NTNU

Norwegian University of
Science and Technology

Automatic Inspection of Cage Integrity with Underwater Vehicle

Rolf Arne Hult Jakobsen

Master of Science in Engineering Cybernetics
Submission date: July 2011
Supervisor: Jo Arve Alfredsen, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

NTNU

Norges teknisk-naturvitenskapelige universitet

Fakultet for informasjonsteknologi, matematikk og elektroteknikk

Institutt for teknisk kybernetikk

MASTEROPPGAVE

Kandidatens navn:	Rolf Arne Jakobsen
Fag:	Teknisk kybernetikk
Oppgavens tittel:	Automatisert overvåkning av not-integritet med undervannsfarkost
Engelsk tittel:	Automatic inspection of cage integrity with underwater vehicle

<u>Oppgavens tekst</u>

I dagens oppdrettsanlegg er fisken omgitt av en not som den eneste barrieren mot det åpne havet omkring. Erfaring viser at det er fare for at noten påføres skader i form av hull og flenger i løpet av en produksjonssyklus. Skader på noten utgjør en latent risiko for rømming og er assosiert med alvorlige økonomiske og miljømessige konsekvenser. I dag brukes dykkere for notinspeksjon, men denne operasjonen kan være både risikofylt, tidkrevende og kostbar. Motivasjonen bak denne oppgaven er et ønske om å robotisere denne operasjonen. Ideen er at en undervannsfarkost (AUV) programmeres til automatisk å utføre periodiske inspeksjoner av notens tilstand, der inspeksjonen i første rekke er basert på bilder av notsegmenter fra et kamera som bæres av farkosten. Bildene prosesseres og analyseres fortløpende av en datamaskin om bord i farkosten for deteksjon av eventuelle notskader.
I denne oppgaven skal det utvikles en konseptprototyp der en mikro-ROV av typen VideoRay brukes til å emulere en AUV-plattform som utfører vertikal traversering og inspeksjon av notvegger. I denne forbindelse skal det vurderes i hvilken grad informasjonen i videobildene, i tillegg til skadedeteksjon, kan brukes til å assistere posisjonsbestemmelse og styring av farkosten. Det skal også vurderes om belysning av notsegmentene, f.eks. med strukturert laserlys, kan forbedre ytelsen til systemet. Oppgaven omfatter følgende punkter:

- Gjennomgang av problemstillingens bakgrunn, bruksscenarier og utarbeidelse av kravspesifikasjon for konseptprototyp
- Utvikling av programvare for sanntids bildeanalyse av notsegmenter (skadedeteksjon), samt et styresystem for automatisk vertikal traversering av notvegger med utgangspunkt i VideoRay-ROV. Vurdering av behovet for utvikling av egen enhet for strukturert belysning av noten.
- Planlegging og gjennomføring av systematiske tester av konseptprototyp, både i ideelle og realistiske omgivelser
- Diskusjon av resultater og kartlegging av systemets muligheter og begrensninger

Oppgaven gitt:	3. februar 2011
Besvarelsen leveres innen:	30. juni 2011
Utført ved:	Institutt for teknisk kybernetikk, NTNU
Veileder:	Jo Arve Alfredsen, ITK, NTNU

Trondheim, 26. januar 2011

Jo Arve Alfredsen

Faglærer

Abstract

This thesis is a preliminary study for the initiative to develop an autonomous AUV for use in aquaculture. A micro-ROV is used to test the concept of inspecting net integrity in fish cages by analyzing video feed from a video camera. Software is developed, including an interactive GUI, thruster control, and the analysis algorithms. A laser module is also developed to measure the distance between the ROV and fish cage using two laser line generators. The conclusion is that checking for net integrity with a camera and computer vision software is a viable option in a future AUV. The laser module works well, providing reliable distance measurements for the ROV.

Preface

This thesis concludes my master studies at the Deptartment of Engineering Cybernetics at the Norwegian University of Science and Technology. The project has given me a unique opportunity to combine my theoretical knowledge with practical work and has been an invaluable experience.

I wish to thank my main supervisor, Associate Professor Jo Arve Alfredsen, for his countless advices and help with all arrangements. I would also like to thank my co-supervisor, Per Rundtop at SINTEF Fisheries and Aquaculture, for providing equipment for testing the system. Gratitude is given to Per Inge Snildal at the department's mechanical workshop for his outstanding work with the laser module, and to John Olav Horrigmo for his help and advice during development of the laser module PCB. I also appreciate the assistance from Guttorm Lange at ACE in arranging tests of the system at a real fish farm. Special thanks are given to earlier student Andreas Carlsen for laying a foundation for this project through his own MSc. thesis. I have been able to re-use many of his findings, saving me a lot of time and frustration. He has also helped with troubleshooting through email-correspondence at times when it was needed the most.

NTNU, Trondheim, July 2011

Rolf Arne Jakobsen

Contents

Nomenclature

CAN	Controller Area Network
CV	Computer Vision
EMI	Electro Magnetic Interference
ESD	Electrostatic Discharge
GUI	Graphical user interface
HSV	Hue Saturation Value: Cylindrical-coordinate representation of the RGB color model
ICB	Integrated Control Box
MCU	Microcontroller Unit: small computer on a single integrated circuit
PAL	Phase Alternating Line: analog color television encoding system
PPHT	Progressive Probabilistic Hough Transform
PRO3S	Micro-ROV developed and distributed by VideoRay LLC
RCA	Phono plug (Derived from Radio Corporation of America)
RGB	Red Green Blue: Additive color model
ROV	Remotely Operated Vehicle
RS-232	Recommended Standard 232, serial binary interface
TDS	Tether Deployment System
TOF	Time of Flight: Principle used in many laser rangefinders
UART	universal asynchronous receiver/transmitter
USB	Universal Serial Bus: specification to establish communication between devices and a host controller

Chapter 1

Introduction

1.1 Motivation

As the world human population continues to grow, the demand for fish has seen a rapid increase in recent years. Unfortunately, this demand has long since passed the limit of sustainable use. In today's fishing industry, overfishing is a major problem which leads to a critical decrease in the fish populations. Some scientists even predict that the commercial fish and seafood harvest will collapse by the second half of this century, should current trends continue (Worm [2006]).

A way to increase fish production without harming the natural population is by the use of aquaculture in the form of fish farming. In much the same way as land animals have been domesticated and bred for thousands of years, fish can be raised in enclosures or tanks and then harvested for food. The worldwide growth rate of aquaculture has been rapid and sustained for a long period. Annual increases in production averaged 8.8% in the period 1970-2004 (FAO [2006]). At the same time the take from wild fisheries has flattened out. In 2005 the total world fish production was 141.3 million tons, of which aquaculture accounted for 48.1 million tons, or about one third. With the aquaculture industry more important than ever before, it is of economic interest to make production as streamlined and efficient as possible.

Norway is among the world's leading exporters of seafood. In recent years, much because of its favorable natural conditions with a long coast and an abundance of fjords, Norway has also attained a highly successful aquaculture industry. In 2009, production of the most bred species was 797,000 tons of Atlantic salmon and 92,000 tons of rainbow trout (Jensen et al. [2010]). Most fish farms are sheltered within bays, sounds, and fjords or scattered amongst islands within archipelagos. The fish are usually kept in square, rectangular or circular cages, ranging from 15 to 48 meters deep (Jensen et al. [2010]). Because most of the cage is underwater, divers are frequently used for routine operations such as inspection of net integrity, inspection of the mooring system, and detection of biofouling. Regular inspection of the cage net is especially important as net damage can lead to fish escaping the cage. Escapes are in fact one of the major problems in fish farming today. Consequences often include severe negative environmental effects. Moreover, the fish farms suffer economic losses from fish escapes, both directly from the loss of fish and from fines issued by the fisheries authorities, and indirectly due to bad publicity. In about two out of three escapes, damage to the net is the cause (Jensen et al. [2010]). To decrease the risk of escapes, the integrity of cage nets should be inspected as often as possible. However, inspection by divers is an expensive and potentially hazardous operation. Safety regulations require at least three divers with professional diving licenses to be present at all times during inspection (Arbeidstilsynet [2011]), and the availability of qualified divers may at times be restricted. The motivation behind this project is the demand for a more efficient and less hazardous solution to inspecting cage nets. If the inspection process could be automated, fish farmers would be able to check net integrity more frequently and at the same time cut costs and reduce the risk on human health.

An AUV (Autonomous Underwater Vehicle) is a free swimming, unmanned submarine. A specialized AUV could potentially do most of the routine operations performed by divers today. The task of inspecting net integrity could be done by equipping the AUV with a video camera and integrated image processing hardware. The AUV could then periodically survey the cage net and check for damage. The Department of Engineering Cybernetics at NTNU, together with SINTEF Fisheries and Aquaculture, Kongsberg Maritime Subsea, and other companies, has started an initiative with the goal of developing such an AUV. Since the development of a specialized AUV is a time-consuming and expensive project, it is desirable to test some of the key concepts in practice first. This MSc. Thesis aims at using a pre-existing ROV (Remotely Operated Vehicle) to check for net integrity in fish cages. An ROV differs from an AUV by being controlled from a control panel on the surface, connected by an umbilical cable. This will be a great advantage when testing the concept. All image analysis, as well as control of the ROV, can be performed on the surface instead of by dedicated hardware in the AUV case. A standard laptop can be used, so no development of new hardware is needed. If the concept proves to be successful, the image analysis algorithms can later be implemented on dedicated hardware in the AUV.

1.2 Previous Work

There have been a few earlier attempts at the use of an AUV in aquaculture. In Klepaker et al. [1987], a finished AUV prototype is presented. This prototype used an acoustic data telemetry system to communicate with a surface control unit. It also had an acoustic navigation system, giving its position with reference to navigation transducers located in the AUV's working area. An onboard camera was installed to provide video feed. An illustration of an example setup can be seen in Figure 1.1. The objective was

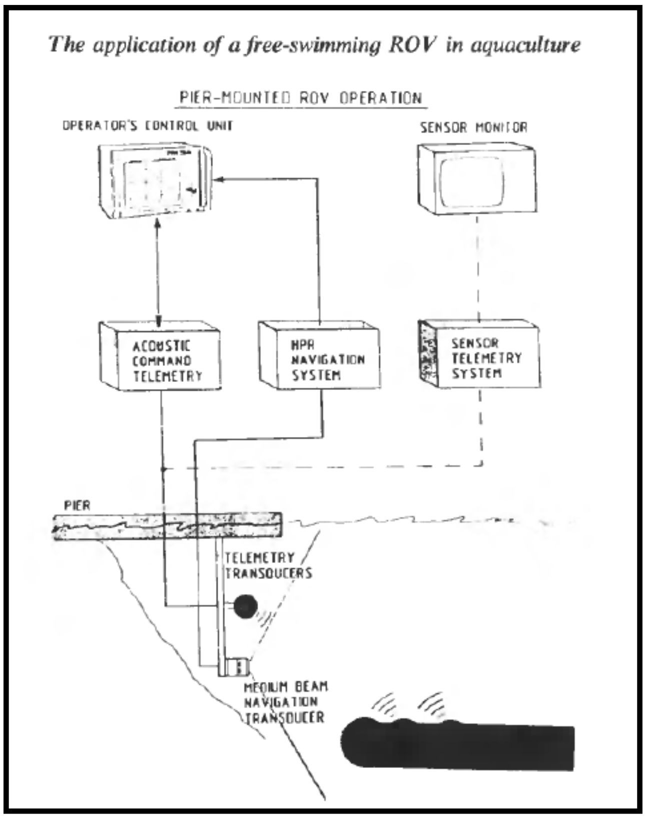

The application of a free-swimming ROV in aquaculture
PIER-MOUNTED ROV OPERATION
OPERATOR'S CONTROL UNIT
SENSOR MONITOR
ACOUSTIC COMMAND TELEMETRY
HPR NAVIGATION SYSTEM
SENSOR TELEMETRY SYSTEM
PIER
TELEMETRY TRANSDUCERS
MEDIUM BEAM NAVIGATION TRANSDUCER

Figure 1.1: Example setup of AUV in Klepaker et al. [1987]

to use such an AUV in a future fish farm where the fish was kept in a fjord, prevented from escaping by acoustic or electric barriers.

Possible roles of a future ROV or AUV in aquaculture are discussed in Balchen [1991]. He mentions inspection of the mariculture facility as the most obvious application. Also more innovative applications such as operating valves, installing equipment and performing repair tasks are mentioned. Balchen even suggests the AUV acting as a "shepherd" or as a "leader" of a school of fish. The disadvantages of using an ROV with a tether cable is also discussed. The tether cable hinders free motion and consumes a lot of energy because of the drag that water currents impose on the cable.

Frost et al. [1996] goes through the development of an ROV for the use in aquaculture. Practical considerations such as power supply and control made the authors chose a tether-cabled ROV instead of an AUV. The ROV is tested in a tank with fish. The fish does not seem to be disturbed by the ROV. They keep a distance of about 1 meter, and no contact between fish and the ROV is observed. The goal of the development is to over time have the ROV undertake operations currently being performed by divers.

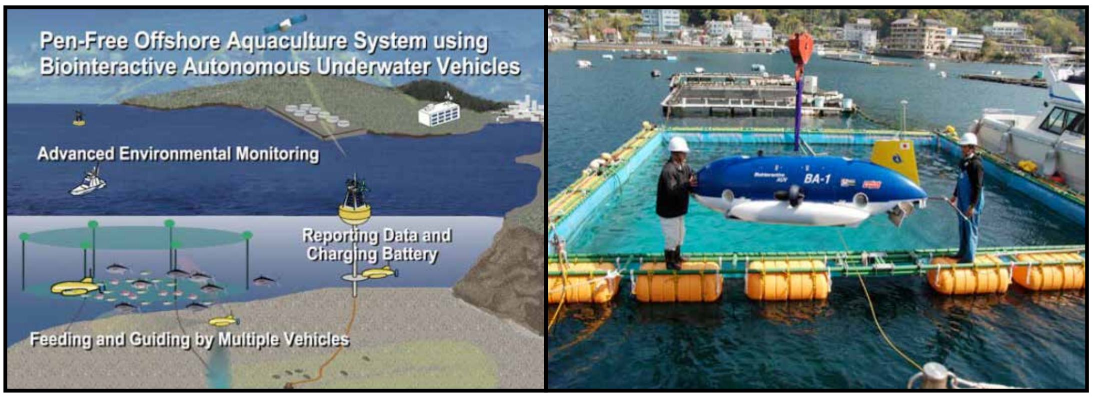

Pen-Free Offshore Aquaculture System using
Biointeractive Autonomous Underwater Vehicles
Advanced Environmental Monitoring
Reporting Data and Charging Battery
Feeding and Guiding by Multiple Vehicles
BA-1

Figure 1.2: Left: Concept of using several AUVs to eliminate the need for a fish cage. Right: The BA-1

The perhaps most ambitious aquaculture AUV project so far is described in Kondo et al. [2010]. An AUV was developed as a part of a larger initiative with the goal of using multiple AUVs to monitor the environment, feed the fish, monitor growth of fish, guide them, and report the data to land via a satelite connection. This is illustrated in the left part of Figure 1.2. The right part of the figure shows the prototype “BA-1”. The AUV is designed to behave as a shepherd by feeding the fish. It is assumed that this will make the fish follow the AUV. Initial tests show that this is indeed the case. After a few days of becoming used to the vehicle, the fish start following the AUV to receive bait. The AUV also carries other instruments to interact with the fish, such as LED arrays and a speaker system. However, these instruments does not seem to have any effect on the fish.

1.3 Outline

The thesis is divided into the following chapters:

VideoRay PRO 3S

This chapter gives information and relevant specifications about the ROV that will be used in the project.

Specifications and Requirements

Specifications are given for current hardware. Detailed requirements are presented for both the hardware and the software that will be developed.

Design

Hardware and software design is described in detail. The concepts used are thoroughly explained.

Computer Vision

This chapter explains the development and theory behind computer vision algorithms used in the project.

Implementation

The complete system implementation is described. Specifications of both hardware and software developed are given.

Test Setup and Results

All system tests are described. Results are presented and briefly commented on.

Discussion

The results of development and tests are discussed in context of both the current project and the initiative to develop an autonomous AUV for use in aquaculture.

Further Work

Ideas and suggestions for continued work towards development of the AUV is discussed.

Conclusion

Overall success of the project is presented and conclusions are made.

Chapter 2

VideoRay PRO 3S

The Department of Engineering Cybernetics at NTNU is in the possession of an ROV, more specifically a VideoRay PRO3S. This ROV will be used throughout the project.

The PRO3S is a Micro-ROV system developed and distributed by VideoRay LLC, a leading ROV manufacturer located in Phoenixville, PA, USA. The system includes the ROV, a Tether Deployment System (TDS), and an Integrated Control Box (ICB), as shown in Figure 2.2. Relevant specifications are listed in Table 2.1.

ROV Dimensions	30.5 x 22.5 x 21 cm
Submersible Weight	3.8 kg
Total System Weight	43 kg
Maximum Speed	1.34 m/s
Depth Rating	152 m
Camera	-Forward facing wide angle PAL camera. 570 lines of resolution and 0.3 lux. -Rear facing black & white camera
Lighting	-Forward - Two 20 W high efficiency halogen lights. -Rear - Ultra high-intensity LED light array.
Propulsion	-Horizontal - Two thrusters with 60 mm propellers. -Vertical - One thruster.
TDS length	76 m
Max voltage in tether	48 VDC

Table 2.1: ROV Specifications (Source: Videoray.)

The propulsion system consists of three thrusters giving the ROV three degrees of freedom as illustrated in Figure 2.1. Horizontal thrusters are placed on the port and starboard side of the ROV, providing surge and yaw abilities. A vertical thruster placed on top of the ROV provides the ability to heave.

