NTNU

Kunnskap for en bedre verden

Bacheloroppgave

IE303612 Bacheloroppgave Automasjon

Computer-aided quality control of whitefish fillets

Forfatter(e): 10038, 10065

Totalt antall sider inkludert forsiden: 99

Innlevert Ålesund, 01.06.18

NTNU

Kunnskap for en bedre verden

Obligatorisk egenerklæring/gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

Du/dere fyller ut erklæringen ved å klikke i ruten til høyre for den enkelte del 1-6:		
1.	**Jeg/vi erklærer herved at min/vår besvarelse er mitt/vårt eget arbeid, og at jeg/vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.**	☒
2.	**Jeg/vi erklærer videre at denne besvarelsen:** • ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • ikke refererer til andres arbeid uten at det er oppgitt. • ikke refererer til eget tidligere arbeid uten at det er oppgitt. • har alle referansene oppgitt i litteraturlisten. • ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	☒
3.	Jeg/vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§14 og 15.	☒
4.	Jeg/vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert i Ephorus, se **Retningslinjer for elektronisk innlevering og publisering av studiepoenggivende studentoppgaver**	☒
5.	Jeg/vi er kjent med at høgskolen vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens studieforskrift §31	☒
6.	**Jeg/vi har satt oss inn i regler og retningslinjer i bruk av** kilder og referanser på biblioteket sine nettsider	☒

Publiseringsavtale

Studiepoeng: 20

Veiledere: Arne Styve og Hans Støle

Fullmakt til elektronisk publisering av oppgaven

Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven §2).
Alle oppgaver som fyller kriteriene vil bli registrert og publisert i Brage HiM med forfatter(ne)s godkjennelse.
Oppgaver som er unntatt offentlighet eller båndlagt vil ikke bli publisert.

Jeg/vi gir herved NTNU i Ålesund en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	☒**ja**	□**nei**
Er oppgaven båndlagt (konfidensiell)? (Båndleggingsavtale må fylles ut) - Hvis ja:	□**ja**	☒**nei**
Kan oppgaven publiseres når båndleggingsperioden er over?	□**ja**	□**nei**
Er oppgaven unntatt offentlighet? (inneholder taushetsbelagt informasjon. Jfr. Offl. §13/Fvl. §13)	□**ja**	☒**nei**

Dato: 31.05.18

Preface

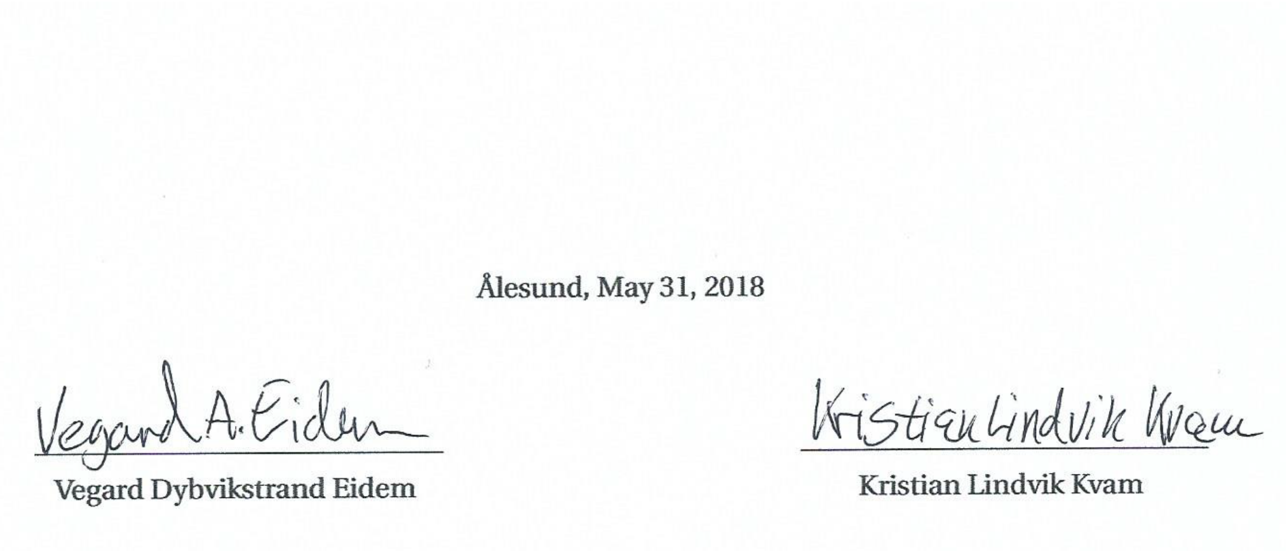
This is a final major assignment by automation students at the Department of Engineering and Science at NTNU in Aalesund. The assignment gives 20 points of credit and includes concept testing of methods for detection of wrongly cut fish fillets and assessment of blood content in a whitefish fillet. The task was given by the company Optimar, which delivers finished fish handling systems.

Both members of the group have taken the course Digital Image Processing, which is the main reason for the choice of assignment.

Thanks to:
Yanran Cao and Lars Christian Gansel for help with acquiring research papers regarding fish blood.

Special thanks to:
Erik Westre and Marius Nedrelid at Optimar and our supervisors Arne Styve and Hans Støle.

We would also like to thank Anders Sætersmoen for taking care of all our orders.



Ålesund, May 31, 2018

Vegard A. Eidem

Vegard Dybvikstrand Eidem

Kristian Lindvik Kvam

Kristian Lindvik Kvam

Executive Summary

Computer-aided quality control of whitefish fillets is a bachelor assignment made for automation students at NTNU in Aalesund. The assignment was given by the company Optimar, a company with vast experience in creating innovative solutions for fish handling.

The task is to:

- Perform concept testing to find methods for detecting wrongly cut fillets.
- Perform concept testing to find methods for assessing blood content in a whitefish fillet.
- Include choice of camera and light source.

To solve these tasks it was done a lot of research, and several tests was performed to test out different theories. In these tests it was experimented with different types of lights and illumination techniques to find a method which could highlight blood spots on a whitefish fillet. It was followed up with testing of several image processing techniques to find suitable methods for finding the area of the fillets and the blood spots.

It was created a java application that could perform the whole process form detecting the fillet to classifying it. In the application it was implemented image processing methods, and a classifying system using the K-Means clustering algorithm. The application controls an illumination system and the whole process can be closely monitored via an HMI.

The project is considered to be successful, but further work and testing is needed to make it ready for an industrial application. It was managed to highlight blood spots on whitefish fillets using one of the illumination techniques tested. The image processing methods and the classification system did well on the small number of images and data accessible, and should be further worked on.

Contents

List of Figures

Terminology

Concepts

class - term used in programming

Symbols

μ - Micro
Ω - Ohm
U_F - Forward voltage
I_F - Forward current
U_T - Supply voltage

Abbreviations

IR - Infrared
NIR - Near Infrared
nm - nanometers
CCD - Charge-Coupled Device
CMOS - Complementary Metal-oxide Semiconductor
mcd - micro candela
LED - Light Emitting Diode
HMI - Human Machine Interface
GUI - Graphical User Interface
VIS - Visible Spectrum
PLC - Programmable Logic Controller
Fps - Frames per second
RGB - Red Green Blue

Chapter 1

Introduction

The fish industry is growing fast and to stay competitive on an international level, we need to minimize loss and maximize revenue. To achieve those goals, quality control is a crucial part of the production. The quality of the fish is mainly determined by the amount of blood in the white muscle tissue. The costumer expects a white fish fillet, not a pink or red fillet [18].

In a kick-off meeting with Optimar, Erik Westre [A.1] gave us a brief introduction on quality assessment of whitefish and what the costumer expect from the product. Fish fillets are categorized into different classes, this is done by the quality controller at the factory. The quality determines if the fillet should be sold as a premium product or a production quality. The poorer quality could be sold to manufacturers of processed fish, hence the poorer quality there are little to no difference in flavor. The premium fish fillet could be sold to restaurants or consumers, that are willing to pay the extra for the crystal white fillet. This means that by performing exceptional quality control, one can eliminate the chances of selling a poor-quality fish as a premium product. As these often ends up as waste at the grocery store, because the costumer always picks out the whitest fish.

So far, the quality control has been conducted manually by an operator at the fish processing plant, but this method brings a lot of uncertainty in to the equation. Because a human individual is performing this task, there are a lot of personal preferences and experience that might count in on the classification of the fillet at hand. Different human individuals have the tendency to appraise the quality different, making the quality control non-consistent. By automating this quality control, one can remove the human factor and classify the fish more consistently.

The problem at hand is solving the quality control issue that occurs by manual control and removing the uncertainty that emerges with the human factor. To do this it's needed to detect the presence and amount of blood residue in the whitefish fillet. Blood residue can be caused by not applying the right bleeding strategy on the fish. Blood can be detected by using a vision system with the right light conditions, a suitable camera and image processing and analysis techniques. By using these techniques, one can segment out the blood spots and calculate the percentage of blood in the fish's muscle tissue. With a functional system one can classify the fish to the associated quality.

It's shown by Olsen [29] and Heia [18] that spectroscopy and light is an efficient way to detect blood residue in the Atlantic Cod. The spectroscopy conducted on the different mutations of hemoglobin, shows the absorption percentage of different wavelengths of light. Absorption of different light is perceived by the eye as dark areas, and this is because no reflected light is emitted back to the eye. The opposition of absorption is reflected light, reflected light would be the one perceived as colors for the human eye.

Another efficient way to detect blood residue is using Near-Infrared Imaging. It's stated by David Boas and Maria Franceschini [11], that you can use NIR-Imaging (Near-Infrared Imaging) and the absorption properties of hemoglobin in the range 650Nm and 950Nm. In this case it's needed to use a camera without a IR cut-off filter, this is designed to block out IR light from the image sensor as elaborated at Optics-Online[32]. A Near-Infrared camera is basically a normal image sensor without the blocking filter used in normal cameras [46].

The problem to be solved is utilizing the knowledge of the light absorbing properties of the hemoglobin in blood. Doing this by using wavelengths that corresponds to the wavelengths specified by [29]. Utilizing the Near-Infrared window in biological tissue could also work for this problem as elaborated by Andrew M. Smith and Nie [5]. For both solutions the wave lengths should in theory be absorbed by the hemoglobin in the blood residue and appear dark on an image. The remaining tasks to be solved is image processing and segmenting out the areas containing blood residue or spots and calculate the percentage of blood in the fish fillet.

1.1 About the assigner

The assignment was handed out by Optimar, a company with vast experience in creating innovative solutions for fish handling. Along side their customers they create advanced automated solutions helping the customers stay competitive in the industry while taking care of both animal welfare and the environment.

Optimars main office is located on Valderøya, outside of Aalesund, inside of the Norwegian maritime and seafood cluster. With 280 employees in Norway, Spain and US, Optimar is stationed to help customers all around the world.

1.2 Objectives

The main objectives of this Bachelor's project are

1. Find papers and articles on how to spot blood residue.
2. Find valid light and hardware (light source and camera).
3. Find image processing techniques.
4. Make test concept.
5. Test and improve.
6. Make java application with HMI.
7. Find out if we can apply some kind of artificial intelligence methods to classify fillets.
8. Test and improve.
9. Document the project well for future work.

1.3 Limitations

The project have many limitations, some of them being lack of: knowledge in biology and maritime biology, the right equipment, image material and fresh enough fish to conduct experiments on.

The main issue in this project is to have the correct light source and a proper camera, not having these could result in a situation where we can't prove that the concept works or not.

The lack of knowledge in biology, and specifically in marine biology is another limitation. As this is a big part of understanding how hemoglobin in the blood acts and work both pre- and post mortem.

As hemoglobin degrades and morphs into another state post mortem, is another limitation. To prove the concept works, relatively fresh fish is needed to capture the different types of hemoglobin in an image.

1.4 Outline

Further in the report one can expect to read about the theoretical foundation of the project and the different materials and methods that have been used. The results are described in a larger section off the report, where one also can read about experimental tests conducted throughout the project. In the last section of report, we have discussed the overall results and made conclusions.

- Chapter 1. Introduction: Structure already discussed in this chapter.
- Chapter 2. Theoretical basis: This chapter presents the theoretical basis which was needed for this project.
- Chapter 3. Materials and Methods: This chapter individually describes how different methods were used to solve the problem at hand. The chapter also describes the different components used.
- Chapter 4. Results: This chapter presents the results developed during the project, and goes into detail describing how the different solutions works.
- Chapter 5. Discussion: In this chapter the results are taken under discussion and opinions are made on the quality of the results.
- Chapter 6. Conclusion: In this chapter an overall conclusion of the project is made, and recommendations for further work on the project is given.
- Bibliography
- Appendix

Chapter 2

Theoretical basis

2.1 Whitefish

The whitefish family includes all species with white meat, such as cod, saithe, haddock and others as explained by WWF.*whitefish* [47] Whitefish is very important for Norwegian fish export, as shown in the article *Yet another record year for whitefish export* [48] by the "Norwegian Seafood Council". As stated here Norway exported cod, haddock and saithe and other whitefish with a value worth around 15 billion NOK in 2017. Equivalent to an 8 per cent increase from 2016.

2.2 Hemoglobin

Hemoglobin is the iron-containing protein found in all red blood cells (RBCs) that gives the cells their characteristic red color. Hemoglobin enables RBCs to bind to oxygen in the lungs and carry it to tissues and organs throughout the body. It also helps transport a small portion of carbon dioxide, a product of cell metabolism, from tissues and organs to the lungs, where it is exhaled. Labtestonline [21]

2.2.1 Methemoglobin

Methemoglobin is a form of hemoglobin that is incapable of carrying oxygen. [25]

2.2.2 Oxyhemoglobin

Oxyhemoglobin is the oxygen-loaded form of hemoglobin, the predominant protein in red blood cells. [26]

2.3 The Electromagnetic Spectrum

The electromagnetic spectrum contains different frequencies and wavelengths of electromagnetic radiation. The electromagnetic spectrum used for spectroscopy lies within the wavelengths 10m to 100pm. At the start of 10meter wavelengths lies the nuclear magnetic resonance(NMR), electron spin resonance(ESR), Microwaves, Infrared, Visible, Ultraviolet, X-ray and then at 100pm the gamma-ray. Although the long wave lengths of NMR and ESR can be harmful, the high-energy radiation of gamma-rays can result in more dramatic situations, such as change of nuclear configuration. [37]

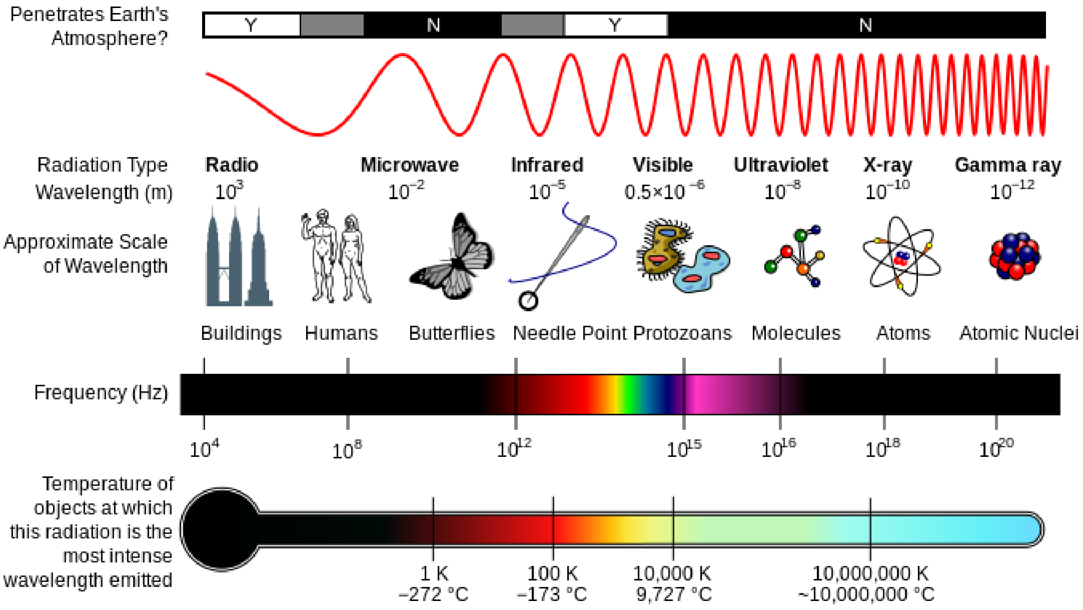


Figure 2.1: Electromagnetic Spectrum [19]

2.3.1 Visible Spectrum

The visible spectrum contains the wavelengths perceived as colors by the human eye between 400-780nm, hence the name visible spectrum as shown by Mehta [27].

2.3.2 Near-Infrared Spectrum

Lies within the Infrared spectrum in the wavelengths 780nm to 2500nm, right above the color red in the visible spectrum as elaborated by Mehta [27].

2.4 Near-Infrared Imaging

The use of near infrared light, most commonly in the range 650 to 950nm to detect levels of oxygenation of the hemoglobin in tissues as shown by Boas and Franceschini [11]

2.5 Near-Infrared Cameras

Near-Infrared Cameras (NIR Cameras) uses the NIR portion of the electromagnetic spectrum above the visible spectrum, to provide light for camera sensors even without sufficient light or poor light conditions. The most commonly used camera sensor until now, is the expensive CCD sensor invented by AT&T Bell Laboratories.

Newer technology with increased sensitivity in the near-infrared range, gives the more common and less expensive CMOS sensor the capability to exploit the NIR spectrum. [46]

2.5.1 CCD

Charge-Coupled Device used store charge in potential wells created at the surface of a semiconductor and moving this charge across the surface by moving the potential minima. Moving the electrical charge to a shift register where it can be converted into a digital value. Boyle and Smith [12]

CCD sensors requires minimal pixel overhead, this makes it possible to make sensors with small foot print, resulting in a image sensor with high density of pixels. The CCD sensors uses a charge transfer readout and this serial readout results in limited readout speed and in need of high-rates to achieve near perfect charge transfer efficiency. Its older technology also requires high-voltage making it less power efficient.El Gamal and Eltoukhy [15]

2.5.2 CMOS

Complementary Metal-oxide Semiconductor consist of four pixels with a color-filter-array (CFA), most commonly red-green-green-blue. The analog data red from the CMOS is then processed through an analog-to-digital converter (ADC). The digital samples from the CFA can now be used to recreate a full color image, this is done by using a spatial interpolation method called de-mosaicing.

The CMOS sensor uses a random-access readout, this provides high speed even with high resolutions. Most digital cameras use the CMOS because of its energy efficiency and speed.El Gamal and Eltoukhy [15]

2.6 IR cut-off filter

Infrared (IR) cut-off filters are used with color CCD or CMOS imagers to produce accurate color images. An IR cut-off filter blocks the transmission of the infrared while passing the visible. This can be done with two optical techniques: absorption or reflection. [32]

2.7 Near-infrared window in biological tissue

Near-infrared light (700–2,500 nm) can penetrate biological tissues such as skin and blood more efficiently than visible light because these tissues scatter and absorb less light at longer wavelengths. Andrew M. Smith and Nie [5]

Noninvasive in vivo imaging with light photons represents an intriguing avenue for extracting relevant biological information. Whereas light in the visible range is routinely used for intravital microscopy2, imaging of deeper tissues (>500 μm to cm) requires the use of NIR light. Hemoglobin and water, the major absorbers of visible and infrared light, respectively, have their lowest absorption coefficient in the NIR region around 650–900 nm. Light photons can be used to measure different native parameters of tissue through which they travel—for example, absorption, scattering, polarization, spectral characteristics, and fluorescence. Weissleder [43]

2.8 CCD Saturation and Blooming

Saturation and blooming are related phenomena that occur in all charge-coupled device (CCD) image sensors under conditions in which either the finite charge capacity of individual photodiodes, or the maximum charge transfer capacity of the CCD, is reached. Once saturation occurs at a charge collection site, accumulation of additional photo-generated charge results in overflow, or blooming, of the excess electrons into adjacent device structures. A number of potentially undesirable effects of blooming may be reflected in the sensor output, ranging from white image streaks and erroneous pixel signal values to complete breakdown at the output amplification stage, producing a dark image. Fellers and Davidson [16]

2.9 Digital Image Processing

Digital image processing and analysis is a field that continues to grow, with new applications being developed at an ever-increasing pace. Digital image processing, also referred to as computer imaging, can be defined as the acquisition and processing of visual information by computer. Umbraugh [41]

2.9.1 Color depth

Gray-scale image

Gray-scale images are referred to as monochrome, or one color, images. They contain brightness information only, no color information. The number of bits used for each pixel determines the number of different brightness levels available. The typical image contains 8-bit per pixel data, which allows us to have 256 (0-255) different brightness (gray) levels. [41]

Binary image

Binary images are the simplest type of images, and can take on two values, typically black and white, or “0” and “1”. A binary image is referred to as a 1-bit per pixel image, because it takes only 1 binary digit to represent each pixel. These types of images are most frequently used in computer vision applications where the only information required for the task is general shape, or outline information. Binary images are often created from gray-scale images via a threshold operation.[41]

2.9.2 Morphological Filtering

Morphology relates to the structure or form of objects. Morphological filtering simplifies a segmented image to facilitate the search for objects of interest. This is done by smoothing out object outlines, filling small holes, eliminating small projections, and with other similar techniques. The two principal morphological operations are dilation and erosion. Umbraugh [41]

Dilation

Dilation allows objects to expand, thus potentially filling small holes and connecting disjoint objects. Umbraugh [41]

Erosion

Erosion shrinks objects by etching away (eroding) their boundaries.Umbraugh [41]

2.9.3 Segmentation

The goal of image segmentation is to find regions that represent objects or meaningful parts of objects. Image segmentation methods will look for objects that either have some measure of homogeneity within themselves, or have some measure of contrast with the objects on their border. Most image segmentation algorithms are modifications, extensions, or combinations of these two basic concepts. The homogeneity and contrast measures can include features such as gray-level, color and texture. [41]

Region Growing and Shrinking

Region growing and shrinking methods segment the image into regions by operating principally in the row and column based image space. [41]

Structuring Elements

An essential part of the morphological dilation and erosion operations is the structuring element used to probe the input image. A structuring element is a matrix that identifies the pixel in the image being processed and defines the neighborhood used in the processing of each pixel. You typically choose a structuring element the same size and shape as the objects you want to process in the input image. For example, to find lines in an image, create a linear structuring element. *Structuring Elements* [38]

2.9.4 Thresholding

Image thresholding is a simple, yet effective, way of partitioning an image into a foreground and background. This image analysis technique is a type of image segmentation that isolates objects by converting grayscale images into binary images. Image thresholding is most effective in images with high levels of contrast. [4]

Otsu Method

The Otsu method is a nonparametric and unsupervised method of automatic threshold selection for picture segmentation. [33]

Adaptive thresholding

In simple thresholding, the threshold value is global, i.e., it's the same for all the pixels in the image. Adaptive thresholding is the method where the threshold value is calculated for smaller regions and therefor, there will be different threshold values for different regions.[40]

2.9.5 Region of interest geometry

Complement Image

In the complement of a binary image, zeros become ones and ones become zeros; black and white are reversed.[24]

Masking

The logic operators AND and OR are used to combine the information in two images. This may be done for special effects, but more useful applications for image analysis is to perform a masking operation. AND and OR can be used as a simple method to extract ROI from an image. For example, a white mask ANDed with an image will allow only the portion of the image coincident with the mask to appear in the output image, with the background turned black; and a black mask ORed with an image will allow only the part of the image corresponding to the black mask to appear in the output image, but will turn the rest of the image white. This process is called masking.[41]

2.9.6 Border clearing

imclearborder

The imClearBorder method suppresses structures that are lighter than their surroundings and that are connected to the image border.[22] The method belongs to the MATLAB image processing toolbox.

floodFill

The function fills a connected component starting from the seed point with the specified color. The connectivity is determined by the color/brightness closeness of the neighbor pixels.[30] The method belongs to the OpenCV library.

2.9.7 Structural Analysis

findcontours

Findcontours is a method that retrieves contours in a binary image using the algorithm [39]. The contours are a useful tool for shape analysis and object detection and recognition.[31] The method belongs to the OpenCV library.

