NTNU
Norwegian University of
Science and Technology

Automate Detection and Weight Estimation of Fish in Underwater 3D Images

Lars-Håkon Nohr Nystad

Master of Science in Computer Science
Submission date: August 2018
Supervisor: Anne Cathrine Elster, IDI
Co-supervisor: Ketil Bø, Trollhetta

Norwegian University of Science and Technology
Department of Computer Science

Problem Description

In this thesis, we investigate the ability to automate detection and weight estimation of fish, by exploiting the 3D camera technology. An image dataset of fish in a tank is given, and tasks include finding a robust segmentation method for detection of fish and further use this as a basis for weight estimation.

Assignment given: 13. February 2018
Supervisor: Dr. Anne C. Elster, IDI
Co-supervisor: Ketil Bø, Trollhetta

Abstract

Being able to determine the weight of fish is important information for fish breeding facilities. Current methods rely on manual measurements, but it is interesting to look into how underwater imaging can be used to automate these measurements.

Underwater imaging is a complex problem due to light attenuation and poor water quality yielding lots of light scattering from an illuminating light source. In collaboration with Trollhetta and with the use of data from a cutting edge range-gated camera, we will tackle the problem of automating the measurement of the weight and size of atlantic salmon (Salmo salar) swimming freely inside a fish tank. The camera is manufactured by SINTEF Digital in collaboration with Odos Imaging and Bright Solutions, and illuminates the scene by the use of green light with a wavelength of $532nm$.

An Active Shape Model (ASM) implementation is used as a backend to handle the problem of segmenting salmon in the given images. The segmentation algorithm detects the contour of the fish, and makes it easy to find both the length and height of the fish in image space.

The segmented image is matched against the corresponding depth component of the image and the mean of all pixels is used as a measurement for the distance from the camera to the fish. By using the principal of similar triangles in the pinhole camera model we project information like length and height from image space into world space.

The method proposed in this thesis can be regarded as a proof of concept, and a baseline for further research into this particular problem.

Sammendrag

Å finne vekten til fisk er viktig informasjon for oppdrettsanlegg. Dagens metoder måler vekt manuelt, men det er interessant å se på hvordan bildetaking under vann kan bli brukt til å automatisere disse målingene.

Bildetaking under vann er et komplekst problem grunnet høyt effekttap av lysenergi og dårlig vannkvalitet som fører til lysspredning fra en opplysende lyskilde. I samarbeid med Trollhetta og ved bruk av et moderne undervannskamera, skal vi undersøke problemet med å automatisere målinger av vekt og størrelse på oppdrettslaks som svømmer fritt inne i en oppdrettsmerde. Kameraet er utviklet av SINTEF Digital i sammarbeid med Odos Imaging og Bright Solutions, og opplyser scenen foran kameraet ved bruk av grønt lys med en bølgelengde på $532nm$.

En ASM implementasjon er brukt for å takle problemet med å segmentere laks i de gitte bildene. Segmenteringsalgoritmen detekterer konturen av fisker, og gjør det lett å finne både lengde og høyde på fisken gitt i piksler.

De segmenterte bildene er sammenlignet mot de tilsvarende dybde-bildene og gjennomsnittet av alle dybdeverdiene blir brukt som et mål på hvor langt unna en fisk er fra kameraet. Ved å bruke prinsippet om formlike trekanter i hullkamera-modellen kan lengde og høyde gitt i piksler projekteres til lengde og høyde i meter.

Metoden som er foreslått i denne oppgaven kan bli sett på som et ”proof of concept”, og en basis for videre forskning til dette spesifikke problemet.

Acknowledgement

I would like to thank my supervisor Dr. Anne C. Elster and my co-supervisor Ketil Bø (Trollhetta) for guidance and feedback in regard to this thesis. I would also like to thank SINTEF for providing the dataset that has been used to run experiments on. I am also grateful for the opportunity to be a part of the HPC-lab group, and all the resources it possesses. Last but not least I would like to thank my fellow master students whom has been a great resource both in the form of academic support and friendship.

Table of Contents

List of Tables

List of Figures

Abbreviations

AAM	Active Appearance Model
AI	Artificial Intelligence
ASM	Active Shape Model
CMOS	Complementary metal–oxide–semiconductor
CNN	Convolutional Neural Network
CPU	Central Processing Unit
GPA	Generalized Procrustes Analysis
GPU	Graphics Processing Unit
PCA	Principal Component Analysis
R-CNN	Region Based Convolution Neural Network
RMSD	root-mean-square deviation
SSM	Statistical Shape Model
SVD	Singular-value decomposition

Chapter 1

Introduction

Mass and size measurement of fish is important for many operations in the production line of salmon breeding facilities. Hao et al. (2015) mentions several reasons to have a good estimation of measurements like these. Transportation of fish for slaughtering can be more effective and the slaughterhouse can know with a good estimate how much fish they actually have. Rules for total fish weight limitations can be easily obeyed with better control, and feeding of fish can be optimized to reduce breeding time and food waste.

The traditional method for estimating mass and size of a fish population has been to take a certain amount of fish out of the cage and do the measurements manually. This is a very cumbersome process and research has been put in to automate this process.

In the last decade, range sensing cameras has found many application for automation problems. Shao et al. (2013) has collected research of different application areas for this upcoming technology. They have collected applications for object classification and tracking, scene reconstruction, robotics, medicine and more.

This thesis will cover a method for automating the measurement of mass and size of fish. State of the art image segmentation will be used to detect fish in images from a given dataset made of images from a 3D camera. Each image in the dataset has a corresponding depth map which will be used to calculate information about the fish.

1.1 Motivation

The problem of automating measurements of mass and size of fish is highly relevant for the fish export industry. According to Statistisk sentralbyrå (2018), fish export is one of the biggest export industries in Norway and lots of research is put into underwater imaging for automation applications.

Although there has been a lot of focus lately regarding the use of deep learning for solving computer vision problems, we will in this thesis use more traditional methods for reaching our goal of segmenting fish from images.

1.2 Contributions

The contribution of this thesis is a proposed method for automatic estimation of the weight of fish from a 3D image. With the research presented and a method that can be viewed as a proof of concept, it is much potential for future work using this thesis as a baseline. Ketil Bø at Trollhetta, who I have been cooperating with from the beginning of this project, were a part of defining the bounds of the problem description stated in this thesis. Ketil will have the opportunity to use the work I have done, and make use of it out in the industry.

1.3 Thesis Outline

This thesis will continue using the following outline.

Chapter 2: Background will cover relevant research material for solving the stated problem description. First theory comprising of 3D camera systems and image segmentation are covered, followed by techniques for estimating fish weight and finishing of with related work.

Chapter 3: Methods will present the theory behind the techniques used for image segmentation and the weight estimation of fish. This includes explaining the algorithms and mathematics behind the ASM segmentation method, and how knowing the depth of objects from a 3D camera can be used to project distances in image space into world space.

Chapter 4: Implementation is where we go into detail of how the methods are made use of. We will mention what tools and frameworks has been used in order to achieve an implementation of all the methods, and give a general overview of the project repository.

Chapter 5: Results will contain what we have achieved with our implementation and show segmentation of fish, both what works and what does not.

Chapter 6: Discussion is where we go through our results and explain why they came out as they did. We will talk both about the good results and the bad, and explain why they are so.

Chapter 7: Conclusions and Future Work is the last chapter in this thesis and is where we finalize and conclude our work.

Chapter 2

Background

This chapter covers the background material relevant for the problem description stated at the beginning of this thesis. The 3D camera technology will be described from early stereo vision systems up until today's 3D camera systems. Next state-of-the-art image segmentation methods will be discussed, focusing on both traditional methods and deep learning methods. Lastly this chapter will cover related work regarding 3D image segmentation, and fish weight estimation from images. Although not been used in this project, some Artificial Intelligence (AI) methods will be presented because it's a subject that is under intense study these days.

2.1 Stereo Vision

A traditional way for finding depth information in an image is by the use of stereo images. The method is based on how humans perceive depth by using two eyes and letting the brain "calculate" the depth. Two cameras are used to represent the eyes, and a computer running some clever algorithms does the work of the brain. Bradski and Daebler (2008) describes the method in four steps when using two cameras.

1. The first step is called *undistortion* and handles the problem of cameras with slightly different lenses. There are two lens problems that arise during manufacturing, *radial distortion* is the result of lenses having slightly different shapes, and *tangential distortion* is when the lenses are placed and oriented differently when assembling the cameras. This step requires two sets of parameters called distortion parameters and camera intrinsic parameters, which is found during calibration of the two cameras.

2. The second step is called *rectification* and handles how the cameras are rotated and translated in relation to each other. The translation and rotation of a camera is called the extrinsic parameters and the result of this step are images that are projected so that the two image planes are co-planar.

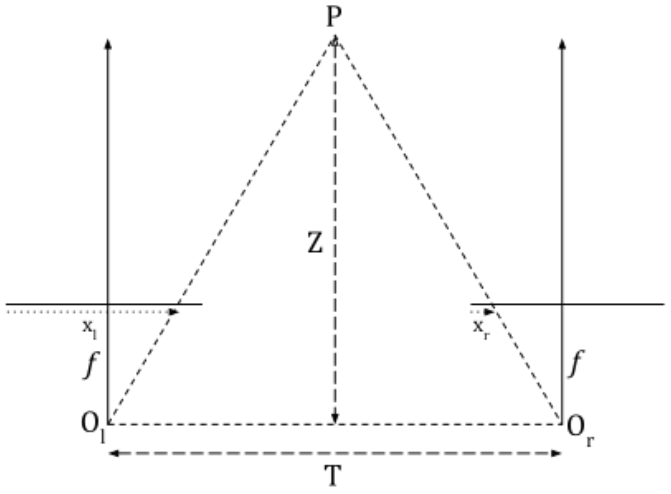


Figure 2.1: Stereo vision pinhole camera model adapted from Bradski and Daebler (2008)

3. The third step is called *correspondence* and finds the differences in x-coordinates of the same feature on the image planes of the left and right cameras. The result is called a disparity map and does not necessarily have to be the differences in x-coordinates. If two cameras are placed on top of each other, then we need to find the differences in y-coordinates. An overview and evaluation of different correspondence algorithms is covered by Scharstein et al. (2001).