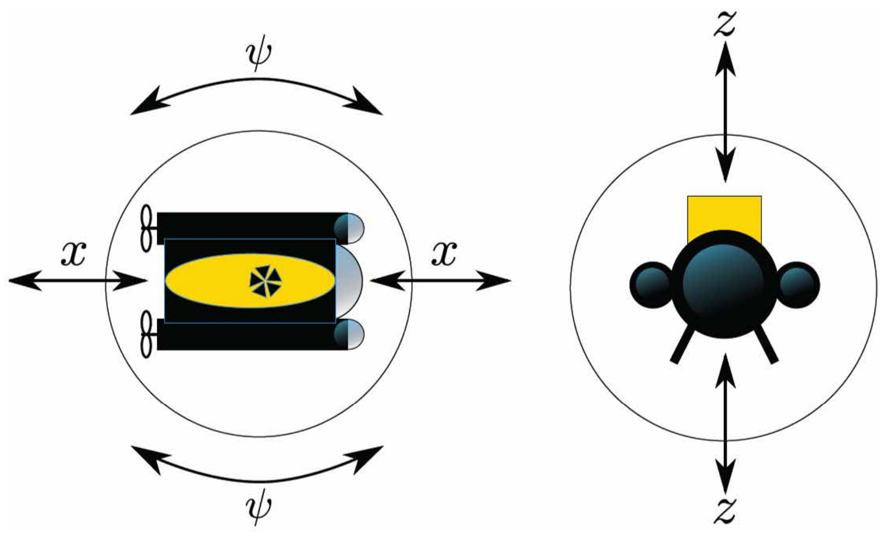


Figure 2.1: Surge (x), Yaw (ψ), and Heave (z). (Image source: Carlsen [2010])

The ROV has two cameras, one in the bow and one in the aft. The bow camera is forward facing with a control tilt giving it a 180° vertical field of view. Two forward facing lights are mounted on the sides of the ROV to illuminate its surroundings. The aft camera is mounted facing backwards with a LED array around its lens.

All the actuators, lights and cameras can be controlled from the ICB. One can also view the video feed by attaching a monitor. The PRO 3 S is built with an open architecture and can alternatively be controlled from an external device, such as a laptop computer, via a serial connection.

Accessories can be connected to the ROV by an accessory port over the aft camera. This port includes two wires for serial communication, a video channel, and power supply.

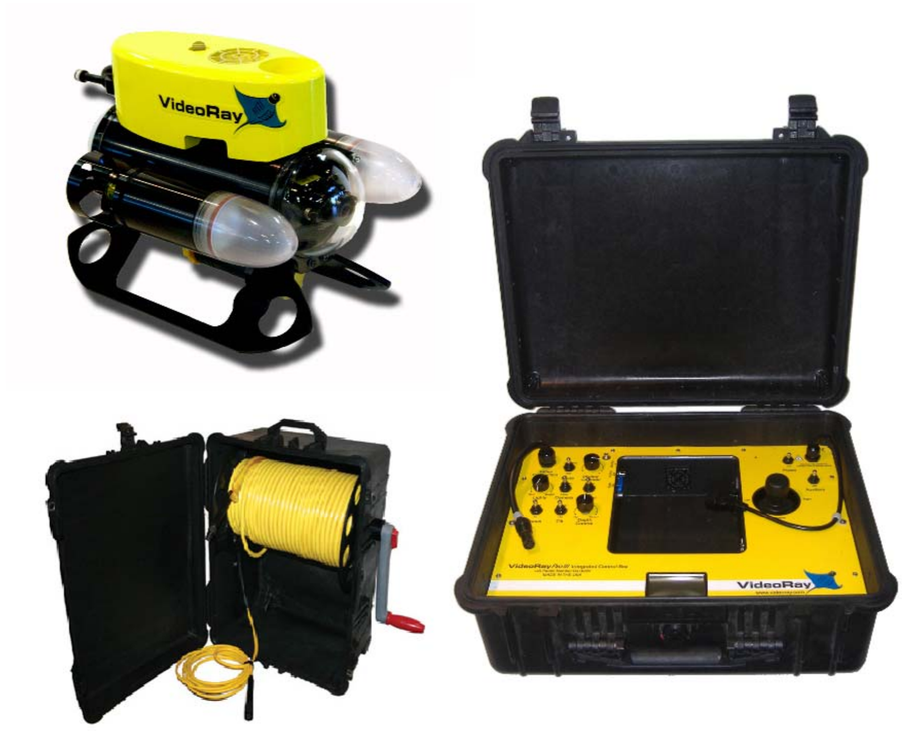


Figure 2.2: Videoray PRO 3S system. Upper-Left: ROV, Lower-Left: TDS, Right: ICB. (Image source: Videoray Videoray)

Chapter 3

Specifications and Requirements

The idea is to have the ROV hovering close to the cage while using its on board camera to provide video of the cage net. This is illustrated in Figure 3.1. The video feed will be supplied to an external laptop computer which will analyze the frames to identify the net. This video feed may also be used for navigation. The laptop will generate control signals and send them back to the ROV, controlling its thrusters.

A laser-module will be constructed and attached to the ROV's belly. It will include laser line generators for distance measurement as well as instruments for better navigation. An overview of the hardware setup can be seen in Figure 3.2. Detailed specifications and requirements are given in the following sections.

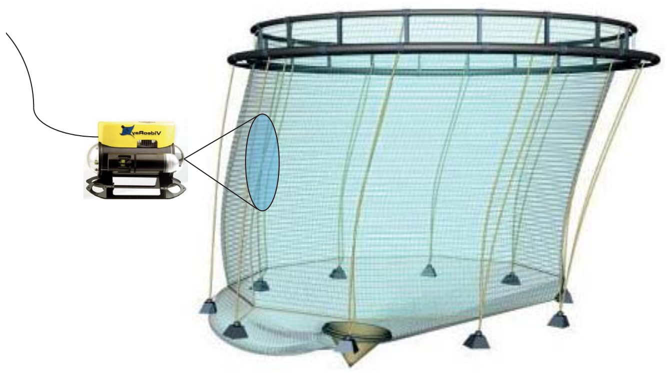


Figure 3.1: ROV surveying fish cage.

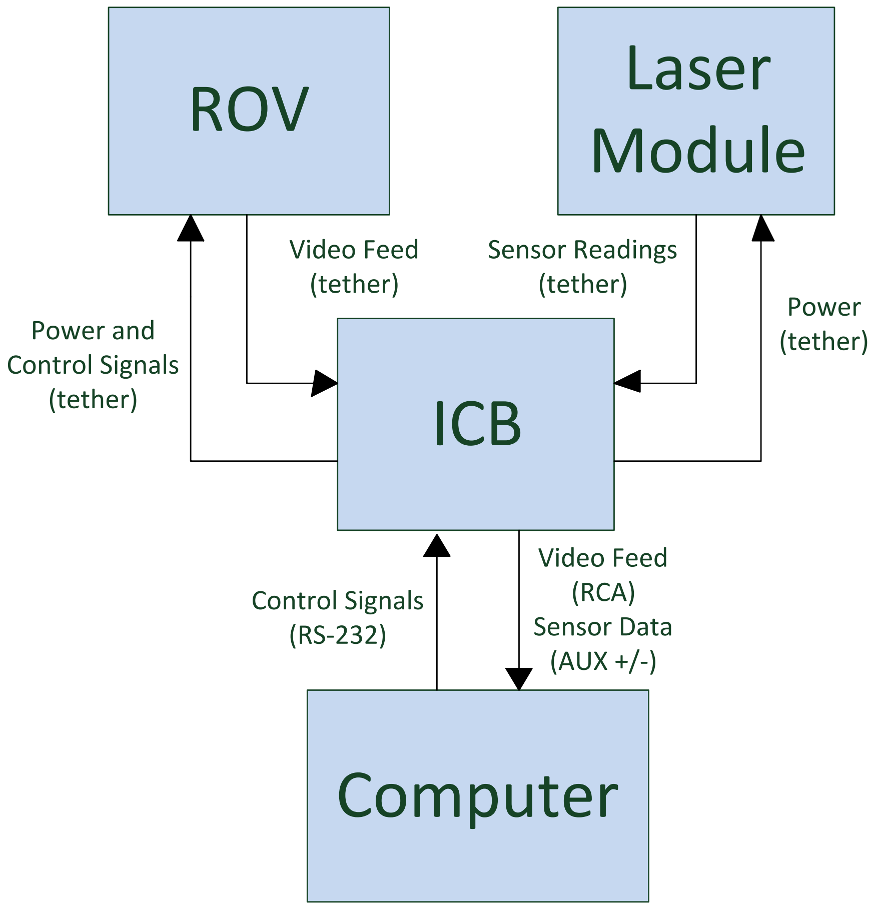


Figure 3.2: Overview of hardware setup. Both the laptop computer, the ROV and the laser module communicates through the ICB.

3.1 Hardware

3.1.1 ROV

Thrusters

The ROV needs a thruster system for propulsion. The thrusters should ideally provide the ROV with four degrees of freedom: surge, sway, heave, and yaw[1]. As can be seen in Figure 2.1, the PRO3S lacks sway capability, i.e. it can not move sideways. This makes it unable to traverse the net horizontally while at the same time pointing the camera towards it. The lack of sway capability may also make it difficult to hold the ROV in a steady position in the presence of current.

Camera

The ROV needs a video camera for filming the cage net. It should have a high enough resolution for effectively analyzing the net, and ideally also provide color video. The PRO3S has two cameras, but the one in the bow is clearly best suited for this project. The bow camera provides analog color video with a resolution of 570 lines. The camera is connected to a tilt system so that it can be adjusted to an optimal angle.

External Connections

The ROV needs to have an input for control signals and an output for video feed. The PRO3S provides an RS-232 interface to control all of its functions and an RCA connector for video feed output.

3.1.2 Laser Module

A laser module will be developed to help the ROV keep a constant distance to the net during surveys. This module can also include additional instruments to help the ROV navigate in the water. The PRO3S has a pre-installed compass and a pressure sensor, but these have shown to be unreliable on previous occasions. Installing more reliable and accurate sensors would be helpful for navigation.

Construction

The ROV will be used primarily in seawater which is corrosive to some materials. Materials used in the laser module housing should, if possible, not be prone to seawater corrosion. Otherwise, protection against corrosion might be needed. The housing should be able to withstand the pressure of 50 meters depth. The electronics need to run on the power supplied by the ROV: 12VDC (6W) or 48VDC (30W).

[1]The two remaining degrees of freedom, pitch and roll, are usually self stabilizing in ROV's, including the PRO3S.

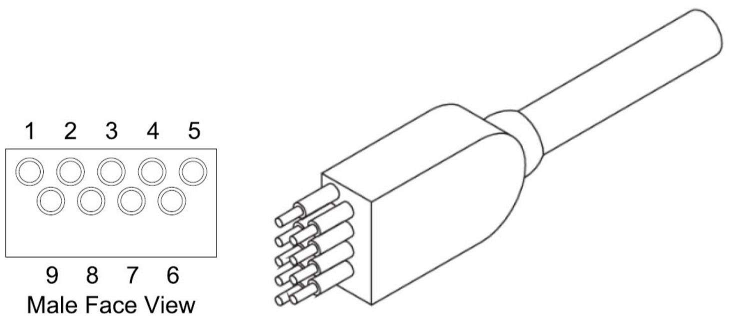


Figure 3.3: Male SubConn Micro Low Profile 9 tether connector. (Image source: SubConn Inc., www.subconn.com)

Data Transmission

There are two free wires in the tether cable, AUX+ and AUX-. These can be used by the laser module for communication with the laptop computer. A communication interface will need to be developed for this. The leads in the 76 meter long tether cable are separately insulated, but not shielded. The communication interface therefore needs to be robust with respect to both noise and long transmission lines. The bandwidth needs to be high enough to support the sampling rates of the sensors.

Cable Connection

The PRO3S has a female SubConn connector for connecting axillary equipment. The laser module will need the male equivalent (Figure 3.3) to use this connection.

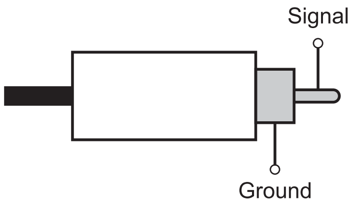
3.1.3 ICB Connection

Three separate connections need to be made between the ICB and the laptop computer: video feed, control signals, and laser module communication.

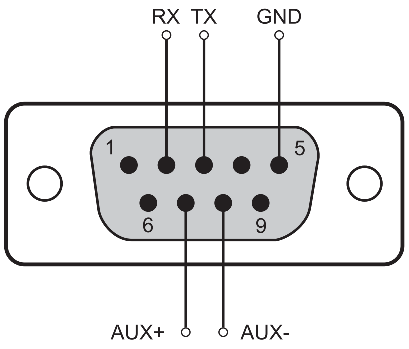
Video Feed

The video feed is accessible with a male RCA-connector (Figure 3.4(a)). The output from the ICB is an analog PAL-signal. This signal needs to be digitalized for processing on the laptop computer. Dedicated hardware, meeting the following specifications, is needed:

- Capture video from RCA connector
- Support 25Hz PAL-signals
- Preferably USB-interface for easy laptop connection.



(a) Male RCA/phono plug.



(b) Male DE-9/DB-9/D-sub connector.

Figure 3.4: ICB-connections. (Image source: Carlsen [2010])

Control Signals

The ROV is controllable through a standard RS-232 connection. Only three wires from the standard are used: RX, TX, and GND. The laptop computer can be connected by a D-sub9 connector, see Figure 3.4(b). The RS-232 interface uses the following parameters:

- 9600 bps baud rate
- 8 data bits
- 1 stop bit
- no parity

Laser Module Communication

The AUX +/- wires to be used for laser module communication are available at pin 7 and 8 of the same D-sub9 connector as the control signals (Figure 3.4(b)). In order to separate the control signals from the AUX-wires a splitter is needed. A cable for this purpose has already been made by earlier student Knut Ove Stenhagen.

Laptop Computer

The computer should be a laptop for system mobility. It needs to meet the following specifications:

- Run Windows XP, Windows Vista or Windows 7
- Have a display that is readable under daylight conditions
- Have USB-/serial-ports for the ICB-connections

The video feed will most likely be connected by a USB-port. The control signals need a serial port (D-sub9). Alternatively, a USB-serial adapter can be used. The connection

Figure 3.5: Getac V100 rugged convertible laptop. (Image source: Getac, www.getac.com)

needed for the laser module connection is dependent on the protocol chosen, but most likely it will be a USB- or serial-port. To summarize, the laptop requires at least one USB-port plus two additional USB- or serial-ports.

A Getac V100 (Figure 3.5) has been purchased by the institute for use with the ROV. This is a “rugged convertible laptop”, made specifically for use in harsh environments. It runs Windows 7 and has a display which, on earlier occasions, has proven to be easily readable outdoors. The display size is 10.4” with a resolution of 1024x768 pixels. The V100 includes a serial port as well as two USB-ports. It thus meets all the required specifications for this project.

3.2 Firmware

The laser module needs a micro controller unit (MCU) to gather, process, and transmit signals from the sensors. It should also control the enabling/disabling of the laser line generators. The MCU will need to be programmed to perform these tasks. This program will from now on be referred to as “the firmware”.

The signals from the different sensors will most likely be in different formats and sampled at different intervals. The firmware should convert the signals to a suitable format and transmit them to the laptop computer as efficiently as possible. There is no need to wait for readings from all the sensors before transmitting, so the firmware should be able to transmit these signals independently.

3.3 Software

The laptop needs a program able to process and analyze the video feed, read and process sensor data, and control the ROV. It should also provide a graphical user interface (GUI). This program will from now on be referred to as “the software”.

3.3.1 Video Feed Analysis

The software should analyze the video feed in real-time. Not every frame needs to be analyzed, but the sampling rate should at least be high enough to allow for all traversed net to be included. If there is enough processing power to increase the sampling rate further, this will create a redundancy and thus improve the robustness of the system.

The anaysis algorithm should be as robust as possible with respect to lighting conditions, angel of view, algae growth, and to foreign objects passing in between the camera and the net. It should also be adaptable to the most common mesh dimensions used in fish farms.

3.3.2 ROV Control

The software should control the ROV so that it is able to maintain its position in the water. Ideally, the software should control the ROV to move in a pattern to search the whole cage net for damage. However, due to the ROV’s inability to move sideways (sway), this might be difficult to achieve. Moving in a search pattern is also of less importance as the purpose of this project is only to demonstrate the concept. Later development of an AUV will likely include thrusters to achieve all four desired degrees of freedom. A sophisticated control system to cover the cage with only three degrees of freedom would therefore become obsolete in later stages of the AUV-project.

3.3.3 Graphical User Interface

The GUI should be informative, interactive, and user friendly. Sensor data should be displayed in a human readable format. Video feed and the processed images should also be displayed. The user should be able to enable/disable the lasers and to adjust ROV control as well as the analysis parameters.

Chapter 4

Design

4.1 Laser Module

4.1.1 Concept

Using laser beams to measure distance is not a new concept. Laser rangefinders have been used for many years in various application ranging from high-precision military use to measuring distances at the golf course. The most common principle used in laser rangefinders is what is known as "time of flight" (TOF) (Amann et al. [2001]). The rangefinder emits a laser pulse in a narrow beam towards an object and then measures the time it takes the reflection on the object to return to the sender. These rangefinders can often make accurate measurements up to several kilometers. They are, however, not ideal for this project for several reasons:

- TOF rangefinders emit only a narrow beam. When measuring the distance to a net in a fish cage, this beam will need to hit exactly at a mesh edge for a correct reading. Since these edges are very narrow, achieving reliable distance measurements would be virtually impossible with a moving ROV.
- Because of the high speed of light, TOF rangefinders work best at distances in excess of one meter, where small errors are acceptable (Amann et al. [2001]).
- Laser rangefinders designed for underwater use are generally very expensive.

The first reason alone is strong enough to reject standard TOF rangefinding for this project. A principle used in many industrial laser distance sensors is triangulation (Dorsch et al. [1994]). A laser pulse is emitted in the same way as in TOF. A sensor close to the laser emitter then measures the angle of the reflection returning from the object. The angle of refraction varies proportionally to the distance of the object. This method achieves high precision at small distances. However, this method also suffers from the fact that the beam needs to hit exactly on a mesh edge for correct measurement.