2.10 Machine learning

2.10.1 K-Means Clustering

The aim of the K-means algorithm is to divide M points in N dimensions into K clusters so that the within-cluster sum of squares is minimized. The algorithm requires as input a matrix of M points in N dimensions and a matrix of K initial cluster centres in N dimensions. [17]

2.10.2 Overfitting

Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the model's ability to generalize.[13]

2.10.3 Data preprocessing

The purpose of preprocessing is simply to transform raw data into a format appropriate for subsequent data mining. Preprocessing consists of three steps. The first step is data selection when the data relevant to the analysis is decided on and multiple data sources are identified. The second step is data cleansing. At this stage all the relevant data is retrieved from the target databases and integrated in a common source. Redundant and inconsistent data is removed from the collection. The third step is data transformation. It is a stage when the cleansed data is transformed into forms suitable for data mining. [28]

Chapter 3

Material And Methods

3.1 Tools and libraries

3.1.1 Java

The java programming language is a high-level language that can be characterized as object orientated and multithreaded along with other features. In the java programming language, all source code is first written in plain text files ending with ".java" extension. Those source files are then compiled into ".class" files by the java compiler. A ".class" file does not contain code that is native to your processor; it instead contains bytecodes - the machine language of the Java Virtual Machine. [2]

3.1.2 Matlab

MATLAB® is a programming platform designed specifically for engineers and scientists. The heart of MATLAB is the MATLAB language, a matrix-based language allowing the most natural expression of computational mathematics. Using MATLAB, you can analyze data, develop algorithms and create models and applications. [45]

Image Processing Toolbox

Image Processing Toolbox™ provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development. You can perform image segmentation, image enhancement, noise reduction, geometric transformations, image registration, and 3D image processing. [23]

3.1.3 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. [1]

3.1.4 Arduino microcontroller

Arduino is a open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs and activating outputs. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the arduino programming language. The arduino board is also inexpensive and both the software and hardware is open source. [44]

Serial communication

Used for communication between the Arduino board and a computer or other devices. All Arduino boards have at least one serial port. It communicates on digital pins 0 (RX) and 1 (TX) as well as with the computer via USB. [34]

3.1.5 Weka

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. [14]

ARFF

An ARFF file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science for the use with the Weka machine learning software. [9]

3.1.6 Multithreading

By definition, multitasking is when multiple processes share common processing resources such as a CPU. Multi-threading extends the idea of multitasking into applications wher you can subdivide specific operations within a single application into individual threads. Each of the threads can run in parallel. [20]

3.2 Materials

3.2.1 Camera

Creative Live VF0520

Web Camera which can capture 30 fps at a resolution of 800*600 pixels.

Logitech c920

Web Camera which can capture 30 fps at full HD 1080p.

Figure 3.1: Logitech c920

3.2.2 Lighting

Table 3.1: LED list.

LED list					
Color	Wavelength	Supply voltage	Forward Voltage	Forward Current	Luminous Intensity
Infrared	880nm	5V	1.3V	20mA	N/A
Infrared	890nm	5V	1.4V	100mA	N/A
Red	630nm	5V	2.6V	50mA	12000mcd
Green	525nm	5V	3.6V	30mA	35000mcd
Blue	428nm	5V	3.8V	30mA	1630mcd

3.2.3 Electronics

Arduino Uno

Arduino uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), and 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. [8]

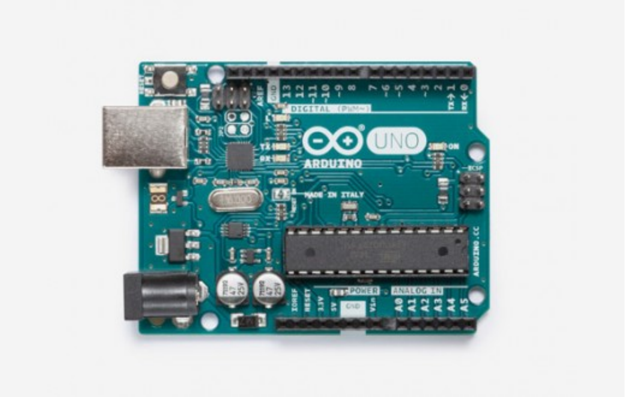


Figure 3.2: Arduino Uno

Arduino Nano

Arduino Nano is a compact board similar to the UNO. It has 22 digital input/output pins (6 of which are PWM) and 8 analog inputs. [7]

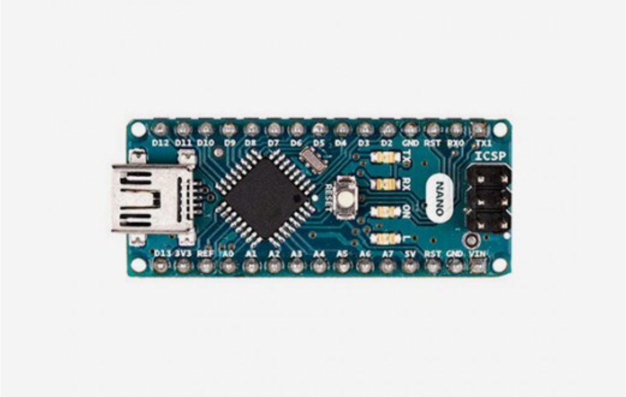


Figure 3.3: Arduino Nano

Panasonic JS1-5V-F relay

A 5V DC coil rated relay with a max switching current of 10A and a max switching voltage of 125V AC. [3] The contacts of the relay is of the type break-make.

Figure 3.4: Panasonic rele

CLIFF USB-adapter

USB contact for panel mounting. USB 2.0 B socket in the front, and USB 2.0 A socket on the back.[42]

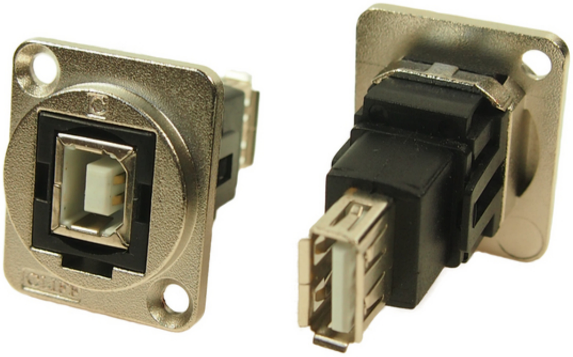


Figure 3.5: USB-adapter

Matrixboard

Single sided matrix board with 1mm holes. Dimentions are 580 x 100 x 1.6mm.[36]

Figure 3.6: Matrix protoboard

Power supply

A 5v DC power supply capable of delivering 4A. Jack plug connection in the end.

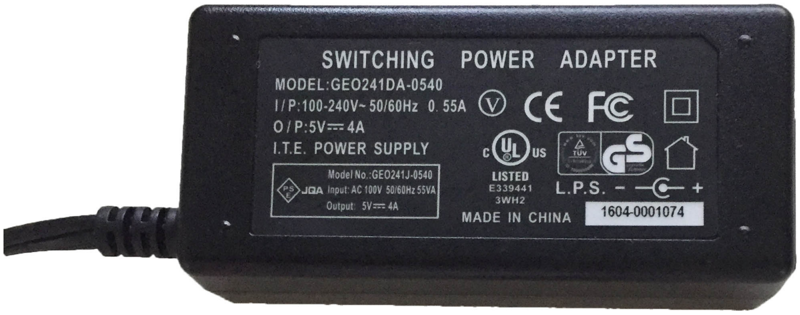


Figure 3.7: Power supply

3.2.4 Other parts

List of different parts used for this project can be seen in Table. 3.2

Table 3.2: Misc. Parts

Miscellaneous parts			
#	Item	Quantity	Specs
1	Jumper wires	N/A	
2	Push Button	1	NO
3	Resistor	40	47Ω
4	Resistor	40	56Ω
5	Resistor	40	68Ω
6	USB-cable	1	Protocol: Serial

3.2.5 Software List

A list of all software used can be seen in Table. 3.3.

Table 3.3: Software.

Software	
#	Name
1	Netbeans IDE
2	MATLAB
3	Arduino IDE
4	PCSCHEMATIC
5	Fusion 360
6	draw.io

3.3 Methods

3.3.1 Project Management

Meetings

The project were carried out on a sprint based form. Meetings were held every other week, where we planned the objectives for the next two weeks. Every meeting with the project group were accompanied by the group's supervisor, Arne Styve. The group supervisor helped managing the work load for every sprint and gave advise along the way.

There were also some meetings held with the assigner, Optimar. To get started with the project, a meeting were held at Optimar's plant at Valderøya. Before the finishing phase of the project, a last meeting with the company were held.

Sharelatex

The report were decided to be written in LaTeX, as editor the group chose sharelatex. Sharelatex.com [35] is an online editor, with a small social section, chat and ability to post little comments connected to the text. Sharelatex uses the macro set LaTeX of the TeXmacro compiler. LaTeX provides several predefined document classes with extensive sectioning and cross-referencing capabilities. [10]

Jira

As project progress tracking and planning the online application Jira by Atlassian were used. The application enables the group to set up the project and its tasks, the tasks can further be assigned to members and used in project sprints.

Confluence

The software used for file shearing, log, minutes and other stuff for managing information. The application also have a social platform making it easy to inform other team members about different things.

3.3.2 Image Processing Techniques

Testing image processing methods were done using Matlab and its image processing toolbox. To be able to test the different methods, an image of a white fish fillet with blood spots was needed. Because of lack of image material, an image of a white fish fillet containing no blood was manipulated. Dark gray spots were drawn on the image to represent blood spots.

RBG image to gray scale

The image is converted from a RGB image to a gray-scale image. This is done so that morphological filtering methods like thresholding can be applied to the image, since morphological filtering methods requires the input image to be a single-channel image.

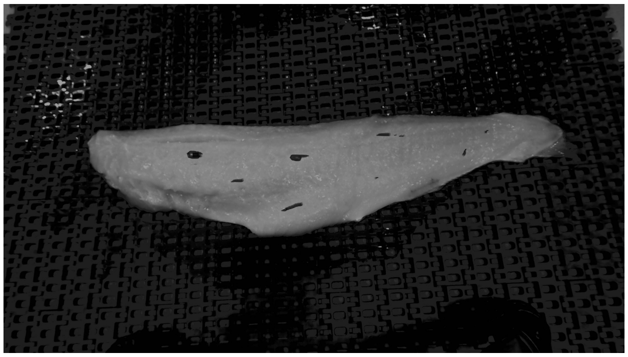


Figure 3.8: Gray scale image

Otsu Threshold

The Otsu Threshold method is used to create a binary image from a gray-scale image. Applying a Otsu threshold lets us segments the white fillet from the much darker background. This makes it easy to find the contours of the objects and extract variables such as size. The method is fast and accurate. The Otsu method was chosen because it is easy to implement and gave satisfying result when testing in Matlab.

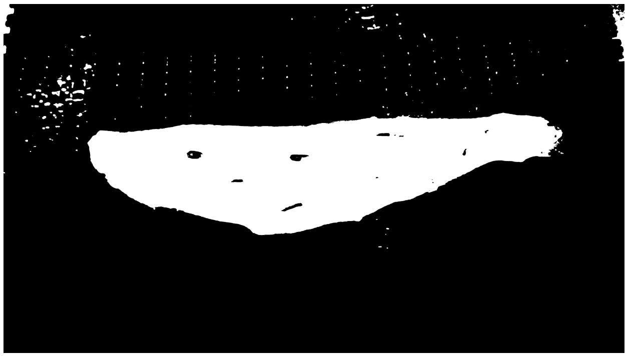


Figure 3.9: Binary image

Eroding

Eroding is used to remove noise and small unwanted objects within the image. Applying a threshold to an image which not has a consistent background will likely create objects not wanted in the image. These objects are removed or downsized depending on their size by eroding the image with a structuring element.

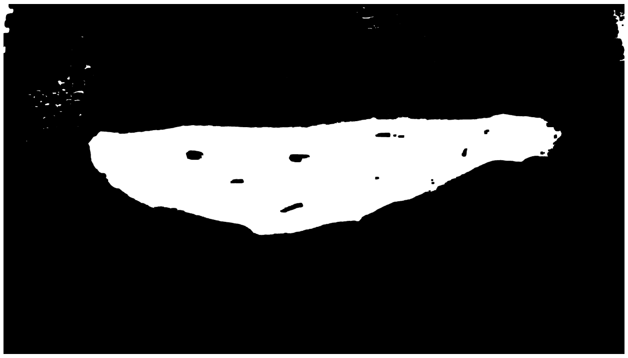


Figure 3.10: Eroded image

Dilating

Dilating is used to rebuild the objects not removed by the eroding method. The dilate method rebuilds the shape of the objects in the image back to their original form using the same structuring element used to erode the image. This makes sure that the fillet is built back to its original form, while the objects removed by the eroding no longer exist and will therefor not be rebuilt.

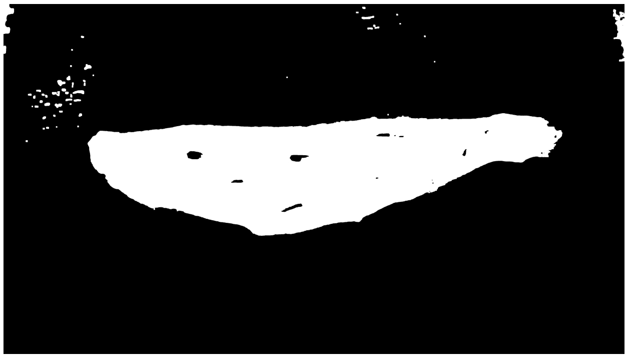


Figure 3.11: Dilated image

Masking

Masking is used to mask out the fillet from the rest of the image. The binary image from fig 3.11 is used as a mask, overlaying the gray image (3.8) and only displaying the areas of the image that are within the white area of the mask. By masking out the fillet from the rest of the image, one eliminates the background. Eliminating the background gives a much cleaner image and makes it possible to apply sensitive threshold methods without the background affecting the region of interest (the fillet).

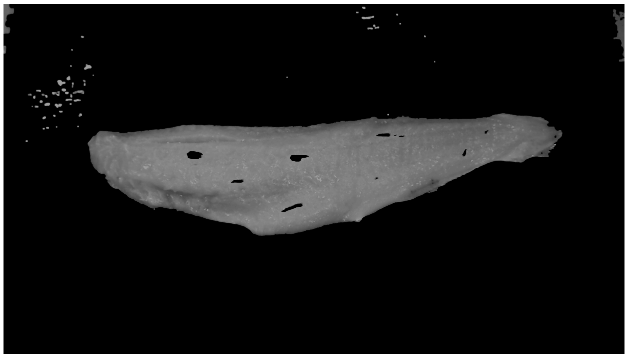


Figure 3.12: Masked image

Adaptive Treshold

Adaptive threshold is used to segment the blood spots on the fillet by calculating the threshold for small regions in the image. By calculating the threshold for small regions and not having a global threshold value, it is possible to segment blood spots on the whole fillet even if the fillet has varying illumination.

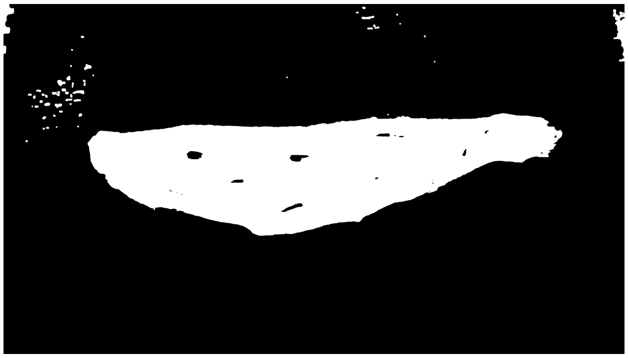


Figure 3.13: Binary image

Complement image

Complementing is used to invert the colors within the image. This is simply done so that another method (imclearborders) will remove the right objects and not the blood spots.

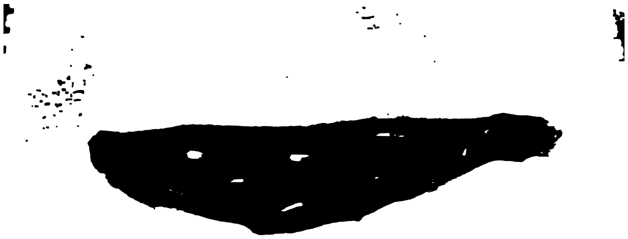


Figure 3.14: Complemented image

Clear Borders

Clearing the border is done using a method from the matlab image processing toolbox called "imclearborders". Applying this method to the complemented image removes all the white areas in the image that are not isolates. Since the blood spots (which are white) are isolated within the fillet (which is black), they will not be removed. The result is a black image where only the blood spots are white, making it possible to measure the amount of blood in pixels.

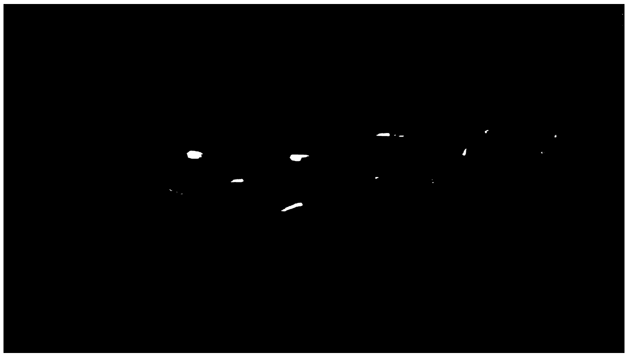


Figure 3.15: Only isolated white pixels reserved

Find contours and area

Finding contours is done using a method from the OpenCV library called "findcontours". The method is used to find contours in two of the binary images. The method is used on the dilated image to find the contours of the fillet, making it possible to find the area of the whole fillet. The method is also used on the image after it has been filtered by the "imclearborders" method to retrieve the area of the blood spots. The contours found in each image is stored in separate arrays.

The area of the fillet is found by iterating through the array containing all the contours, retrieving only the largest contour.

The area of the blood spots is found by iterating through the array containing all the contours, summing together all the contours.

3.3.3 Machine learning

K-Means Clustering

Clustering is used to classify fillets by assigning each fillet to the cluster with the most resemblance. A previously trained cluster model is loaded at startup. The cluster model has a total of 4 clusters where each cluster holds data from different fish fillets. Each fillet has four attributes; the area of the fillet, the area of hemoglobin, the area of methemoglobin, and the area of oxyhemoglobin. When a fillet has gone through all filtering and data extraction methods, the data is written to a ARFF file. The ARFF file is then evaluated with the cluster model and the fillet gets assigned to a cluster. Based on the attributes extracted from the fillet, it will be classified as either small, bad, medium, or good.

Training the cluster

Training the cluster is done so that fillets will get assigned to the cluster with the most resemblance. When training the cluster, one needs a dataset containing training data. It is important that the dataset only contains relevant data to avoid overfitting. The dataset should therefore be cleaned to remove redundant and inconsistent data 2.10.3 so that unwanted objects won't get assigned to clusters they should not be in. By training the cluster with a clean dataset one will get a better result, and objects with great inequality could be discarded instead of being assigned to a cluster.

3.3.4 Communication

The communication between the java application and the Arduino is done using serial communication over USB. For this an RxTx library made for java programming was used. The RxTx library holds simple methods such as serialRead() and serialWrite(), making it easy to send and receive data. When the java application is started, in the Server class, a new Arduino object is made where parameters such as COM and baud rate are set. The Server class then tries to establish connection with the Arduino, using the command arduino.openConnection(). The Arduino constantly reads the input from the java application, and it also writes data to the java application when a certain input is triggered.

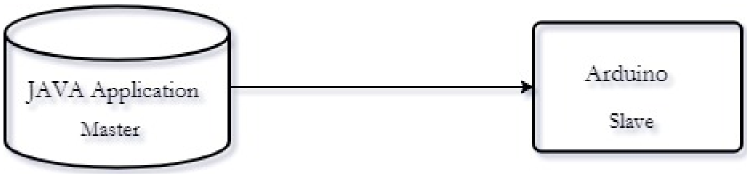


Figure 3.16: Communication diagram

Chapter 4

Results

4.1 Design

In this section the different design solutions are described in detail, from choice of construction and LED's. The design of a valid light source and its control system was an iterative process. Going from a simple housing for Infrared LED's to a more sophisticated dome diffusion tunnel. Form first to last iteration, the use of different illumination strategies was important to get the best lighting conditions. The strategies commonly used in all iterations were a reflector and a light diffusion method.

4.1.1 Infrared LED housing

The infrared light housing was made out of wood and diffused acrylics. The main frame 4.1 could hold seven 5mm LED's and the inside of the box was covered in aluminum tape to reflect the light out of the box. To diffuse the light exiting the box, a 4mm plate of diffused acrylic was used as the cover, see 4.2. The LED's used was 890nm Infrared LED's (3.1), all connected to a Arduino Uno for power.

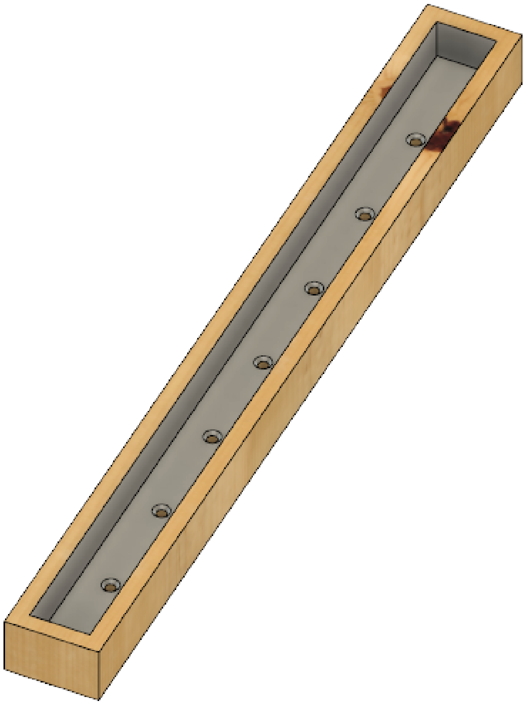


Figure 4.1: LED housing frame

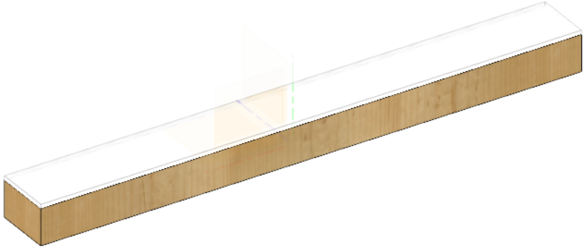


Figure 4.2: LED housing with cover

4.1.2 3D-printed housing

This small 2x2 LED housing was 3D printed, and the inside was covered with aluminum tape as a reflector. The small light source were printed into 2 different parts, the back lid with GoPro mount in Fig. 4.3 and the actual reflector in Fig. 4.4.

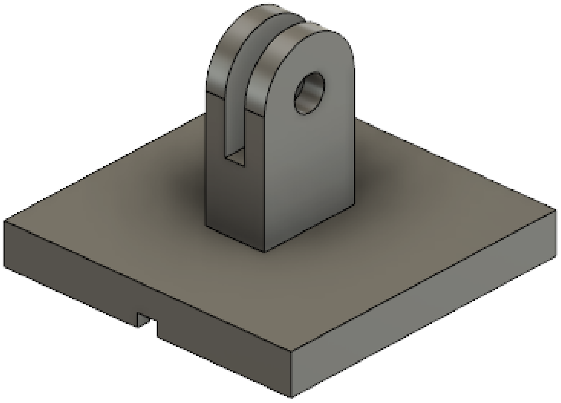


Figure 4.3: Backlid with GoPro mount

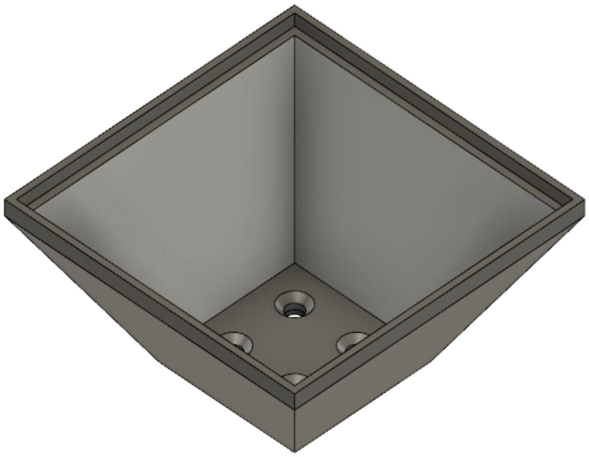


Figure 4.4: Reflector

4.1.3 Light dome

To achieve more desirable illumination, a strategy commonly used in industrial applications called 'dome diffuse' were implemented. The technique is used in machine vision to minimize specular reflection on shiny surfaces. The reason behind this choice was the article; '[6]'.