4. The fourth step is called *reprojection* and uses the disparity map to calculate the actual depth of each pixel. The triangulation technique will be covered in Section 2.1.1.

2.1.1 Triangulation

Given that we have an undistorted, aligned stereo camera setup with known camera arrangement, we can calculate a depth map from a disparity map using triangulation. Fig. 2.1 shows a perfect stereo setup with equal and known focus length f for both cameras and known distance T between camera origins. The disparity d is found in the disparity map and is calculated from $x_l - x_r$. The depth Z is calculated from the properties of similar triangles using the equation:

$$\frac{T-(x_l-x_r)}{Z-f} = \frac{T}{Z} \implies Z = \frac{fT}{x_l - x_r}$$

The depth is inversely proportional to disparity as we can see from the formula, which gives a nonlinear relationship between the two terms. Fig. 2.2 shows a plot of this relationship with arbitrary values for focal length and distance between the cameras. The consequence of this relationship is that objects nearer the cameras has higher depth resolution than objects further away from the cameras.

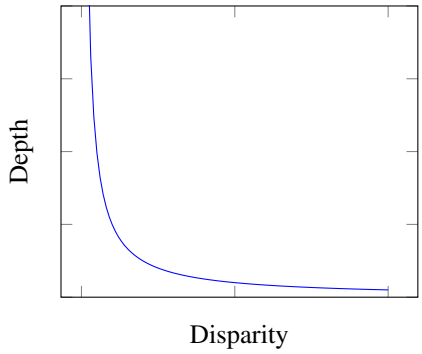


Figure 2.2: Relationship between depth and disparity in a stereo camera setup

2.2 3D Vision

Cameras with a built-in depth feature is a relatively new technology which has gained a lot of attention, especially in the robotics and computer vision areas. Many commercial 3D cameras such as Intel RealSense and the Kinect is available at affordable costs and has been found to be very useful for automation. The typical approaches for finding depth is by the use of *structured light* or *time of flight* which both are explained by Sarbolandi et al. (2015).

2.2.1 Structured Light

The structured light approach is quite similar to the traditional stereo vision approach only that one of the cameras are replaced by a projector. A known pattern, typically vertical stripes are projected onto the scene and a camera sees how the pattern is projected. Correspondence between the left and right image planes are easily found because of the projected pattern and from this a disparity map is created. Depth is calculated using the same method as for traditional stereo vision seen in Section 2.1.1. Fig. 2.3 shows a simplistic model of a structured light setup and the idea of projecting a known pattern.

2.2.2 Time of Flight

Foix et al. (2011) describes how time of flight cameras work, specifically those that use the continuous-wave modulation technology. Distance is calculated by emitting a signal towards the object and look at the signal reflected back at the camera. By measuring the phase shift of the reflected signal, the time the signal travelled can be calculated. The phase shift is calculated by using a method called four-bucket sampling (Kaufmann et al. (2004)), and is done by sampling the incoming signal four times (A_0, A_1, A_2, A_3) distributed equally over the period. Using this the phase shift φ is calculated.

$$\varphi = \arctan \frac{A_3 - A_1}{A_2 - A_0}$$

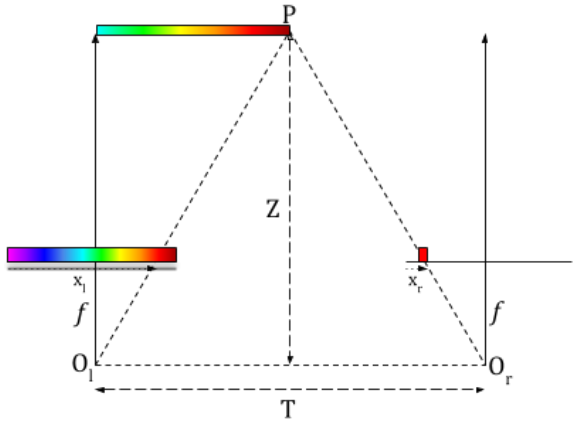


Figure 2.3: Structured light model. A projector located at O_l sends a known pattern onto the scene. A camera located at O_r sees the scene and how the pattern is projected onto it and a disparity map can be created.

The time travelled Δt is calculated by dividing the phase shift by the angular frequency of the emitted signal.

$$\Delta t = \frac{\varphi}{2\pi f}$$

The distance D is then easily found by using the distance formula with velocity equal to the constant c which is the speed of light and the time the signal travelled Δt.

$$D = \frac{1}{2} c \Delta t$$

We divide by 2 since the time the signal travelled is both ways, which is twice of what we are interested in.

2.3 Image Segmentation

Image segmentation is the process of dividing an image up so that every pixel is categorized into a group. If we would like to segment a specific object, the goal would be to gather the pixels that make up the object and put them into the same group. Smistad et al. (2015) has made an overview of state of the art image segmentation methods, and shown how well they are suited for optimization on the Graphics Processing Unit (GPU). We will begin this section by describing some of these methods and then cover a state of the art method using deep learning.

2.3.1 Level Set Method

The level set method introduced by Sethian (1999), finds boundaries of objects by evolving an initial curve to converge to the edges of objects. The level set equation is a partial

(a) An image of coins to be segmented **(b)** The speed at which the curve will evolve

Figure 2.4: Level set example

differential equation that can be solved numerically and looks like this.

$$\frac{\partial \phi}{\partial t} = -F|\nabla \phi|$$

This equation describes how a curve evolves over time with F being the speed at which the curve will evolve and $|\nabla \phi|$ is the gradient image of the current iteration of the curve ϕ. Given an initial curve we can apply an update rule to evolve the curve over iterations, which looks like this.

$$\phi_{n+1} = \phi_n - F|\nabla \phi_n|$$

As an example we can show how to segment the coins from the image in Fig. 2.4a. Our initial curve will be a rectangle following the border of the image and the speed F will be the gradient image passed through a sigmoid function to enhance the edges. The speed function looks like the image in Fig. 2.4b where black areas corresponds to a slow evolution of the curve and white areas is a fast evolution of the curve. Fig. 2.5 shows the evolution of the curve through several iterations.

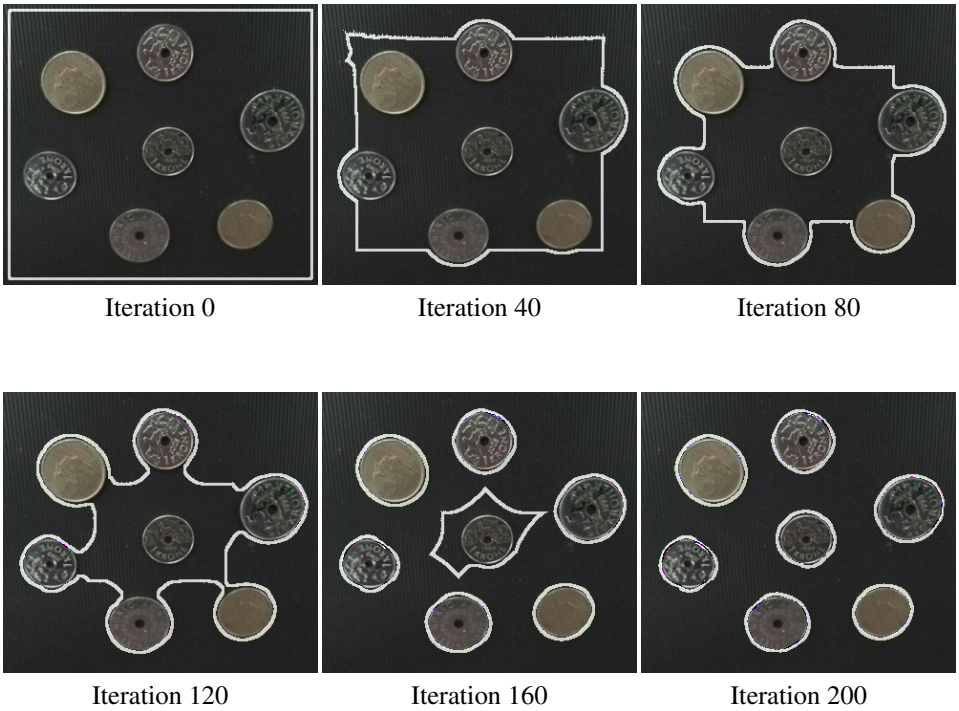


Figure 2.5: Curve evolution through iterations

2.3.2 Active Shape Models

ASM introduced by Cootes et al. (1995) is a technique for finding known shapes in an image. By labeling shapes in an image dataset, a Statistical Shape Model (SSM) can be trained to recognize similar shapes in new images. This is not to be mistaken with deep learning approaches using neural networks, since the SSM creation is based on statistical analysis to be trained. The training dataset is labeled by placing landmark points at protruding places around a shape. Each landmark point needs to be in the same anatomic position for all the training shapes. When labeling is done all shapes need to be aligned, which is typically done by using the Generalized Procrustes Analysis (GPA) introduced by Gower (1975). When the shapes are aligned a mean shape is calculated and Principal Component Analysis (PCA) (Flury (1988)) is used to calculate shape variations. The mean shape and its variations make up a SSM, which can be used by an ASM algorithm to find new shapes. ASM is a local search algorithm that displaces each individual landmark point to fit a new shape, to then use the SSM to verify that all the landmark points make up a valid variation of a shape. Both the creation of the SSM and the ASM algorithm will be covered more thoroughly in Chapter 3 and Chapter 4. Another algorithm that uses a SSM is the Active Appearance Model (AAM) algorithm (Cootes et al. (2001)), which in addition to using the SSM also uses texture information in the shapes.