A much simpler approach to laser distance measurement will be undertaken in this project. The plan is to use two line laser generators attached underneath the ROV camera and pointing straight forward. The laser beams will form lines, instead of single points, when they hit an object. In this way, parts of the line will always hit a mesh edge on the cage net. The laser generators will be placed horizontal with respect to each other and generate two parallel vertical lines. Since the laser lines both point straight forward, the distance between the lines will be the same at all distances from the ROV.

The camera on the ROV will be used as the measurement sensor. When the ROV is pointed towards an object, the laser lines will show up in the camera feed. The horizontal distance between the lines in the image can be measured as a number of pixels. This distance will vary with the distance to the object. Specifically, the distance between the lines in the image should be halved when the distance from the object is doubled. Mathematically, the relationship can be formulated as:

$$d = A \times x^{-1}$$

where d is the distance between the ROV and the object, A is a constant, and x is the number of pixels between the lines in the image. The constant A is decided by the physical distance between the laser line generators, the shape of the lens, and by image resolution. The power of -1 may not be absolutely accurate due to "fisheye" effect on the ROV camera. The exact formulation will be determined empirically later in the project.

4.1.2 Communication

The ROV system has two auxiliary wires that have no defined purpose and can be accessed from the ROV and the control panel. That makes them perfect for serial communication between the laser module and the computer. The communication signal will go through the 76 meter long TDS cable together with supply power for the ROV, so the communication protocol will have to be robust with regard to distance and noise.

The Controller Area Network (CAN) protocol is ideal for this purpose. CAN was originally released by Robert Bosh GmbH in 1986, and an updated version, CAN 2.0, was released in 1991. It was originally developed for the automobile industry to make cars more reliable, safe, and fuel-efficient, while decreasing wiring harness, weight, and complexity. It has also gained popularity in other industries such as automation, medical equipment and mobile machines (Pazul [2002]). Several nodes can be connected to one bus, but in our case only two nodes will be used. The CAN protocol can send information at bit rate of up to 1 Mbit/s, so latency should be of no concern. In addition, it has a sophisticated error handling system, making sure that all messages are delivered to the receiver. The maximum bus length is specified as 1000m. Two versions of CAN 2.0 are available: A and B. The B version supports more nodes on the same bus, but since only two nodes will be used in this project, CAN 2.0A is used. The CAN-bus consist

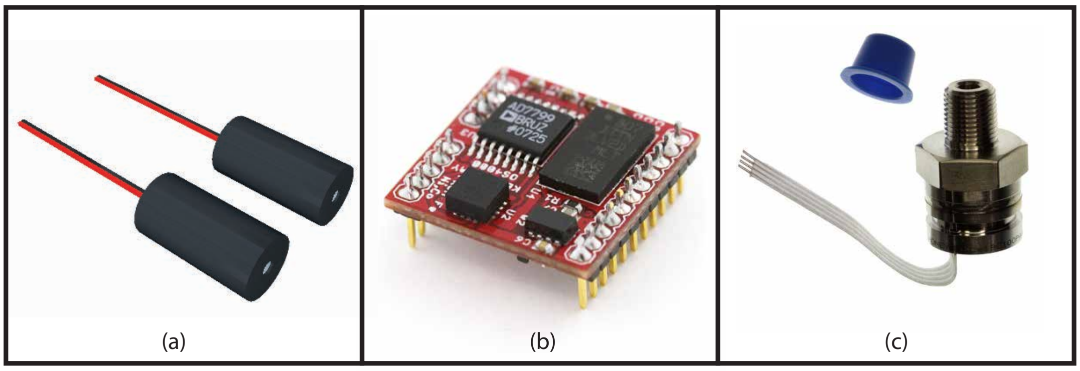


Figure 4.1: a) Uniform Line Generators from Diode Laser Concepts Inc. b) OceanServer Technology OS4000-T compass module c) Honeywell S&C 19C100PA4K Pressure Transducer

of two wires: CANH and CANL. They need to be terminated in both ends of the bus by a 120Ω resistor.

4.1.3 Electronics

Laser line generators

Two 5mW laser line generators from Diode Laser Concepts Inc. have been chosen for use with the laser module. They were chosen because of their industrial quality and easy availability. They have a wavelength of 635nm (red) and a fan angle of 45°. The lasers were tested in a water tank to be visible underwater during daylight conditions. A datasheet for these specific laser generators could not be found, but the model series have an input voltage of 5VDC and a power consumption of up to 150mA. A quick test revealed that these specific laser generators consume about 60mA. An illustration of the lasers can be seen in Figure 4.1(a).

Sensors

The module will also contain two sensors for navigation; a compass and a pressure sensor. The sensors are chosen primarily because they have both been used by earlier student Andreas Carlsen during his MSc. thesis (Carlsen [2010]), and has given satisfactory results in use with the ROV. As the sensors can be collected from his project, no purchase of new sensors is needed.

The compass is an OceanServer Technology OS4000-T, depicted in Figure 4.1(b). It is “tilt compensated”, meaning that it will show the correct orientation even if pitch or roll is applied to the ROV. It provides the compass heading in clear text (ASCII) as a number between 0-359.9°, so the signal does not need any further processing. Selected specifications:

- Compass accuracy: maximum 1.5° rms error when roll and pitch angles are $< \pm 60°$
- Bandwidth: maximum 40 Hz
- Output resolution: 0.1°
- Operating temperature: -40°C to +80°C
- Operating voltage: +3.3V to +18V DC
- UART Baud Rate: 4800 $^{bit}/_{s}$ to 115200 $^{bit}/_{s}$

The baud rate is set to 19200 $^{bit}/_{s}$ and the sample rate to 10 Hz, which should be sufficient for the current application.

The pressure sensor is a Honeywell S&C 19C100PA4K, depicted in Figure 4.1(c). This sensor has been used by both Carlsen [2010] and Skjaeveland [2009]. It was shown to provide good accuracy when used with proper signal processing. Its pressure range is 0-100 psi (0-689400$^{N}/_{m^2}$), relative to vacuum. This means that the transducer will measure one atmosphere, or 14.696 psi (101325 $^{N}/_{m^2}$), at sea level. The pressure range left to measure water depth is then 85.304 psi. Hydrostatic pressure in seawater can be calculated by the following equation (White [1998]):

$$p = p_{atm} + \gamma z_t$$

where p is the pressure in $^{N}/_{m^2}$, p_{atm} is the surface atmospheric pressure, γ is the specific weight of seawater (10050 $^{N}/_{m^3}$), and z_t is the depth. We can use this formula to calculate the maximum operating depth of the pressure transducer:

$$z = \frac{p_{100psi} - p_{atm}}{\gamma} = \frac{689400^{N}/_{m^2} - 101325^{N}/_{m^2}}{10050^{N}/_{m^3}} \approx 58.5m$$

Since fish farms rearly go deeper than 50 meters, the depth rating should be sufficient for this project. The output signal is an analog low voltage signal which will need to be amplified before being transmitted to the MCU.

MCU

An Atmel AT90CAN128 (Figure 4.2(a)) has been chosen as the module's MCU. It has a built in CAN-controller (explained in the communication sub-section), UART for compass communication, 10-bit ADC for reading pressure measurements, and general I/O-pins to control the lasers.

Power

The tether connection from the ROV supplies 12VDC up to 6W. The module has been chosen to run on 5VDC which is an acceptable voltage for all included components. A TracoPower TES2N-1211 switching regulator (Figure 4.2(b)) will convert the power

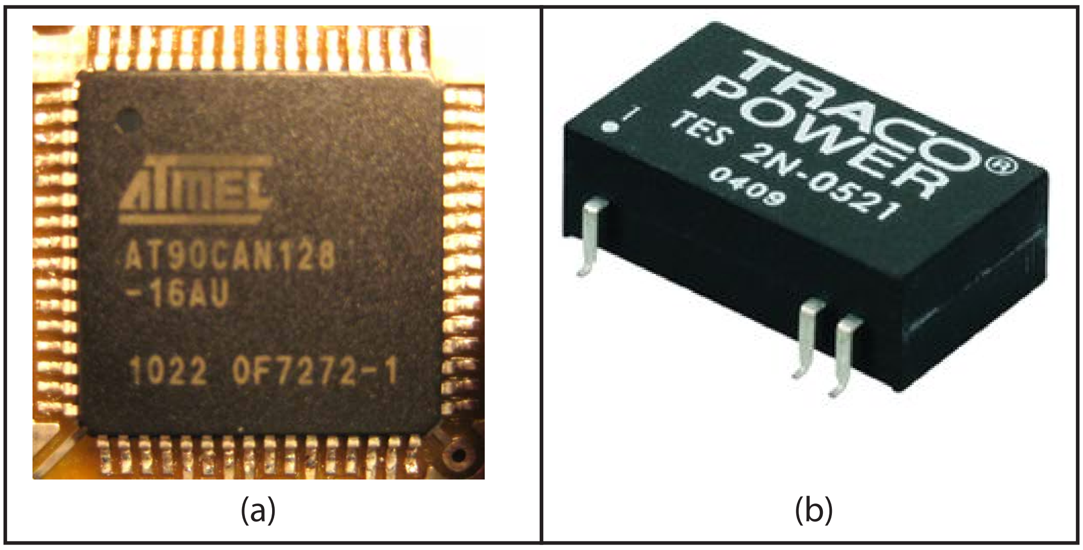


Figure 4.2: a) Atmel AT90CAN128 b) Tracopower TES-2N1211 switching regulator (picture depicts different unit, but with same package. Image source: Tracopower).

down to this voltage. It has a 9-12VDC input rating and a 5VDC 2W output. The electronics in the laser module have been calculated to use maximum 250mA or 1.25W, so the switching regulator should provide more than enough power. The TES2N-1211 also has a built-in EMI filter to provide a clean output. The typical efficiency is 77%, which makes its maximum power consumption 2.6W, far below the 6W provided by the ROV.

Pressure signal amplification

As described earlier, the pressure transducers output is an analog low-voltage signal that will need to be amplified before it is processed by the MCU. This has been done previously, first by Skjaeveland [2009] and later by Carlsen [2010]. Their steps and choice of components will be used in this project as they have already proven to give good results.

The 19C100PA4K input terminals are connected to the 5VDC power supply. The output signal is specified as $^1/_{100}$ of the of the input voltage so the output will be in the range 0mV to 50mV. A single-supply INA155UA instrumental amplifier from Texas Instruments is used to amplify this signal. The unit gives a linear output close down to 0V. Because the pressure transducer will measure atmospheric pressure at sea level, input voltage for the instrumental amplifier will always be in the linear amplification area.

Low-pass filtering will remove high-frequency noise on the input signal. An RC-filter

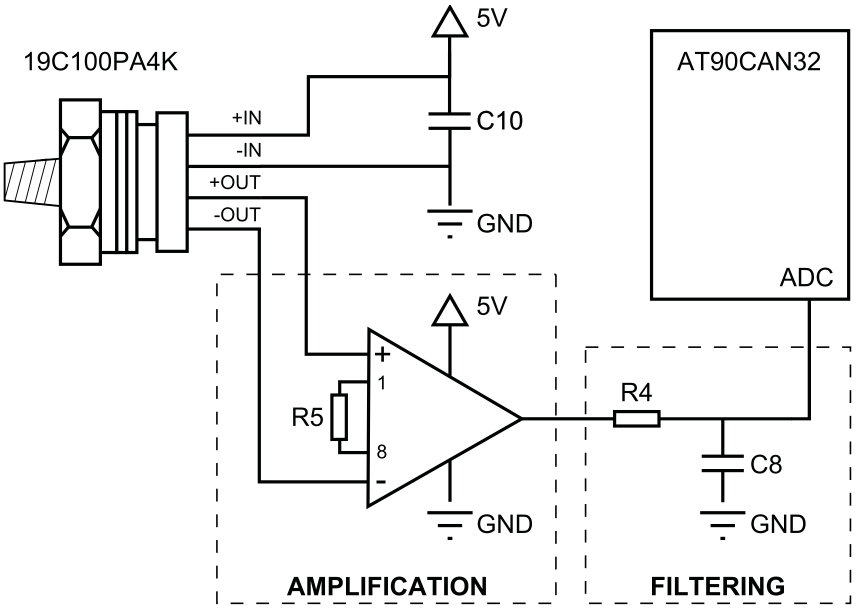


Figure 4.3: Pressure measurement processing chain. (Image source: Carlsen [2010])

with a cutoff frequency of 10Hz is used for this purpose. This will allow for rapid depth changes while at the same time removing most disturbances. The resistor (R) and capacitor (C) values control the cutoff frequency ($\omega_c = 2\pi f_c$) by the following relation (Nilsson and Riedel [2005], eq. 14.19):

$$f_c = \frac{1}{2\pi RC}$$

By connecting a 0Ω resistor between pin 1 and pin 8, the instrumental amplifier is set to amplify the input voltage by a factor of 50. This will give an output voltage range of 0-2.5V. The complete pressure signal processing chain can be seen in Figure 4.3.

CAN-signal processing

A CAN-transceiver is needed as a conversion-medium between the CAN-bus and the MCU. The transceiver converts digital MCU-signals to differential signals for transmission and vice-versa. It also protects the MCU from voltage peaks, Electro Magnetic Interference (EMI) and, Electrostatic Discharge (ESD). Carlsen [2010] successfully used a Mictrochip MCP 2551 CAN-transceiver in his project. This same transceiver with the same configuration will be used in this project.

The transceiver slew-rate controls the rise and fall times of the bus-signal. On the MCP2551, the slew-rate is controllable by connecting a resistor between pin 8 and ground. A chart of slew-rate as a function of resistor value can be seen in Figure 4.4. Using a 22kΩ resistor will give a slew-rate of about $17 V/\mu s$. This will allow for the fast rise and fall times required by the application.

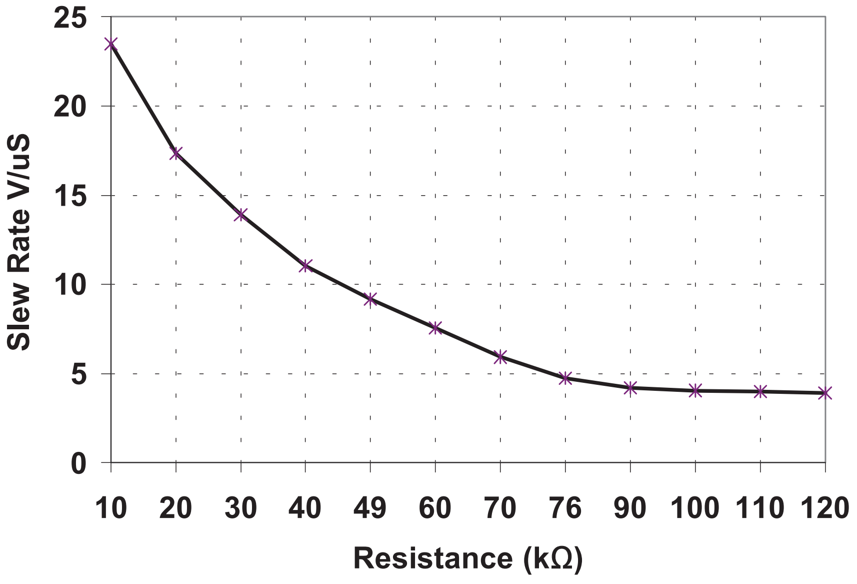


Figure 4.4: MCP2551: Slew-rate as a function of resistance. (Image source: Microchip MCP2551 Data Sheet)

Circuit board

The circuit board was designed in CadSoft Eagle 5.10. It is a two layer circuit board designed to be as small as possible while at the same time minimizing noise and electrical interference on the compass. The following steps have been taken to achieve these goals:

- The switching regulator is assumed to be the biggest source of electromagnetic noise and is placed on the opposite side of the board from the compass.
- Power tracks have been made wide and as short as possible.
- 0.1 uF[1] decoupling capacitors have been connected in parallel with the power input on all components.
- Both layers of the circuit board has been made with a ground plane covering all free space.
- Vias have been placed strategically to minimize ground paths.

The board design can be seen in Figure 4.5. Full schematics and board design can be found in Appendix A. Four LEDs are included to indicate the module's status. A red LED indicates power, while three green LEDs indicate CAN-communication, ADC-communication, and UART-communication. A pushbutton is connected to the MCU

[1]This capacitor value has previously been used with good results

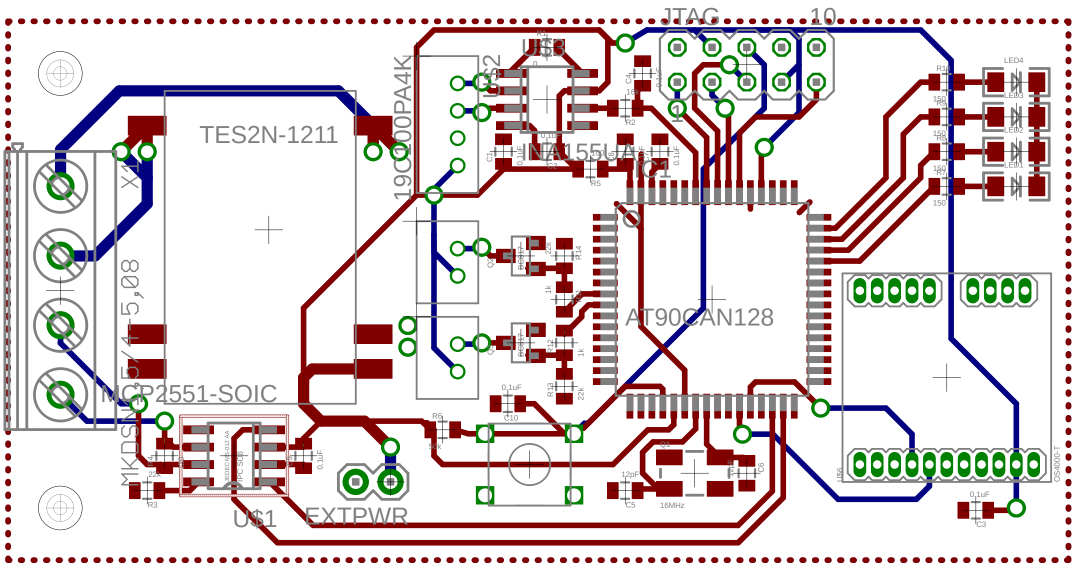


Figure 4.5: Circuit board design

reset line. Together with the LEDs, this will provide a powerful tool during initial testing and debugging.