Dome

This illumination device is using a strategy called dome diffusion, and has the light directed upwards into the dome rather than directly onto the object. It was built out of cardboard and some spare wood. Fig. 4.5.

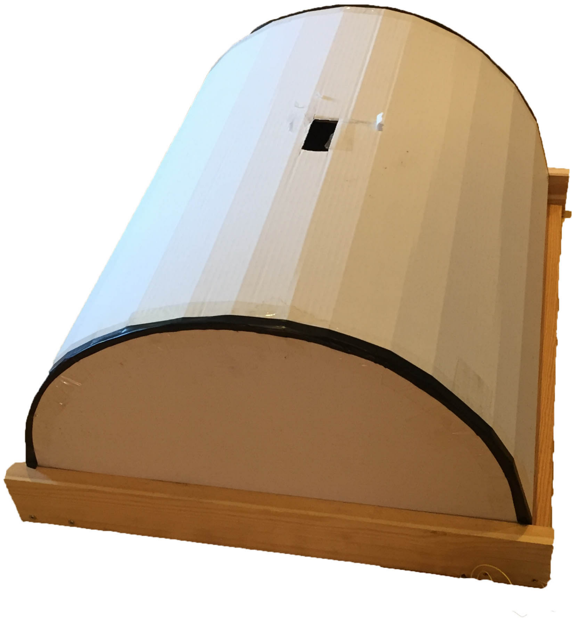


Figure 4.5: Dome tunnel

Reflector

The material used as reflector is aluminum tape, covering the whole inside of the dome as seen in Fig. 4.6. Although the reflector has some wrinkles it still delivers great lightning conditions, despite its imperfections as seen in Fig. 4.7, 4.8 and 4.9.

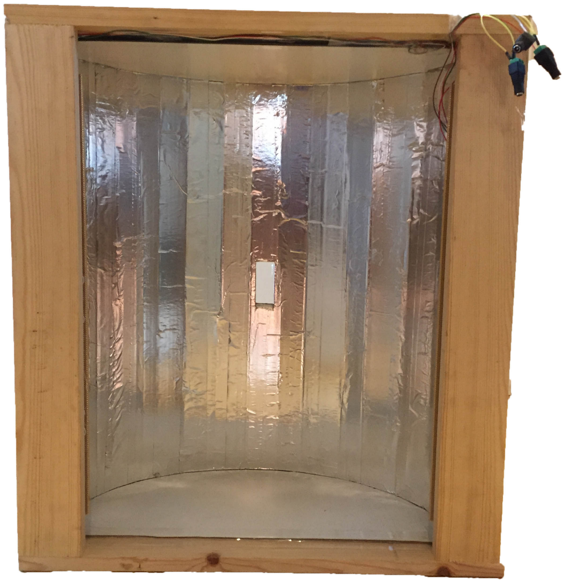


Figure 4.6: Underside of dome

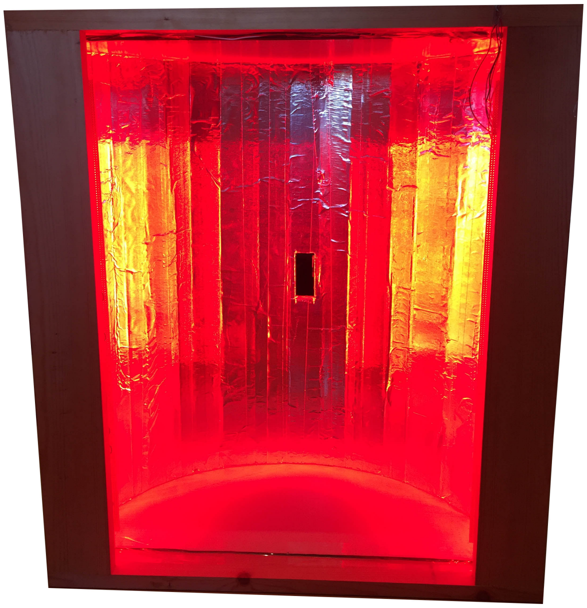


Figure 4.7: Red illumination from the dome

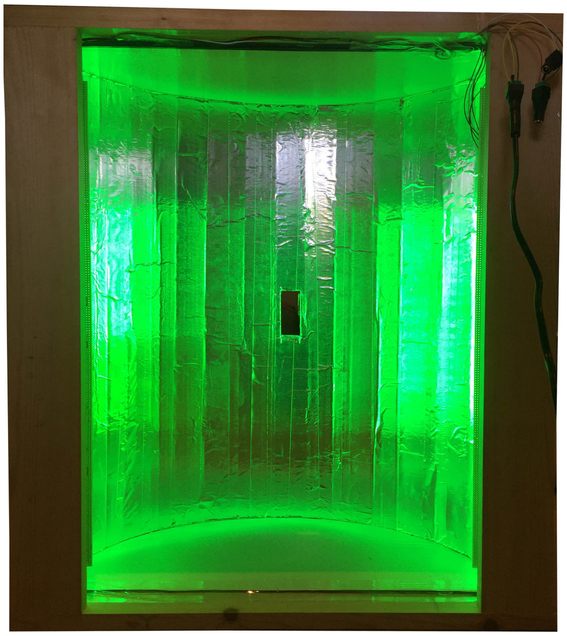


Figure 4.8: Green illumination from the dome

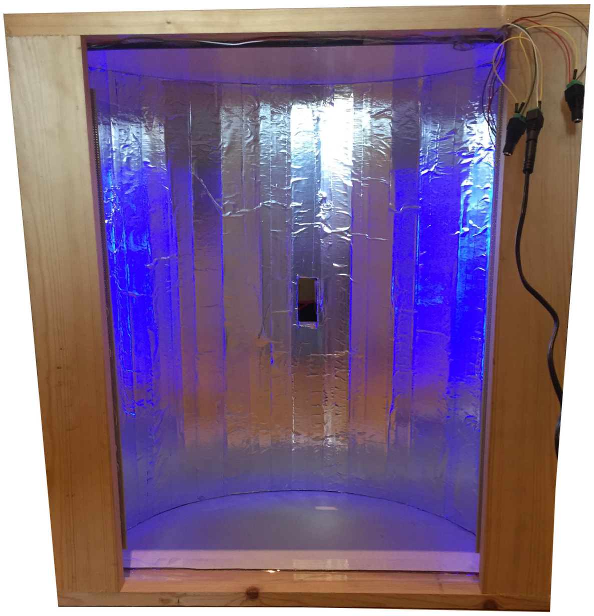


Figure 4.9: Blue illumination from the dome

Light racks

The dome is illuminated by two light racks made out of 120 LED's, resistors and two long strips of perfboard as seen in Fig.4.10.

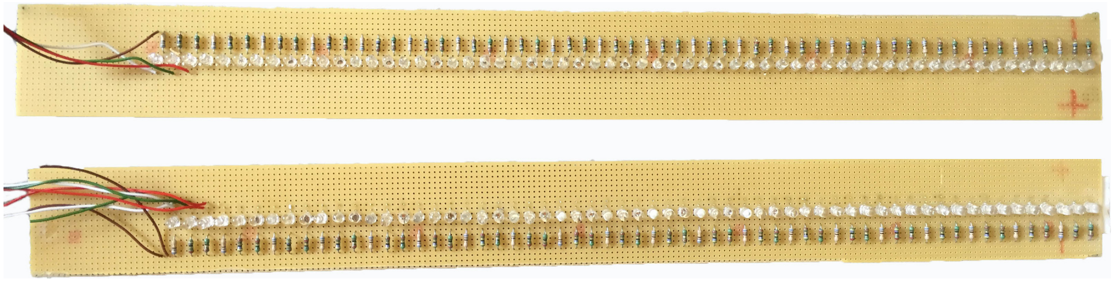


Figure 4.10: Light racks

Each light rack consists of 60 LED's, 20 of each of the wavelengths closest to the values specified by the spectroscopy done by Olsen [29]. The types of LED's chosen for this light rack was red at 630nm, green at 525nm and blue at 428nm found in the LED Table. 3.1. All the individual colors have their own parallel circuit, where each LED have its own resistor. The red LED's have individual resistors at 56Ω, green at 47Ω and blue at 68Ω as seen in Fig. 4.11 or in the full schematic A.3.

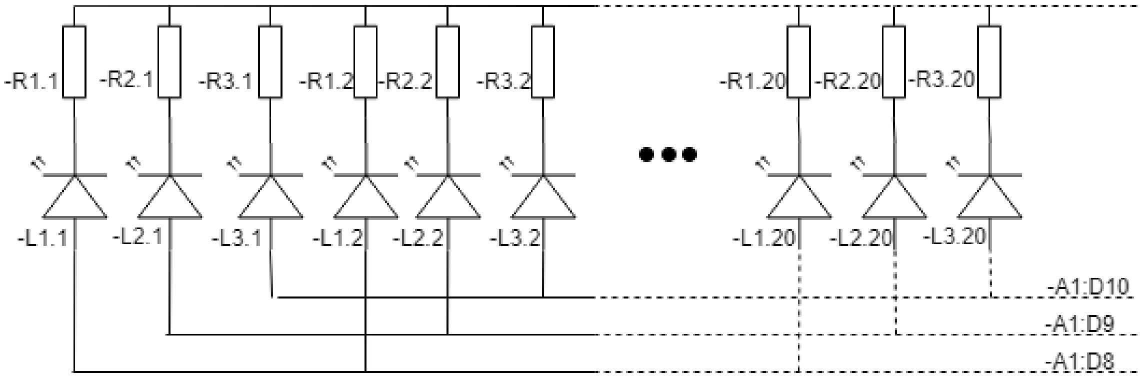


Figure 4.11: Alternating LED and resistors

Each LED's resistor was calculated by using the information from the datasheet as seen in Table. 3.1. Knowing the forward voltage U_F, forward current I_F and the supply voltage U_T provides an opportunity to calculate the resistors value by applying Ohm's Law as shown in Eq. 4.1 and Kirchhoff's voltage law as shown in Eq. 4.2. The values for forward voltage U_F and forward current I_F are chosen to be as little as possible, to achieve better lifespan and dimmer illumination from the LED's.

$$U = R * I \tag{4.1}$$

$$\begin{gathered}\sum_{n=1}^{n} I_n = 0\\ U_T = U_R1 + U_R2 + \cdots + U_{Rn}\end{gathered} \tag{4.2}$$

Red LED

For the Red LED the F_v= 2.6V and I_F=50mA and the U_T = 5V. The desired voltage over the LED is equal to the forward voltage, the resistors job is therefor to serve as a voltage divider. Using Kirchoff's voltage law in Eq. 4.2 and get:

Solving for U_R and get the voltage across the unknown resistor R_X:

$$U_R = U_T - F_v \Rightarrow U_R = 5V - 2.6V = 2.4V$$

Using the voltage U_R and current I_F as input in Ohm's Law 4.1. Solving for R_X:

$$U_R = R_X * I_R \Rightarrow R_X = \frac{U_R}{I_R} \Rightarrow R_X = \frac{2.4V}{0.05V} = 52\Omega$$

Green LED

For the Green LED the F_v= 3.6V and I_F=30mA and the U_T = 5V. The desired voltage over the LED is equal to the forward voltage, the resistors job is therefor to serve as a voltage divider. Using Kirchoff's voltage law in Eq. 4.2 and get:

Solving for U_R and get the voltage across the unknown resistor R_X:

$$U_R = U_T - F_v \Rightarrow U_R = 5V - 3.6V = 1.4V$$

Using the voltage U_R and current I_F as input in Ohm's Law 4.1. Solving for R_X:

$$U_R = R_X * I_R \Rightarrow R_X = \frac{U_R}{I_R} \Rightarrow R_X = \frac{1.4V}{0.03A} = 46.67\Omega$$

Blue LED

For the Blue LED the F_v= 3.8V and I_F=20mA and the U_T = 5V. The desired voltage over the LED is equal to the forward voltage, the resistors job is therefor to serve as a voltage divider. Using Kirchoff's voltage law in Eq. 4.2 and get:

Solving for U_R and get the voltage across the unknown resistor R_X:

$$U_R = U_T - F_v \Rightarrow U_R = 5V - 3.8V = 1.2V$$

Using the voltage U_R and current I_F as input in Ohm's Law 4.1. Solving for R_X:

$$U_R = R_X * I_R \Rightarrow R_X = \frac{U_R}{I_R} \Rightarrow R_X = \frac{1.2V}{0.02A} = 60\Omega$$

The selection of resistors doesn't fully match the ones calculated.
Some might vary because of the availability of resistors.

4.1.4 Light controller box

The light controller box is a control unit containing components for controlling the three different LED circuits and communication with the application. The circuit inside the box contains an Arduino Nano; 3.2.3 and three brake-make relays; 3.2.3. The Arduino is used as both IO and as serial communication, making it a viable choice for prototyping. It's using serial communication with software and is powered via the micro USB connection. The relays are operated by the Arduino's IO, but its contacts are supplied by an external 5v power supply capable of delivering 4A. The peripheral of the light controller box contains a DC barrel Jack as feed-through for the external power supply and a USB socket as feed-through to the Arduino. For a closer inspection of the controller box, check A.3.

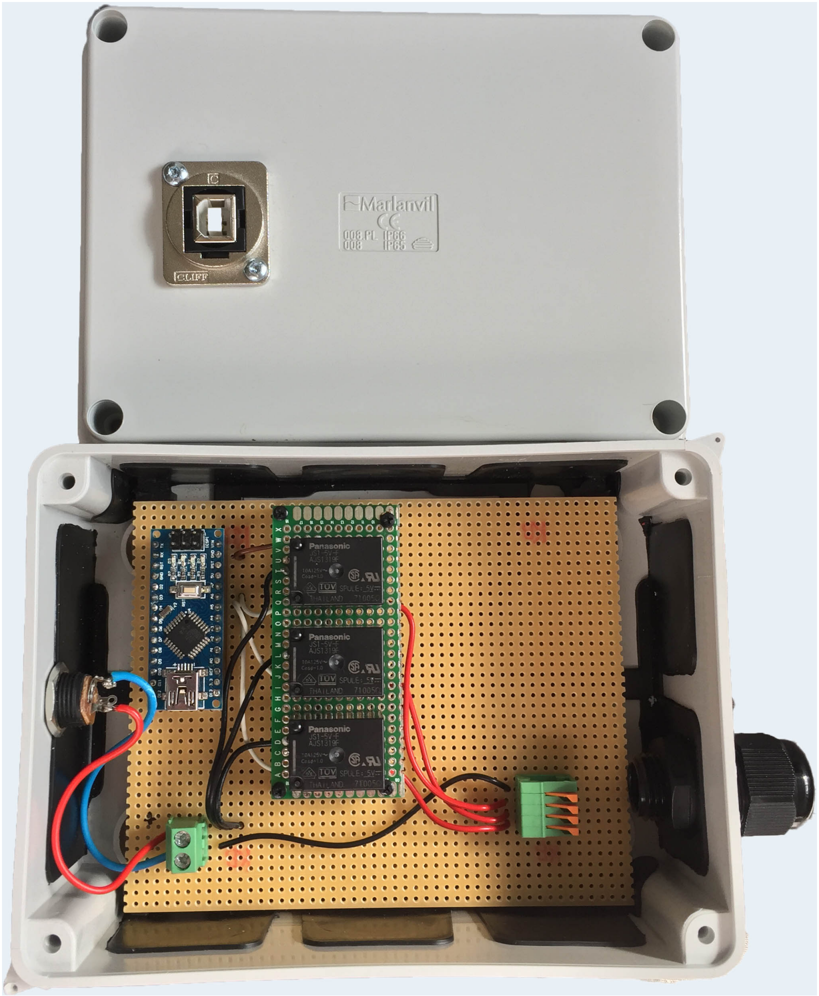


Figure 4.12: Light controller box

4.2 Experimental Results

In this section it is described in detail how the tests completed throughout the project was performed, and the results given from each test. In these tests it is tried out several types of lights and illumination techniques.

4.2.1 Experimental test 1

The first test conducted was to try out the theories of using Near-Infrared light to highlight blood vessels on a human arm. As shown by Smith and Nie, [5] there's an optical window between 650nm and 950nm for optical imaging of live animals. It was therefor decided to order two types of infrared LED's with a wavelength of 880nm and 890 which is in the near infrared spectrum. [27] As for the camera, a web camera of the type Creative Live VF0520 was used. This camera was available at hand and it was free of charge, so there would be no consequences if the camera got destroyed in the process. To be able to view infrared light, a disassemble of the camera was needed to remove the IR cut-off filter. Removal of the IR cut-off filter makes it possible for the camera to absorb light outside the visible spectrum. A quick test was done by capturing an image of an infrared light source, confirming that the camera now could perceive infrared light.

When performing the test, there was a need for the environment to be as dark as possible, as the natural light would disturb the image. For this, a simple box made of cardboard was used. The box had holes for the infrared LED's, a hole to fit the camera, and a larger hole so that a human arm could be inserted into the box. In Fig 4.13 one can see an illustration of the setup used during the test. A human arm was inserted into the intended hole of the cardboard box with the infrared LED's pointing down towards the arm. It was then captured images of the arm using a PC connected to the camera. Various parameters such as brightness, contrast, and exposure was adjusted to get the best possible contrast between the vessels and the skin. The test was performed two times, first using the 880nm LED's and then using the 890 LED's.

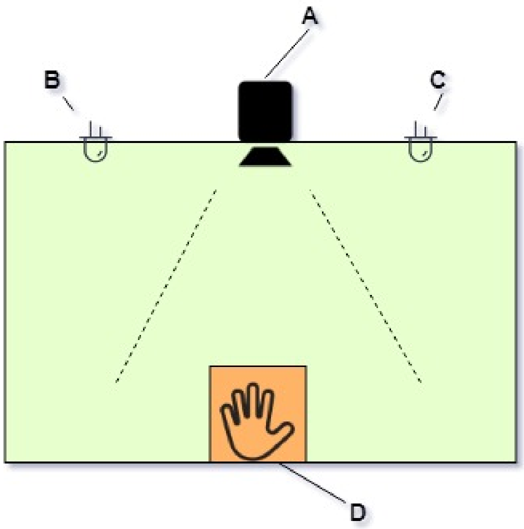


Figure 4.13: Test setup. (A) indicates the camera, (B,C) indicates the Infrared LED's, and (D) is the cut out opening of the cardboard box.

The test results were promising and showed an increased contrast between the blood vessels and the surrounding skin, se Fig 4.14. The resolution and overall quality of the image were not that good, looking blurry and noisy. Both the infrared LED's seemed to increase the contrast between the blood vessels and the surrounding skin. The 880nm LED's did not emit that much light, only delivering 15 mW (see Fig 4.14), and the 890nm LED's emitted a little too much light, delivering 150 mW (see Fig 4.15).This made the images taken using the 880nm LED's look to dark, and the images taken using the 890nm LED's having blooming effects.

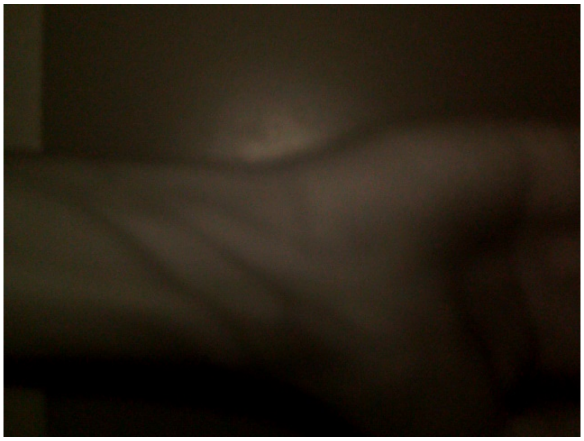


Figure 4.14: Test results, 880nm LED's.

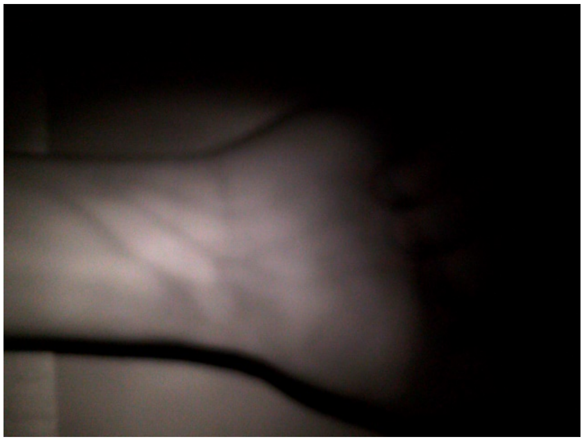


Figure 4.15: Test results, 890nm LED's

4.2.2 Experimental test 2

For the second test the intentions were to capture images with higher resolution and see if the contrast between the blood vessels and the surrounding skin could be enhanced. To achieve this, the camera was replaced by a newer and better camera of the type Logitech C920 HD Pro. This camera can capture images in HD up to a resolution of 1920x1080. Same as in the first test, the IR cut-off filter had to be removed from the camera. This was a bit trickier with the Logitech camera, because it required soldering off the lens housing to get to the IR cut-off filter. Again, a quick test was performed to check that the camera now could perceive infrared light.

The setup was relatively similar to the setup used in the first test, but some changes were made to distribute the light more evenly. In Fig 4.16 one can see an illustration of how normal printing paper was used to diffuse and distribute the light inside the test box. The paper helps diffuse the light, avoiding spot lights inside the box and provide for a more uniform light condition. This reduces the chance of experiencing an unwanted light condition that might cause over exposure and blooming. The placement of the LED's was changed to minimize spot light on the test subject. Just like the first test, we inserted a human arm into the hole of the cardboard box and captured images of the arm. Parameters such as brightness, contrast, and exposure was adjusted to get the best contrast between the blood vessels and the surrounding skin possible.

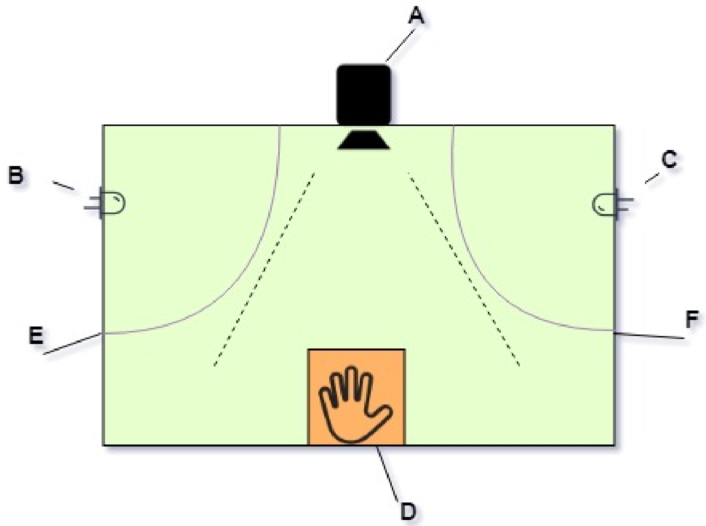


Figure 4.16: Test setup. (A) indicates the camera, (B,C) indicates the Infrared LED's (890 nm), (D) is the cut out opening of the cardboard box, and (E,F) indicates regular white A3 paper.

The results from this test showed a drastic improvement in image quality and the contrast between the blood vessels and the surrounding skin were heavily improved, see Fig 4.17. The hemoglobin in the blood vessels absorbs the light, meanwhile the surrounding skin reflects the light, making the blood vessels look dark. One can see the blood vessels running all the way from the palm to further up the arm. It was also tested capturing images without the use of the paper as diffusion, and in Fig 4.18 one can see how the image loses details when too much light is focused on one specific place.