2.3.3 Machine Learning Methods

Mask R-CNN He et al. (2017), is a machine learning method that is built upon the Region Based Convolution Neural Network (R-CNN) approach of detecting objects. The R-CNN method has two outputs in its last layer of the neural network, which is an object class and a bounding box surrounding the object. Mask R-CNN outputs the objects mask as a third element, which is the task of instance segmentation in computer vision. Mask R-CNN was tested on the COCO dataset which is an image dataset with over 300K images 1.5 million object instances. According to the paper, Mask R-CNN outperformed every other instance segmentation methods ran earlier, so it would be interesting to test its application towards a similar problem description as the one in this thesis.

2.4 Related Work

Here will related work in regards to image segmentation and weight estimation of fish be presented. The related work of image segmentation will mainly be focused on what has been done in the area of segmenting 3D images that has both color and depth components. The part regarding fish weight estimation will cover techniques for estimating weight using information like length, height and surface area, all values we can find by using the methods for this project.

2.4.1 3D Image Segmentation

Rother et al. (2006) introduced the term co-segmentation in image analysis for segmenting common parts of an image pair. Two color images containing the same foreground objects were used for the background subtraction. For finding the common objects in both images a histogram matching approach was used, and the ease of generalizing this method has motivated follow-up work.

The depth data of an image can also be used in a co-segmentation method as presented by Fu et al. (2015). Saliency models are used to find foreground objects in images by creating a saliency map from a combination of saliency maps from the color and depth images. The saliency map for the color component are created from a series of images of the same scene, which makes the resulting map a so called co-saliency map. This is not the case for the depth component where the saliency map is created from a single depth image. This led to not fully exploiting the depth information in the scene, and was later improved by Song et al. (2016).

Toscana and Rosa (2016) proposes a segmentation method mainly designed for robot grasping. A modified canny edge detector combining information from both color and depth maps is used to build an undirected graph of the image. The graph is then partitioned using internal and external differences between graph regions.

2.4.2 Fish Weight Estimation

O. et al. (2010) estimated weight by analyzing an image of a fish in a controlled environment. Surface area of the fish was extracted, and width and length was found by fitting

the smallest possible rectangular bounding box around the fish. These variables was then used and compared in different regression models, where a simple regression model based on power curves was found to give the smallest error. This also made the calculation of the bounding box obsolete, since surface area was the only variable used. The regression model looked like this.

$$Y = AX^B$$

Where $Y =$ weight(kg), $X =$ view area(cm2) and A and B are coefficients related to the fish species.

Viazzi et al. (2015) used the same regression models as O. et al. (2010), but found that the linear model

$$Y = A + BX$$

gave the best result.

Chapter 3

Methods

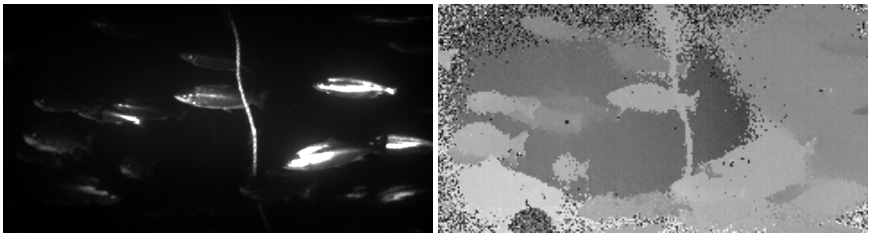
In this chapter, we will look at which methods have been used in order to solve the problem of estimating fish weight from 3D images. SINTEF Digital in cooperation with Odos Imaging Limited and Bright Solutions, has developed an underwater 3D camera that estimates depth in real-time. This camera has been used to create an image dataset of salmon swimming inside a tank, which is the data the methods in this thesis is testing on. Both the 3D camera and the dataset will be described in this chapter, followed by the techniques used for segmentation and weight estimation.

3.1 3D Camera

The 3D camera developed by Risholm et al. (2018) uses a time of flight technique called range gating to estimate depth. A laser with a wavelength of $532nm$ is used to illuminate the scene in front of the camera and a Complementary metal–oxide–semiconductor (CMOS) sensor is used to capture and convert the reflected photons. In front of the CMOS sensor is a fast shutter mechanism that can open and close in $1.67ns$. By having the shutter mechanism open and close at different time periods after the laser emits a signal, the distance the received light has traveled can be estimated. Because of the shutter time being $1.67ns$ and the light has to travel back and forth, the minimum spatial sample increment is $\Delta_z = \frac{1}{2}\Delta_t C_w = 18.8cm$ when the light speed constant C_w is set to $22.5cm/ns$ which is the speed light travels underwater. The intensity image and depth image in the dataset, is created by taking images of 25 ranges with four exposures per frame for a total of 100 images per frame. This results in a camera that takes pictures 10 times a second with depth estimates up to $8m$.

3.2 Dataset

The dataset is a video made up of 809 images of salmon swimming inside of a fish tank. All images has an intensity component and a depth component and the frame rate of the video



(a) Example intensity image **(b)** Example depth image

Figure 3.1: Dataset frame

is 10 frames per second giving 81 seconds of footage. Fig. 3.1 shows an example of a frame in the video where Fig. 3.1a shows the intensity component of the image and Fig. 3.1b shows the depth component of the image. All images from the dataset added in this thesis, are added with permission from SINTEF.

3.3 Statistical Shape Model

The fish we would like to segment from the dataset all have similar shape with some variations. Some fish may be larger than others, some can swim perpendicular to the camera direction while some swim slightly directed towards the camera. SSMs are a statistical model of a shape that catches these slight variations in shape. A SSM needs to be trained from a labeled dataset and can be used to segment shapes in new images. The labeled dataset consists of several fish where their shape has been represented by a set of landmark points shown by Fig. 3.2. Before creating the SSM, all shapes needs to be aligned using GPA. After alignment PCA can be performed to create the SSM, which will contain a mean shape and its variations. Both GPA and PCA will be described in detail below.

3.3.1 Generalized Procrustes Analysis

Procrustes analysis introduced by Gower (1975), takes a set of shapes and finds the translation, scale and rotation to align them all. Procrustes analysis minimizes what is called the Procrustes distance, which for two shapes with k points each, $(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)$ and $(z_1, w_1), (z_2, w_2), ..., (z_k, w_k)$ can be computed using:

$$d = \sqrt{(x_1 - z_1)^2 + (y_1 - w_1)^2 + ... + (x_k - z_k)^2 + (y_k - w_k)^2}$$

To achieve minimization we start by translating the mean of the shapes to the origin $(0, 0)$

$$\bar{x} = \frac{x_1 + x_2 + ... + x_k}{k}, \bar{y} = \frac{y_1 + y_2 + ... + y_k}{k}$$

All points are translated in the following way:

$$(x', y') = (x - \bar{x}, y - \bar{y})$$

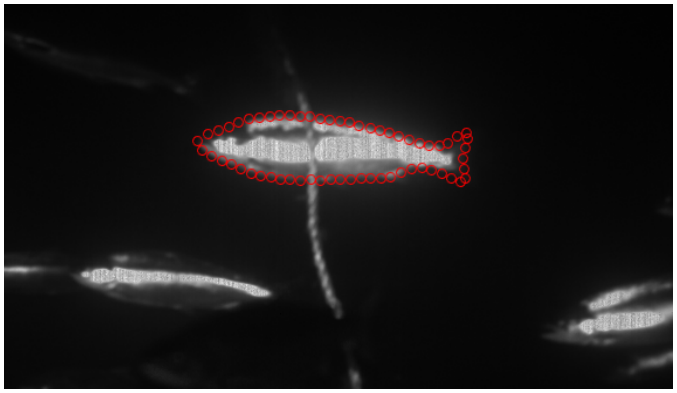


Figure 3.2: A fish represented by landmark points

Scaling is performed by finding the scaling factor that will set the root-mean-square deviation (RMSD) to 1.

$$s = \sqrt{\frac{(x_1 - \bar{x})^2 + (y_1 - \bar{y})^2 + ... + (x_k - \bar{x})^2 + (y_k - \bar{y})^2}{k}}$$

All points are scaled in the following way:

$$(x', y') = ((x - \bar{x})/s, (y - \bar{y})/s)$$

Rotating is trickier since there is no standard reference shape. GPA solves this by first choosing a arbitrary reference shape and then perform regular Procrustes analysis. When done if the Procrustes distance is larger than some threshold between the mean shape and the reference shape, then the mean shape is used as reference shape and the method is repeated. To find the angle θ to rotate a shape towards a reference shape, Singular-value decomposition (SVD) is used.

3.3.2 Principal Component Analysis

Principal component analysis is widely used to statistically analyze multidimensional data. Here we will describe PCA with relevance to SSM creation and ASM search, but a more thorough explanation can be found in numerous sources like Flury (1988).

