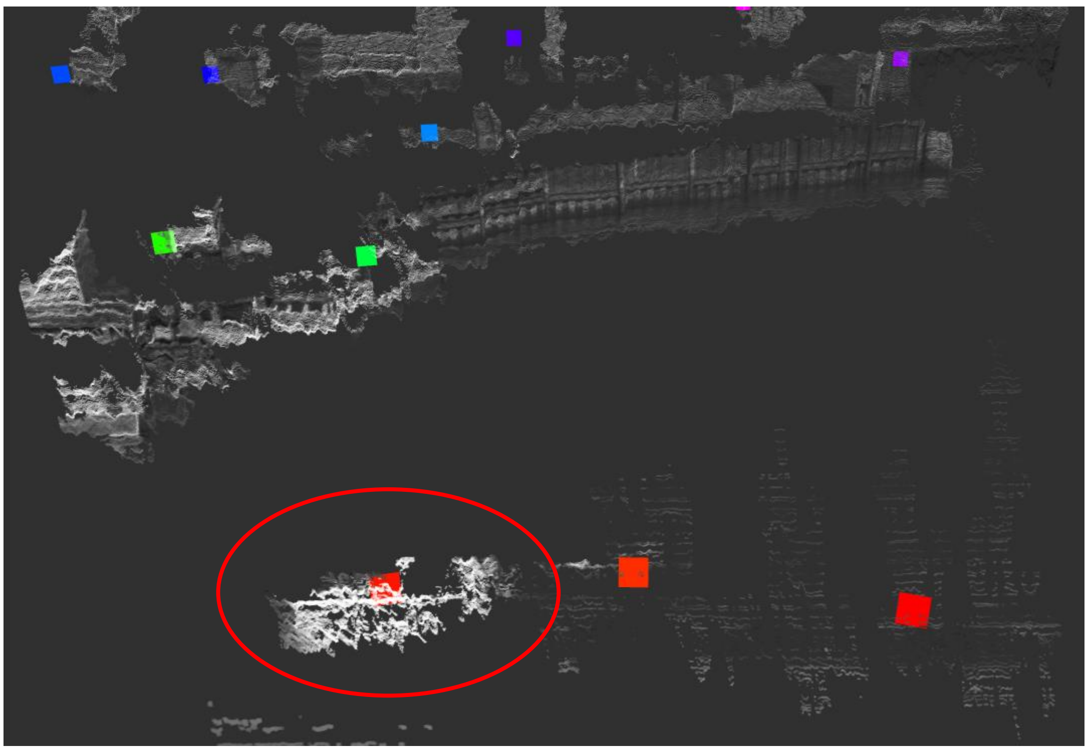


Scenario 4





Scenario 5



NTNU

Kunnskap for en bedre verden