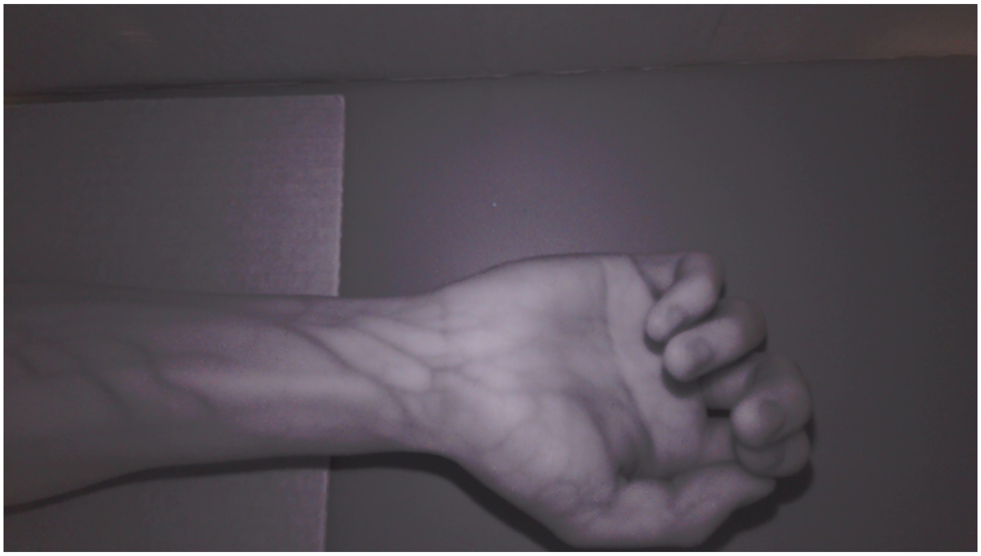


Figure 4.17: Test result.

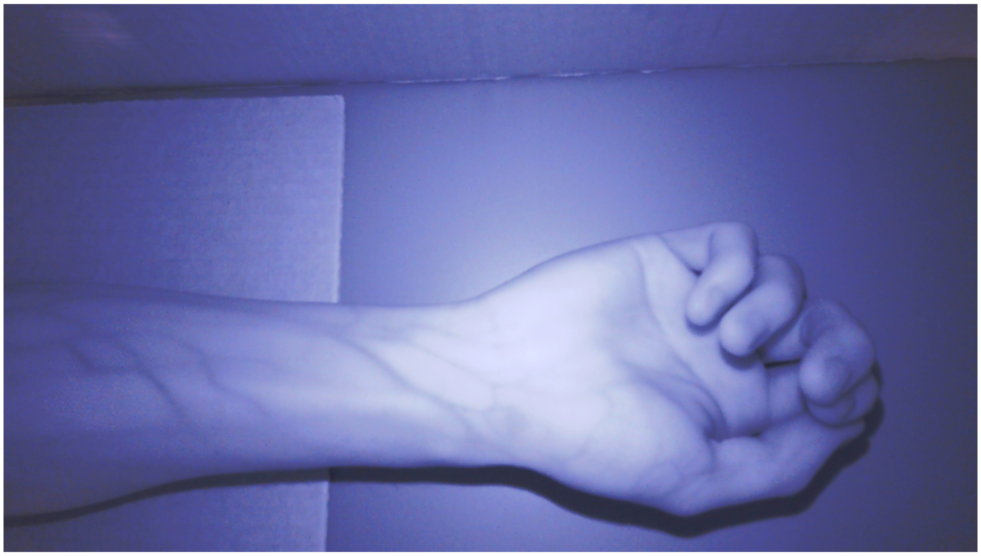
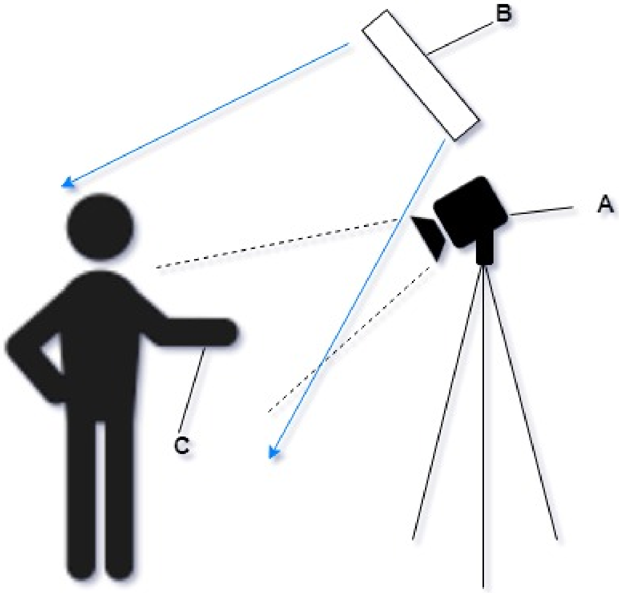


Figure 4.18: Example of blooming.

4.2.3 Experimental test 3

For the third test it was conducted an experiment to block out the visible light from the electromagnetic spectrum. It would make the camera capture only the reflected light outside of the visible light spectrum, making this a Near-Infrared Camera. To achieve this an optical blocking filter was necessary, and for this an exposed negative film from an old camera was used. As this absorbs all visible light and passes all light outside the visible spectrum, this gave the needed properties without breaking the bank. The exposed negative film is supposed to have the same properties as an expensive filter, and is a hack commonly used by armature photographers (see Fig 4.20). This setup rendered it possible to take photos in an open environment without the need of the cardboard box.

Since the filter only passes the wavelengths outside the visible spectrum, we could now take pictures without the cardboard box. To achieve the same results as the ones from the prior tests, a new and brighter lights source was needed. A new custom light housing with reflectors and diffusing was built for this purpose. 4.1.1. For the test, we exposed a human hand with the new light source and captured a image with the modified camera as shown in the Fig. 4.19. The room the test was conducted in was exposed by both light from the sun, and by light tubes in the ceiling.



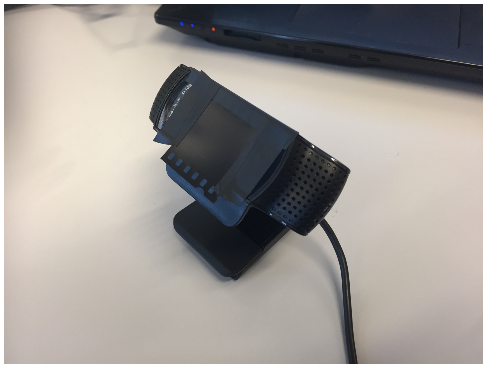


Figure 4.20: Camera with filter

Figure 4.19: Test setup. (A) indicates the camera, (B) indicates the IR light source, and (C) indicates a human arm.

The results showed the same good contrast between the blood vessels and surrounding skin as the previous test, but visible light were no longer an issue, see Figure 4.21. The filter seemed to work surprisingly well and almost no light from the visible spectrum were absorbed by the image sensor.

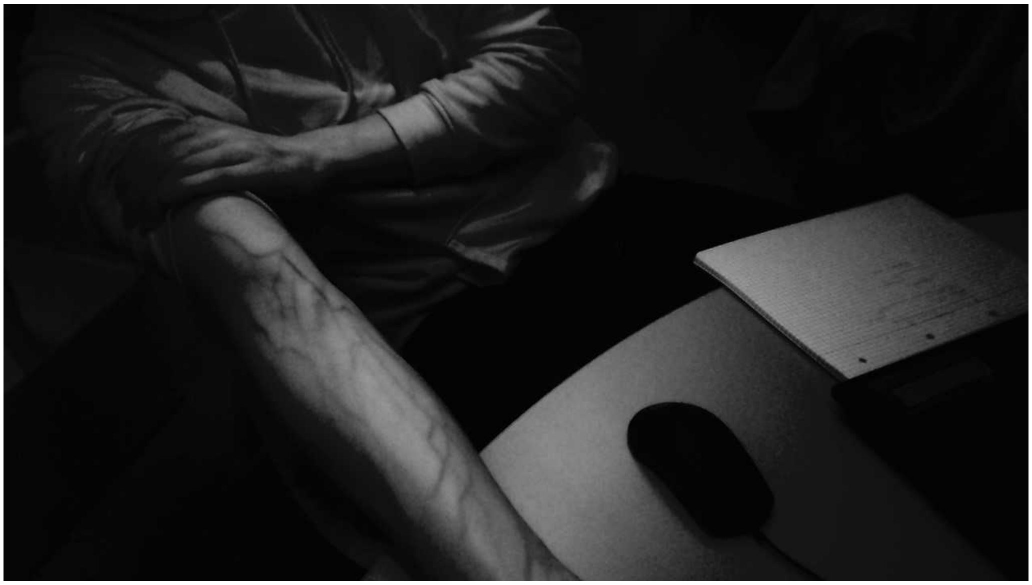


Figure 4.21: Image taken using negative film as filter

4.2.4 Experimental test 4

For the fourth test, it was now time to test the methods used in the previous test on real fish fillets. The experiment took place at Optimar's facilities who had acquired a box of cod for the occasion. The fish had been dead for several hours and were stored in a refrigerator. Setting up the test environment, a piece of dark blue conveyor belt was used as the underlay for the fillet. The conveyor belt was used to that the test images would have correlation with a real-world application so that the images could later be used for testing image processing techniques. It was chosen a dark conveyor belt to get a good contrast between the white fillet and the background. This will make it easier to segment the fillet from the background when testing image processing methods.

The fish was then filleted, and one of the halves were placed on the conveyor belt. Using the same camera and light source from the previous test, it was captured images of the fillet. Both the camera and the light was pointing at the fillet with a slight angle as seen in Fig 4.22.

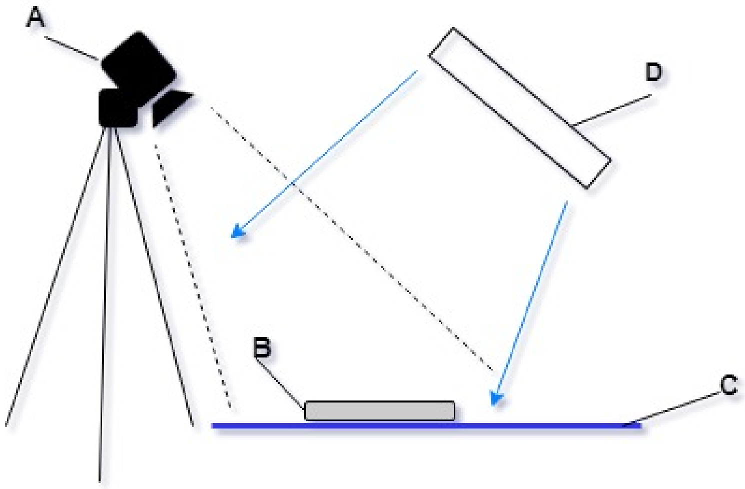


Figure 4.22: Test setup. (A) indicates the camera, (B) indicates the white fish fillet, (c) is the conveyor belt, and (D) is the IR LED housing.

The fish acquired were of premium quality and did therefor have very little or no blood spots. Looking at the images, none of the blood spots on the fillet appeared to be enhanced by the infrared light. In Fig 4.23 a red circle and a black line marks the area of the fillet where one could see a blood vein by eye, but the camera were not able to pick it up. The test was repeated several times, but the results stayed the same.

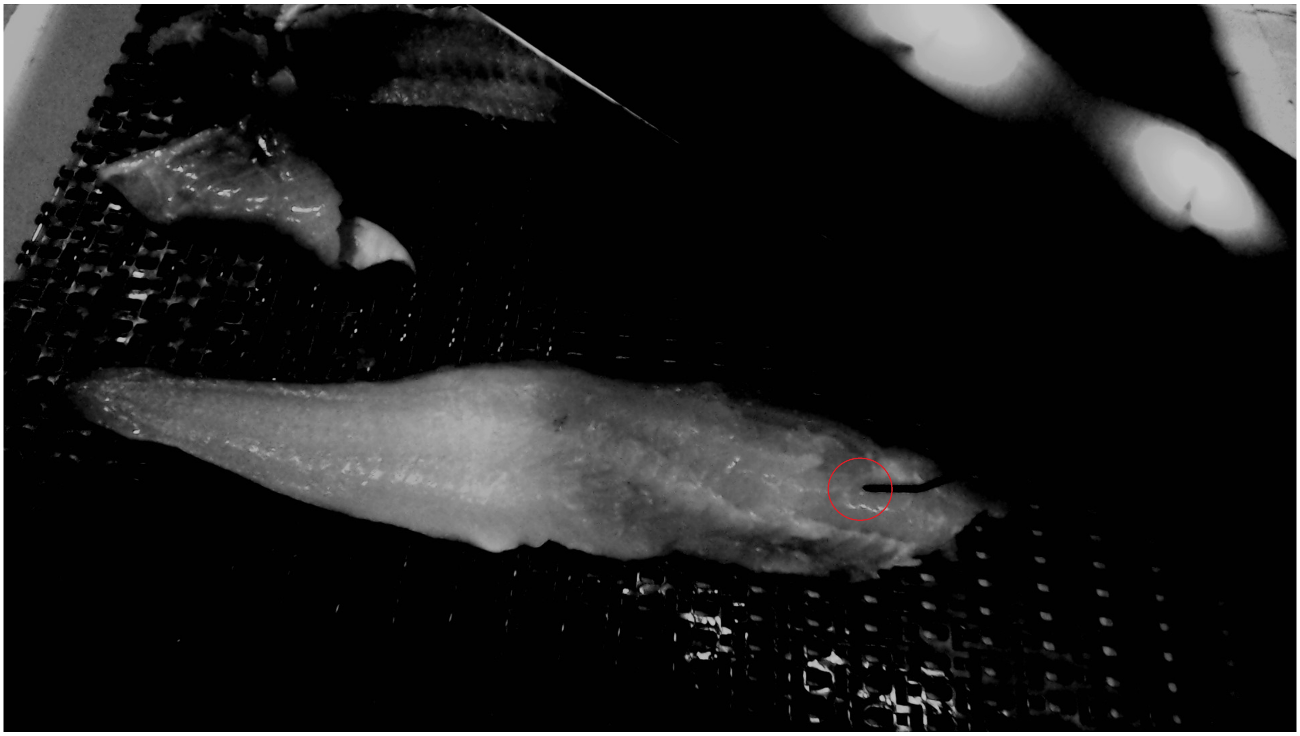


Figure 4.23: Red circle marks the area where a blood spot is visible to the eye.

4.2.5 Experimental test 5

For the fifth test, a practical test using lights with the same wavelengths found in the documents by Olsen [29] was conducted. The goal for this test was to illuminate freshly cut fillets and see if there was any enhancement of the blood spots in the fillet. To properly illuminate the fillets, it was decided to use a technique called "dome diffuse" [6]. For the test it was acquired two fish of type pollock which still was alive to keep them as fresh as possible. The fish was killed in a human way and then filleted, leaving two fillets from each fish. The fillets from the first fish was placed in a dome as illustrated in fig 4.24, using a black plastic bag as the underlay. Using a pc, it was captured several images of the fillets with each of the colored lights. Because of all the auto adjustments in the camera software, it was taken some additional images where the exposure was manually adjusted. The same procedure was then executed for the second fillet.

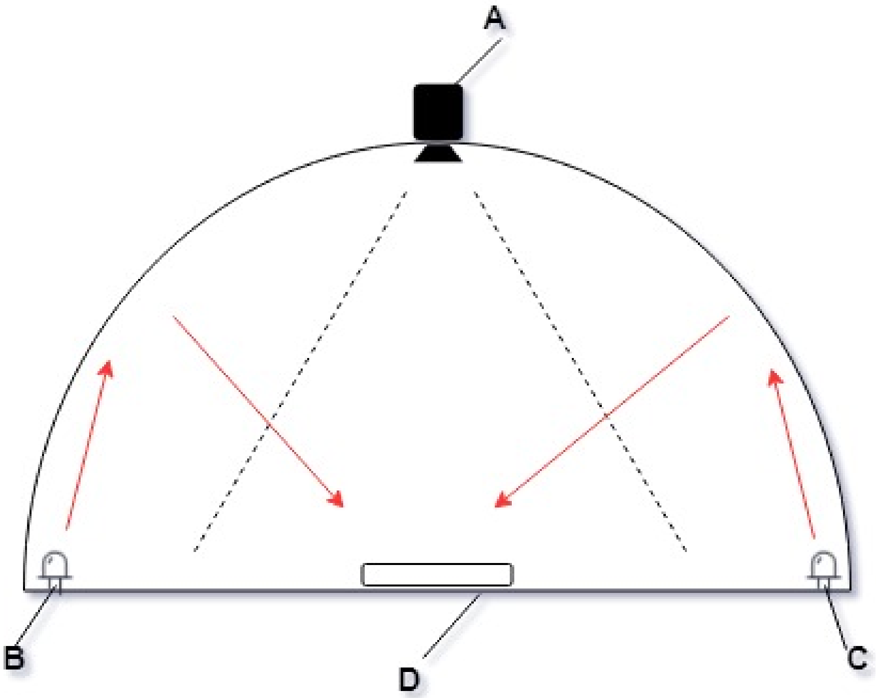


Figure 4.24: Test setup. (A) indicates the camera, (B,C) indicates the LED light sources, and (D) indicates the fish fillet.

Fig 4.25 shows an image of the first fish filleted. The image show that both the fillets contains areas where blood spots are visible, and some areas of the meat has a red tone to it. One can also see how the light causes the fillet to reflect the light, creating white areas on the fillet.

Fig 4.26 shows an image of the fillets illuminated by the 630nm red LED's. One can see that both fillets have been properly illuminated, and there are no shadows around the edges of the fillets. There are no reflections causing unwanted white spots in the image, but the image seems to a little bit blurry. Looking at the fillets one can clearly see some black spots which seems to be blood that have emerged.

Fig 4.27 shows an image of the fillets illuminated by the 525nm green LED's. Same as the red illuminated image, there are no shadows around the edges of the fillets or unwanted reflections. The image clearly shows an enhancement of what appears to be blood spots.

Fig 4.28 shows an image of the fillets illuminated by the 428nm blue LED's. The fillets are not that well illuminated, making the image look dark and blurry. This makes it hard to look for details, but one can see that some areas contains dark spots and lines that appears to be blood.

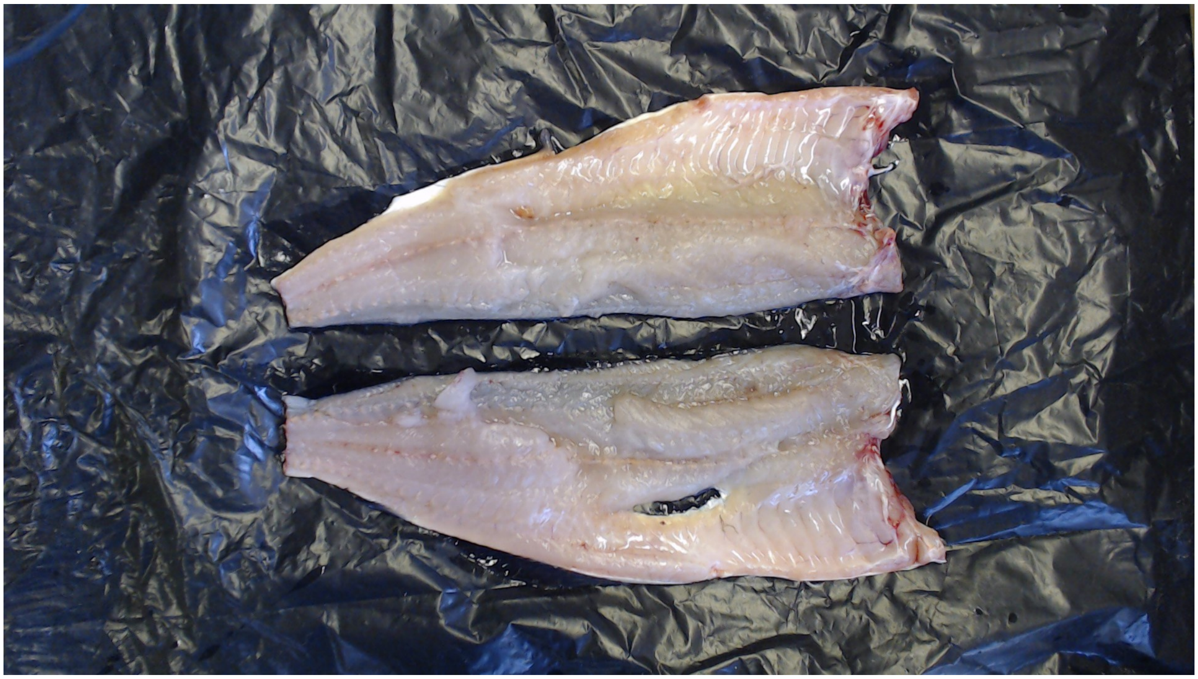


Figure 4.25: Image of the fish fillets.

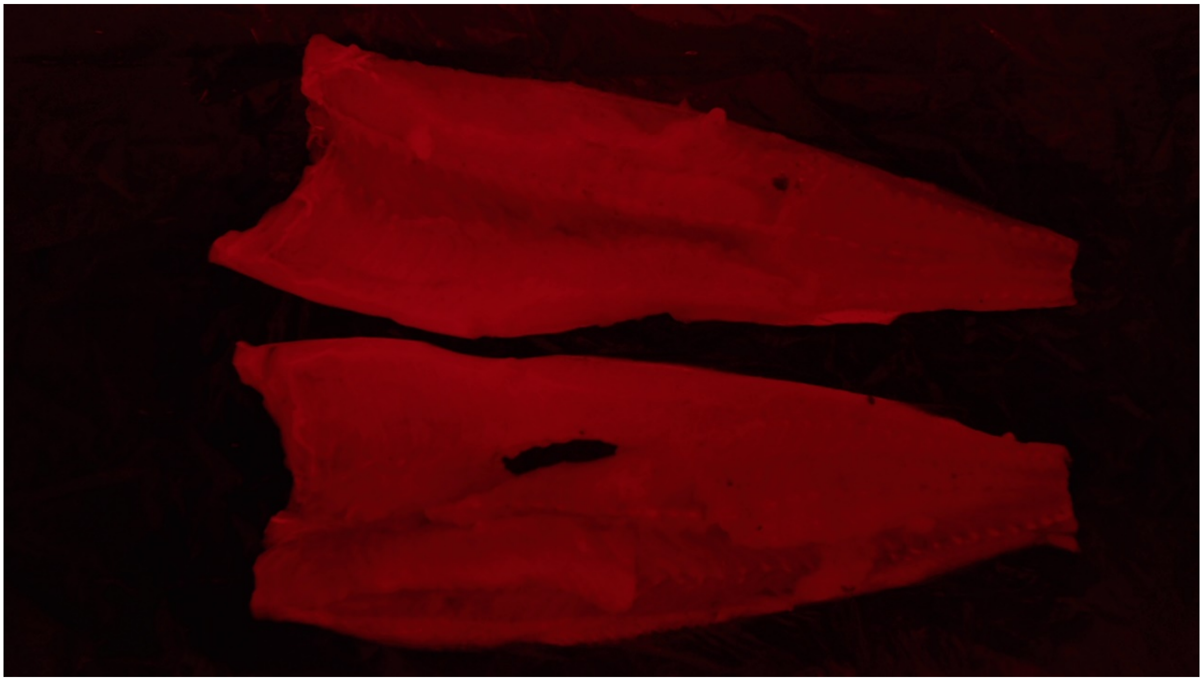


Figure 4.26: Red illuminated fillets (640nm).

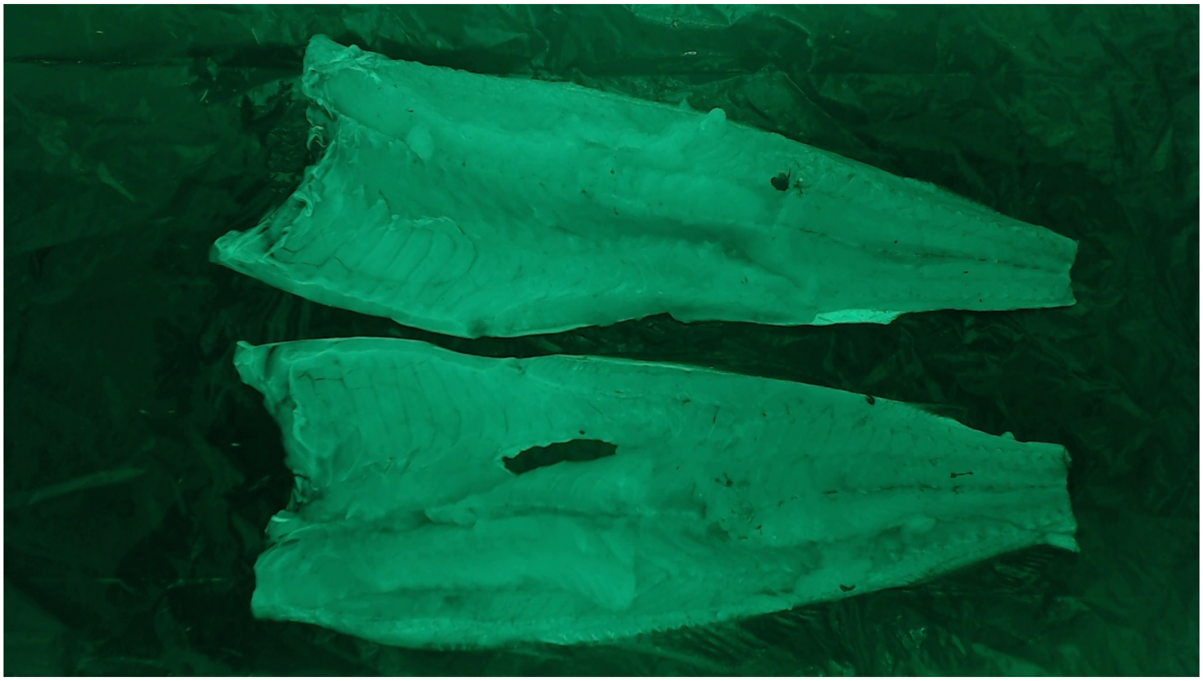


Figure 4.27: Green illuminated fillets (525nm).