If we consider a set of data points in two dimensions as shown in Fig. 3.3, the eigenvectors of the covariance matrix plotted in blue and green, will be the directions of variance in the data. These eigenvectors together with their eigenvalues are what's called principal components. The blue vector is the first principal component since it has the largest possible variance. This is given by the eigenvalue, and the vector is scaled by a factor of the square root of the eigenvalue. This method of computing the PCA is called the covariance method, which is further explained by Shlens (2014). In the case of ASM, the data we want to analyze will be in $2k$ dimension, where k is the number of landmark points we use to represent the fish shape. We multiply by 2 because of the points themselves being in

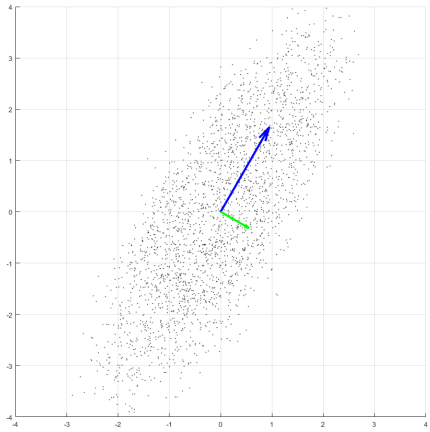


Figure 3.3: Eigenvectors of the covariance matrix of the distributed points

two dimensions. If we use the notation $(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)$ for the landmarks in the first shape, $(z_1, w_1), (z_2, w_2), ..., (z_k, w_k)$ for the landmarks in the second shape and $(u_1, v_1), (u_2, v_2), ..., (u_k, v_k)$ for the landmarks in the nth shape, our data matrix will look like the following with dimensions $2k \times n$:

$$A = \begin{bmatrix} x_1 & z_1 & ... & u_1 \\ y_1 & w_1 & ... & v_1 \\ x_2 & z_2 & ... & u_2 \\ y_2 & w_2 & ... & v_2 \\ ... & ... & ... & ... \\ x_k & z_k & ... & u_k \\ y_k & w_k & ... & v_k \end{bmatrix}$$

This matrix can be used to find the covariance matrix $C = cov(A^T)$, which will have the dimension $2k \times 2k$. The principal components will be the eigenvectors and eigenvalues of the covariance matrix which are defined by:

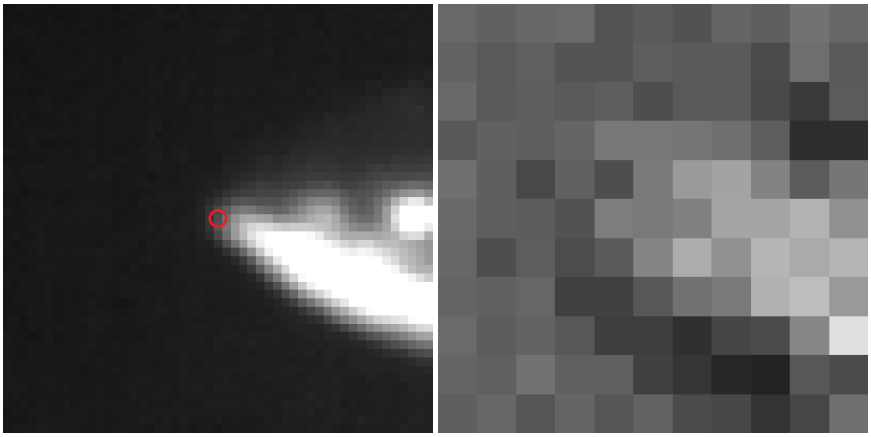
$$Cv = \lambda v$$

where v are the eigenvectors and λ are the eigenvalues. Variations of a fish shape can then be found by a linear combination of the mean shape $\bar{S}$ and weighted principal components b:

$$S' = \bar{S} + vb$$

where only the most significant principal components is used and b is a vector of weights equal to the number of principal components constrained by the particular principal component's eigenvalue.

$$-3\sqrt{\lambda_k} \leq b_k \leq 3\sqrt{\lambda_k}$$



(a) Landmark point of the tip of a fish's head

(b) Gradient of 11×11 region around landmark

Figure 3.4: Gray-level profile of landmark

3.4 Active Shape Model Search

The search algorithm used in this project follows the work done by Milborrow (2016). Before the actual search, every landmark point needs to build a gray-level profile model. This model is used during the search to find which direction the individual landmark point needs to be moved to converge into the new shape. A profile model is built from finding the gradient of a $s \times s$ region around each landmark point in all of the training shapes. Fig. 3.4 shows an example landmark on the tip of a fish's head and a 11×11 normalized gradient image around the landmark. All the gradient images are thereby reshaped into a 1D vector of size s^2, and the mean and covariance matrix is calculated for each landmark point across all training shapes. As seen for the statistical shape model, PCA is also performed on the gray-level profiles and stored with the profile model to test the validity of a landmark point during search.

When the gray-level profile model is built, The mean shape of the SSM is placed at an initial point where it is estimated that a fish will be. This is done by smoothing the image we would like to search with a very high smoothing value. By doing this we look at the image as a height map, where there is likely to be fish at high intensities. By finding all local maxima, we have a set of points we can use as initial points when starting the search.

The search is an iterative search that begins with going through all the landmark points of the current shape and look at a 3×3 square region and use all these pixels as a center to calculate 9 gradient images. This is illustrated by Fig. 3.5 using a square region of size 11×11, only the center pixel (the current landmark point) is left out. All these gradient images are tested towards the mean gray-level profile, and the one that is most equal to the mean is the direction of which the current landmark will be shifted. To calculate which gradient image is most equal to the mean, we use the information we stored in the gray-level profile model when we performed PCA. If $\bar{g}_i$ is the mean gray-level profile for the

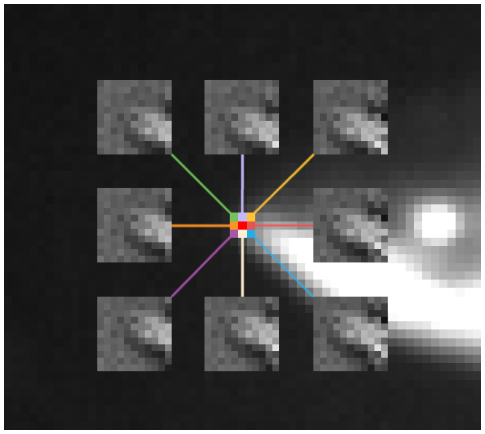


Figure 3.5: Gradient images of a square region around pixels surrounding a landmark point shown in red

current landmark i and g is a reshaped gradient image from one of the 3×3 surrounding pixels, we can calculate the distance metric which is the Mahalanobis distance mentioned by Milborrow (2016).

$$m = \sqrt{(g - \bar{g}_i)^T C_g^{-1} (g - \bar{g}_i)}$$

where C_g^{-1} is the inverse covariance matrix of the gray-level profile data. The pixel surrounding the current landmark that minimizes m is chosen as the new position of the current landmark.

When all landmarks are shifted to their new positions, GPA is done to fit the current shape s_curr to the new landmark points. From this we can deform the fitted shape S into a new shape variation using PCA, by calculating the weighted principal components and constraining them by the eigenvalues:

$$b = v^T (S - \bar{S})$$

and then calculating the new shape variation:

$$S' = \bar{S} + vb$$

3.5 Salmon Weight Estimation

The weight estimation technique is using a fairly straight forward approach. There are three measurements we can derive for a fish using the depth image and the segmented intensity image, length L, height H and surface area A. First we need to find the distance the fish is from the camera. We do this by checking what pixels in the segmented intensity image corresponds to in the depth image. These points are stored as a vector and we find the standard deviation of the set of points. If the standard deviation is bigger than some

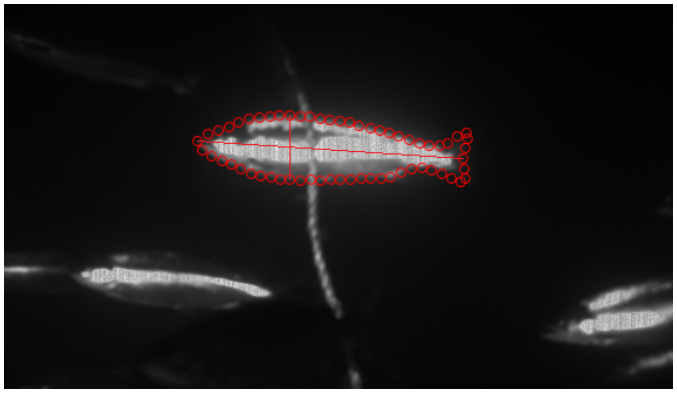


Figure 3.6: Exploiting the landmark points to get easy access to the fish's length and height in pixels.

threshold we say that the fish are not valid for finding depth data because the depth image is to noisy. If we pass the test we take the mean of all the points and use that as our distance Z to the fish.

The neat feature of active shape models that appear during this part of the process is that we can easily find how many pixels long a fish is. The same goes for the height in pixels, because we know what landmark point is at the tip of the fish and we know what landmark point is at the tail of the fish. We say that the length between these landmark points is the length of the fish, which is equally true for the height shown in Fig. 3.6.

We use the notation L_p, H_p for the length and height measured in pixels, and the area A_p of the fish is just the number of pixels the shape consists of. To project these values from image space into world space we need to know the focal length of the camera measured in pixels. By looking at the pinhole camera model shown by Fig. 3.7 we can use some trigonometry to find the focal length.

If we know the horizontal angle of view θ and the image width in pixels W, the focal

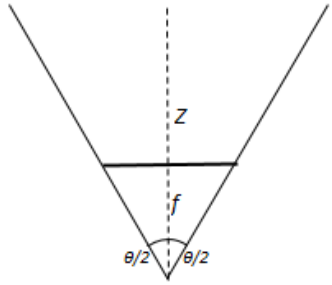


Figure 3.7: Simple pinhole camera model

length in pixels is given by:

$$f = \frac{W}{2tan(\frac{\theta}{2})}$$

When we know the focal length in pixels it is straight forward to project from image space to world space by the use of similar triangles.

$$L = Z\frac{L_p}{f}, \quad H = Z\frac{H_p}{f}, \quad A = Z^2\frac{A_p}{f^2}$$

Using these calculations in reality is most likely not sufficient for accuracy. This will be further explained in Chapter 6.