4.1.4 Mechanical design

The laser module will be attached underneath the ROV. An attachment has already been made for an earlier project and can be re-used for the laser module. The main part of the module will have to be a cylinder of 600mm outer diameter to fit this attachment.

Distance between the line laser generators is the most crucial part of the mechanical design. Increasing distance between the lines will increase the resolution of the measurement as variation in pixels with distance to the object will be greater. However, if the lines are too far apart, they may not show up in the video when the ROV is close to the object being measured. The optimal line distance is found where both lines are just in the image at minimum working distance. This optimal distance can be found by placing the ROV in front of a wall at minimum working distance and measuring the horizontal length of wall that shows up in the image. Unfortunately, the ROV was damaged in mid March, and had to be sent for repair. Calculation of optimal distance was therefore somewhat inaccurately calculated by looking at old footage from the ROV and approximating the distance and length of objects in the image. The minimum working distance was set to 100mm, as the ROV will probably search for net integrity at far greater distances from the net. Optimal line distance between the laser lines was calculated to 265mm. The inaccuracy of this calculation is however of less importance as the mechanical design will limit the line distance further, as explained below.

Per Inge Snildal at the department's mechanical workshop has created the mechanical design with the following specifications in mind:

- The laser module should fit the attachment already made for the ROV and should therefore include a cylinder with outer diameter of 600mm.
- The housing should be waterproof and be able to withstand the pressure at 50 meters depth. It should also not be prone to seawater corrosion.
- The line laser generators should be as far apart as practically possible (up to 260 mm), and should point straight forward from the ROV. They need to be isolated from water but they also need some sort of window to allow the beams to pass through.

The resulting design has been drawn in Google SketchUp and can be seen in Figure 4.6. The full 3D model can be found in the electronic attachment included with this report. The module is nose-heavy and have a slight positive buoyancy, but a manual weight-adding system on the ROV can easily compensate for this. The cylinder is connected to the front part of the module by four bolts. An inlaid o-ring ensures a tight seal. The "laser-windows" also have inlaid o-rings and can easily be removed to adjust the lasers. The PCB is attached to the front part of the module by a bracket. The entrance of the SubConn cable has been sealed by insulation tape.

The distance between the lasers is 164.5mm, 100.5mm less than the calculated optimal distance. This will reduce the measurement resolution but was the result of practical manufacturing considerations. A beneficial side-effect is that the reduced line distance will allow the ROV to measure object distance at even closer range than the chosen minimum working distance of 100mm.

4.2 ICB connection

Two external hardware devices are needed to connect the ICB to the Getac V100 laptop computer: A frame grabber to digitalize the video feed and a CAN-adapter to communicate with the laser module's CAN-interface. Such devices were acquired for Carlsen [2010] and can be re-used for this project.

A TerraTec Grabby frame grabber will be used to digitalize the video. The product is cheap, can interface RCA composite video, and delivers output via USB. Selected specifications are given in Table 4.1.

The laser module's CAN-interface will be connected to a Kvaser Leaf SemiPro HS to make the signals available via USB. Selected specifications are given in Table 4.2.

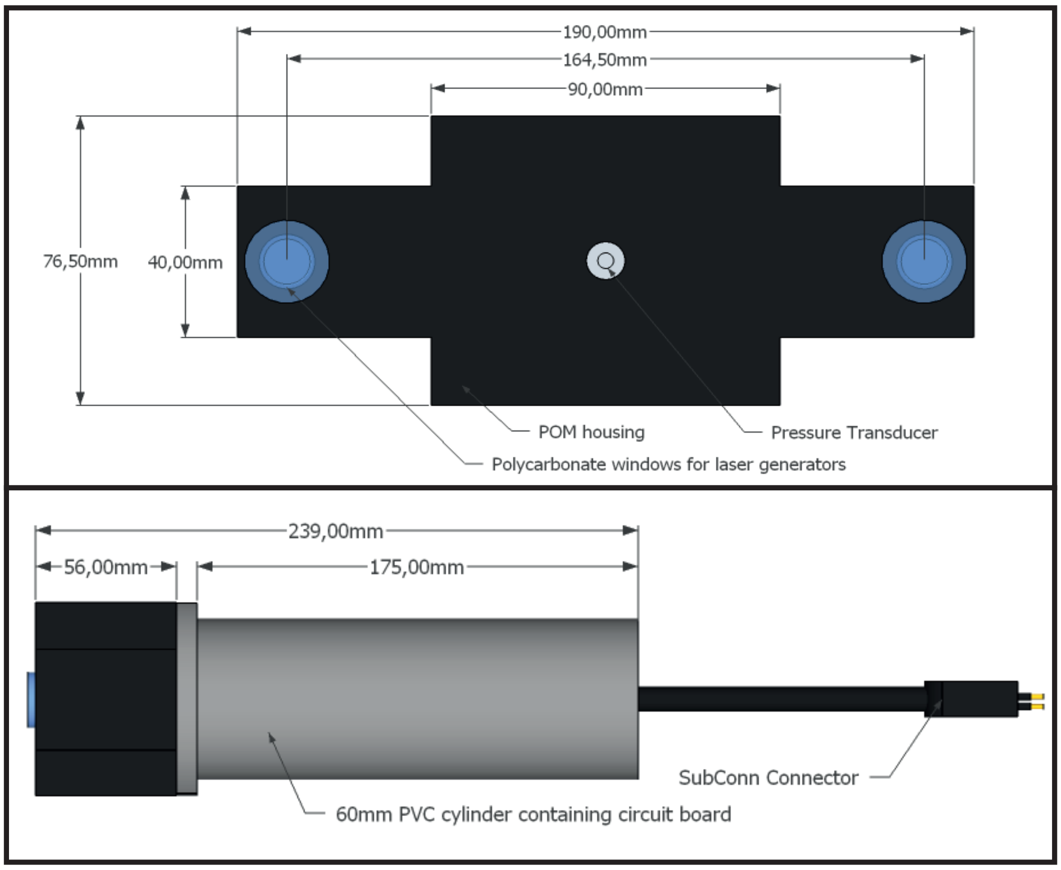


Figure 4.6: Laser module, mechanical design. Top: Front view. Bottom: Side view.

Data Interface	USB 2.0
Analog Video Input	Composite- and S-Video
PAL Capture Resolution	Up to 720x576 pixels
PAL Capture Rate	Up to 25 fps at maximum resolution
Power Source	USB

Table 4.1: TerraTec Grabby specifications.

Data Interface	USB 2.0
Galvanic Isolation	Yes
Supported Message Formats	2.0 A and 2.0 B
Bitrate	$5^{kbit}/_{s}$ to $1000^{kbit}/_{s}$
Maximum Message Rate	15000 Hz
Power Source	USB

Table 4.2: Kvaser Leaf SemiPro HS specifications.

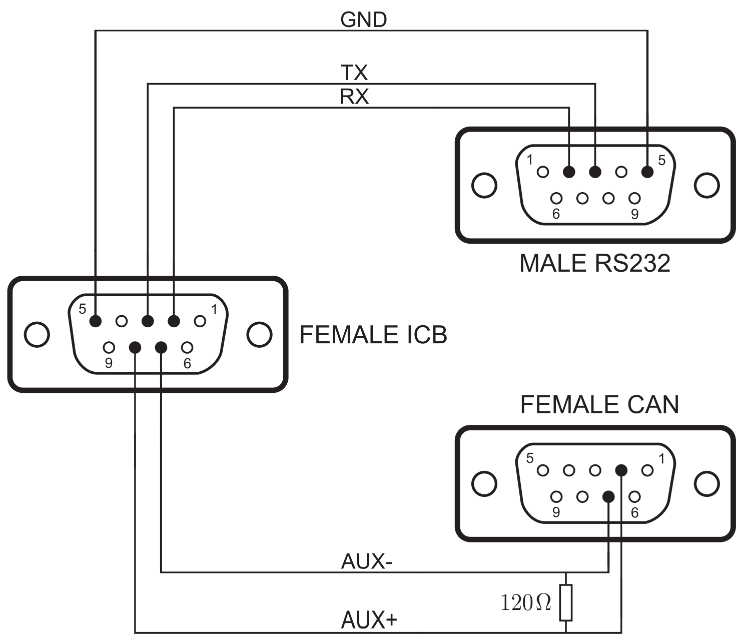


Figure 4.7: D-sub9 splitter wiring. (Image source: Carlsen [2010])

As mentioned in the specifications chapter, both the CAN-bus and the RS-232 interface are accessed by the same Dsub9-port on the ICB. Earlier student Knut Ove Stenhagen has made a splitter to connect both interfaces simultaneously. The wiring of this splitter is shown in Figure 4.7. A 120Ω resistor is included inside the housing of the CAN-part of the splitter to terminate CANH/CANL (AUX+/AUX-).

4.3 Firmware

The firmware is the program that will be stored in the flash memory of the MCU. The program has been designed so that three routines control program behavior: UART (compass) interrupt, ADC (pressure sensor) interrupt, and the main control loop.

Compass interrupt

The compass can deliver data at many different formats. The chosen one is referred to as "$C" and the chosen outputs[2] are azimuth (compass heading), pitch, roll, and temperature. The resulting output is on the form "$Cxxx.xPxx.xRxx.xTxx.x*cc" and is interpreted as follows:

- $ indicates the start of a new sentence

[2]Clearly, the most important output for our purpose is azimuth. The other outputs are included so that they can easily be accessed, should they be needed in the future.

- C, P, R, or T, means that the following value is the azimuth, pitch, roll, or temperature, respectively.
- xx(x).x is the measured value. Azimuth is between 0.0 and 359.9.
- *cc is an XOR checksum sent at the end of each sentence.

Compass communication uses UART1 on the MCU with baud rate 19200, 8bit, 1 stop bit, no parity. Every time a new message is received, the interrupt routine is run. The routine reads the message which is a character in ASCII format. It interprets where the character belongs and store appropriate values. Whenever a full reading (e.g. azimuth) is stored, a flag is set to indicate that this new reading is ready. The compass is set to operate at 10Hz so that 10 full data words are sent each second.

ADC interrupt

The MCU system clock runs at 16MHz. The ADC is set with a prescaler of 128, so that it runs at 125kHz. Every time a new reading is ready, the ADC-interrupt routine is run. This routine stores the reading and sets a flag indicating that a new reading is available. The reference voltage is set to 2.56V and the readings have a resolution of 10 bits (1024). The maximum voltage from the instrumental amplifier is 2.5V, so the true resolution of the depth readings are $\frac{1024*2.5V}{2.56V} = 1000$. A reading change of one bit represents a change of 6.86cm in depth.

Main Routine

This is the first routine that is run when the system starts up. It first initiates ADC, UART, and CAN communication. It then enters an infinite control loop which does the following:

1. Check sequentially if new readings are received from the sensors (pressure, azimuth, pitch, roll or temperature). When a new reading is found, transmit it by CAN and proceed to check the next reading.
2. Check if a new message is received by CAN and take the appropriate action. CAN-messages received by the MCU contain information to enable/disable the lasers/status LED's, and also if the compass should be configured[3].

As long as no new CAN-messages are received, the loop runs at approximately 10-15Hz. This means that the compass readings are transmitted at about the same speed as they are received (10Hz). The ADC-readings are transmitted much slower than they are read (125kHz), but still more than fast enough for navigation. The loop continues to run as long as power is present and is only interrupted by CAN-, UART-, and ADC-interrupts. The data format for all CAN-messages can be found in Appendix B.

[3]There has been no need to configure the compass after installation of the laser module. Compass configuration has therefore not been implemented in the software. The firmware still responds to CAN-messages indicating compass configuration and configures the compass accordingly. Software enabling of compass configuration can thus easily be implemented for future use if needed.

4.4 Software

During his work with the ROV, Carlsen [2010] used the Nokia/Trolltech Qt Framework. Qt is a cross-platform application framework widely used for making applications with a GUI. Carlsen achieved a good result using this framework and it will also be used in this project.

Qt's use of "signals" and "slots" has been central to the development of the software design. These are mechanisms provided by Qt to help creating loosely coupled components. A signal is triggered by an event which can be user-defined to almost anything. E.g., it can be a button click in the GUI, a timed trigger, or any trigger defined in a function. A slot is the signals counterpart. The signals can be connected to slots and pass arguments along with the trigger signal. Whenever a slot is triggered, it initiates a function specified in the program. In this way, functions can communicate without knowing of each others existence.

The software has an object oriented design and contains a series of intertwined objects. A large group of signals and slots connect the objects together, but the catalyst can be found in two objects: Video Feed and CAN.

Video Feed

The video feed gathers frames from a user specified source. The user can specify the ROV-camera, an AVI-file, or an image file. When enabled, the video feed uses a timer to grab frames each 100ms (10 frames per second). This frame rate can later be adjusted according to the processing power of the computer used to run the software. As soon as a new frame is ready, the video feed emits a signal containing the frame. This signal is then picked up by three other object slots:

- The GUI receives the frame so that it can be viewed by the user.
- An analysis object receives the frame for analysis. The frame is analyzed according to user-specified options.
- A video recorder also receives the frame so that the video feed can be recorded if desired.

Capturing and emitting frames is the sole purpose of the video feed, but with each new frame, a cascade of events is triggered. The slots triggered in GUI and video recorder are dead end, meaning that they do not lead to new signals being emitted. The analysis, on the other hand, emits several new signal based on the analysis result. When a new analysis is ready, the analysis frame is passed on to the GUI so that it can be viewed by the user. The frame is also passed on to the video recorder so that the analysis feed can be recorded. If laser line distance is enabled, the calculated distance is sent to the GUI for display. The distance is also sent to the navigation object, which in turn calculates and emits the ROV's position compared to the desired position.

CAN

The CAN object is responsible for communication with the laser module. When a CAN-frame is received, the CAN object checks the frame identifier to see what the frame contains. If the frame contains a new pressure reading or compass heading, the value is transmitted to the sensor filter for processing. The compass heading is received in clear text and does not need much processing, while the pressure readings are received as ADC-values from the MCU and need to be converted to a depth in meters. The depth readings are also passed through a moving average low pass filter to remove noise. As soon as the values are processed, they are transmitted to the GUI for display and to the navigation object.

ROV Control

The navigation object gathers distance, compass heading, and depth measurements and compares these to the set desired values. Error estimates are then calculated and sent to the controller. All desired values can be updated from the GUI.

A PI-controller controls the ROV thrusters based on the error estimates received from the navigation object. It uses a timer to update integrals and control signals every 25ms. The control signals are then sent to a ROV communication object which generates the appropriate RS-232 signals to control the ROV.

Analysis

The analysis object is responsible for taking in video frames and interpreting what they contain. This is clearly most complex part of the software and the part that has been most time consuming to develop by far. A lot of theory lies behind the methods used, and initial testing has been performed during the development. The next chapter has been devoted to elaborate on the development process and the theory behind it in detail.

Chapter 5

Computer Vision

Computer vision (CV) is the science of having a computer extract information from an image to help it solve a task. A big part of this MSc. Thesis is to make useful algorithms for inspecting net integrity based on digital images. This would be a daunting task if the algorithms were to be made from scratch. Luckily, there has been a lot of research in the field of CV during the last decades. Hundreds of CV-algorithms have been developed, many of which may prove useful in this project. This chapter will present a brief description of CV as a research field, followed by a detailed description, along with initial testing, of relevant algorithms and methods.

5.1 Computer Vision

As humans, we perceive the three-dimensional world around us with little effort. When looking at a family portrait, we can separate the people from the background, identify them, tell what they are wearing, and even tell their emotions with ease. After decades of research, scientists are still far away from fully understanding how the human visual system works.

In the late 70's, computers had enough computing power to do simple analysis on digital images. Since that time, a lot of research has been made towards the ultimate goal of having computers replicate the human visual system. We now have reliable techniques to achieve specific results, such as tracking an object moving against a background, identifying objects in an image, and even creating accurate 3D-models from thousands of images taken from different angles. However, despite all these advances, we are still far away from having computers capable of interpreting images at the same level as human beings.

A problem with CV is that there is no standard formulation of how CV-problems should be solved. Instead, there exist an abundance of methods and algorithms to solve specific tasks where the solution can seldom be generalized over a wide range of applications. A related problem is the inherit difficulty in making methods that are robust and adaptable to change. A method that works well in one scenario might fail utterly if the circumstances, say the lighting conditions, are slightly changed. Comprehensive testing is therefore important when implementing CV-algorithms.

The main purpose of using CV in this project will be to analyze the cage net for damage. The focus will be on mesh analysis, i.e. to identify each mesh in the cage net and check for damage. In this way, net damage can be identified at an early stage before holes are large enough for fish to escape. The alternative would be to analyze the net at a larger distance, treating the net as a solid object and looking for large holes. This method would allow for greater areas to be searched at a time but would not allow detection of early stage damage. A secondary purpose of CV will be to track two laser lines on the net for distance measurement. Since there is no general solution to a CV-problem, a variety of algorithms will be tested, and the promising ones will be implemented in the final software. The following sections will give a detailed description of how the problem is approached, as well as the theory behind the methods used.

5.2 Separating Image

The video feed from the ROV will consist of a stream of 10 still images per second. We want to sample and analyze images from this video feed. During surveying, these images will consist of a foreground (cage net) and a background (water, and possibly fish, foreign objects, etc.). We are only interested the foreground, and so a logical first step is to separate the foreground from the rest of the image. If this is done successfully we don't have to take the background into consideration when further processing the image.

When creating a method to separate the image, it is important to consider how the foreground differs from the background. To understand this, we need to take a look at how digital images are stored. We can think of a grayscale image as an n-by-m matrix[1], where n and m are the vertical and horizontal resolution, respectively. Each element in the matrix holds the value of the pixel at that location. In an 8-bit image, this value ranges from 0-255, where 0 is black and 255 is white.