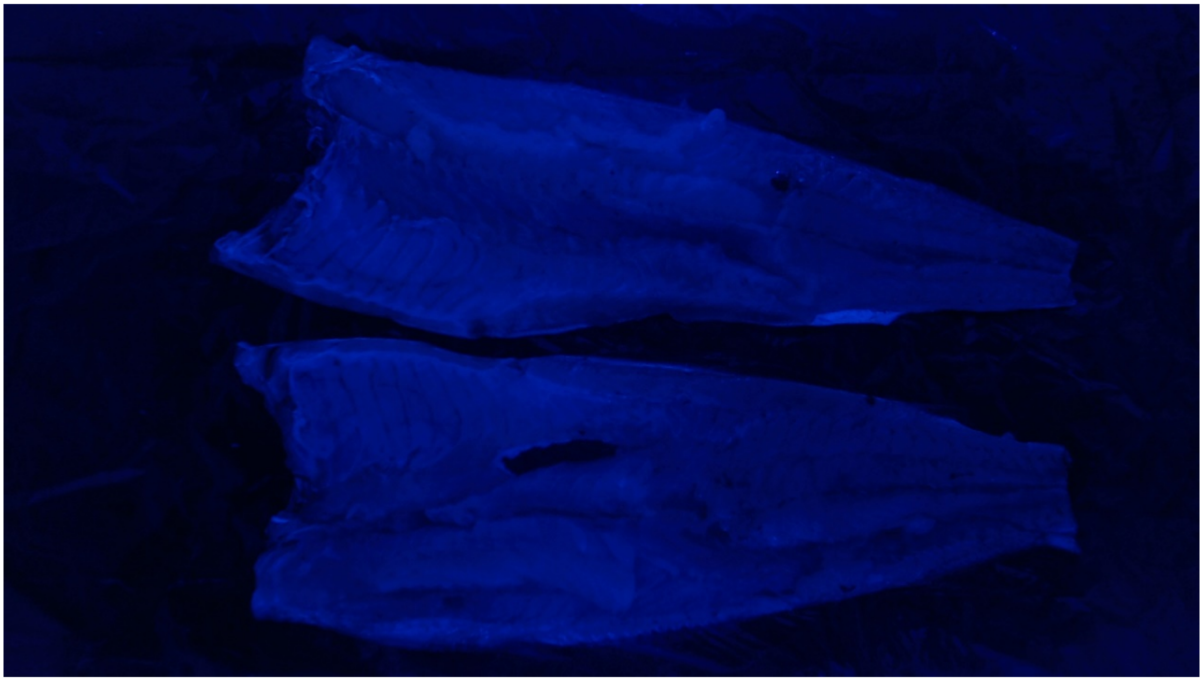


Figure 4.28: Blue illuminated fillets (428nm).

4.3 Image Processing

4.3.1 Object Detection

To be able to capture images of the fillets at the right time, a detection method was made using image processing methods from the OpenCV library. When creating the object detection, it was assumed that every fillet on the conveyor would lay in a more or less horizontal position. The object detection method constantly grabs images from the camera. The image then goes through a cropping process where two rectangles are cropped out from the image. These two rectangles are then converted into grayscale images and then a binary threshold is applied to both rectangles. Both rectangles are then checked for nonzero pixels (white pixels in this case since it's a binary image) and if the number of nonzero pixels in both rectangles exceeds a given number, it gets registered as an object. Fig. 4.29 and fig 4.30 demonstrates how a fillet coming in on a conveyor would be detected using the rectangles.

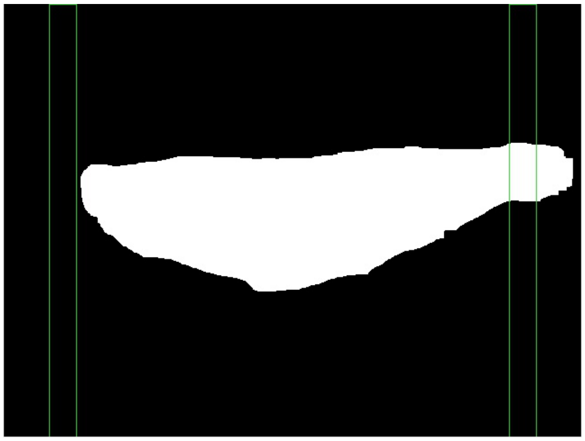


Figure 4.29: No object detected

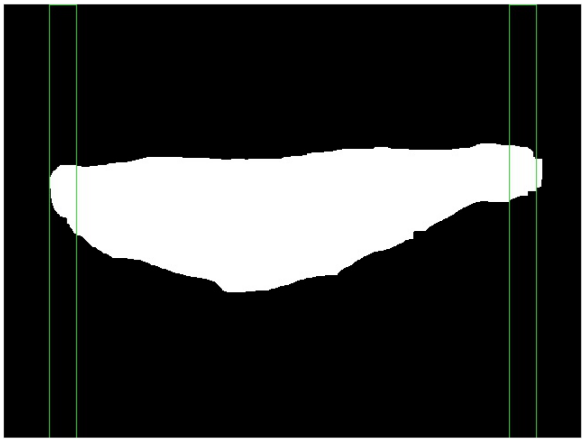


Figure 4.30: Object detected

4.3.2 Segmenting and finding Area of the Fillet

In figure 4.31 one can see the result given from using several image processing methods. One can see the fillet has been segmented from the background using the Otsu threshold method 3.3.2. The eroding 3.3.2 and dilating 3.3.2 methods have removed almost all the noise and unwanted objects, leaving some objects in the top left and right corners.

Fig 4.32 shows a representation of all the contours found in fig 4.31 using the findContours method 3.3.2 from the image processing library opencv. One can see that all the white objects in the image has a green line closing around them. The largest contour is then found, and the area is calculated.

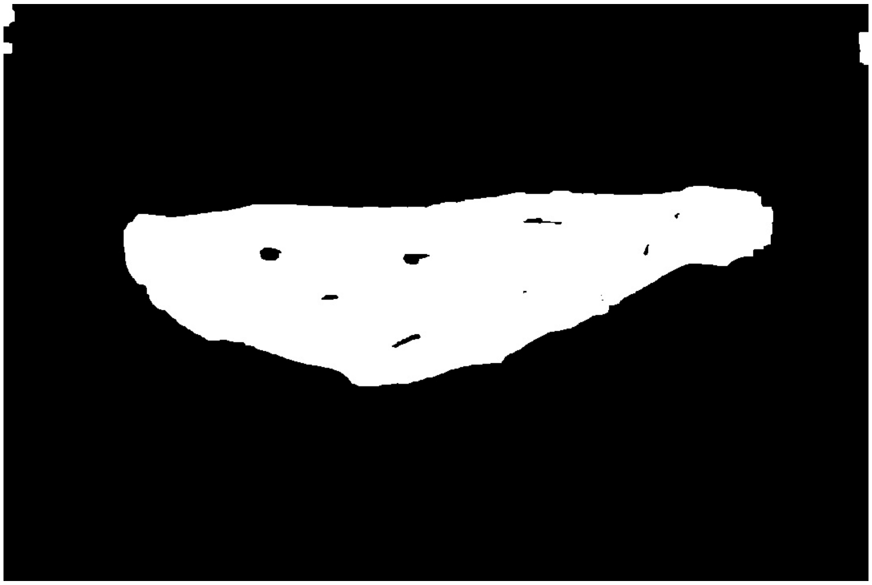


Figure 4.31: Segmented fillet

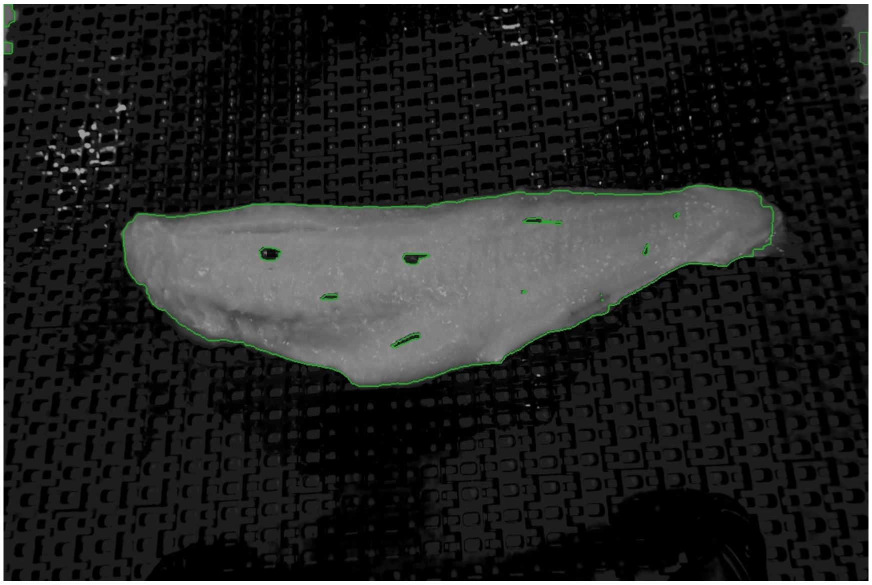


Figure 4.32: Fillet contours

4.3.3 Segmenting and finding Area of the Blood Spots

In fig 4.33 one can see the result from using all the image processing methods in section 3.3.2. All blood spots have been segmented from the fillet, leaving a black background. The adaptive threshold method 3.3.2 segmented all the dark spots on the fillet, and the imclearborders method 3.3.2 isolated the blood spots in the image, leaving no noise or unwanted objects.

Fig 4.34 shows a representation of all the contours found in fig 4.33 by using the findcontours method 3.3.2. One can see the green lines close around the blood spots. The area of all the closed contours are calculated and the total area is found.

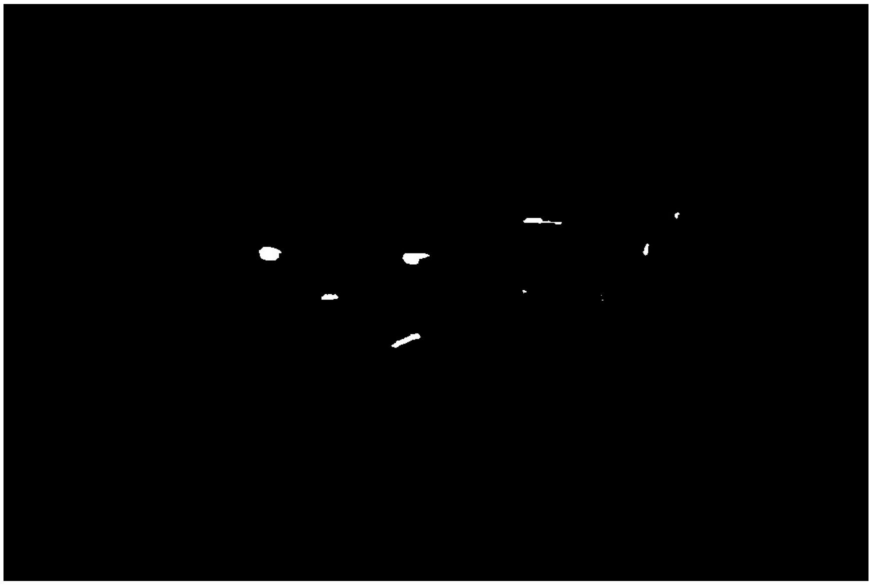


Figure 4.33: Segmented blood spots

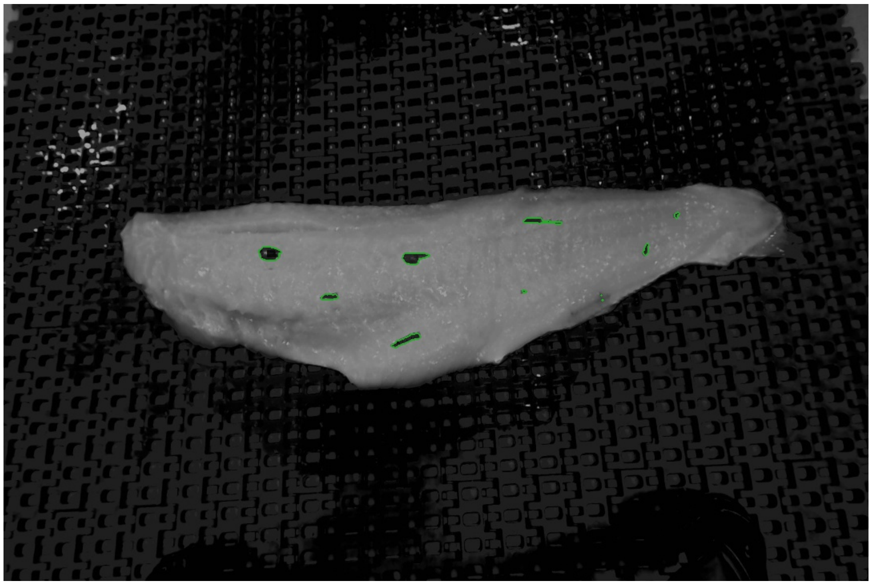
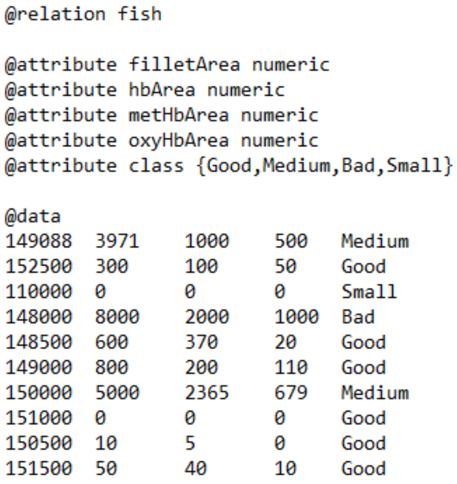


Figure 4.34: Blood contours

4.4 Classification

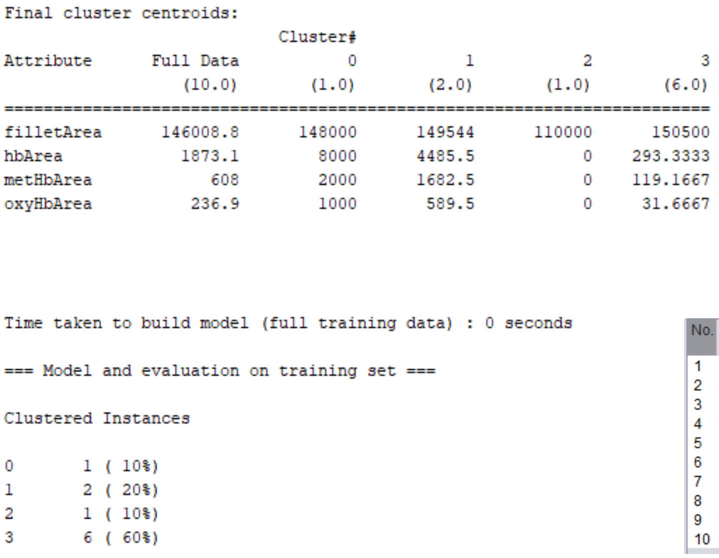
To be able to classify a fillet, the data extracted from the fillet needed to be evaluated with a trained dataset. This meant it needed to be collected data from a larger number of fillets to create a training dataset. However, getting access to a large amount of fresh white fish fillets turned out to be impossible at the time. It was therefore improvised a small dataset containing manipulated test data. In Fig 4.35 one can see an representation of the dataset used for training the cluster. The dataset contained 5 attributes, the first four being numeric values representing data extracted from the fillet. The fifth attribute only describing what each fillet should be classified as and is therefore ignored when running the K-Means clustering algorithm.



```
@relation fish

@attribute filletArea numeric
@attribute hbArea numeric
@attribute metHbArea numeric
@attribute oxyHbArea numeric
@attribute class {Good,Medium,Bad,Small}

@data
149088  3971    1000    500     Medium
152500  300     100     50      Good
110000  0       0       0       Small
148000  8000    2000    1000    Bad
148500  600     370     20      Good
149000  800     200     110     Good
150000  5000    2365    679     Medium
151000  0       0       0       Good
150500  10      5       0       Good
151500  50      40      10      Good
```

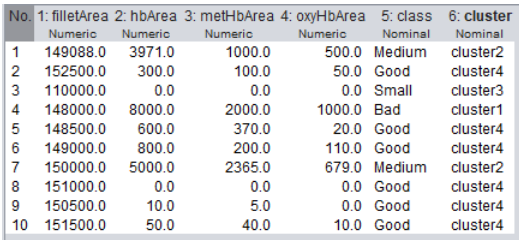
Figure 4.35: ARFF file containing training data.

Fig 4.36 shows the output data represented after running the K-Means algorithm. One can see the number of clusters and how many instances are assigned to each cluster. The output also shows the final cluster centroids and the time taken to train the cluster. The output data does not show which instance is assigned to which cluster, so therefore the Weka application was used to visualize all the instances and its assigned cluster. In Fig 4.37 one can see a table containing all five attributes from the dataset and which cluster each instance has been assigned to, confirming that the instances with the most resemblance have been assigned to the same clusters. Fig 4.38 shows a confusion matrix of the cluster results.



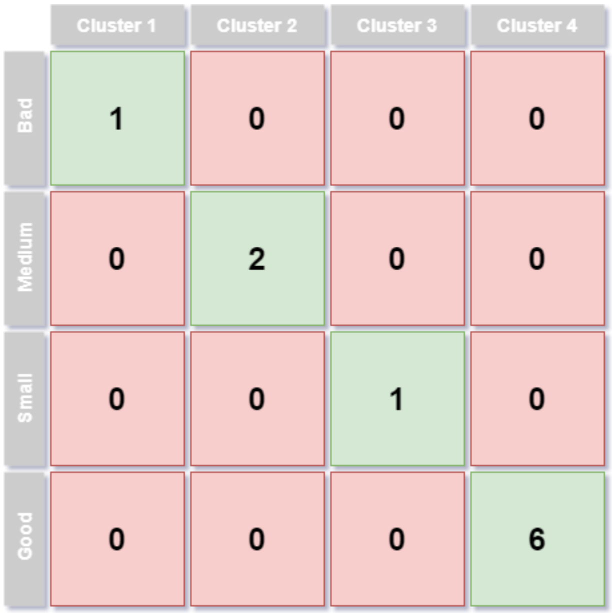
```
Final cluster centroids:
                                 Cluster#
Attribute       Full Data               0            1            2            3
                   (10.0)           (1.0)        (2.0)        (1.0)        (6.0)
==========================================================================
filletArea       146008.8          148000       149544       110000       150500
hbArea             1873.1            8000       4485.5            0     293.3333
metHbArea             608            2000       1682.5            0     119.1667
oxyHbArea           236.9            1000        589.5            0      31.6667




Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       1 ( 10%)
1       2 ( 20%)
2       1 ( 10%)
3       6 ( 60%)
```

Figure 4.36: Output data from clustering.



No.	1: filletArea Numeric	2: hbArea Numeric	3: metHbArea Numeric	4: oxyHbArea Numeric	5: class Nominal	6: **cluster** Nominal
1	149088.0	3971.0	1000.0	500.0	Medium	cluster2
2	152500.0	300.0	100.0	50.0	Good	cluster4
3	110000.0	0.0	0.0	0.0	Small	cluster3
4	148000.0	8000.0	2000.0	1000.0	Bad	cluster1
5	148500.0	600.0	370.0	20.0	Good	cluster4
6	149000.0	800.0	200.0	110.0	Good	cluster4
7	150000.0	5000.0	2365.0	679.0	Medium	cluster2
8	151000.0	0.0	0.0	0.0	Good	cluster4
9	150500.0	10.0	5.0	0.0	Good	cluster4
10	151500.0	50.0	40.0	10.0	Good	cluster4

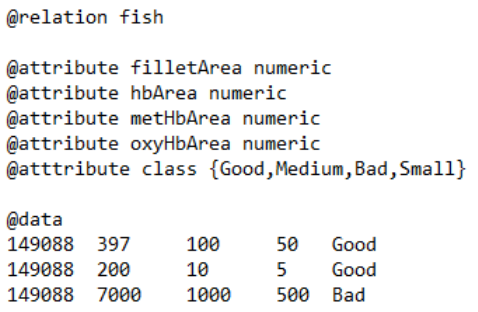
Figure 4.37: Table showing the cluster results.



	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Bad	1	0	0	0
Medium	0	2	0	0
Small	0	0	1	0
Good	0	0	0	6

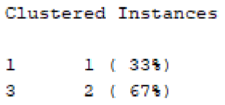
Figure 4.38: Confusion matrix of the clustering result.

Classifying an instance was done by loading a dataset containing test data and evaluating it with the trained cluster. Fig 4.39 shows a dataset containing three instances. Evaluating the dataset with the cluster model gives an output showing which cluster each instance belongs to, see Fig 4.40. Again, the fifth attribute was ignored when running the K-Means algorithm, and by looking at the dataset and the output, one can see that the instances has been classified correctly. One instance being classified as Bad, and two instances being classified as Good.



```
@relation fish

@attribute filletArea numeric
@attribute hbArea numeric
@attribute metHbArea numeric
@attribute oxyHbArea numeric
@atttribute class {Good,Medium,Bad,Small}

@data
149088  397     100     50   Good
149088  200     10      5    Good
149088  7000    1000    500  Bad
```

Figure 4.39: Dataset containing test data.



```
Clustered Instances

1       1 ( 33%)
3       2 ( 67%)
```

Figure 4.40: Cluster evaluation result.

Because of lack of knowledge regarding the appearance of a wrongly cut fillet, it was assumed that a wrongly cut fillet would mostly go on the change in size. It was therefore decided to use clustering to classify fillets as small when the area of a fillet was significantly smaller than a normal fillet. This method was taken up for discussion at a later meeting with Optimar, and it was concluded that the method would probably not work as intended. It was then explained how a wrongly cut fillet may look like and a proposition was given to look into RST-invariant.

4.5 Software architecture

The overall software architecture consists of a java application and an Arduino. The java application operates as the master, doing all the main work. Using serial communication, the java application communicates with an Arduino operating as slave. The Arduino acts as the physical link in the system, performing tasks like turning on lights when the master sends information to do so.

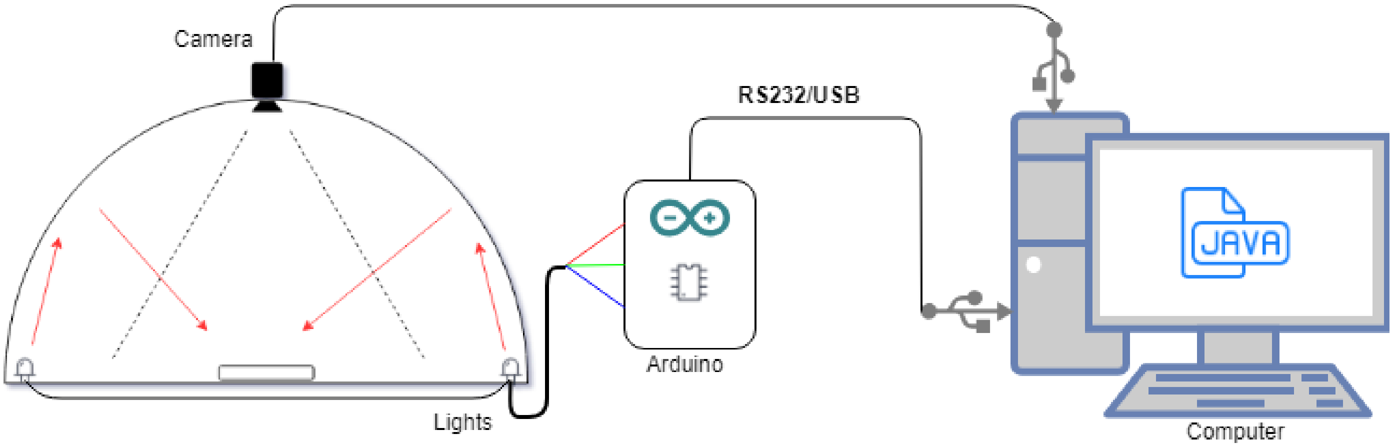


Figure 4.41: Graphical illustration of the physical setup.