Chapter 4

Implementation

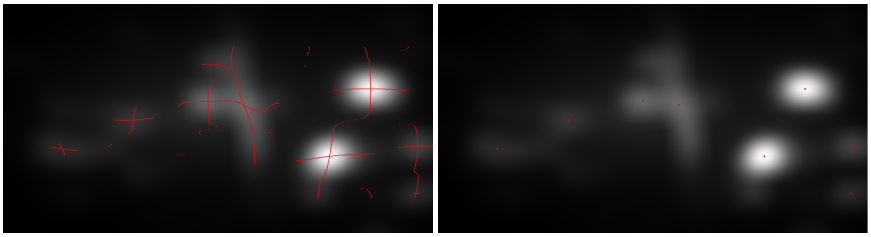
In this chapter, the implementation of the methods described in Chapter 3 will be covered. The main tool used for developing has been MATLAB, with its high level functionality for image processing. The first part of this chapter will explain the image dataset and its labeling. Next we will cover the segmentation part and the active shape model implementation. And lastly the implementation of the weight estimation method will be explained.

```
root/
├── Data/
│   ├── images/
│   ├── images_labeled/
│   ├── images_segmented/
│   ├── landmarks/
│   ├── FishGrayModel.mat
│   ├── FishInCage.nc
│   └── FishShapeModel.mat
└── Matlab/
    ├── Labeling/
    ├── LoadData/
    ├── Segmentation/
    └── WeightEstimation/
```

Figure 4.1: Folder structure of the project

4.1 Dataset and Labeling

The dataset of the 809 frames video is stored in the NetCDF file format, which is a sort of standard in the scientific community for storing array-oriented data. The people behind the 3D camera the dataset is created by, has also their own project UTOFIA (UTOFIA (2018)) for testing underwater applications of the camera technology. On their web page a loader



(a) Local maxima of all rows and all columns

(b) The local maxima intersection between rows and columns

Figure 4.2: Local maxima of highly smoothed intensity image

for the NetCDF file, can be downloaded and is included in the LoadData/ folder in the project. The loader is written in MATLAB and reads all the raw data from the file FishInCage1.nc to then do the calculations necessary to generate an intensity image and depth image for every frame. All the images are read through this loader and stored in a MATLAB friendly .mat format in the images/ folder.

The labeling of the images are done partially manual and automatic. 30 fishes has been labeled manually by creating a binary image where all the pixels that make up the fish are set to one and zero elsewhere. An automatic landmark generator is written inside the Labeling/ folder to create 60 landmark points around the contour of the fish and store them in a .csv file inside the images_labeled/ folder.

4.2 Active Shape Model Implementation

The ASM implementation is based on the work done by Miller (2017), who has made a MATLAB implementation of the ASM technique from the method described by Milborrow (2016). The original implementation was meant for locating faces in images, and trained its shape model using the MUCT database (Milborrow et al. (2010)).

The first problem we encounter when implementing the ASM search, is how do we know where there is likely to be a fish in an image. The way we find our potential points to search for fish, is by taking the Gaussian blur of the image using a very large smoothing value to view the image as a height map. Then we find all the local maxima along all the rows and all the columns as shown in Fig. 4.2a. We can see that we get several cross sections, which is the intersection points shown in Fig. 4.2b. The ASM search is run for every intersection point we find in an image, but before the search starts we need to build the SSM and the gray-level profile model.

The SSM is built by first aligning all training shapes using GPA without taking consideration to the scaling between the shapes, because we need the search to able to find fish of different sizes. Fig. 4.3 shows the landmark points of all training shapes after Procrustes analysis is performed. PCA is performed on the training shapes by using MATLAB's built in functionality for finding eigenvectors and covariance matrices. Fig. 4.4 shows the variation of shapes by tweaking the three principal components that has the most influence on the shape (where the variance is largest).

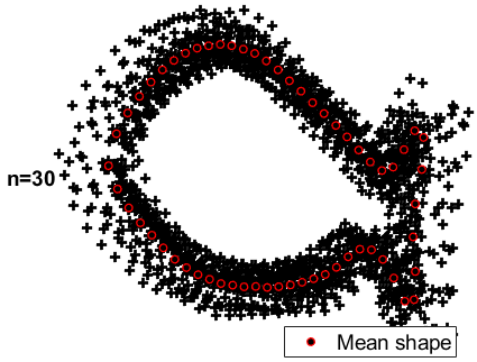


Figure 4.3: The training shapes aligned using Procrustes analysis.

The gray-level profile model follows the method for building as described in Chapter 3, only we need to build gray-level profiles for different resolution. The search will start of at a very coarse resolution of $\frac{1}{4}$ times the original resolution, then the search will move on to do the same operations at only half the resolution, and lastly the full resolution will be used. This means that we need to generate gray-level profiles for all these three resolution for every landmark point in every training shape. The size of the square region around each landmark point will also vary between each resolution ranging from 10×10 down to 5×5. To find the gradient image of these images, the Laplacian kernel is used in the convolution process which looks like the following:

$$L = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

The gradient image is then normalized and passed through a sigmoid function to limit very high and low values.

The ASM search itself does not contain any implementation specific details other than what is already explained in Chapter 3. The search is ran for every detected peak in the highly smoothed image that finds potential fish positions. The search is run over three different resolutions and five iterations for each resolution. For a fish to be validated to be a true positive we check the Mahalanobis distance metric and if it is below a certain threshold we accept it as a valid segmented fish. The final contour of the fish is found from drawing lines between the final landmark points and all the pixels containing the fish is found from running a region growing algorithm using 4-connectivity starting from the mean pixel. The segmented fish is saved as a binary image in the images_segmented/ folder. The length and height in pixels of the fish are stored in the filename of the binary image and are found from taking the distance between known landmark points. The length is defined as the length between the landmark points at the head and the tail, and the height is found from the landmark points marking the top and bottom of the fish.

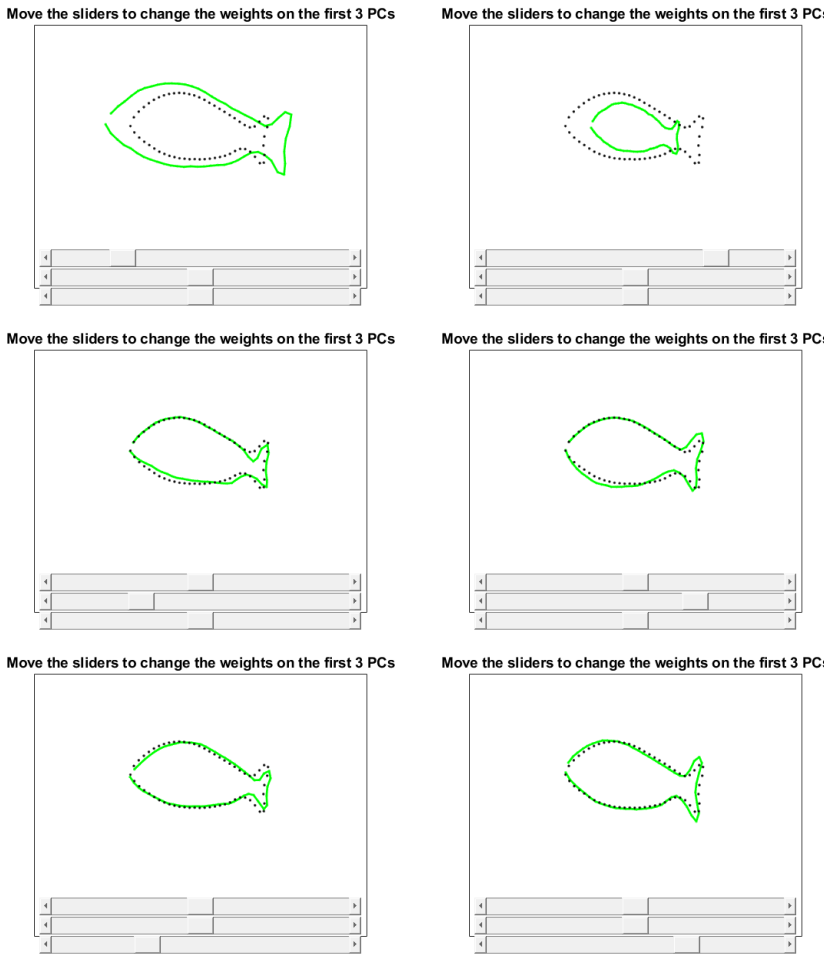


Figure 4.4: Variations of the 3 most influencing principal components

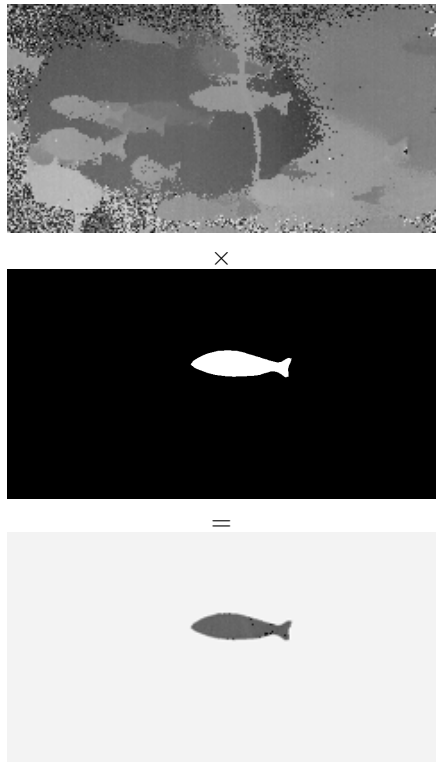


Figure 4.5: Extraction of depth pixels from depth image and the binary segmented image

4.3 Weight Estimation Implementation

The weight estimation implementation is made simple by manually selecting segmented images that have succeeded in converging to a fish contour. Each of these images is used in combination with the corresponding depth image to extract the depth pixels of the fish. This can be visualized by Fig. 4.5. From these depth values we extract the mean depth and the standard deviation of the depth points. The standard deviation tells us how noisy the data points are, and we use the mean as the distance to the whole fish.