A color image can be seen as an n-by-m-by-c matrix, where the extra dimension, c, is the number of color channels. Usually, $c = 3$, but it depends on the color model. There are many different color models, but we will only consider two for this project, RGB and HSV.

[1]In practice, images are often stored as information about the image dimensions, format, etc., along with a one-dimensional data array containing all the pixel values.

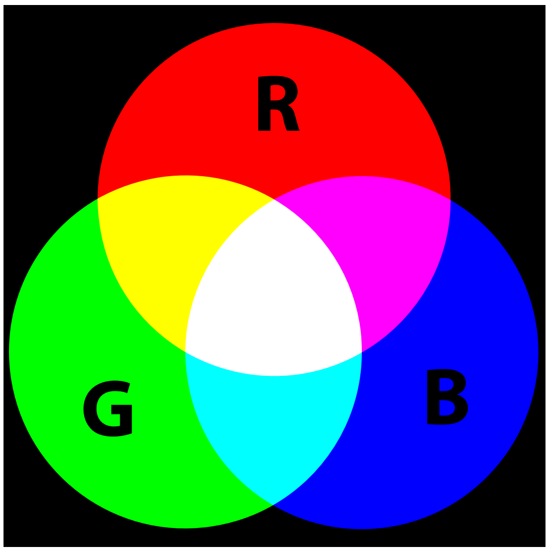


Figure 5.1: Red, Green, and Blue, and the combinations of the three. Each circle represents a maximum value of its color. Note that a mixture of all three colors creates white. (Image source: Wikimedia Commons repository)

RGB

RGB is the most common color model for use with cameras and displays. It is an additive model where red, green and blue light are added together. In an RGB-image, each pixel has three channels: R, G, and B. Each of the channels contains a number representing its respective color. By adding these channels together, new colors are made (Figure 5.1). By adjusting the value of each channel, a wide array of colors can be represented.

HSV

HSV stands for hue, saturation, and value. It is a cylindrical-coordinate representation of points in the RGB color model. Just like RGB, HSV is a three channel representation of each pixel. However, the channels represents different aspects of the image, as shown in Figure 5.2.

The hue channel is the angle around the central vertical axis and decides the color. It starts with red at 0°, passing through green at 120°, blue at 240°, and wraps back to red at 360°.

The saturation is the distance from the vertical axis of the cylinder. It represents the color intensity. A high saturation value gives a strong color, while a saturation of zero gives a grayscale pixel.

The value is the position along the vertical axis. A low value gives a dark pixel, while a high value gives a bright pixel. A value of zero corresponds to black, no matter the

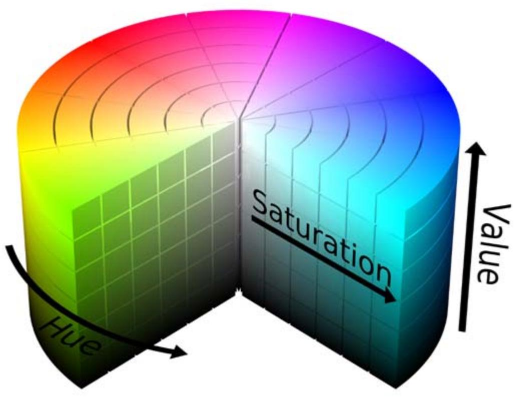


Figure 5.2: The HSV color space. Hue is the color, saturation the intensity of the color, and value represents the brightness. (Image source: SharkD, Wikimedia Commons repository)

value of the other channels. Note however that to create white, the value has to be at maximum and the saturation at zero.

Image channels

The total of six channels in the RGB and HSV color model all represent different aspects of an image. When studying the video feed, it will be an advantage to study each of these channels separately. Some of them will show the transition from water to net more clear than others. This can be demonstrated by an example.

Erik Høy at SINTEF Fisheries and Aquaculture has provided a few sample images of a cage net in water, for experimental purposes. Figure 5.3 is a patch of such an image. This patch has been split into all six channels of the RGB and HSV color model in Figure 5.4. In this example, the red layer (a) shows a clear transition from background to the net. The background has a red value close to zero, while the net has a high red value. The saturation channel (e) also gives a clear transition, showing the background as distinct, bright values. The hue channel (d), on the other hand, does not show a clear transition from background to net. Parts of the net has a hue value close to that of the background. Separating the net from the background would therefore be difficult based on this layer alone. Each layer has been inspected for all the example images, and the red layer and saturation layer shows the best overall performance with these images. However, the layers may show different performance if the circumstances are changed.

Thresholding

In the example above, the foreground can be separated from the rest of the image by using a threshold on the red layer. First the image should be smoothened to remove

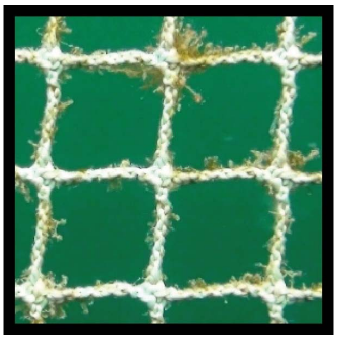


Figure 5.3: Patch of cage net image.

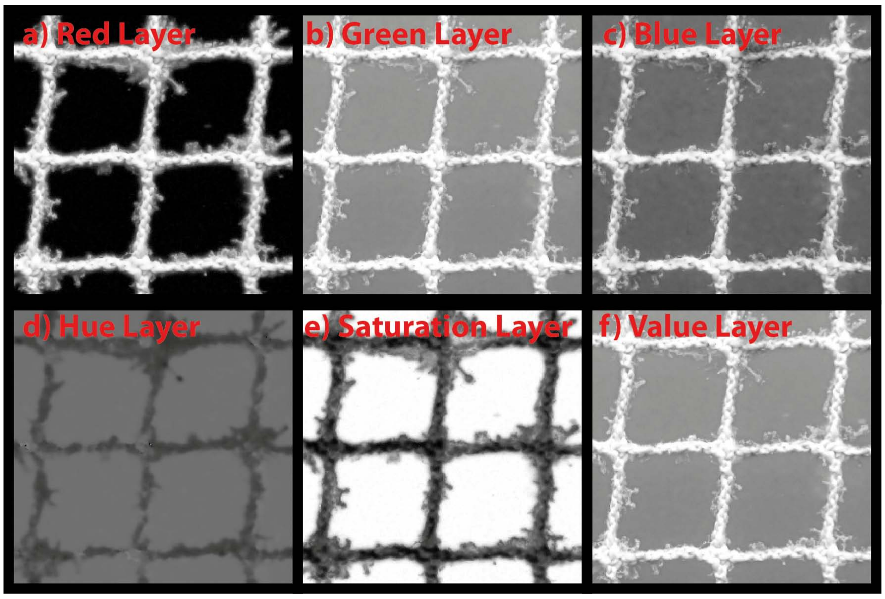


Figure 5.4: The original patch split into the six color channels of the RGB and HSV color model.

high frequency noise. Then, all pixels with a value below a constant, t, are set to zero, while the other pixels are set to maximum. This creates a binary mask which can be combined with the original image to erase the background. The challenge is to set t to an appropriate value. If t is set to low, not all of the background will be filtered away. If, on the other hand, t is set too high, parts of the net will be filtered away. Since the goal is to check for net damage, removing parts of the net at this stage is far worse than leaving some of the background behind. t should therefore be set with a safety margin to avoid removing parts of the net. To help us set t, we can look at the layers histograms.

Histogram

In digital imaging, a histogram is a graphical representation of the tonal distribution in the image. If we plot the histogram of an 8-bit grayscale image, the x-axis represents the tonal values, ranging from zero (black) on the left to 255 (white) on the right. The y-axis plots the number of pixels with each tonal value. Figure 5.5(a) shows the histogram of such an image, the red layer from Figure 5.4(a). This image has a resolution of $280x280$ giving a total of 78400 pixels. As previously noted, the background is very dark in the red layer. This is also reflected in the histogram. Figure 5.5(b) shows the tonal values 0-9 (darkest 10 pixel values) of the histogram. They contain ∼54% of the total pixels in the image. It is fair to assume that these pixels contain most of the background. Figure 5.6 shows binary masks created with different values of t. We can see that setting t to 10 will remove most of the background but leave areas close to the net. This is probably because of blur in the picture. If we increase t, the threshold will start to eat away from the edges of the net. However, it is not before t apporaches 200 that net discontinuities appear in the mask. This example layer is thus very robust with respect to variations in t. Setting t in the range 10-150 will create a binary mask clearly showing the contour of the net.

An algorithm has been developed to threshold out a certain percentage of the pixels in an image by the use of histograms. The algorithm first counts the total number of pixels in the image. It then calculates a histogram and sets t to the tonal value closest to threshold away the desired percentage of the pixels.

In this example we used a global histogram, i.e. we calculated one histogram for the whole image. This posed no problem on the small image patch in the example. In a larger image a global histogram may, however, not be sufficient. Lighting conditions may change along the picture and make the histogram unsuitable for choosing a single threshold, t. A typical example is the sun making a lighting gradient with the top of the image clearly brighter than the bottom. A solution is to divide the image into many sub blocks and using the algorithm described above on each of the sub blocks. Appropriate block size will have to be determined by experimentation.

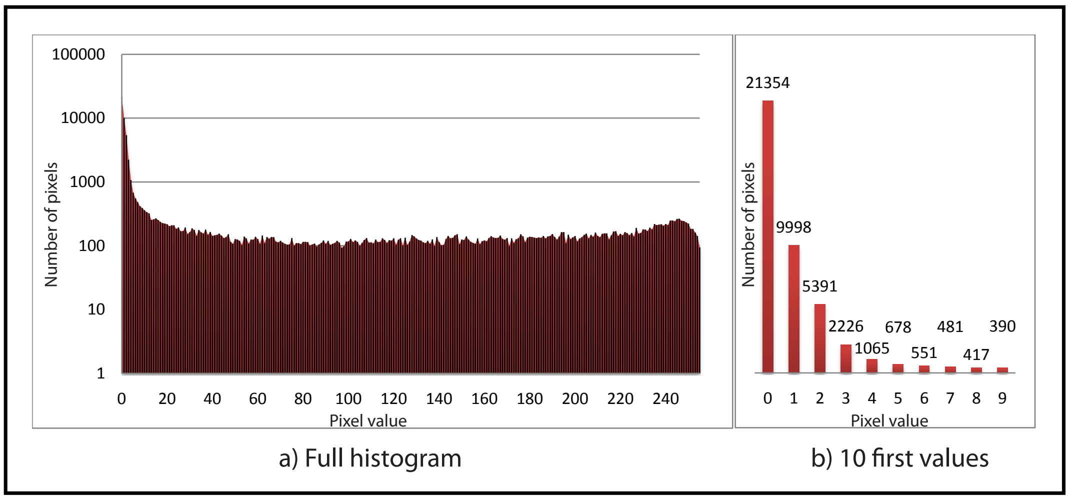


Figure 5.5: Histogram of the previously shown red layer. Total number of pixels in the image are 78400. a) shows the full histogram. Note that the y-axis is logarithmic for better readability. b) shows the first 10 values of the histogram in a linear scale.

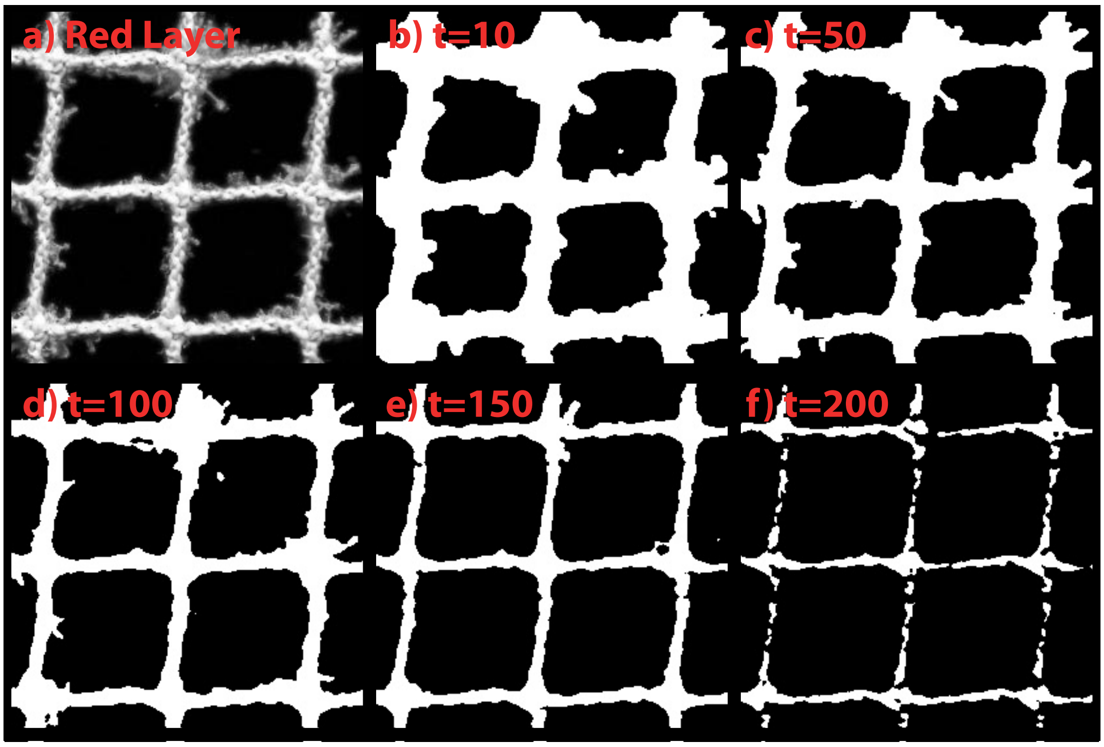


Figure 5.6: Red layer and binary masks created with different threshold (t) values.

Edge Detection

So far, our methods for separating the image has been solely based on the foreground having a different layer brightness from the background. One other possible method for identifying the net in the image is by edge detection. Strong edges will most likely occur at the transition from water to net. The background will hopefully be monotone and not contain to many edges.

Most edge detection algorithms have been made for grayscale images, so we need to convert the images from RGB to grayscale. An alternative is to run edge detection algorithms on each of the color layers separately. The resulting edge maps can then be combined for better results.

We define an edge as a location of rapid intensity variations. If we think of a grayscale image as a height field where bright areas are peaks and dark areas are valleys, edges occur at locations of steep slopes. Mathematically this can be formulated as:

$$\mathbf{J}(\boldsymbol{x}) = \bigtriangledown I(\boldsymbol{x}) = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})(\boldsymbol{x}) \tag{5.1}$$

The gradient vector J points in the direction of the steepest ascent with a magnitude corresponding to slope steepness. Unfortunately, this method accentuates high frequencies and hence amplifies image noise. It is therefore a good idea to smooth the image with a low-pass filter before calculating the gradient. Because of its independence of orientation, the Gaussian filter (Equation 5.2) is often used for this purpose [Szeliski, 2010, p. 239]. σ indicates the width of the filter.

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{5.2}$$

Both the gradient and the Gaussian filter are linear operations and thus commute. We can therefore convolve the image with the horizontal and vertical derivatives of the Gaussian kernel function:

$$\bigtriangledown G_\sigma(\boldsymbol{x}) = (\frac{\partial G_\sigma}{\partial x}, \frac{\partial G_\sigma}{\partial y})(\boldsymbol{x}) = [-x - y]\frac{1}{\sigma^3} e(-\frac{x^2+y^2}{2\sigma^2}) \tag{5.3}$$

$$\mathbf{J}_\sigma(\boldsymbol{x}) = \bigtriangledown[G_\sigma(\boldsymbol{x}) * I(\boldsymbol{x})] = [\bigtriangledown G_\sigma](\boldsymbol{x}) * I(\boldsymbol{x}) \tag{5.4}$$

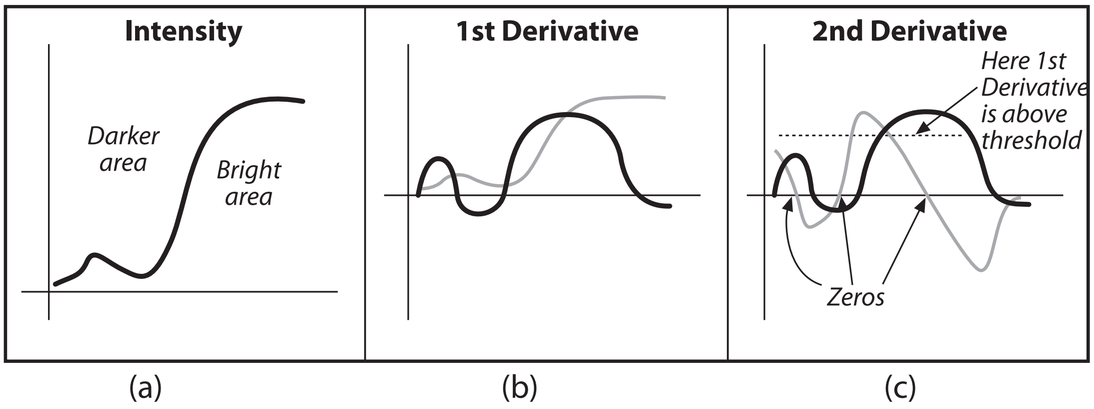


Figure 5.7: (a) A graph of pixel values in a one-dimensional transition from a dark to a bright area. (b) The first derivative (black) of this transition. (c) The second derivative (gray). Its zeros indicate local maxima. Only one maxima is above the threshold and thus corresponds to a "strong" edge. (Image source: Bradski [2008])

5.2.1 Laplacian

We now have a map of slopes on the smoothened image. We are however only interested in isolated edges, i.e. single pixels along the slope contours. These are found as local maxima in the slope map, with directions perpendicular to the edge directions. We find these maxima by taking a directional derivative of the slope map in the direction of the gradient and then identifying its zeros. This directional derivative is equivalent to taking the dot product between a second gradient operator and our slope map:

$$S_\sigma(\boldsymbol{x}) = \bigtriangledown \cdot \mathbf{J}_\sigma(\boldsymbol{x}) = [\bigtriangledown^2 G_\sigma](\boldsymbol{x}) * I(\boldsymbol{x}) \tag{5.5}$$

$S_\sigma(x)$ is called the Laplacian. We now need to find the zeros of the Laplacian. An easy way to do this is to look for adjacent pixels where there is a sign change, i.e. $[S(x_i) > 0]! = [S(x_j) > 0]$.