4.5.1 Java Application

The java application has a total of 15 classes.

- **Camera** class holds method for capturing images.
- **Cluster** class holds methods for training and loading clusters, as well as evaluating data with the cluster.
- **dataDistribution** class sets the classification parameters in the GUI, and sends the classification value to the arduino.
- **GUI** class holds methods for displaying data in the GUI. The class also holds the graphical representation of the GUI and methods for handling interaction with buttons and such.
- **ImageData** class grabs images from the StorageBox class, applies image processing methods to the images, writes the data from the images to an ARFF file, evaluates the file with the cluster and logs the data.
- **ImageMethods** class holds image processing methods.
- **LiveStream** class grabs videoframes from the camera and displays them in the GUI.
- **Logging** class holds methods for logging classification data.
- **Main** class initializes all the threads and starts them, as well as loading the trained cluster model.
- **ManualMethods** class holds image processing and drawing methods for the manual control page in the GUI.
- **ObjectDetection** class constantly grabs frames from the camera and checks for objects.
- **Server** class establishes connection with the arduino and reads and writes data.
- **StorageBox** class stores data and images. Some in synchronized methods.
- **TakeImage** class tells the camera when to take images and times the lights connected to the arduino.
- **WriteData** class holds method for creating ARFF file.

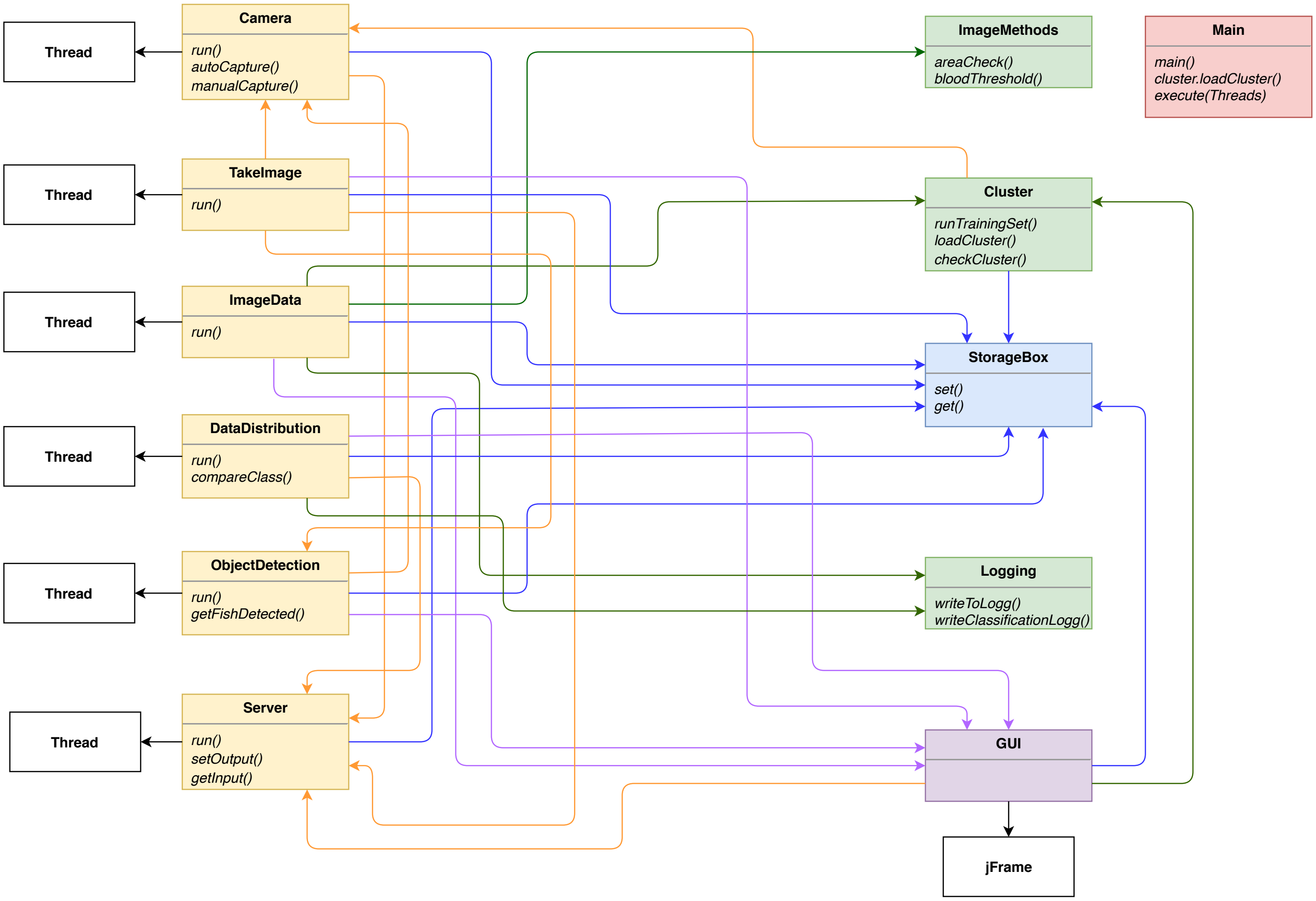


Figure 4.42: Class diagram of the most important classes in the java application

Application process

When the java application is launched, the main class starts all the threads and the GUI is set visible, displaying the main window. By turning on object detection in the tools menu and pressing the start button the application will start looking for fillets. The ObjectDetection class constantly grabs images from the camera and checks for fillets. When a fillet is detected the class sets a flag indicating so.

The TakeImage class constantly checks the flag in the ObjectDetection class. When the flag indicates a fillet has been detected, the TakeImage class sets the output parameter in the server class to be written to the Arduino client. It simultaneously calls the autoCapture method in the Camera class to take images.

The autoCapture method in the Camera class captures three images which are timed with the Arduino client to illuminate each image with three different lights (red, green, blue). The images are then stored in synchronized methods in the StorageBox class. The methods are synchronized so that the images won't get overwritten by new images before they have been utilized. The ImageData class constantly checks for new images in the StorageBox class. When there are new images, the class retrieves the images and applies image processing methods from the ImageMethods class. The red illuminated image is checked to find the area of the fillet, and the area of any blood spots. The green and blue illuminated images are also checked to find the area of any blood spots. An ARFF file containing the four parameters extracted from the images is then created and evaluated using the checkCluster method in the Cluster class. The checkCluster method classifies the fillet based on the data in the ARFF file and stores the result in the StorageBox class.

After evaluation, the DataDistribution class retrieves the result from the StorageBox class. The DataDistribution class then checks the result and displays the right information in the GUI. It also sets the output in the Server class to write the result to the Arduino client.

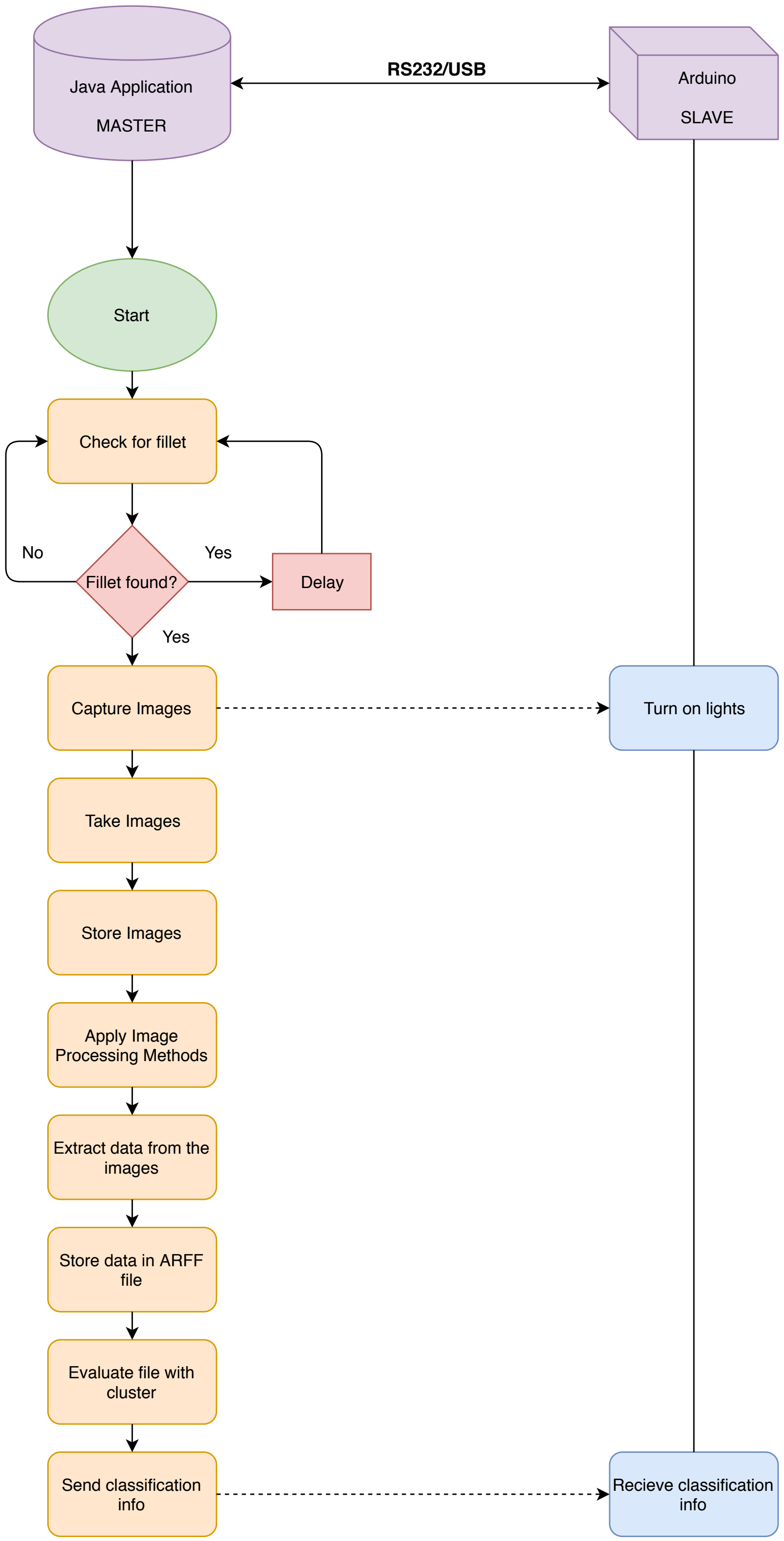


Figure 4.43: Flowchart showing the system processing

4.5.2 HMI

The HMI consists of several windows which lets one observe and control the application. The HMI was designed to be easy to understand and lets one do anything from observing the main process to manual testing.

Main

The main page is shown when the application is started. This page holds the start and stop buttons which lets you start and stop the application. The larger section of the page displays the first image taken of each fillet, and one have the option to turn on visualization of the fillet segmentation by checking the “show contours” button. A section at the far right of the page displays all data extracted from each fillet and the classification of the fillet. The section also displays how many fillets has been checked and how many has been classified as what. At the top of the page, there is a bar menu where one can choose between other features.

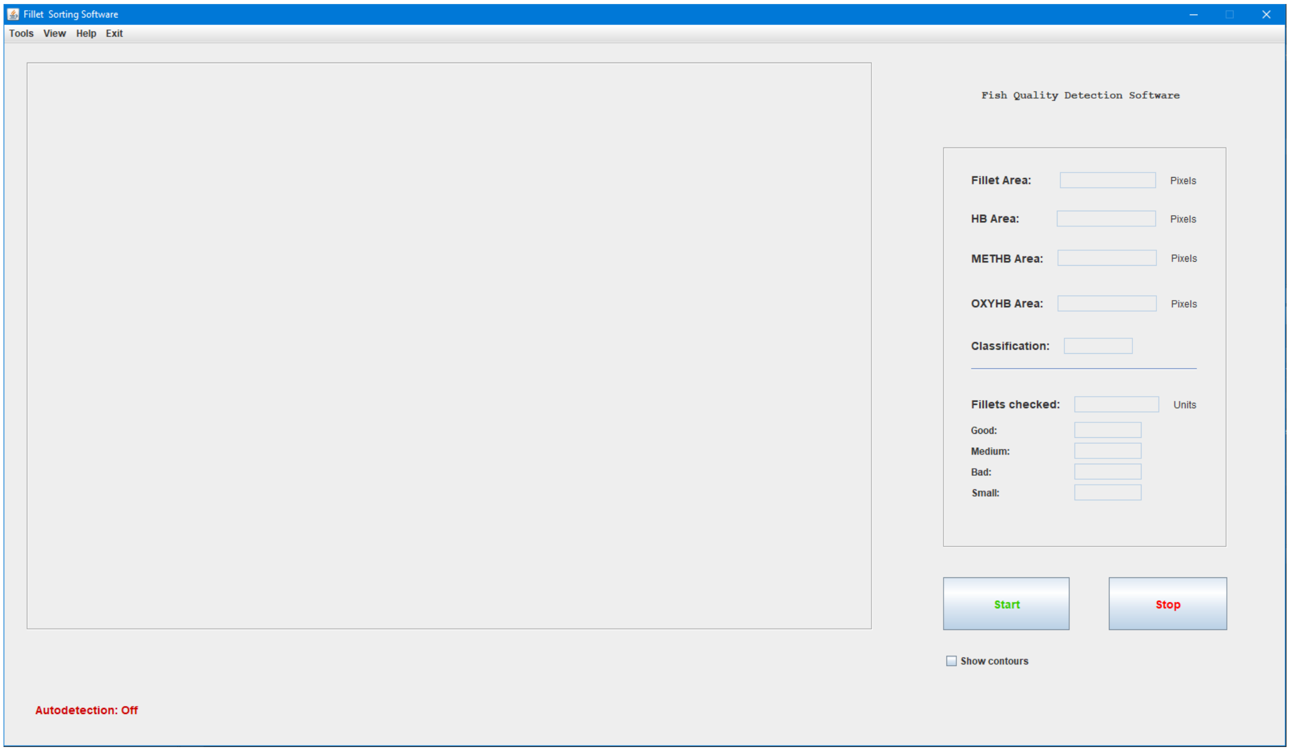
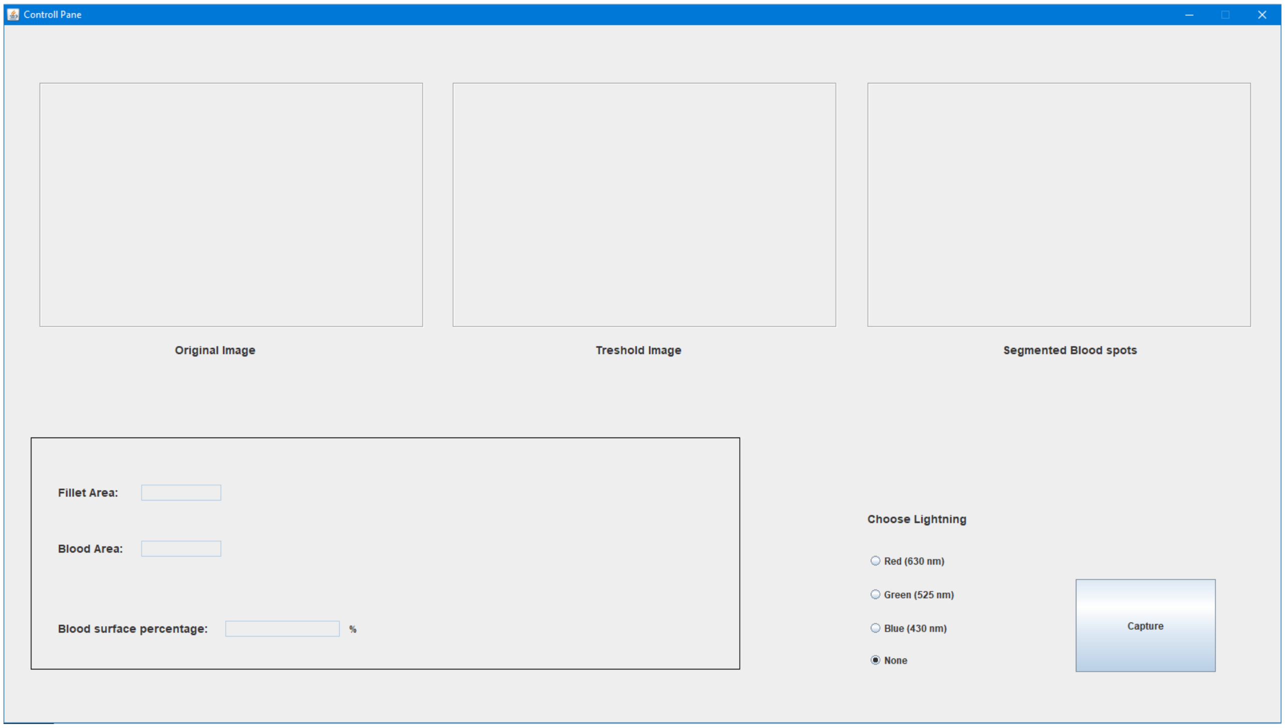


Figure 4.44: Main window of the HMI

Manual test window

The manual control page lets you manually check if the lights and the image processing methods works as they should. At the bottom right of the page one is given an option to choose between light sources, and a button to capture the image. The down left section displays the area of the fillet and the blood spots, as well as the percentage of the fillets surface containing blood. At the top of the page one can see the original image, the segmented fillet, and the segmented blood spots.



○ Red (630 nm)
○ Green (525 nm)
○ Blue (430 nm)
● None

Figure 4.45: Manual testing window of the HMI

Illumination control window

The illumination control page displays the three images captured of each fillet. This lets one check if the images are illuminated by the correct color while the application is running. One can also turn on visualization of the segmented blood spots in each image by checking the checkbox "Show segmented blood spots" at the downright corner of the page.

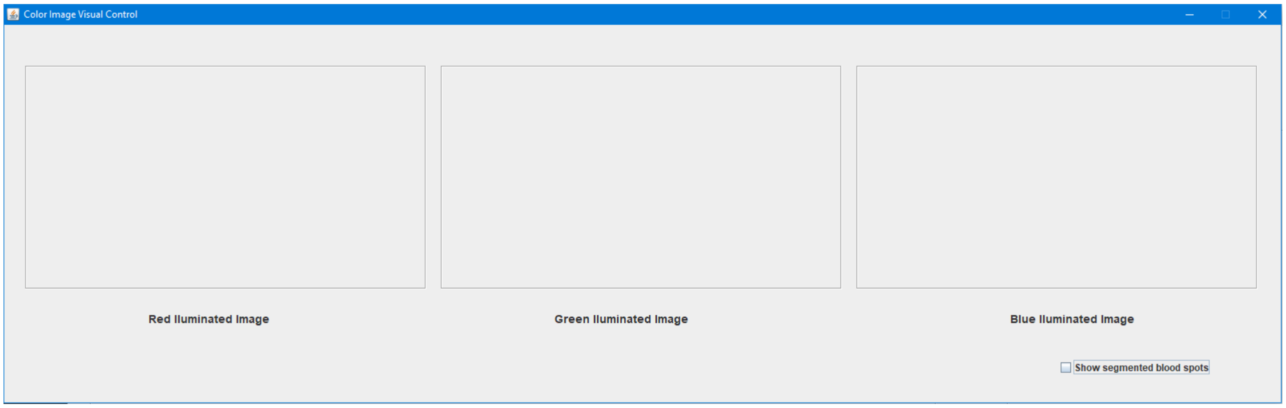


Figure 4.46: Illumination control window of the HMI

Live stream window

The live video feed page lets one see live video feed from the camera. By checking the "Show boxes" checkbox, two rectangles will overlap the image. These two rectangles are placed at the same place where the object detection 4.3.1 crops out its rectangles to give a good visualization of the object detection.

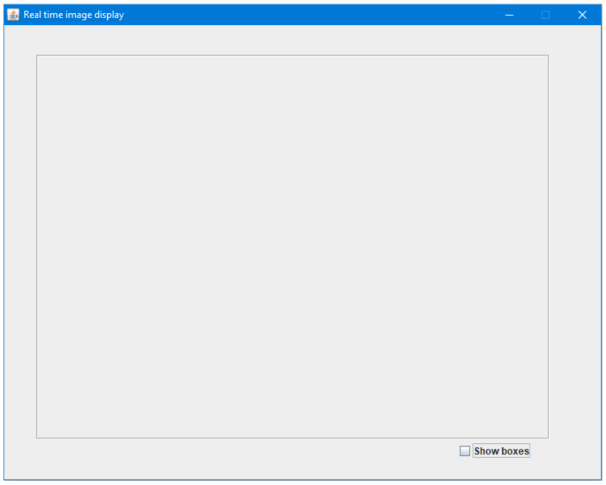


Figure 4.47: Live stream window of the HMI

4.5.3 Arduino client

When the Arduino is connected to a power source it will constantly check for a serial connection in the main loop. If there is a connection, the Arduino will read data from the java application and call the two methods **lightSource()** and **qualitySignal()** . The two methods will check the input data and execute a task if the input data matches any of the cases. The main loop also checks an input which can be connected to a button, and sends a variable to the java application if the input is triggered. The purpose of this button is to make it possible to manually capture images in the java application. Capturing images manually will make it more controllable to capture individual images of fillets when collecting data for datasets.

- **Lightsource()** This method is used to turn on the lights connected to the Arduino. It will turn on the red, green, or blue light for 500 ms depending on the input data.

- **qualitySignal()** This method is used to simulate the classification result from the java application by setting different outputs on the Arduino depending on the input data.

Chapter 5

Discussion

5.1 Design

Every experiment conducted, needed a source of illumination. For the first three iterations 4.2.1, 4.2.2 and 4.2.3, we used a simple cardboard box to block out light from the surroundings in combination with the LEDs mentioned in the respective tests. The lighting in all three test seems poor and are not suited for a real world industrial application as seen in Fig. 4.23, 4.17 and 4.21. However, they provide a basis for further testing and research.

The tests mentioned above, gave the basis for the test conducted on real fish fillets 4.2.4, provided by Optimar. This illumination source was built as elaborated in the design section in the result chapter 4.1.1. This design gave somewhat decent lighting, but was mainly discarded because we needed more consistent light throughout the image.

To achieve better illumination another design alternative called 'dome diffuse' from section 4.1.3 were initialized. Dome diffusion is commonly used to illuminated reflective surfaces as described by the article '*A Practical Guide to Machine Vision Lighting* [6]' and were therefore a very appealing solution. The dome is illuminated by LEDs derived from a spectroscopy done by [29]. As seen in the Fig. 4.5, 4.7, 4.8 and 4.9, one can see the imperfections in the reflective material used in the reflector. Although these imperfections is visible in the images above, they didn't have any noticeable impact on the actual test in the result section 4.2.5. Comparing the image without dome diffusion in Fig. 4.5 with the ones with illumination in Fig. 4.7, 4.27 and 4.28, it's easy to see how efficient this method is. The dome diffusion technique gives less reflection from the moist muscle tissue of the fillet, and this supports the claims made in the article [6]

A control unit for the lights were made as seen in the section 4.1.4. The Arduino used might not be the best for this application, in an industrial implementation of such system a PLC would be a better choice. Eliminating poor communication speeds, providing better performance and versatility to the system.