We need to calculate the focal length to be able to project distances into world space. This is done using the formula derived in Chapter 3, and we know from the camera documentation that the screen width in pixels are $W = 960$ and the horizontal angle of view is $\theta = 63.42°$. This gives the focal length

$$f = \frac{960px}{2tan(\frac{63.42°\pi}{2*180})} = 777px$$

In Chapter 5 we will use this focal length to project distances in image space into distances in world space and show the results.

Chapter 5

Results

In this chapter, we will show the results from running our implemented methods using the given dataset. The results of the image segmentation is presented, with both successful and failed segmentations. The advantages of the ASM algorithm will be shown alongside its disadvantages and this will be discussed thoroughly in Chapter 6. We will also look at the results from our weight estimation method, but will sadly lack an estimate of precision of the results since it's not given how large the fish in the dataset actually are.

5.1 Image Segmentation Results

The results of the image segmentation depends a lot on what threshold we choose for the acceptance of a valid segmentation. We threshold the Mahalanobis distance metric, which tells us how well all the landmark points fit in relation to the gray-level profile. If we set a low threshold we get acceptable segmentations, but very few fishes are found. If we set a higher value, lots of fish are found but many are false positives and the contour is not completely accurate. For all segmentation images shown in this chapter, the red dotted shape is the initial mean shape that guesses where a fish is. The green contour is the final shape retrieved when the ASM algorithm has finished.

The implementation is not very optimized and timings say that it takes 3-4 seconds to process a single search for a fish in a single image. The implementation is tested on a system having an intel i7-3770 Central Processing Unit (CPU) and a NVIDIA GeForce GTX 980 GPU.

Fig. 5.1 shows a segmented image that has used a very low threshold value for the Mahalanobis distance. We can see that the fishes found have a very accurate contour, but there is many fish in the image that doesn't get segmented. Fig. 5.2 however uses a high threshold which means that more segmentations pass the validation test. This leads to us finding more fish, but most of the contour estimates are inaccurate.

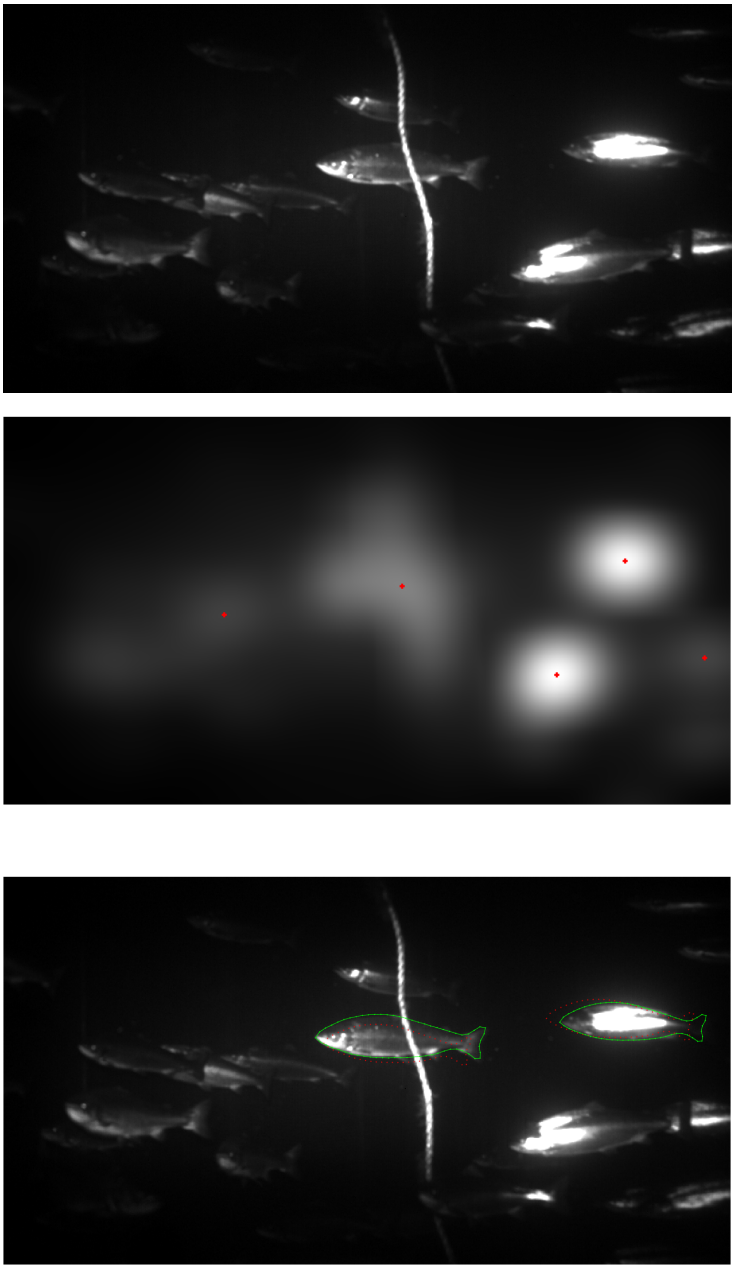


Figure 5.1: Original image, highly smoothed image with potential fishes and segmented image when using a low threshold for the Mahalanobis metric.

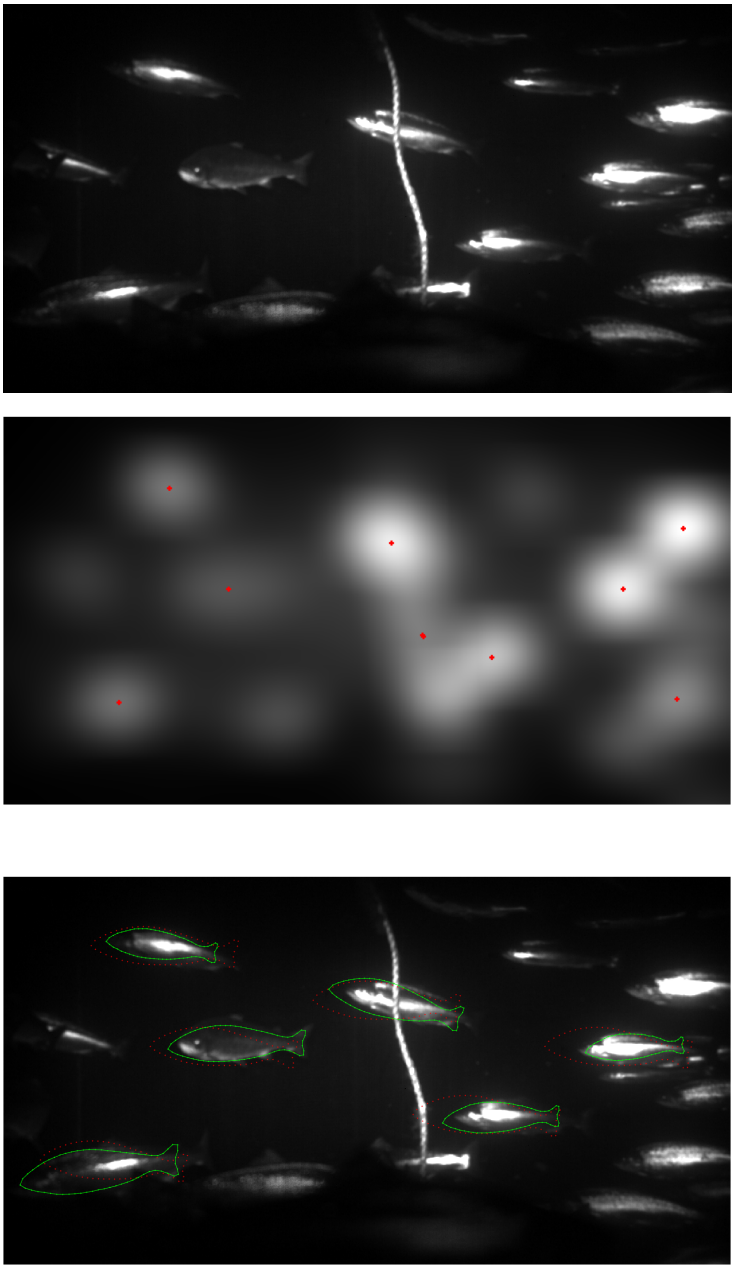


Figure 5.2: Original image, highly smoothed image with potential fishes and segmented image when using a high threshold for the Mahalanobis metric.