We now have a map of all the edges in the image. To filter out the most significant edges, we can compare the edge map to the first derivative $\mathbf{J}_\sigma(\boldsymbol{x})$ and choose only edges where the first derivative is above a threshold, as shown in Figure 5.7.

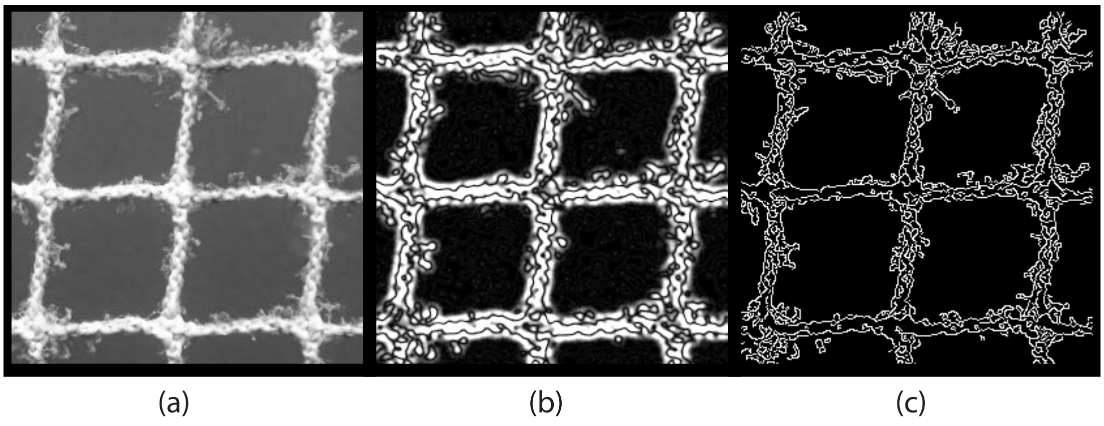


Figure 5.8: a) The original image patch converted to gray-scale. b) Laplacian after smooting with a gaussian filter with $\sigma = 3$ c) Result after using Canny Edge Detector on the same image. Thresholds of $t1 = 75$ and $t2 = 225$ were used.

5.2.2 Canny Edge Detector

This simple method for finding edges in images was further refined by John Canny (Canny [1986]). The Canny Edge Detector computes the first derivatives in x and y direction and then combine them into four directional derivatives. The local maxima of these directional derivatives are marked as "edge-candidates". The Canny algorithm then tries to assemble the edge-candidates into contours. It does this by applying a hysteresis threshold. An upper and a lower threshold are chosen. All candidates above the upper threshold are chosen as edges, while all candidates below the lower threshold are rejected. The candidates in between the two thresholds are chosen only if they are connected to a pixel above the upper threshold. Canny recommended a ratio of high:low threshold between 2:1 and 3:1. Figure 5.8(a) shows a grayscale version of the image patch used earlier. (b) is the result after calculating the Laplacian, and (c) is the result after using the Canny Edge Detector. The Canny-image (the result of using the Canny Edge Detector on the image) shows the net edges in the image much clearer than the Laplacian. To convert this result into a binary mask as we did earlier by thresholding, we can flood the space between adjacent edges. An algorithm was developed for this purpose. In brief, what the algorithm does is as follows:

1. Look for edge-pixels in the Canny-result
2. For each edge pixel, look for neighbor edges in a user-defined radius
3. For each neighbor edge, flood the rectangle made by setting the two edge-pixels as diagonal corners

Figure 5.9 shows the result after using this algorithm on the Canny-image. Not surprisingly, this result is very similar to the results when using a threshold on individual layers. The fact that these two very different approaches lead to similar results can be

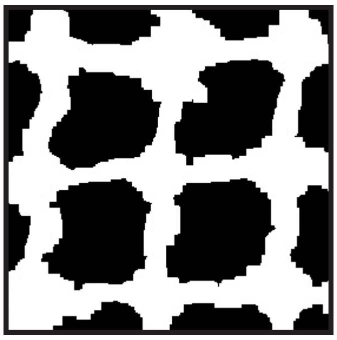


Figure 5.9: The result after using the flooding algorithm on the Canny-image in Figure 5.8(c). A search-radius of 10 pixels was used.

taken advantage of. Since the results are in the same form, we do not have to consider which of the two approaches was used when further processing the image. If one fails, we can switch to the other one without it affecting the rest of the program.

5.2.3 Hough Transform

A cage net is made up of mashes where each mash edge will appear as a straight line in the image. In the ideal case, the edge detectors will identify all of these straight lines. However, due to imperfections in the image and/or the detector, edge detectors are unlikely to identify the whole mesh edges. Instead, the edges will be divided into small edge segments as shown in Figure 5.8(c). If the edge segments are too sparse to show a clear contour of the net, it would be useful to combine these segments into representations of complete lines.

The Hough Transform (Szeliski [2010] pp.251-254) is a well known algorithm for finding lines in an image.[2] First we use an edge detector, e.g. Canny, to pre-process the image. The Hough Transform then lets each edge “vote” for plausible lines in the image. Each point in the pre-processed image (x_0, y_0) (Figure 5.10(a)) votes for a whole family potential lines (Figure 5.10(b)). For computational reasons, the lines are represented in an accumulator array by the polar coordinates (ρ, θ) of the point where the implied line is perpendicular to the origin. The accumulator corresponding to each (ρ, θ) is then incremented. When all edges have been processed, the algorithm choose the lines above a threshold in the accumulator array.

An extension of the Hough Transform is the Progressive Probabilistic Hough Transform (PPHT). This algorithm re-fits the chosen lines to the constituent edges (as opposed to having each line go from one end of the image to the other). It is ‘probabilistic‘ because, instead of accumulating every possible line in the accumulator array, it calculates only

[2]There also exist other variations of the Hough Transform used to find circles or ellipses, but they will not be useful in this project.

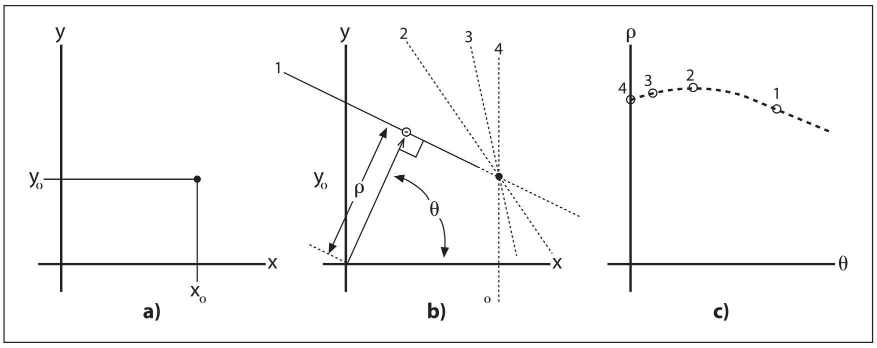


Figure 5.10: Original Hough Transform: A point (x_0, y_0) in the result of an edge detection algorithm (a) can represent many lines parameterized by different ρ and θ (b) which taken together forms a curve (c). (Image source: Bradski [2008])

a fraction of them. The idea is that if a line is strong enough, it will be found even if it is hit only a fraction of the time. This method significantly reduces computation time.

Figure 5.11(a) shows a Canny-image where the edges are sparse in some areas. In (b), the Hough Transform has been used to identify lines in the image. It does a good job at this. A problem with the Hough Transform is that if the threshold is set too low, false lines will be detected. Moreover, all lines go across the whole image. If a net mesh is broken, this algorithm might overlook it, as it does not make sure the whole line is covered with edges. This problem is solved by using PPHT (c). The PPHT algorithm allows for the setting of a minimum and a maximum line length. E.g., if the maximum length is set to half of a mesh length, a broken mesh will not be overlooked. However, the PPHT does not provide a lot of additional information in this example. After testing both the original Hough Transform and the PPHT on a number of example images, the results are not very promising. Better results are attained by using the flooding algorithm on the Canny-image alone. The Hough-algorithm will therefore not be implemented in the final software.

5.3 Detecting meshes

When we have a good binary mask filtering away the background in the image, the next step is identify the individual meshes. This can be done in many ways, but an approach that has proven to be effective is to look for consistent lines across the binary mask. If there are no broken meshes, lines should be detected in both horizontal and vertical directions at approximately regular intervals. If a mesh edge is broken, the corresponding line will not be found. An algorithm was developed in order to find lines in the binary mask through the following steps:

1. Starting at the top-left corner of the binary mask image, search for areas with

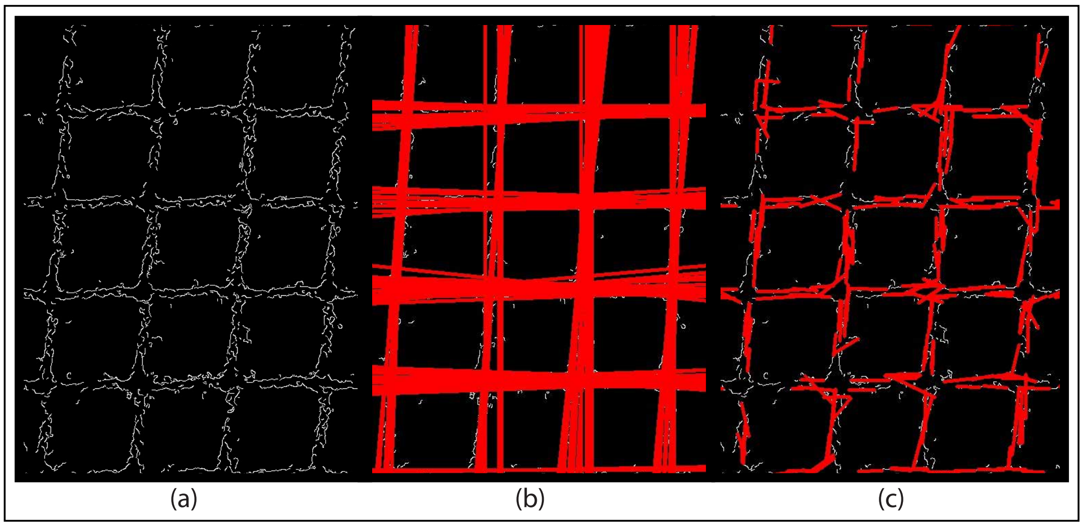


Figure 5.11: a) Result with sparse edges after using the Canny Edge Detector on an image. b) The red lines are the result of using the Hough Transform with a threshold of 150. c) The result on the same image after using PPHT with a threshold of 50.

active (white) pixels along the top/left edge of the image.

2. When an active area is found, do a depth first search (DFS) to see if the pixel is part of a line going vertically/horizontally across the whole binary mask. If a line is found, store its location as an array of pixel points.

3. If a line is found, search for a new area of active pixels and repeat the procedure until the bottom/right edge of the image is reached.

The DFS needs to detect lines even if they are not completely horizontal/vertical. It should also search all possible routes to find consistent lines across the whole image. An iterative function was developed for this purpose. For each iteration it first searches straight down, but then both to the left and to the right if the path is blocked. In this way, lines with an angle up to 45° can be found. The function also marks all dead-end paths discovered in order to save computation time. The result of using this line search algorithm on the example image can be seen in Figure 5.12.

When the lines in the image has been found, the distance between all lines can be calculated. Horizontal and vertical lines should be treated separately. Ideally, an input image of a net with no broken meshes should produce lines of equal distance. However, due to various factors such as imperfections in the net, algae growth, lighting conditions, etc., the lines will show up with small distance variations. The algorithm should therefore tolerate small variations in line distance. If large variations occur (specifically, if two consecutive lines have a line distance much larger than the rest), this should be interpreted as a missing line, and thus, a broken mesh.

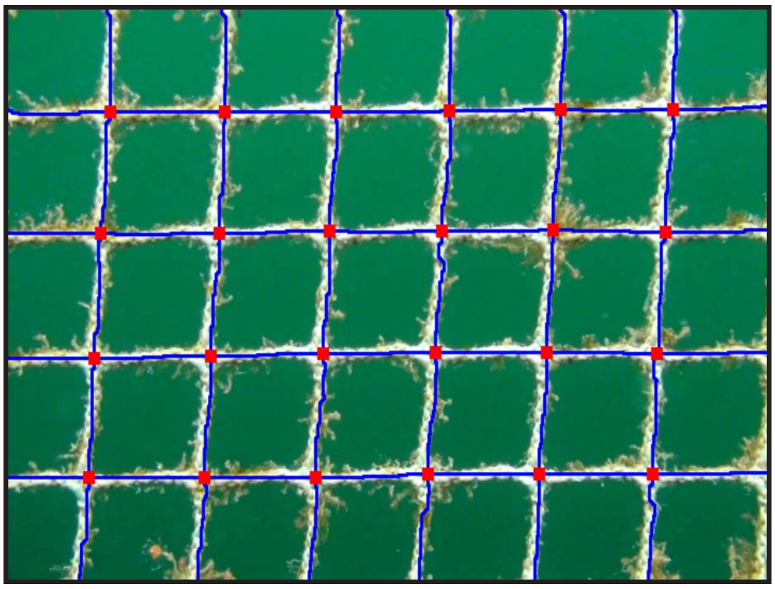


Figure 5.12: The result of the line search algorithm put on top of the image the algorithm was used on. A threshold on the red layer was used to create the binary mask for the algorithm. The red dots mark all crossings of horizontal and vertical lines.

5.4 Detecting Laser Lines

Robust detection of the laser module's laser lines is important in order to provide a reliable distance measurement between the ROV and the fish cage. Since the laser lines will lie in the red part of the visible spectrum, it is fair to assume that they will create very bright areas in the red color layer. If so, we can separate the laser lines from the rest of the image by thresholding on the red layer. When we have a binary mask, calculating the distance between the two lines should pose little difficulty. A simple algorithm has been developed for this:

1. Starting at the left edge of the binary mask, count the number of white pixels in the first column, mark the value as "maximum" and mark the column number.
2. Count the number of white pixels in the second column. If this number is greater than the maximum value, update the maximum value and mark the column.
3. Repeat for each column, until the center column of the image is reached. The left line should now be in the marked column.
4. Do the same for the right half of the image to find the right line.

This algorithm should find the horizontal location of both lines, providing that the laser lines show up in their respective halves of the image. When the lasers are pointed at a cage net, the lines should show up on the binary mask as spots rather than lines. The algorithm should still work, given that the lines are vertical in the image. This will need to be adjusted for by fine positioning the laser generators manually.

When the column number of both columns are located, a simple subtraction gives the pixel distance between the lines.

Chapter 6

Implementation

6.1 Laser Module

6.1.1 Mechanics

The mechanical work on the laser module was done by Per Inge Snildal at the department's mechanical workshop. The front of the housing is made from POM (Polyoxymethylene) plastic, while the cylinder is made from PVC (Polyvinyl chloride). The "windows" for the laser line generators are made from polycarbonate. The module should be able to withstand the pressure at the required 50 meters depth. A photography of the finished module can be seen in Figure 6.1. The ROV with the laser module attached at its belly can be seen in Figure 6.2.

6.1.2 Line distance calibration

As mentioned earlier, the relationship between the distance from the ROV to an object, and the number of pixels between the laser lines on the ROV camera feed should ideally be:

$$d = A \times x^{-1}$$

To calibrate the equation, the ROV was put in front of a white wall so that the laser lines could be easily identified on the ROV camera feed. The number of pixels between the lines were measured at intervals of 25cm, ranging from 25cm to 200cm. The captured camera images contained 720 columns of pixels. The results are listed in Table 6.1. Based on these measurements, a best fit power equation was calculated using Microsoft Excel. The resulting equation is:

$$d = 5173.6 \times x^{-0{,}996}$$

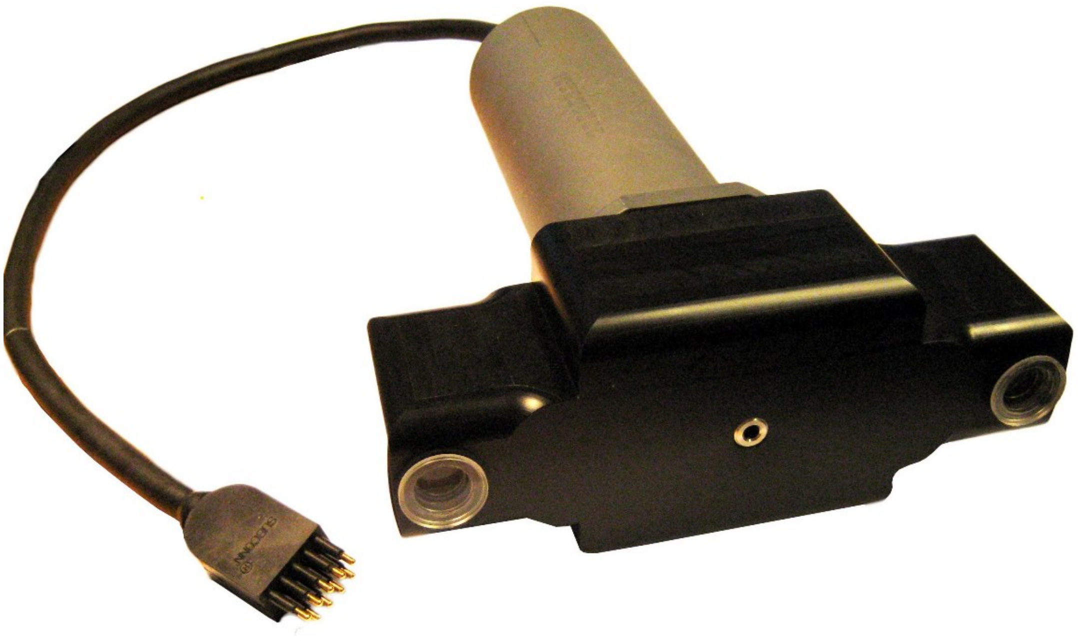


Figure 6.1: Finished Laser Module

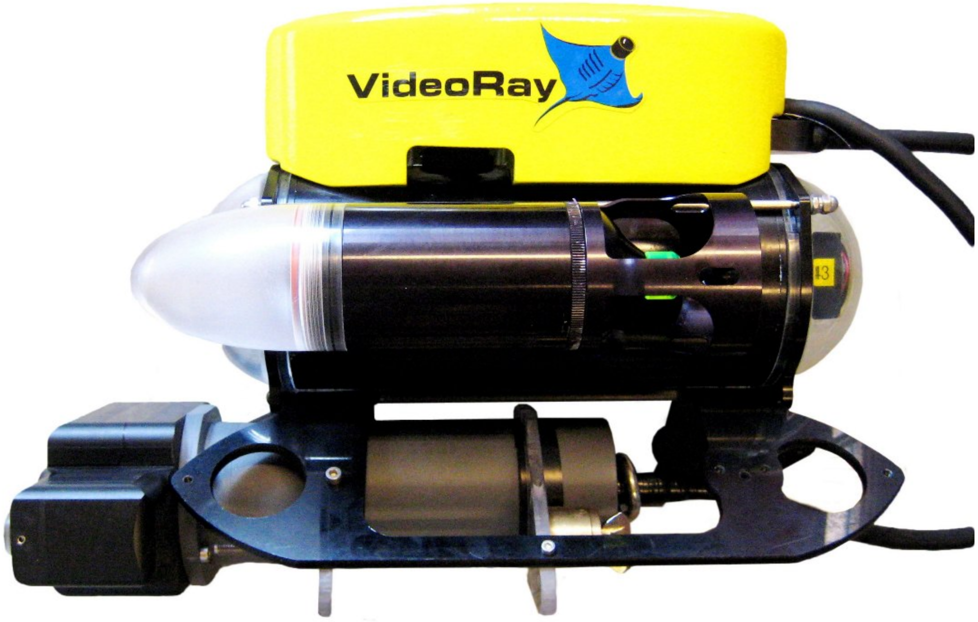


Figure 6.2: Laser module attached to ROV

Distance between ROV and wall	Number of pixels between lines
25 cm	205
50 cm	105
75 cm	73
100 cm	53
125 cm	43
150 cm	36
175 cm	29
200 cm	26