5.2 Experimental Tests

The experimental tests were conducted to see how different wavelengths of light, responded to blood residue. For the two first tests, the light source was selected based on the principle of Near-Infrared imaging. Boas and Franceschini [11] explains that the dominant absorbers in the human body both in the near infrared and visible wavelengths are oxyhemoglobin, deoxyhemoglobin and water. Wavelengths in the NIR region are also less prone to scattering, and can therefor penetrate deeper into biological tissue than visible light. As elaborated by Weissleder [43] near infrared light is needed for imaging deeper than 500 μm. Based on the theory above we chose two types of LED in the near infrared region, at 880nm and 890nm as shown in Table. 3.1. These were used for the four first tests in section 4.2. The theory applied in a real world application showed some promising results on human blood vessels in 4.2.1, 4.2.2 and 4.2.3, but not as good for the 4rd iteration 4.2.4 conducted on a whitefish fillet. Seeing that the test was unsuccessful for the whitefish fillet, left some questions regarding the use of NIR lighting in an application for fish. The first reason why this might not work in such application, might be the extremely good quality on the fish the experiment was conducted on. Secondly the theories of NIR light might not be applicable on the hemoglobin in fish, possibly because human have another mutation of hemoglobin than the whitefish. The fish used in the test were bleed out and slaughtered some time before the test was conducted, the degrading of hemoglobin post mortem might influence the results.

After the Near infrared lighting hypothesis was disregarded, another one emerged, based on the previous studies of [29]. The choice of LEDs for the 5th test 4.2.5 was derived from his spectroscopy of hemoglobin in this paper. From three peaks in the spectroscopy, the wavelengths 630nm, 525nm and 428nm were chosen as illumination. After noticing how poor the illumination was on the prior test, a technique called dome diffusion [6] was implemented. This illumination strategy is designed to suppress shadows around the edges, better light spread and diffuse the light. The results in the test speaks for themselves, the illumination strategy seems to spread the light as intended. However not all the test images show the results desired. The image where we used the red LEDs at 630nm are too bright as seen in Fig.4.26. There are many reasons that can cause an over illuminated image, that might suggest that the image is overexposed or that the light source is to intense. The red LED used has a luminous intensity of 12000mcd, but the green LED has almost triple the intensity of 35000mcd, yet the green image in Fig.4.27 has none of the same issues. Another reason might be the muscle tissue absorbing all light, making it look soaked and over overexposed. The green image shows promising results, blood spots that weren't noticeable in the original image in Fig.4.25 is now visible. In addition to the blood spots, some of the blood vessels are now visible. The blue illuminated image has poor light conditions, gives very little contrast between blood residue and the white muscle tissue as seen in Fig.4.28.

The image could use some more illumination, either more or brighter LEDs.

5.3 Image Processing

Finding the area of fillets and the blood spots worked well. The image processing methods used, managed to remove most of the unwanted objects and noise from the image. This left a clean binary image which gave good results when finding the contours of the fillet and the blood spots. Using a dark underlay for the fillet gave good contrast between the fillet and the underlay. This proved to be mandatory in order to segment the fillet from the background. Due to lack of image material, it has not been possible to test the image processing methods properly. It is therefore unknown how the methods would perform in different scenarios, and should be further tested to make adjustments.

The object detection was never tested on actual fillets. When creating the method, it was assumed that every fillet on the conveyor would lay in a more or less horizontal position. It was later learned that the position of the fillet could vary, meaning that the object detection would not work properly.

5.4 Classification

Using K-Means clustering for classification worked good when testing with synthetic data. As mentioned in the results, it was impossible at the time to get access to a larger amount of fish fillets. This made it impossible to test the clustering algorithm with actual data collected from fish fillets. A proper dataset would have to be made to be able to test the system in a real-world application.

Detecting wrongly cut fillets was done using clustering. The intentions were to classify fillets as wrongly cut if the size of the fillet was drastically smaller than a normal fillet. We were later informed that a wrongly cut fillet would not specifically change in size, but should also be determined by shape of the fillet. This method should therefore be further developed, and Optimar gave a proposition to look into RST-invariant.

5.5 Software

The overall software design turned out good and worked very well as a concept application. It made it possible to test out the different methods and observe how they performed. In a real-life situation the application would more likely operate on an industrial computer without the need for a HMI, but as a concept application it has worked excellent.

Using an Arduino as the controller unit for the LED's worked ok for testing, but the communication speed of the Arduino slowed down the overall process a bit. Because of the delay it introduced, the triggering of the camera had to be deled to be able to time the camera and the lights. In an industrial application it would me more realistic to use a PLC.

Chapter 6

Conclusion

The main objectives of this assignment were to find applicable methods, to determine the quality and if a whitefish fillet was cut correctly. Fish quality is mainly determined by blood residue in the muscle tissue and the shape of the fillet. To achieve such goal a vision system to detect blood residue and cutting pattern were needed to be developed. This included detection of fish, image processing methods to segment blood residue, a proper illumination strategy and a viable camera.

Fish quality assessment is currently done by visual inspection by an operator, the method is therefore prone to human errors. To eliminate errors and automate the quality assessment, we've designed a software that will calculate the blood content of the total area of the whitefish fillet. The application for quality takes an image of the fish, using illumination with specified wavelengths that are absorbed by hemoglobin in the blood residue. The absorption increases the contrast between the white muscle tissue and the blood present in the fillet, this increase leads to easier segmentation of the blood. We also implemented K-means clustering, this gives the system the ability to be trained to classify the fish fillet's quality based on its blood content. Testing and training the system on an real-world application remains to be done.

Finding methods for whether the fish was cut correctly were not focused on, the methods made for this part were only dependent on the size of the fish. Fish that were noticeably smaller than the once in the training set would be classified as small. Finding right illumination have been a huge part of this assignment, as this could benefit the image processing methods by giving the image greater contrast. Research iterations were made through experimental test of different light sources, with the focus on the absorption capabilities of hemoglobin. Concluding that some wavelengths are more efficient to use than others, as seen in the result section. This method might be applicable in an industrial application, but further testing should be done to reinforce the results from this thesis.

We consider the project to be success full, as it creates great foundation for further research and more questions to be answered.

Bibliography

[1] *About* [n.d.], https://opencv.org/about.html. Online; accessed 08-February-2018.

[2] *About the Java technology* [n.d.], https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html. Online; accessed 03-May-2018.

[3] *AJS1319F | JS Relays* [n.d.], https://www3.panasonic.biz/ac/ae/search_num/index.jsp?c=detail&part_no=AJS1319F. Online; accessed 22.may-2018.

[4] *Analyzing images using image thresholding techniques* [n.d.], https://se.mathworks.com/discovery/image-thresholding.html. Online; accessed 19-March-2018.

[5] Andrew M. Smith, M. C. M. and Nie, S. [2009], 'BioImaging: Second window for in vivo imaging', https://www.nature.com/articles/nnano.2009.326. Online; accessed 08-February-2018.

[6] *A Practical Guide to Machine Vision Lighting* [n.d.], http://www.ni.com/white-paper/6901/en/. Online; accessed 27-May-2018.

[7] *Arduino Nano* [n.d.], https://store.arduino.cc/arduino-nano. Online; accessed 03-May-2018.

[8] *Arduino UNO* [n.d.], https://store.arduino.cc/arduino-uno-rev3. Online; accessed 03-May-2018.

[9] *Attribute-Relation File Format* [n.d.], https://www.cs.waikato.ac.nz/ml/weka/arff.html. Online; accessed 19-March-2018.

[10] Beeton, B. [n.d.], 'Potinters to Frequently Asked and Answered Questions', http://tug.org/tex-ptr-faq. Online; accessed 23-May-2018.

[11] Boas, D. and Franceschini, M. A. [2009], 'Near infrared imaging', http://www.scholarpedia.org/article/Near_infrared_imaging. Online; accessed 09-February-2018.

[12] Boyle, W. and Smith, G. [1970], *Charge Coupled Semiconductor Devices*, volume: 499, issue: 4 edn, Nokia Bell Labs.

[13] Brownlee, J. [2016], 'Overfitting and underfitting with machine learning algorithms', https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/. Online; accessed 27.may-2018.

[14] *Data Mining Software in Java* [n.d.], https://www.cs.waikato.ac.nz/ml/weka/. Online; accessed 19-March-2018.

[15] El Gamal, A. and Eltoukhy, H. [2005], 'Cmos image sensors', *IEEE Circuits and Devices Magazine* **21**, 6–20. Online; accessed 13.February-2018.

[16] Fellers, T. J. and Davidson, M. W. [n.d.], 'CCD Saturation and Blooming', http://hamamatsu.magnet.fsu.edu/articles/ccdsatandblooming.html. Online; accessed 13-February-2018.

[17] Hartigan, J. A. and Wong, M. A. [1979], 'Algorithm as 136: A k-means clustering algorithm', *Journal of the Royal Statistical Society. Series C (Applied Statistics)* **28**(1), 100–108.
URL: *http://www.jstor.org/stable/2346830*

[18] Heia, K. [2017], 'Using light to find blood in fish', *Reliability Engineering and System Safety* .

[19] Inductiveload [n.d.], 'EM Spectrum Properties', https://commons.wikimedia.org/wiki/File:EM_Spectrum_Properties_edit.svg. Online; accessed 27-May-2018.

[20] *Java - Multithreading* [n.d.], https://www.tutorialspoint.com/java/java_multithreading.htm. Online; accessed 03-May-2018.

[21] Labtestonline [2015], 'Hemoglobin', https://labtestsonline.org/tests/hemoglobin. Online; accessed 07-February-2018.

[22] MATLAB [n.d.*a*], 'Clearing border components', https://blogs.mathworks.com/steve/2007/09/04/clearing-border-components/. Online; accessed 21.may-2018.

[23] MATLAB [n.d.*b*], 'Image processing toolbox', https://se.mathworks.com/products/image-processing-toolbox.html?adobe_mc_ref=https%3A%2F%2Fwww.google.no%2F. Online; accessed 11.may-2018.

[24] MATLAB [n.d.*c*], 'imcomplement', https://se.mathworks.com/help/images/ref/imcomplement.html. Online; accessed 21.may-2018.

[25] Medicinenet [n.d.*a*], 'Medical definition of methemoglobin', https://www.medicinenet.com/script/main/art.asp?articlekey=4366. Online; accessed 31.may-2018.

[26] Medicinenet [n.d.*b*], 'Medical definition of oxyhemoglobin', https://www.medicinenet.com/script/main/art.asp?articlekey=15772. Online; accessed 31.may-2018.

[27] Mehta, A. [2011], 'Introduction to the Electromagnetic Spectrum and Spectroscopy', https://pharmaxchange.info/2011/08/introduction-to-the-electromagnetic-spectrum-and-spectroscopy/. Online; accessed 08-February-2018.

[28] Negnevitsky, M. [2011], *Artificial Intelligence - A guide to intelligent systems*, "third" edn, Pearson.

[29] Olsen, S. H. [2011], 'Quantification and characterization of residual blood in fish muscle', *Reliability Engineering and System Safety* .

[30] OpenCV [n.d.*a*], 'floodfill', https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html. Online; accessed 21.may-2018.

[31] OpenCV [n.d.*b*], 'Structural analysis and shape descriptors', https://docs.opencv.org/3.1.0/d3/dc0/group__imgproc__shape.html#ga17ed9f5d79ae97bd4c7cf18403e1689a. Online; accessed 21.may-2018.

[32] Optics-online [n.d.], 'IR cut-off filter', http://www.optics-online.com/irc.asp. Online; accessed 08-February-2018.

[33] Otsu, N. [1979], 'A Threshold Selection Method from Gray-Level Histograms', http://web-ext.u-aizu.ac.jp/course/bmclass/documents/otsu1979.pdf. Online; accessed 19-March-2018.

[34] *Serial* [n.d.], https://www.arduino.cc/reference/en/language/functions/communication/serial/. Online; accessed 19-March-2018.

[35] *sharelatex* [n.d.], https://www.sharelatex.com. Online; accessed 23-May-2018.

[36] *Single Sided Matrix Board* [n.d.], https://no.rs-online.com/web/p/matrix-boards/1595537/. Online; accessed 22.may-2018.

[37] Skoog, Douglas A., W. D. M. H. F. J. and Crouch, S. R. [1999], *Analytical Chemistry: An Introduction*, 7th edn, Emily Barrosse, Hampshire, United Kingdom.

[38] *Structuring Elements* [n.d.], https://se.mathworks.com/help/images/structuring-elements.html. Online; accessed 19-March-2018.

[39] Suzuki, S. and be, K. [1985], 'Topological structural analysis of digitized binary images by border following', *Computer Vision, Graphics, and Image Processing* **30**(1), 32 – 46.
URL: *http://www.sciencedirect.com/science/article/pii/0734189X85900167*

[40] Tutorialspoint [n.d.], 'Opencv - adaptive threshold', https://www.tutorialspoint.com/opencv/opencv_adaptive_threshold.htm. Online; accessed 21.may-2018.

[41] Umbraugh, S. E. [1998], *Digital Image Processing And Analysis*, 2nd edn, Wiley.

[42] *USB-adapter i XLR-hus* [n.d.], https://www.elfadistrelec.no/no/usb-adapter-xlr-hus-usb-usb-cliff-cp30207nm/p/30030433?q=cliff+usb&page=22&origPos=22&origPageSize=50&simi=99.2. Online; accessed 22.may-2018.

[43] Weissleder, R. [2001], 'A clearer vision sor in vivo imaging', https://www.nature.com/articles/nbt0401_316. Online; accessed 08-February-2018.

[44] *What is Arduino?* [n.d.], https://www.arduino.cc/en/Guide/Introduction. Online; accessed 19-March-2018.

[45] *What is MATLAB?* [n.d.], https://www.mathworks.com/discovery/what-is-matlab.html. Online; accessed 08-February-2018.

[46] *What is unique about NIR(Near-Infrared) cameras?* [n.d.], http://multipix.com/whitepaper/what-is-unique-about-nir-near-infrared-cameras/. Online; accessed 09-February-2018.

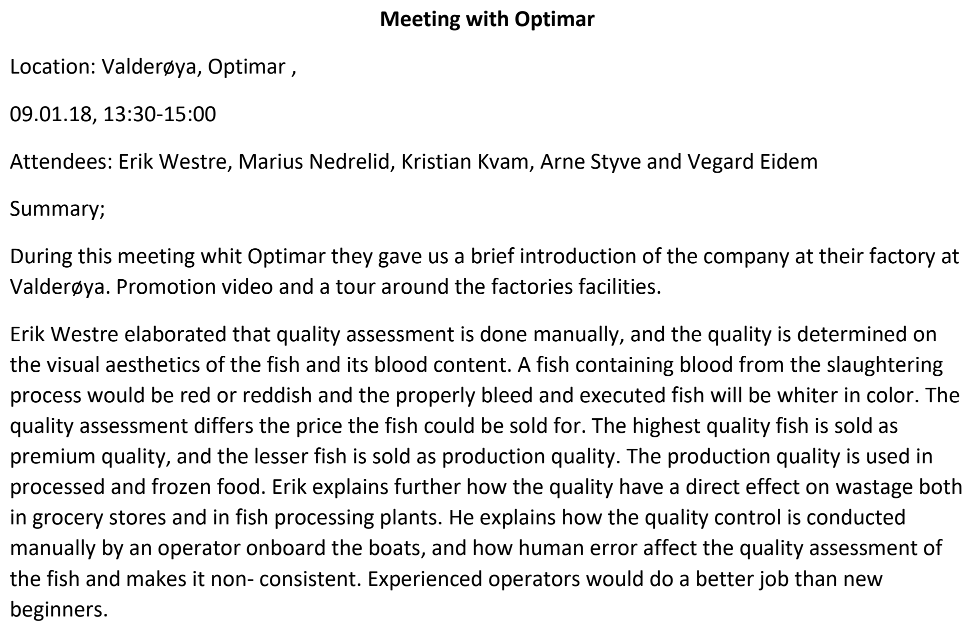
[47] *whitefish* [n.d.], https://www.worldwildlife.org/industries/whitefish. Online; accessed 27-May-2018.

[48] *Yet another record year for whitefish export* [n.d.], https://en.seafood.no/news-and-media/news-archive/yet-another-record-year-for-whitefish-exports/. Online; accessed 27-May-2018.

Appendix A

Appendix

A.1 Minutes



Meeting with Optimar

Location: Valderøya, Optimar ,

09.01.18, 13:30-15:00

Attendees: Erik Westre, Marius Nedrelid, Kristian Kvam, Arne Styve and Vegard Eidem

Summary;

During this meeting whit Optimar they gave us a brief introduction of the company at their factory at Valderøya. Promotion video and a tour around the factories facilities.

Erik Westre elaborated that quality assessment is done manually, and the quality is determined on the visual aesthetics of the fish and its blood content. A fish containing blood from the slaughtering process would be red or reddish and the properly bleed and executed fish will be whiter in color. The quality assessment differs the price the fish could be sold for. The highest quality fish is sold as premium quality, and the lesser fish is sold as production quality. The production quality is used in processed and frozen food. Erik explains further how the quality have a direct effect on wastage both in grocery stores and in fish processing plants. He explains how the quality control is conducted manually by an operator onboard the boats, and how human error affect the quality assessment of the fish and makes it non- consistent. Experienced operators would do a better job than new beginners.

Møte med Lars Christian Gansel.

Location: NTNU, Ålesund

20.02.18, 12:00-

Attendees: Lars Christian Gansel, Kristian Kvam and Vegard Eidem

Arne Styve hooked us up with a meeting with Lars Christian Gansel, in hope that he could help us out with some of the questions we had regarding to hemoglobin in whitefish.

He gave us some pointers on people we could contact that had more experience on the biology behind this topic:

He gave us a tip that we should contact Yarnran Cao, she is a doctor that have specialized herself in fish biology. He further gave us a tip to contact Ekrem Mismi at SINTEF Ocean. Ekram has previous experience in image analyzing and research on salmon.

The last tip he gave us was to conduct the tests on fresh fish, newly bleed out fish and stressing them up beforehand. He thought that these approaches might give us better results than the one conducted at Optimar's facility

Optimar

Location: Optimar, Valderøya

27.04-2018, 12:45-15:00

Attendes: Erik Westre, Marius Nedrelid, Kristian Kvam, Vegard Eidem, Arne Styve

In this meeting with Optimar we showed them our progress we had made throughout the project. We showed them our application and how it worked.

After we showed them the java application we got some input regarding the k-means clustering used. How do we handle the data logged by the application, that are meant for retraining the cluster? How do we handle abnormalities in the training data and bad data?

We asked about input on how a wrong cut fillet looked like. And Marius gave us some input on how to handle it. He mentioned RST invariance, and use that to find the invariant between multiple fishes.

Other tips from Marius Nedrelid:

- Fast image processing with the right equipment.
- Simulate the test whit fish fillets
- Speed of conveyor belt and 3x images. Installation need have a small footprint.
- External communication to the system. IP (Ethernet) TCP/IP
- Avioid sensors, error and keep process time to a minimum.
- Distance between camera and conveyor. Size of cell?
- Movement on the fillet, stitch together images.
- The things not tested, needs assumptions in the report.

A.2 Source code

The source code for the java application and the Arduino can be found in two separate gitgub repositories.

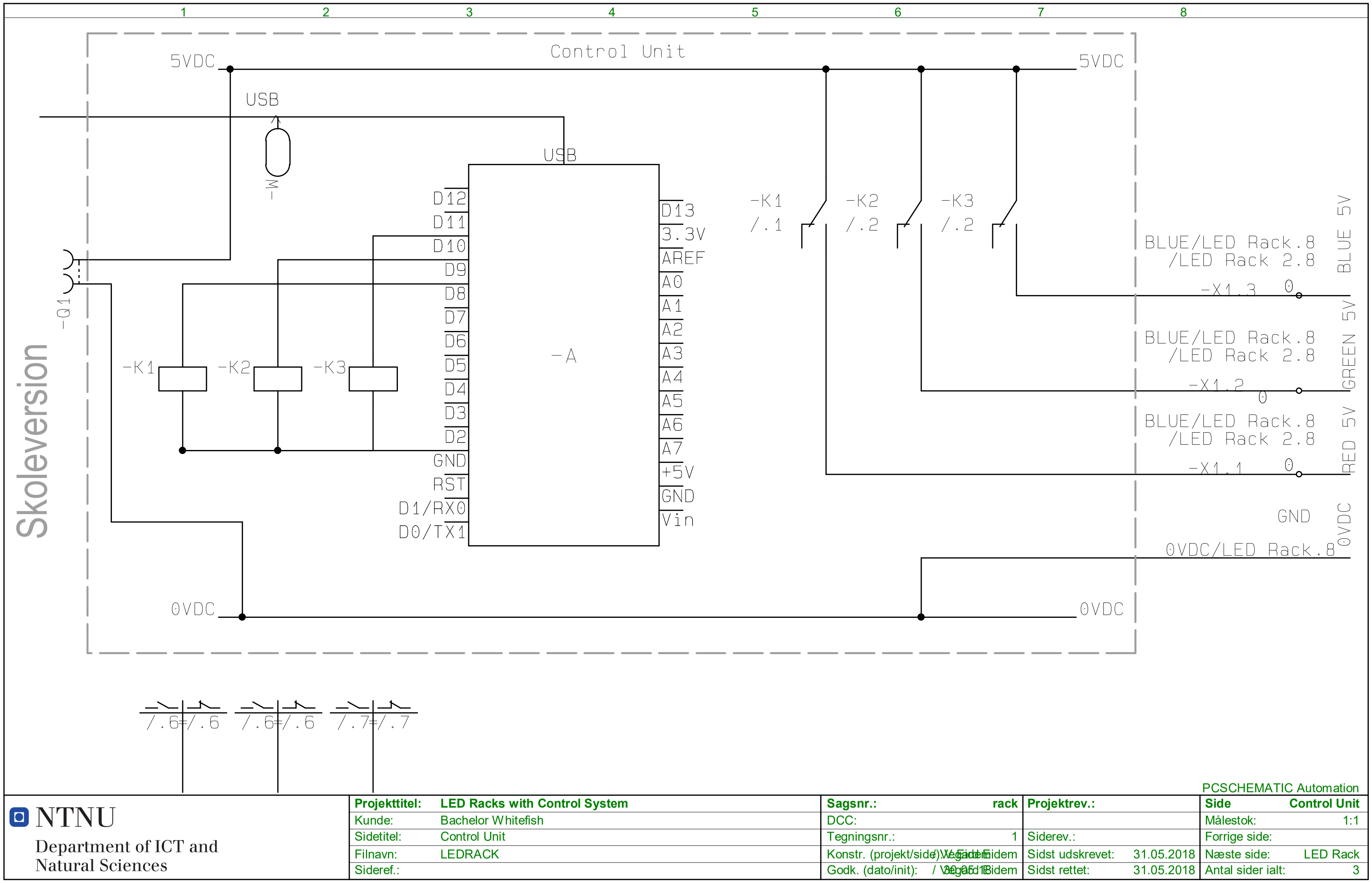
Link to java source code:

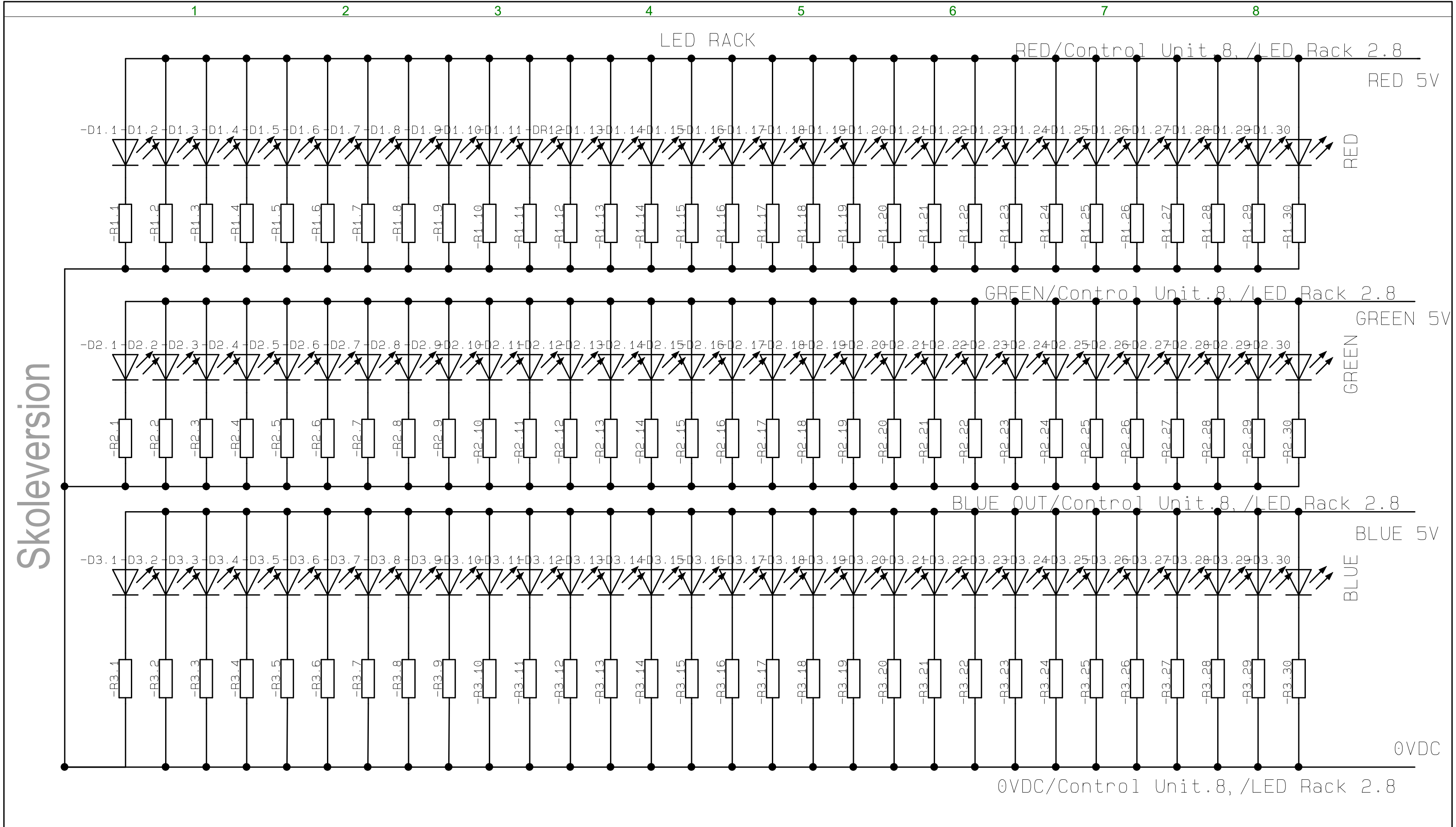
https://github.com/KriKvam/Quality-detection-of-white-fish-fillets

Link to Arduino source code:

https://github.com/KriKvam/Quality-detection-Arduino-client

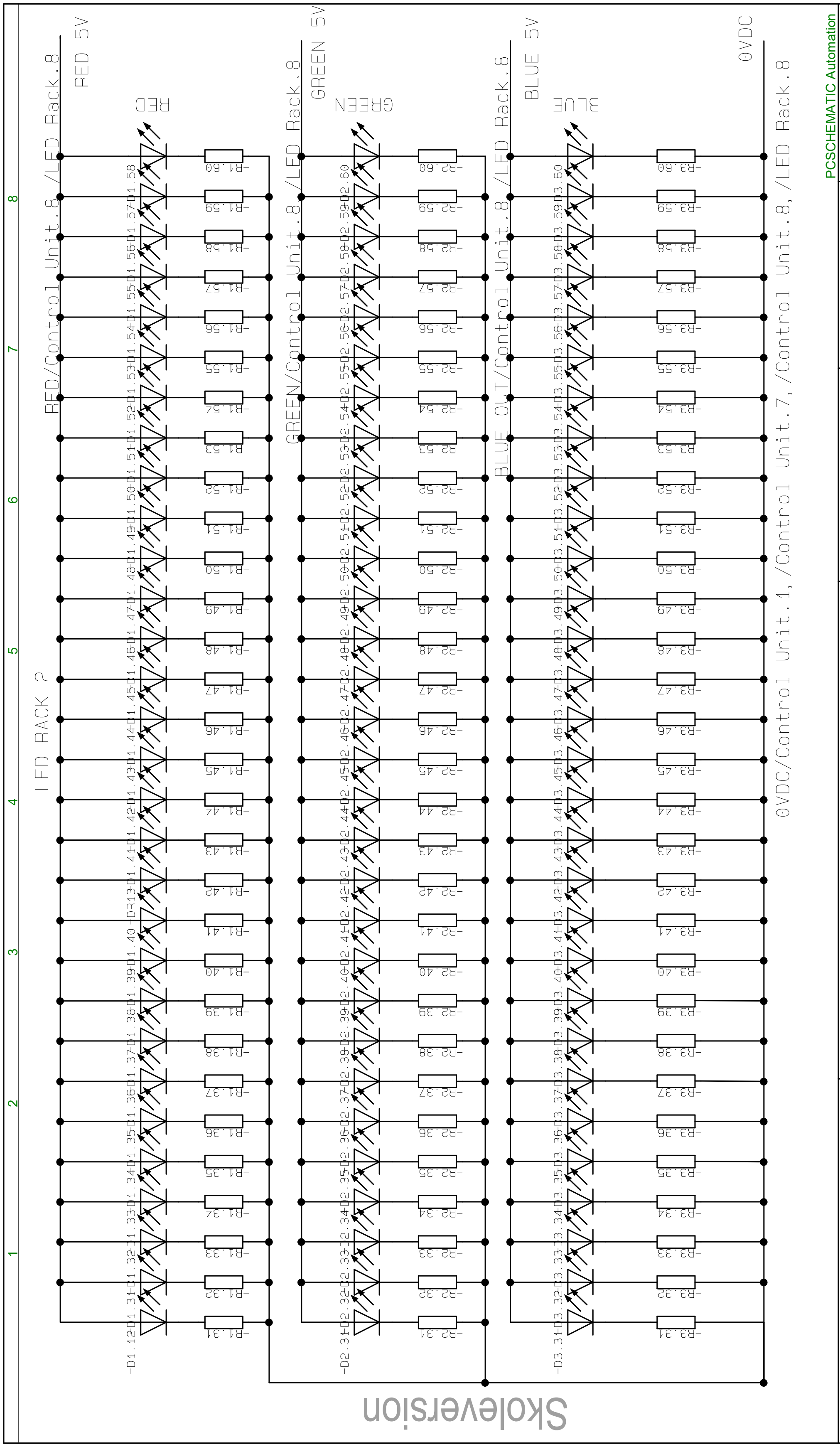
A.3 Schematics





PCSCHEMATIC Automation

NTNU Department of ICT and Natural Sciences	**Projekttitel:**	**LED Racks with Control System**	**Sagsnr.:**	**rack**	**Projektrev.:**		**Side**	**LED Rack**
	Kunde:	Bachelor Whitefish	DCC:				Målestok:	1:1
	Sidetitel:	LED RACK 1	Tegningsnr.:	1	Siderev.:		Forrige side:	Control Unit
	Filnavn:	LEDRACK	Konstr. (projekt/side):	V.Eidem/V.Eidem	Sidst udskrevet:	31.05.2018	Næste side:	LED Rack 2
	Sideref.:		Godk. (dato/init):	31.05.18/V.Eidem	Sidst rettet:	31.05.2018	Antal sider ialt:	3


RED 5V
GREEN 5V
BLUE 5V
0VDC
RED
GREEN
BLUE
LED RACK 2
RED/Control Unit.8 /LED Rack.8
GREEN/Control Unit.8 /LED Rack.8
BLUE OUT/Control Unit.8 /LED Rack.8
0VDC/Control Unit.1, /Control Unit.7, /Control Unit.8, /LED Rack.8
1
2
3
4
5
6
7
8

NTNU
Department of ICT and Natural Sciences

PCSCHEMATIC Automation

Projekttitel:	**LED Racks with Control System**	**Sagsnr.:**	**rack**	**Projektrev.:**		**Side**	**LED Rack 2**
Kunde:	Bachelor Whitefish	DCC:				Målestok:	1:1
Sidetitel:	Diagram	Tegningsnr.:		Siderev.:		Forrige side:	LED Rack
Filnavn:	LEDRACK	Konstr. (projekt/side):	V.E/Eidem	Sidst udskrevet:	31.05.2018	Næste side:	
Sideref.:		Godk. (dato/init):	31.05.2018 Eidem	Sidst rettet:	31.05.2018	Antal sider ialt:	3

A.4 Pre project report

FORPROSJEKT - RAPPORT

FOR BACHELOROPPGAVE

TITLE: **Quality control white fish fillet**

CANDIDATE(S): **Vegard Eidem –** **Kristian Kvam -**			
DATE: **19/01/18**	COURSE CODE: **IE303612**	COURSE: **Bacheloroppgave**	RESTRICTION: - Open
STUDY PROGRAM: BACHELOR IN AUTOMATION TECHNOLOGY		PAGES/APPENDIX: /	LIBRARY NO.: - Not used-

EMPLOYER(S)/SUPERVISOR(S): Optimar/Arne Styve

SUMMARY:

In this project, we are going to prototype a software that sorts white fish fillets by their quality. The task is given by the corporation Optimar and our task is to find a good solution to check the quality of fish fillets on a conveyor using image processing. The main task is to evaluate the amount of blood in each fillet using the light absorption qualities of hemoglobin. We will be doing a lot of research and testing before we start creating our software. Throughout this report you can read how we're planning to go forward to achieve our goals and how we will be documenting our work.

Denne oppgaven er en eksamensbesvarelse utført av student(er) ved NTNU i Ålesund.

CONTENTS

1 INTRODUCTION

For our Bachelor assignment, we have chosen the assignment “white fish fillet” because of the interest we gained in image processing during a previous course at the university. We were both fond of the idea of using image processing to control an industrial system, and were therefor thrilled to see this as a listed assignment.
This assignment is given by the corporation Optimar which specializes in designing and building complete automated systems for fish handling. We are in this assignment to evaluate certain qualities of a white fish fillet using only image processing. In more detail, we are to evaluate the amount of blood in the fillet, the volume of the fillet and check if it has been properly cut. These factors will then be used to sort the fillets based on their quality. The system also needs to be able to evaluate several fillets a second to keep up with the rest of the production.
The whole purpose of this assignment is to use image processing to collect and process the data which is needed to sort the fillets based on their quality. By only using image processing one also eliminates the need for single purpose components such as weights.

2 TERMS

- Hemoglobin – An iron-containing oxygen-transport metalloprotein in the red blood cells of all vertebrates
- (Digital) Image Processing – Is the use of computer algorithms to perform image processing on digital images.
- White fish – A fisheries term for fish with white flesh.
- JIRA – Is a proprietary issue tracking software, which provides bug tracking, issue tracking and project management function.
- Confluence – An enterprise knowledge management system.
- Risk Assessment Matrix – A risk matrix is a matrix used during risk assessment to define the level of risk by considering the grade of likelihood against the grade of consequence severity.
- MATLAB – A multi-paradigm numerical computing environment that allows matrix manipulations, plotting of functions and data and implementation of algorithms
- OpenCV – Library of programming functions mainly aimed at real-time computer vision.
- Java – Computer programming language that is concurrent, class-based and object-oriented.

3 PROJECT ORGANIZATION

3.1 Project group

Studentnummer(e)
460002 (Vegard Eidem) 460005 (Kristian Kvam)

Tasks for the project group – organization

Project leader – Vegard Eidem

Secretary – Kristian Kvam

Tasks for project leader

- Head of Research
- Set up meetings
- Keep track of progress
- Organize project

Tasks for secretary

- Head of programming
- Design the overall system in java
- Find solutions in OpenCV to match the research methods
- Help project manager when needed

3.2 Steering group (supervisor and contact person/employer)

Supervisors:

- Arne Styve
- Hans Støle

Contact for employer:

- Erik Westre

4 AGREEMENTS

4.1 Agreements with employer

There has not been any need for a contract for this project, but there might be need for a NDA later in the project.

4.2 Workplace and resources

The workplace is set at the University's project Lab and will be used regularly until we need to do some fieldwork at Optimar.

Optimar will cover the expenses on hardware and software needed to complete the assignment.

The human resources we have access to is Erik Westre and Marius Nedrelid. Erik Westre is the company’s Technical Manager and Marius is a System and Control Engineer. We also have access to our supervisors: Arne Styve and Hans Støle. Arne Styve is the main supervisor and Hans Støle is the project’s Image analysis/processing asset.

4.3 Group norms – cooperation rules – attitudes

The workday should normally start at 08:00 and end at 16:00, unless there have been other agreements made in the group. If the group is delayed, overtime must be initiated at an early stage to avoid further delay.

There should be two main breaks during the day, the first at 10:00-10:15 and the second from 13:00-13:30.

Every morning the group should start with a 5-10min recap and strategic meeting before starting. This to reflect over where the group left of the day before and to gain control over the present days 'to-do-list'.

The individuals in the group should treat each other with respect and equality, but give constructive critic where constructive critic is fit. The team members shall respect each other's ideas, and reflect over the possibilities rather than reject them.

5 PROJECT DESCRIPTION

5.1 Problem - objektive – purpose

One of the biggest disadvantages of manual quality control of white fish, is that the person performing the task doesn't always make the right decisions, but might base the current evaluation based on last evaluation or other emotions. The issue at hand is to make the control of the white fish quality easier and reduce cost, this by implementing a camera solution and a light source. The main goal is to find good methods and hardware that can do this in an efficient and consistent way. One purpose of this is to make the quality control automated and consistent, this means lower expenses for the costumer and gives higher revenue.

5.2 Requirements for solution or project result - Specification

Since this project is more a research project, there have not been given any specific demands other than good documentation.
Since we are to design a system, there have been given some specifications which needs to be for filled for the system to be usable. The system should be able to detect all fish fillets on a conveyor and sort them based on how much blood the fillet contains. The processing time of the system should also be relatively low to be able to process several images a second to keep up with the rest of the system.

Parts needed for the project will be ordered by Optimar themselves. We are responsible to find parts suitable for the project and parts should not exceed unreasonable prices.

The project will be considered finished when all specifications are for filled and the system provides results good enough for production. The system must also be well documented.

5.3 Scheduled procedure (s) for development work – method (s)

The assignment is almost a pure software development task, so use of regular Gant diagram will not be used, we will use JIRA [3] that is known for its good compatibility with software development as it has its own version control. The project management will be done every other week, and we will assess what we can do next two weeks period on every meeting with our supervisor. On these meetings, we will make an Sprint in JIRA, and this will become the next 2 weeks to-do list. The strength of a Sprint

in JIRA is that we don't have to delay the whole project because of a small delay in one of the points in the Sprint. We can simply move it over to the next two-week sprint.
We will also use Confluence as a wiki and a place to store documents and research documents, we will also publish our meeting minutes in Confluence. In confluence you can also comment and assign different side tasks to members in the group, for example if something from the meeting minute needs to be taken care of, one can assign it to a member.

5.4 Information gathering – finished and planned

During the first phase of the project there will be an own segment dealing with research. Here should the necessary literature and information be collected from other similar trials or matters. When planning an assignment, it's crucial that you collect some basic knowledge before even starting on the research process.

Optimar provided a slide which contains information about how the different states of hemoglobin absorbs different wavelengths of light. This is valuable information for further research on software and camera setups. Information on how the blood in the fish fillet absorbs light will also give us a pointer on how we can build our test algorithms in Matlab.

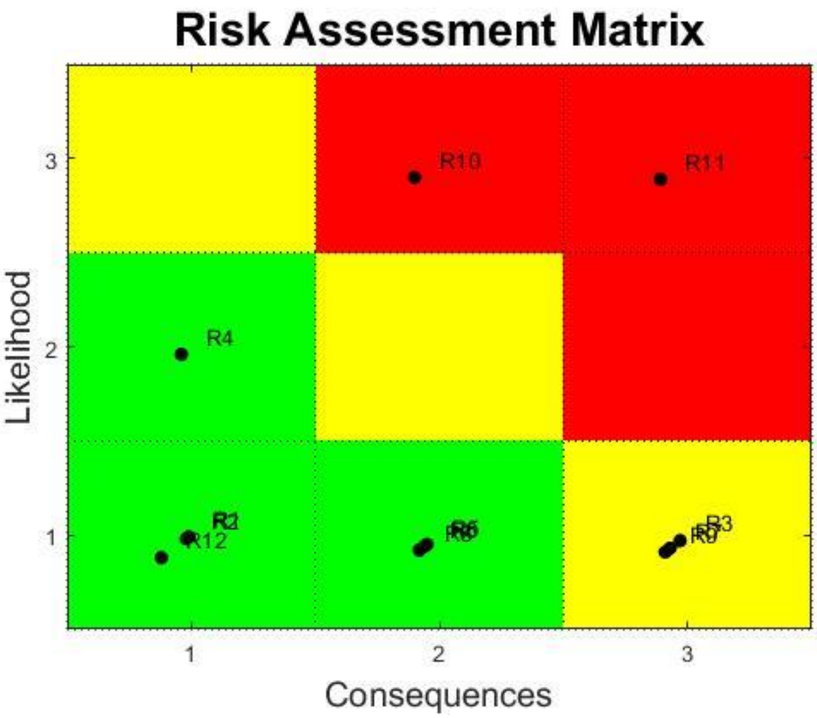
When the thesis is handed in and the project starts one can proceed to collect information on different adjustable light sources and suitable cameras. It will also be necessary to collect information on different image processing methods that can help us extract information from the image material provided by Optimar.

5.5 Assessment – analysis of risk

In the Risk Assessment, we set up a 3x3 Risk Assessment Matrix with Likelihood 1-3 (low, medium and High) and same for the consequences. By doing a risk assessment we can analyze the probability of an event to occur and the consequences of that event. By doing a Pre-project risk assessment, we should be able to foresee how these events could harm us timewise. The following risks that are considered is:

Risk	Consequences	Likelihood	Description
R1	1	1	Member oversleeps
R2	1	1	Misses the bus
R3	3	1	Terminal Illness
R4	1	2	Cold
R5	2	1	Flu
R6	2	1	Minor delay
R7	3	1	Severe delay
R8	2	1	Computer crash
R9	3	1	Quarrels
R10	2	3	Small "undetectable" bugs
R11	3	3	Large tedious bugs
R12	1	1	Minor bugs

We made a script in MATLAB [1] that scatter plotted the different risks into a 3x3 risk assessment matrix, the image is exported and displayed here.



5.6 Main activities in further work

Nr	Activity	Person in charge	Cost	Time /Scope
A1	Do research on how the project can be done.	VE	0,-	
A11	Find papers on white fish and image processing techniques for fish.	VE	0,-	
A12	Find out if similar projects have been done before.	VE	0,-	
A121	Gather and analyze data from similar projects.	VE	0,-	
A13	Use the information we have gathered to design a test concept.	VE	0,-	
B1	Create image material using techniques from the research.	VE/KK	0,-	
C1	Test the concept and various image processing methods.	VE/KK	0,-	
C11	Analyze the data and find solutions to reduce eventual errors.	VE/KK	0,-	
C111	Finish a test concept that gives reliable data and minimal errors.	VE	0,-	
D1	Design the structure and processes of the java runnable.	KK	0,-	
D11	Find OpenCV methods equal to the test methods used in Matlab.	KK	0,-	
D12	Build the java classes and design the GUI.	KK	0,-	
D121	Build the program and do tests to get a working prototype.	KK	0,-	
D122	Optimize the program to improve performance.	KK	0,-	

5.7 Progress plan – management of the project

Main plan

The main focus in the start of the project will be in research and information gathering. During this phase (phase A) we will be gathering information and data that can give us ideas of how we need to go forward to get the results that we want. When the necessary information has been acknowledged we will move towards the second phase (phase B) where we will be creating image material. During this phase, we will be using methods learned in phase A to capture images that can later be used to extract the necessary data. The third phase (phase C) can quickly become the most time consuming one. This phase is where we develop a reliable way to extract and analyze the data. This will require a lot of testing and tuning to find the most optimal solution. The last phase (phase D) is building the main software. This also requires finding image processing methods similar to the ones used during testing. Most of the time during this phase will be put into designing and programming the structure of the software.

Management Utilities

For controlling and keeping track of the project, we will be using Jira [3]. Jira is a software used in project managing and includes many features like bug tracing. It also lets us assign tasks and keep track of the progress by creating deadlines and milestones. We will also individually be logging our daily progress so that other members of the group can give helpful feedback if needed.

Development aids

During the research process, one would like to test different image processing methods and analyse the data. Matlab [1] is great for research purposes because you can easily plot histograms and other helpful diagrams. Matlab will be used to test out image analysis algorithms before implementing them into a more suitable programming language.

Internal control - evaluation

Internal evaluation will take place at a meeting hosted every 2 weeks where both group members and supervisors are present. We will be using the software Confluence [5] to keep track of our progress and assigning new tasks. Through this software, every member/supervisor will have full access to all files and schedules and a can easily bring up things for discussion during the meeting.
When evaluating if a task is done, certain criteria’s needs to be for filled. The task must have been properly executed so that it won't causes problems further down the road. The task must also provide good and reliable results so that the next task can get started right away without hesitations.

5.8 Decisions – decision-making process

During the project, there may be need for restrictions. Restrictions will be made if members of the group find the time consumption or complexity of a task inflecting the rest of the project. The restriction made should not affect the quality of the project in a bad manner.

Making the final decision will be done by the person in charge of the task. If there is to occur a disagreement, an input from the supervisor will be used to resolve the problem.

6 DOCUMENTATION

6.1 Reports and technical documents

- Documentation is an important part of our project since it’s heavily based on research. During the whole project, we will be documenting our progress and our discoveries to keep everything in detail. This documentation amongst other will later be used to write a project report at the end of the project.

- We have implemented a daily routine where we log everything we do at the end of the day. These daily logs will be used to write a summary every two week which again will be used in the project report.
- All our code will also be well documented. When writing our code, we will be keeping the code clean by using relevant names on variables and such. This will make it easier for others to do eventual changes to the code in the future. We will also be generating Javadoc for the java code and documentation for the MATLAB research code.
- All papers and documents will be published by NTNU at the end of the project where they will be available for the public.

7 PLANNED MEETINGS AND REPORTS

7.1 Meetings

Meetings with the steering group

- There will be a meeting every two week where both the project group and supervisors are present. Time and date will be decided short time in forehand for each meeting to not interfere with other obligations. During these meetings, we will be discussing our recent progress, maintaining our schedule and if needed give input on how something can be done differently.

Project meetings

- Project meetings will take place every day and last for about five minutes. The meeting starts at beginning of the day, and one can extend the time of the meeting if needed. During these meetings, one will update each other on recent progress and discuss todays goals.

7.2 Periodic reports

Progress reports (including milestones)

- We will be writing a daily log to keep everything in detail, and then write a summary every two week which will be gone through during the border meeting. When there has been any progress on a certain milestone, the progress will be updated through the dedicated software.

8 PLANNED DEVIATION TREATMENT

For the planning of the project the JIRA [1] platform is used, and target process is planned every two weeks with supervisor, Arne Styve. The goal is to plan a suitable amount of work to do every other week. If progress doesn't go as planned, we shuffle the remaining tasks over to the next two weeks. If the project is delayed there's initiated overtime to get back on track.

Every change and progress are documented in JIRA, this platform provides a to-do-list instead of a Gant diagram. JIRA is much more suitable in Software development then a Gant diagram, and provides an overlook of the tasks and when they are due.

Every morning the group have a short debrief and recap meeting, this is necessary to track progress and go over the days to-do-list. It's the projects leaders' responsibility to overlook these things and prevent the group from being delayed.

9 EQUIPMENT/REQUIREMENTS FOR IMPLEMENTATION

In this project, one will need MATLAB [1] for research purposes, NetBeans IDE [2] for Java programming and the OpenCV [4] library for image processing in Java. To emit the necessary light

waves that are absorbed by hemoglobin's, an adjustable light source is needed and a proper camera to capture this.
MATLAB will we used as a research tool and testing different image processing and analyzing methods. MATLAB is relatively easy to use and has many good methods for visualizing, plotting graphs and other diagrams.
When it comes to programming software is all about preference, we decided to use NetBeans IDE because that's what we are most familiar with. We will use the OpenCV library for image processing in Java.
We also know that we need a light source with adjustable wavelength to detect hemoglobin in it's different states. We also need a proper camera that can detect this and with a reasonable resolution.

10 REFERENCES

[1] https://www.mathworks.com/products/matlab.html
[2] https://netbeans.org/
[3] https://jira.atlassian.com/
[4] https://opencv.org/
[5] https://www.atlassian.com/software/confluence

ATTACHMENTS

Attachment 1 - MATLAB Script: RiskAssessmentMatrix .m

Attachment 2 – Excel-file: risktest.xlsx (For RiskAssessmentMatrix.m)