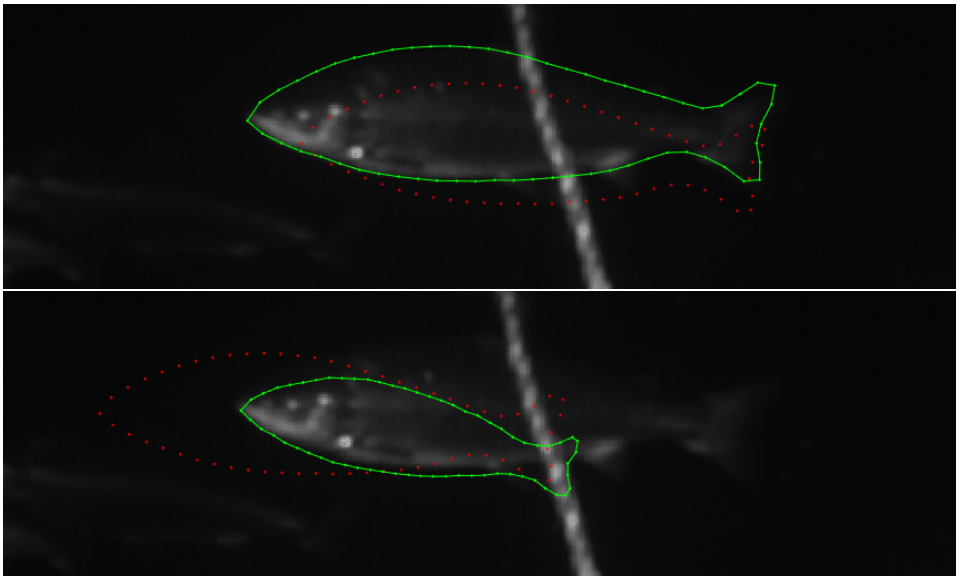


Figure 5.3: Importance of the placement of the initial shape.

Fig. 5.3 shows an example of the importance of the initial shape. The top image shows a successful convergence to the contour of the fish, but the bottom image had a different initial shape which caused the search to create a false positive of the convergence.

Fig. 5.4 and Fig. 5.5 shows an advantage and a disadvantage of using the ASM algorithm. The advantage is that even though the contour of the fish is not all that clear, the ASM algorithm is able to find a part of the contour to then estimate the rest, by constraining the landmarks to a valid shape according to the SSM. A disadvantage is when one fish partly occludes another. The final contour contains two fishes, but is very similar to only one.

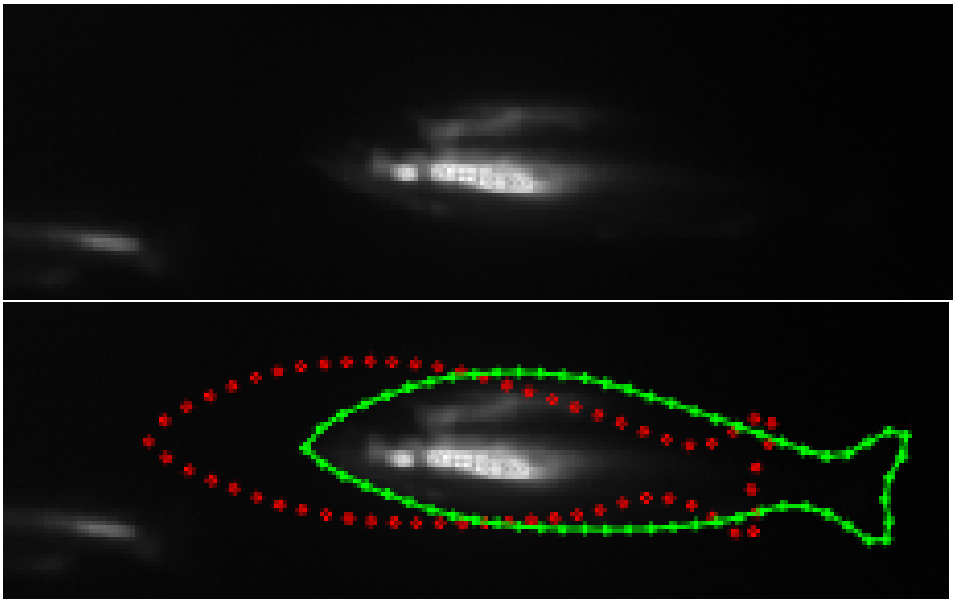


Figure 5.4: Advantage of the ASM algorithm.

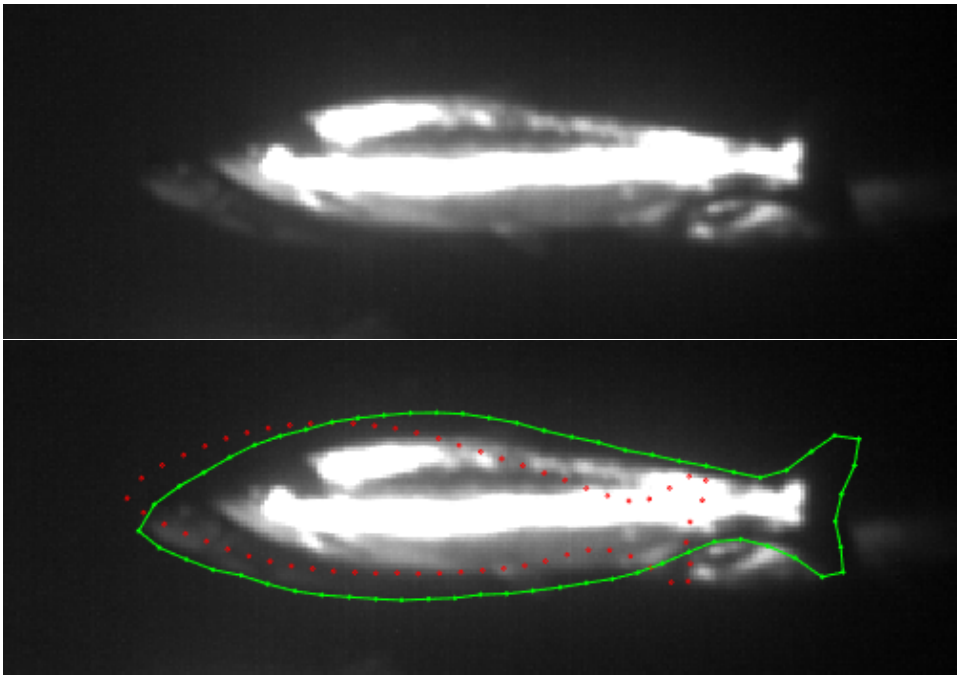


Figure 5.5: Disadvantage of the ASM algorithm.

5.2 Weight Estimation Results

Table 5.1 shows the mean distance measured μ, the standard deviation σ, the length, height and area in pixels L_p, H_p, A_p and the length, height and area in meters L, H, A, of 10 different well segmented fishes.

Table 5.1: Results of doing weight estimation on ten different fishes.

μ	σ	L_p	H_p	A_p	L	H	A
2.989	2.261	188	50	6769	0.723	0.192	0.100
2.637	1.837	218	57	9049	0.740	0.193	0.104
3.054	2.281	189	50	6825	0.743	0.197	0.105
2.996	2.243	193	51	7084	0.744	0.197	0.105
2.870	2.203	188	49	6724	0.694	0.181	0.092
2.791	2.008	219	57	9043	0.787	0.205	0.117
2.702	1.994	197	52	7405	0.685	0.181	0.090
3.206	2.477	182	49	6515	0.751	0.202	0.111
3.079	2.349	176	47	6022	0.697	0.186	0.095
2.567	1.848	218	57	8973	0.720	0.188	0.098

The values L, H, A are what we want to use when estimating the weight of a fish. In the related work from Chapter 2 these values are passed through functions which has been found through regression analysis. I have decided to not do these measurements on the data I have found since there is no way to tell what weight the fish in the tank really are, and therefore no way of telling how accurate the estimation is. What we can observe however, is that L, H and A comes out to be similar even though L_p, H_p and A_p varies. This comes down to how accurate the distance μ is, which again is something that needs to be tested when the filming of the fish is done.

Chapter 6

Discussion

In this chapter, we will go through our results from Chapter 5 and elaborate on what we found from running our implementation. We will keep the discussion around the segmentation and its improvements as well as the weight estimation results.

6.1 Image Segmentation

From the results of the image segmentation we can see that it has potential for improvement. If we set the threshold value for the Mahalanobis metric too small, we are only able to segment a minority of the fishes. If we set the threshold value too high we get a lot of false positives that has not fully found the contour of the fishes. There are ways of solving this issue and improving the segmentation method itself is one of them. We can split the proposed solutions into three categories:

- Incorporate more and better training data into the SSM and the gray-level profile model.
- A better technique for positioning the initial shape before the beginning of the search.
- Improving the search algorithm itself

6.1.1 Better Training Data

Labeling datasets are a slow and cumbersome process that takes a lot of time. Good training data is important for building a SSM that generalizes well across the dataset. The SSM created for the face segmentation in the original implementation, was built from 228 training shape. The implementation presented in this thesis only had 30, which is few.

6.1.2 Better Initial Positioning of Mean Shape

The method used to find positions of potential fish, is a very naive approach which was shown by Fig. 5.3. It is based on the idea of where the most light is reflected back onto the CMOS sensor of the camera, is where there are the highest probability of being a fish. This is a reasonable assumption only that we do not know what part of the fish the light were reflected. It is equally probable that it is the tail than it is the head. This is not optimal for placing the initial shape to hope for convergence to the contour of the fish. An interesting idea that I believe is worth looking into, is the combination of ASM search with deep learning methods like Convolutional Neural Network (CNN). If a CNN could do the coarse work of finding the bounding box containing a fish, the ASM search could have a better starting point for the fine work of extracting the contour.

6.1.3 Improving the Search

ASM is a well known segmentation algorithm that has undergone extensive research, and many improvements to the algorithm has been proposed. The similar method to ASM is AAM (Cootes et al. (2001)), and is one of these improvement methods. Instead of just using a SSM and some information of the region around each landmark point, AAM also uses texture information about the objects.

Another proposed improvement technique by Eguizabal and Schreier (2017). is the idea of weighting the landmark points that are considered to have a good placement more than those that do not. This can be done by measuring the Mahalanobis distance metric for each landmark point in every iteration and make the points with lower scores have higher influence on the generated shape for the next iteration.