Table 6.1: Measurements when calibrating laser module

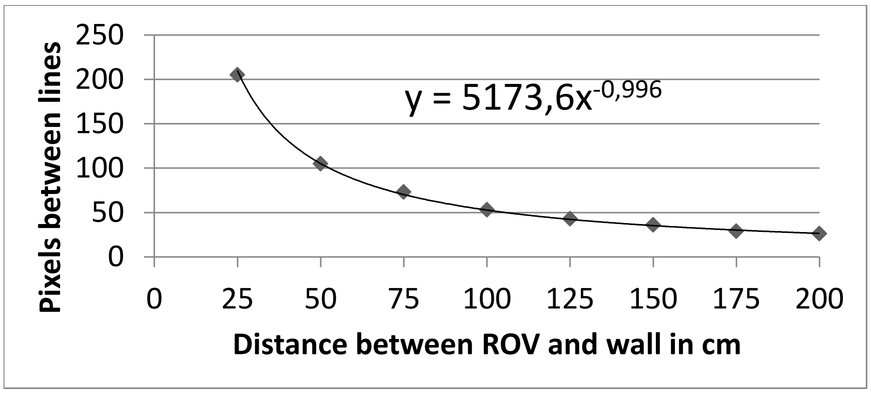


Figure 6.3: Test data plotted together with resulting equation

This is very close to the expected equation. The small deviation in the exponent is most likely due to inaccuracies in the measurements. The “fish eye” effect on the camera was expected to make close measurements seem further away, but this is not noticeable in the test data. The reason is probably that even at the closest measured distance, 25cm, the lines are relatively close to the center of the image. The “fish eye” effect is only noticeable close to the edges of the image. A plot of the measurements together with the power equation can be seen in Figure 6.3. As the figure shows, the equation fits the measured data very well.

6.1.3 Electronics

All electronic components used in the project were either already in stock at the institute or purchased over the Internet. Table 6.2 lists the components and their source. Capacitor and resistor values are not included but can be found in the board schematics in Appendix A. The frame grabber and the CAN-adapter can be seen in Figure 6.4.

Component description	Product name	Source
CAN Adapter	Kvaser Leaf SemiPro HS	in stock
CAN Transceiver	Microchip MCP2551-I/SN	farnell.com
Capacitors	Various 0805 SMD	farnell.com
Compass	OceanServer Technology OS4000-T	in stock
DC-DC Converter	Traco Power TES 2N-1211	farnell.com
Frame Grabber	TerraTec Grabby	in stock
Instrumentation Amplifier	Texas Instruments INA155UA	farnell.com
Internal Cable Connectors	JST (JAPAN SOLDERLESS TERMINALS)	farnell.com
Laser Line Generators	2 x Diode Laser Concepts 211322-0007	mouser.com
MCU	Atmel AT90CAN128	in stock
Oscillator	Euroquartz ceramic crystal 16.000MHZ	farnell.com
Pressure Transducer	Honeywell S&C 19C100PA4K	in stock
Resistors	Various 0805 SMD	farnell.com
SubConn tether connector	SubConn MCLPIL9M	macartney.com
Tactile Switch	TE CONNECTIVITY - FSM4JH	farnell.com
Tether Connector	Phoenix Contact MKDSN1,5/4-5.08	in stock
Transistor	2 x NXP - BC817	farnell.com

Table 6.2: List of electronic components used.

Figure 6.4: Left: TerraTec Grabby, Right: Kvaser Leaf SemiPro HS.

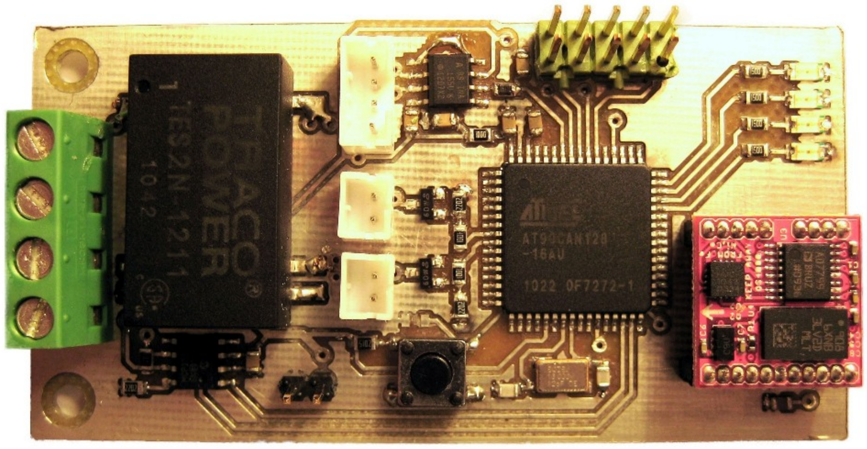


Figure 6.5: Finished circuit board with components soldered on. The compass is detachable.

Circuit board

The circuit board was manufactured with the departments new PCB plotter. The complete circuit board with components attached can be seen in Figure 6.5. The pressure transducer and laser generators can be connected by the white connectors at the center of the board.

6.1.4 Firmware

The firmware was developed in Atmel AVR Studio 4 with GNU GCC compilers and WinAVR libraries. The programming language used is “C”. Programming of the MCU and debugging were performed with the embedded JTAG interface using an Atmel JTAGICE mkII.

A pre-made CAN-driver, developed by Infidigm, was used. It is an interrupt based driver which provides a simple interface to the built-in CAN-controller. The sourcecode is found at: www.infidigm.net/projects/avrdrivers/.

6.2 Software

The software was developed using Qt Creator 2.0.1 together with the Qt 4.7.0 (the Qt framework). Several external libraries were also used to extend Qt’s functionality:

- Qwt 6.0.0: A variety of graphic elements made for Qt. Used to create the compass.
- QextSerialPort 1.2a: A serial port interface made for Qt. Used to control the ROV.

- OpenCV 2.2.0: A vast open-source library for image processing. Used to analyze the video feed.
- Kvaser CANlib SDK 4.2: Library to interface the CAN-adapter.

A program for use with the ROV was made by Carlsen [2010]. His code has been an excellent foundation for this software. The code for CAN- and serial-communication has been re-used with only slight modifications.

6.2.1 Analysis

Image analysis was developed using the OpenCV image analysis library. Version 2.2.0 of the library was primarily used. However, for unknown reasons, this version of OpenCV was unable to communicate correctly with the TerraTec Grabby frame grabber. After weeks of troubleshooting a solution was found. The library-file responsible for communication with I/O devices, “Highgui”, was replaced with the same file from the older version OpenCV 2.1.0, in the part of the code that access the frame grabber. With this older library-file, the software was able to access the frame grabber without any problems.

6.2.2 GUI

The program GUI can be seen in Figure 6.6. The GUI has been divided into several group boxes, each explained below, starting from left to right, top to bottom:

Video Feed shows the video feed chosen in real-time. The frame rate is set to 10 fps. When no video feed is running, the section is black.

Video Control lets the user start video feed from the ROV, an AVI-file on the computer, or an image file. If an image file is chosen, the image will be repeated at 10 fps, thus simulating a real video feed. Each button can be pressed when a feed is running in order to stop the feed. The user can also record the video feed or save frame captures from either the video feed or the analysis on the right.

Full Screen lets the user view either the video feed or the analysis in full screen. In order to exit full screen mode, the user simply clicks a mouse button.

Analysis shows the analysis of the video feed in real time. The analysis is updated with each frame and turns black if no video feed is running.

View Analysis Layers contains a series of check boxes to control what is shown on the “Analysis” screen. Each color layer can be viewed separately, the threshold on the color layers, Canny analysis, line search, or laser line search. A combination of the layers can also be viewed by checking two or more boxes.

Laser Module Control lets the user enable/disable the laser module's on-board laser generators. Status LEDs on the laser module PCB can also be enabled for debugging. They are by default switched off to conserve power.

ROV-control is used to control the ROV thruster system. The user should push "COM-connect" within five seconds of powering up the ROV-system in order to enable COM-communication. The user can also enable/disable the PI-controller, tell the ROV to maintain its current position, or configure controller parameters.

Depth, Azimuth, and Distance are graphic displays of the sensor readings. The scale of the depth bar is progressive and automatically adjusts to the current depth. The range of the distance bar has a fixed maximum value of 2 meters, as distance measurements work very poorly at larger distances.

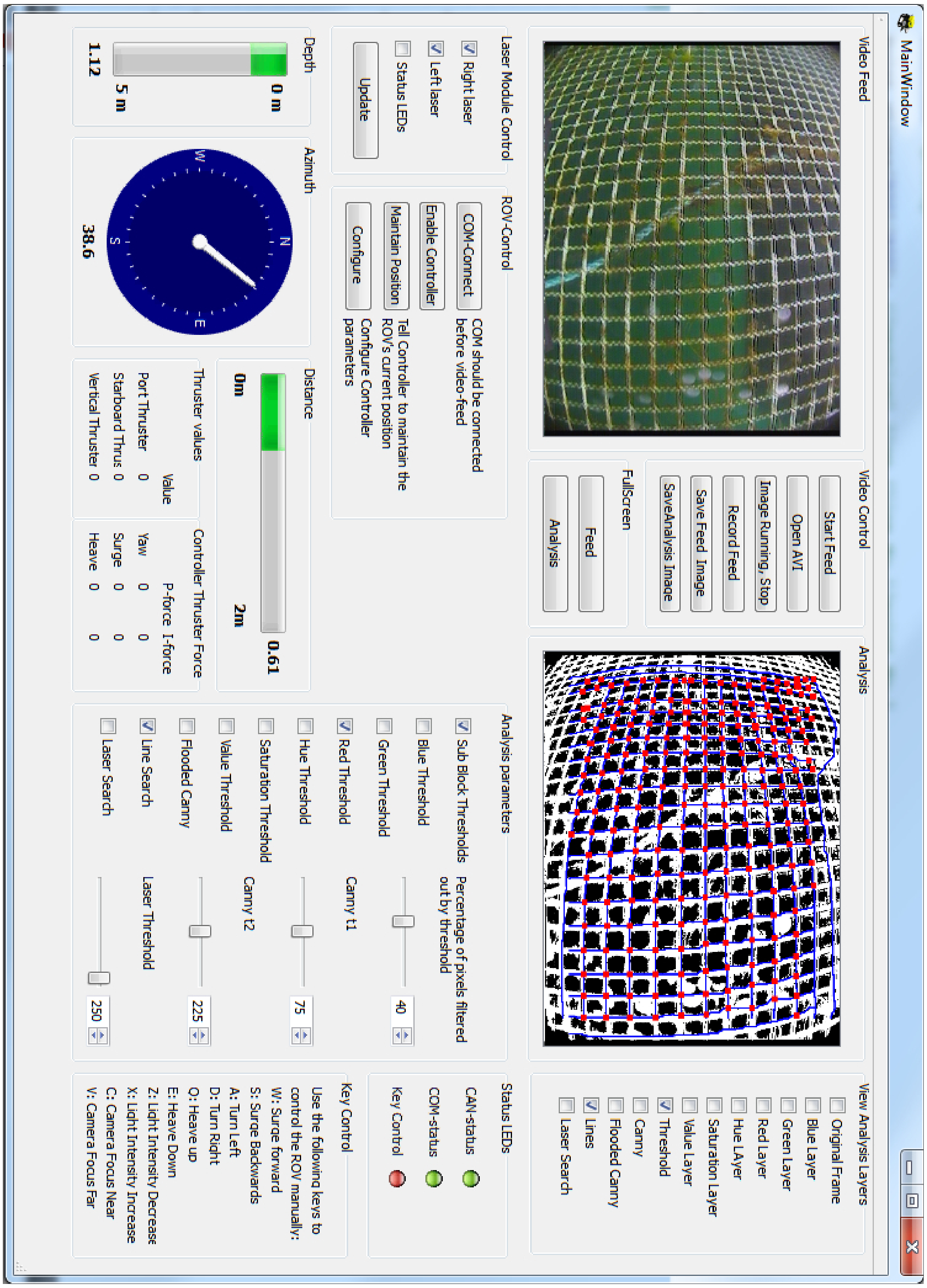
Thruster Values shows the current value of each thruster. The values are based on the last control command sent from the software to the ROV.

Controller Thruster Force is only active when the PI-controllers are running. It shows how much each part of the PI-equations contribute to the total thruster values. This can be very useful when calibrating the controller parameters for new environments.

Analysis Parameters controls the analysis performed on each frame from the video feed. The check boxes control which types of analysis that should be performed while the sliders are values used in the analysis. The values are updated in real time, so that the user can see the effect of changing slider positions immediately. The analysis has been implemented in the same way as discussed in the chapter "Computer Vision".

Status LEDs indicates whether communication with the CAN-bus and the serial interface to the ROV (COM) is active. It also indicates if the ROV is controlled from the keyboard. This last function can be useful because key control will only work if the program window is in focus. Green means active and red means inactive.

Key Control is a summary of how to use the keyboard to control the ROV. COM-connection has to be active in order for key control to work.



COM should be connected before video-feed
Tell Controller to maintain the ROV's current position
Configure Controller parameters

Laser Module Control: [✓ Right laser] [✓ Left laser] Status LEDs

Analysis parameters: [✓ Sub Block Thresholds] Blue Threshold, Green Threshold, [✓ Red Threshold] Hue Threshold, Saturation Threshold, Value Threshold, Flooded Canny, [✓ Line Search] Laser Search

Percentage of pixels filtered out by threshold: 40
Canny t1: 75
Canny t2: 225
Laser Threshold: 250

View Analysis Layers: Original Frame, Blue Layer, Green Layer, Red Layer, Hue LAyer, Saturation Layer, Value Layer, [✓ Threshold] Canny, Flooded Canny, [✓ Lines] Laser Search

	Value
Port Thruster	0
Starboard Thrus	0
Vertical Thruster	0

	P-force	I-force
Yaw	0	0
Surge	0	0
Heave	0	0

Use the following keys to control the ROV manually:
W: Surge forward
S: Surge Backwards
A: Turn Left
D: Turn Right
Q: Heave up
E: Heave Down
Z: Light Intensity Decrease
X: Light Intensity Increase
C: Camera Focus Near
V: Camera Focus Far

Figure 6.6: Software GUI

Chapter 7

Test Setup and Results

Testing was performed at two different locations. Initial tests were performed over several days in a water tank. A final test was then performed at a real fish farm, thus providing an excellent opportunity to test all aspects of the system in a realistic environment.

7.1 Initial tests

Initial tests were performed at a water tank on campus normally used for acoustics experiments. The tank dimensions are 2 by 3 meters with a depth of 1.7 meters. A piece of authentic fish cage net was provided by Per Rundtop at SINTEF Fisheries and Aquaculture. It was attached to a plastic frame to simulate a fish cage. This provided a good foundation for initial testing.

7.1.1 Sensor tests

The first tests performed were of the sensors: depth, azimuth, and distance. The depth sensor was tested in the water tank, while the two other sensors were tested outside of the tank because of practical considerations.

Depth sensor

The depth sensor was tested by standing over the tank with a measuring band, lowering the measuring band to various depths. The ROV was then lowered so that the top of the ROV was at approximately the same depth as the end of the measuring band. The measured depth was compared to that calculated by the software. As expected, a depth bias was present because of atmospheric pressure. This bias was measured to an ADC-reading of 129 (out of 1024). This value was then implemented in the software to be subtracted from all ADC-readings. After adjusting for this bias, the depth sensor

Real distance	Distance calculated by software	Deviation
31 cm	31 cm	0 cm
48 cm	49 cm	1 cm
72 cm	70 cm	-2 cm
104 cm	103 cm	-1 cm
165 cm	168 cm	3 cm
180 cm	184 cm	4 cm

Table 7.1: Testing of laser sensor

performed very well. Exact measurements with a measuring band was difficult, but all readings were within ±5cm of the measured depth. Because of tank limitations, no measurements were made at depths greater than 1.6 meters.