In addition to the ASM algorithm, other methods can be incorporated as well such as object tracking using the kalman filter (Kalman (1960)). If a shape is detected with high accuracy in one frame, the kalman filter technique can track the shape's movement across the image and estimate where it will be in case the detection score is low in another frame.

6.2 Advantages of ASM

Cootes et al. (1995) mentions that the advantages of the ASM technique is its possibility of segmenting objects in noisy and cluttered environments. Fig. 5.4 is an example of exactly this, where not all of the fish is seen in the camera, but the algorithm is still able to estimate the contour well.

Another advantage is the ability for easily extracting the length and height of a fish based on known landmark positions. It's not all segmentation algorithms that is also able to find this type of information about the segmented object, which means another method needs to search for both the head and tail tip of a fish to then find the length.

6.3 Disadvantage of ASM

ASM is an edge based segmentation algorithm, which means that we perform our search on gradients in the image. This loses information about the intensity inside of the contour

of the object, which means we are not exploiting all of the available information in the image when performing the segmentation. This is shown by Fig. 5.5 when one fish are occluding another so both are find to be inside of the contour.

6.4 Weight Estimation

The weight estimation is really a matter of finding relations in lots of data. Ideally it would be known the exact weight of every fish in every image for then trying to fit a regression model to the data. This is a process that needs to be done when the fish is at different stages in life to see how length, height and surface area is related to the weight of a fish. Norsk Institutt for Naturforskning (NINA) (2016) has done this by not fitting a regression model, but rather creating a look-up table that maps fish length to fish weight.

It must be emphasized that the equations used for calculating the focal length f in Chapter 3, is not the ideal way of finding this value. Although the formula is correct, it is prone to error due to small manufacturing inaccuracies. The way to estimate the focal length is by having the camera look at a known pattern from different angles, and letting existing software calibrate the camera parameters for us.

Chapter 7

Conclusions and Future Work

In this thesis, we have investigated the possibility of automating the measurements of fish weight by the use of the information we can be get from an underwater 3D camera. We have found that this is a hard problem which have a lot of use out in the industry at fish breeding facilities. What makes the problem hard is a combination of finding a robust segmentation algorithm for detecting fish in images, and handling noisy depth information to project the size of the fish from image space to world space.

7.1 Conclusions

We have proposed an image segmentation method based on active shape models to extract the contour of fish in images from an underwater range-gated 3D camera. The method have lots of room for improvement, but can be seen as a proof of concept for segmenting these types of images.

Even though the weight and size estimations are lacking results to prove that the proposed method can be of use, this thesis can still be of use as a baseline for future research in this field. Results are lacking due to not knowing the correct camera parameters to project from image space to world space, and the fact that we do not know exactly how big the fish in the dataset are in reality.

7.2 Future Work

We proposed several ideas for improving the segmentation algorithm in Chapter 6. One that is easy to do but takes a lot of time is to label more training data. The training data used in the implementation for this project is very small, and it would be interesting to see what would happen to the segmentation with the use of 50, 100 and even 200 more training shapes.

An optimization of the ASM implementation is possible as described by Smistad et al. (2015). The suggestion is to exploit the parallelism of the GPU by making use

of many more processing cores. The update of each individual landmark point can be ran in parallel with synchronization between iterations. The paper reports that a large amount of landmark points are needed to gain any significant speedup, but something along these lines are needed if the algorithm should be used for real-time applications, which is not strictly needed for the problem this thesis is built on. It would be interesting to test the possibility of not only parallelize all the landmark points, but also search for several fish in the same image at the same time.

Another possibility is to work in the same field only on a different project regarding surveillance of fish in a tank, such as population control by counting the fish or detect if the Salmon is infested with lice.

Bibliography

[1] Bradski, G., Daebler, A., 01 2008. Learning opencv. computer vision with opencv library, 415–453.

[2] Cootes, T., Taylor, C., Cooper, D., Graham, J., 1995. Active shape models-their training and application. Computer Vision and Image Understanding 61 (1), 38 – 59. URL http://www.sciencedirect.com/science/article/pii/S1077314285710041

[3] Cootes, T. F., Edwards, G. J., Taylor, C. J., Jun 2001. Active appearance models. IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (6), 681–685.

[4] Eguizabal, A., Schreier, P. J., 2017. A weighting strategy for active shape models. CoRR abs/1707.09233.
URL http://arxiv.org/abs/1707.09233

[5] Flury, B., 1988. Multivariate Statistics: A Practical Approach. Chapman & Hall, Ltd., London, UK, UK.

[6] Foix, S., Alenya, G., Torras, C., Sept 2011. Lock-in time-of-flight (tof) cameras: A survey. IEEE Sensors Journal 11 (9), 1917–1926.

[7] Fu, H., Xu, D., Lin, S., Liu, J., June 2015. Object-based rgbd image co-segmentation with mutex constraint. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[8] Gower, J. C., Mar 1975. Generalized procrustes analysis. Psychometrika 40 (1), 33–51.
URL https://doi.org/10.1007/BF02291478

[9] Hao, M., Yu, H., Li, D., Sep. 2015. The Measurement of Fish Size by Machine Vision - A Review. In: 9th International Conference on Computer and Computing Technologies in Agriculture (CCTA). Vol. AICT-479 of Computer and Computing Technologies in Agriculture IX. Beijing, China, pp. 15–32.
URL https://hal.inria.fr/hal-01614170

[10] He, K., Gkioxari, G., Dollár, P., Girshick, R. B., 2017. Mask R-CNN. CoRR abs/1703.06870.
URL http://arxiv.org/abs/1703.06870

[11] Kalman, R. E., 1960. A new approach to linear filtering and prediction problems. Journal of basic Engineering 82 (1), 35–45.

[12] Kaufmann, R., Lehmann, M., Schweizer, M., Richter, M., Metzler, P., Lang, G., Oggier, T., Blanc, N., Seitz, P., Gruener, G., Zbinden, U., 09 2004. A time-of-flight line sensor - development and application, 192–199.

[13] Milborrow, S., 2016. Multiview Active Shape Models with SIFT Descriptors. Doctoral Thesis. University of Cape Town (Department of Image Processing).

[14] Milborrow, S., Morkel, J., Nicolls, F., 2010. The MUCT Landmarked Face Database. Pattern Recognition Association of South Africahttp://www.milbo.org/muct.

[15] Miller, J. W., 2017. Active shape models for face detection. https://github.com/johnwmillr/ActiveShapeModels.

[16] Norsk Institutt for Naturforskning (NINA), 2016. Lengde og vekttabell for laks. [Online; accessed 08-August-2018].
URL https://lakseelver.no/news-2016/gjenutsetting-av-laksefisk

[17] O., B. M., Ünal Şengör Gülgün F., Gil, S. M., Guillén, R. E., 2010. Using image analysis to predict the weight of alaskan salmon of different species. Journal of Food Science 75 (3), E157–E162.
URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1750-3841.2010.01522.x

[18] Risholm, P., Thorstensen, J., Thielemann, J., Kaspersen, K., Tschudi, J., Yates, C., Softley, C., Abrosimov, I., Alexander, J., Henrik Haugholt, K., 05 2018. Real-time super-resolved 3d in turbid water using a fast range-gated cmos camera 57, 3927.

[19] Rother, C., Minka, T., Blake, A., Kolmogorov, V., June 2006. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). Vol. 1. pp. 993–1000.

[20] Sarbolandi, H., Lefloch, D., Kolb, A., 2015. Kinect range sensing: Structured-light versus time-of-flight kinect. Computer Vision and Image Understanding 139, 1 – 20.
URL http://www.sciencedirect.com/science/article/pii/S1077314215001071

[21] Scharstein, D., Szeliski, R., Zabih, R., 2001. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In: Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001). pp. 131–140.

[22] Sethian, J. A., 1999. Level Set Methods and Fast Marching Methods, second ed. Cambridge University Press.

[23] Shao, L., Han, J., Xu, D., Shotton, J., Oct 2013. Computer vision for rgb-d sensors: Kinect and its applications [special issue intro.]. IEEE Transactions on Cybernetics 43 (5), 1314–1317.

[24] Shlens, J., 2014. A tutorial on principal component analysis. CoRR abs/1404.1100.
URL http://arxiv.org/abs/1404.1100

[25] Smistad, E., Falch, T. L., Bozorgi, M., Elster, A. C., Lindseth, F., 2015. Medical image segmentation on gpus – a comprehensive review. Medical Image Analysis 20 (1), 1 – 18.
URL http://www.sciencedirect.com/science/article/pii/S1361841514001819

[26] Song, H., Liu, Z., Xie, Y., Wu, L., Huang, M., Dec 2016. Rgbd co-saliency detection via bagging-based clustering. IEEE Signal Processing Letters 23 (12), 1722–1726.

[27] Statistisk sentralbyrå, 2018. Utenrikshandel med varer. [Online; accessed 23-July-2018].
URL https://www.ssb.no/muh

[28] Toscana, G., Rosa, S., 2016. Fast graph-based object segmentation for RGB-D images. CoRR abs/1605.03746.
URL http://arxiv.org/abs/1605.03746

[29] UTOFIA, 2018. Underwater time of flight image acquisition. [Online; accessed 03-August-2018].
URL https://www.utofia.eu

[30] Viazzi, S., Hoestenberghe, S. V., Goddeeris, B., Berckmans, D., 2015. Automatic mass estimation of jade perch scortum barcoo by computer vision. Aquacultural Engineering 64, 42 – 48.
URL http://www.sciencedirect.com/science/article/pii/S0144860914001150