Compass

The compass was tested by placing a standard handheld liquid-filled compass on top of the ROV. The ROV was then rotated to several different compass headings while comparing the azimuth readings from the software GUI to that of the handheld compass. The compass in the laser module has a much higher precision than the handheld compass which had to be read by visual estimates, so small deviations were hard to measure. However, no deviations between the two compasses could be found during this test.

An observation made was that the compass heading changed when moving the ROV around the room while holding it at a constant direction compared to the room itself. Still no deviations were observed between the the compass heading of the laser module and the liquid compass. Apparently, magnetic disturbances were present, affecting the two compasses equally.

Distance sensor

Distance measurements were tested by placing the ROV in front of a white wall at more or less random distances, all less than 2 meters. Each distance was measured and compared to the distance read from the software GUI. The results can be found in Table 7.1.

The software had no problems identifying the laser lines on a white wall. However, detecting the lines on the net frame proved to be more challenging. The net frame was put in the tank and the ROV placed in front of it. Instead of detecting line segments on the net, the software would detect the lines on the white tank wall behind the frame. These lines showed up a lot more clearly on the camera feed than the small segments on the net itself. A black refuse bag was taped onto the frame, behind the net, to reduce this effect. The lines were a lot more dim on the refuse bag than on the white tank wall. Still, mostly due to light reflections from the refuse bag, the software had problems detecting the line segments on the net. Since the problem would not occur in a realistic scenario, this test was postponed untill the trip to the fish farm.

7.1.2 Thruster control

The next test performed was of the PI-controller developed to control yaw (azimuth), heave (depth), and surge (distance). The white walls of the tank provided excellent conditions for distance measurement as the laser lines showed up very clearly in the video feed. Calibration of the controller was done by using Ziegler-Nichols method (Balchen et al. [2003] pp.334-335). The results were not optimal, and improvements were made by manual tuning. Each regulator was calibrated separately and then tested together with the other regulators for fine tuning.

The heave regulator was the easiest to calibrate by far. The laser module, with its low pass filter design, provided accurate and stable depth readings with little noise. The ROV is normally fine-tuned by weights to have a small positive buoyancy before dives. This is a safety precaution, so that if the tether cable for some reason should be disconnected or snap during dives, the ROV will surface. The positive buoyancy provides a large but stable disturbance for the depth controller. The integrator had no problem dealing with this and made the ROV stable within ± 1 ADC-reading (± 6.86cm) of the desired depth. The result of the calibration was a very aggressive yet surprisingly stable heave controller.

Calibrating the controllers for yaw and surge were a lot more complicated. The controllers clearly affected each others behavior, thus making it difficult to get both controllers to work at the same time. After a lot of manual parameter tuning, both controllers were stable without any steady state bias. The end result were stable but somewhat slow yaw and surge controllers.

A demonstration video of the calibrated controllers is provided with the electronic attachments included with this report. As evident in the video, the heave controller is the most aggressive, reaching a steady state long before the other two controllers.

7.1.3 Mesh analysis

Tests were also performed to check the performance of the mesh analysis algorithms. The refuse bag was taped behind the net during the tests, as this background was thought to be more realistic than a white wall. Maintaining a steady position in front of the net using the software controller, or even by manual control of the thrusters, were next to impossible. Partly because of the relatively narrow frame, but mostly because of the lack of sway capability in the ROV. Because of cable drag and small movements in the water, the ROV would quickly move either to the left or the right of the desired position. This problem was solved by attaching the ROV to a metal rod and manually positioning it in front of the net.

Experiments were performed to check the effects of thresholding based on local histograms, by dividing the images into several sub blocks, as mentioned in the chapter: “Computer Vision”. The optimal number of sub blocks turned out to be 64. If the image

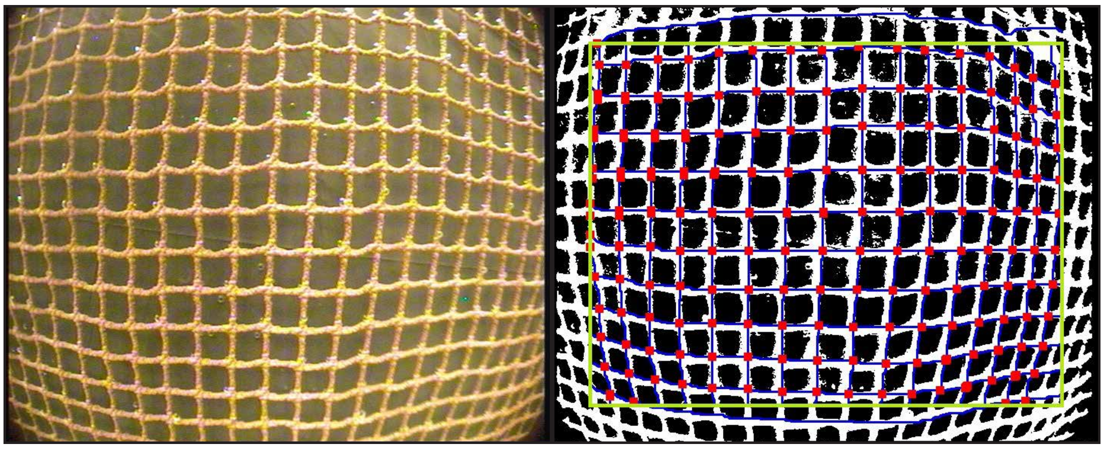


Figure 7.1: Left: Frame grab from mesh analysis test Right: Analysis output for the same frame. A threshold of 56% on the red layer and 64 sub block histograms was used to create the binary mask.

was divided into more than 64 sub blocks, problems occurred when some sub blocks contained little, or no net, and still only a certain percentage of the block was filtered away.

Mesh analysis worked well at distances of 15 to 40 cm. If the ROV was moved further away than this, the resolution of the ROV camera made the algorithms unable to detect each single mesh.

A frame capture from the testing together with the analysis output can be seen in Figure 7.1. The frame dimensions are 720x576 pixels. Due to a combination of fisheye effect and vignetting[1], the algorithms did a bad job at detecting meshes at the edges of the frames. To cope with this, a boarder of 50 pixels was made along the edges of the frames. The software did only initiate line searches within these borders, which have been marked by a green rectangle in Figure 7.1. However, the line searches were not restricted by the boarders which helps explain the horizontal lines stretching outside the top and bottom border line.

7.2 Fish Farm Tests

The tests performed in the water tank were done because of easy access over a long period of time. However, the conditions were not realistic for several reasons:

- There was no water movement in the tank water (other than that created by the

[1]Vignetting is a reduction in image brightness at the periphery compared to the center of the image.

ROV itself). This made the ROV a lot easier to control then what will be the case in the ocean with current and waves.

- The tank walls were white. When analyzing the video feed from the ROV, these white walls shows up very clearly in the video. A fish cage in the ocean will provide an "infinite" body of water as a background.

- The room was artificially lit by fluorescent tubes. This provides a much "cooler" and dimmer light than daylight conditions. This might affect camera performance.

- The tank was only 1.7 meters deep. The effect of depth on sensors and lighting conditions could thus not be tested.

- There were no algae growth on the net used, or any other foreign obstacles in front of the ROV.

Luckily, ACE (AquaCulture Engineering) was kind enough to put one of their fish cages at disposal for testing the system. Testing was performed in the morning of July 1st 2011 in an empty fish cage at Tristein in Bjugn, Sør-Trøndelag. The installation is described as a wave exposed large scale salmon production site. A "walking ring" encircles the fish cage, and the original plan was to place the ICB and TDS on this ring and to deploy the ROV on the outside of the cage. During the tests there were big waves washing over the fish cage, making the planned arrangement unsafe. The ICB and TDS were instead placed in a working boat moored to the fish cage. To protect the tether cable from being pinched in between the boat and the fish cage the ROV was deployed inside the fish cage.

7.2.1 Sensor testing

The first test performed was to check if the distance sensor performed well in a realistic environment. The same test had been inconclusive during initial tests in the water tank. The distance sensor was tested close to the surface, and at depths of approximately 5, 10, and 15 meters. The sensor worked very well at distances of 60 cm and less. When the ROV was moved further away from the net the laser lines became too dim to be effectively distinguished from the rest of the frames. An interesting observation made was that in areas of high algae growth the distance sensor worked at distances larger than the mentioned 60 cm. This is probably due to the effect of algae growth increasing the perceptive diameter of the mesh threads, giving the laser lines a larger area to light up. A video of the distance sensor testing can be found in the electronic attachment included with this report. The video shows the software being able to track the ROV's distance to the net continuously over a longer period of time at distances ranging from 10 to 60 cm, and with the ROV making rapid movements.

7.2.2 Thruster control

The next test was of the position controller. The controller was tested by maneuvering the ROV to a position in front of the cage net and then instructing the controller to hold the current position. The depth control was able to maintain the depth without any problems. The desired depth was changed manually several times, and the ROV moved to the this depth within short time as long as enough tether cable was available.

The two other controllers, azimuth and distance, failed utterly at this task. The controllers were first tested close to the surface. The combination of waves, current, and cable drag, made the ROV quickly drift off the desired compass heading, losing camera view of the cage net and thus losing track of the distance. By maneuvering the ROV to greater depths, the effect of waves was eliminated and the current strongly reduced. Unfortunately, the cable drag increased with depth. The controller parameters had been tuned in a water tank without any notable water movement and the controller was thus not nearly aggressive enough to cope with the increased disturbances. Attempts were made to change the controller parameters and make the controller more aggressive, but a fully functioning controller was not achieved. During the remaining tests the depth controller was enabled while the horizontal thrusters were controlled manually from the laptop keyboard.

7.2.3 Mesh analysis

Mesh analysis was tested by using threshold on all six color layers. The red layer gave the best result by far. Using Canny edge detection was also successful to some degree, but did not give the same performance as thresholding. After a quick calibration it was discovered that thresholding away approximately 40% of the pixels and using 64 sub block histograms gave the best result. As with the previous tests, mesh analysis was tested near the surface and at depths of 5, 10, and 15 meters. The software was able to detect meshes effectively at all these depths. Even in areas with high algae growth, most meshes were detected within the search area. Because of the previously mentioned problems with maintaining ROV position, countinous mesh analysis over longer periods of time were not achieved.

Analysis of single frames, captured when the ROV was in position, worked well at distances up to approximately 60cm. The increased distance compared to the 40cm when testing in the water tank was probably due to a larger mesh size in the cage net on the fish farm.

The first image in Figure 7.2 shows a frame from the mesh analysis tests. The second and third images show a binary mask created using a global histogram and using 64 sub block histograms respectively. The fourth image shows the result of mesh analysis on the third image. The green rectangle marks the search boarder.

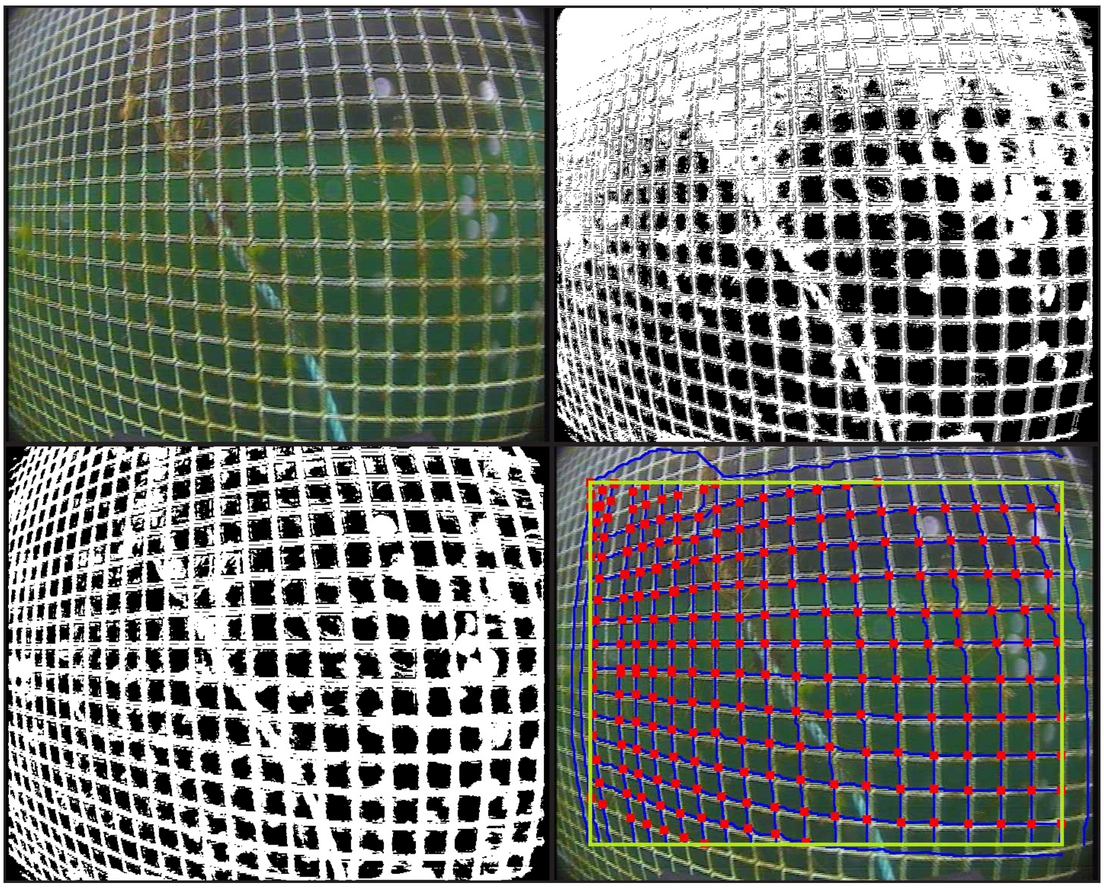


Figure 7.2: Mesh analysis using a threshold of 40% on the red layer. Upper left: Original video feed. Upper right: Binary mask created with global histogram. Lower left: Binary mask created using 64 sub block histograms. Lower right: Result of mesh analysis using the sub block binary mask.

Chapter 8

Discussion

The goal of this project is to test the concept of using an AUV to check for net integrity in fish cages. The following sections will discuss the test results in relation to both the current project and the long-team goal of developing an autonomous AUV to check for net integrity without human interaction.

8.1 Sensor tests

There were no problems with the laser module during any of the tests. Per Inge Snildal at the department's mechanical workshop did an outstanding job at creating a well functioning as well as aesthetically pleasing design for the module housing. The housing worked as expected and did not take in any water during the tests. The electronics also functioned perfectly. An error in any of the components or unstable communication between the laser module and the laptop could easily have lead to frustration during the tests, but luckily this was not experienced.

8.1.1 Depth Sensor

The depth sensor performed very well in all tests. This came as no surprise, as the same sensor had already been used by two previous students with good results. The sensor was used with the same hardware setup and software calculations as by these previous students, so it should theoretically behave in exactly the same way.

The software was calibrated to calculate depth in saline water while depth measurements were made in fresh water. Despite this, no measurable bias was noticed. This is most likely because of inaccurate measurements. Since all measurements were made from the top of the tank using visual estimates, the result of the readings giving small deviations within ±5 cm of the measured depth does in no way indicate sensor inaccuracy. More precise tests are needed to decide the exact accuracy of the sensor.

The tank tests were also performed at relatively shallow depths compared to the sensor and ROV depth range. During the tests at the fish farm, the sensor was used at (indicated) depths of up to 15 meters. These depths were not physically measured but seemed reasonable judging from the camera feed. If nothing else, this shows that the pressure sensor is functioning at greater depths.

Despite not being tested for absolute accuracy, the pressure sensor proved to be more than precise enough for this project. Its main purpose when surveying a fish cage will most likely be to measure depths relative to previously measured depths. An autonomous AUV will probably have to be calibrated before use with a new fish cage by being maneuvered to the bottom of the cage and registering the depth. As such, precise depth measurements will not be as important as consistent measurements and a high depth resolution. The current resolution is 6.86 cm which was sufficient for this project. If higher precision should be required, it can be accomplished at the expense of depth range, which is currently 0 to 58.5 meters.

8.1.2 Compass

Carlsen [2010] reported the compass being very sensitive to magnetic disturbances during indoor use. This was also experienced during initial compass tests in this project. When using the ROV outdoors, magnetic disturbances should be less of a problem as there are less sources of disturbance. The ROV itself is a big source of magnetic disturbance but this effect was dealt with when calibrating the compass during its first use with the laser module. As with the pressure sensor, absolute azimuth is not as important as consistent measurements when surveying a fish cage.

The measured compass heading was never compared to that of a handheld compass during testing on the fish farm. The heading was fairly consistent with regards to maneuvering the ROV to different depths. This indicates that there were little magnetic disturbances, or at least little variation in the disturbances. The working boat could potentially be a big source of magnetic disturbance, but this was not notably affecting the compass.

8.1.3 Distance sensor

The initial tests demonstrated the distance sensor to be very accurate when used on a white wall. This comes as not surprise, as the sensor was calibrated using a similar wall. After the inconclusive testing of the distance sensor on a net in the water tank, it was not certain that the sensor would work on a fish net in a more realistic scenario. However, during the fish farm tests the distance sensor was surprisingly robust with regards to distance, angle between ROV and net, algae growth, and rapid movements. As long as the laser line hit the cage net and the distance did not exceed 60cm, the sensor had no problem continuously keeping track of the distance. The absolute distance was not measured during tests on the fish farm but there is no reason to believe that the sensor was less accurate here then when used on the white wall during initial testing.

The maximum distance of approximately 60cm may be low for future applications, but in theory, this distance can easily be increased by changing the 5mW laser generators for generators of higher output power. The current laser generators each consume approximately 60mW of power. The PCB on the laser module is dimensioned to supply laser generators of up to 135mW each, so the laser generators can be changed to generators operating at twice the strength of the generators currently used as long as they fit in the module housing.

8.2 Thruster Control

All thruster controllers worked fairly well during initial tests in the water tank. Calibration of the surge/yaw controllers were a lot harder than calibrating the heave controller. The performance of these controllers are very dependent on each other because of several factors. Most importantly, they use the same thrusters. The effect of this can be explained by an example. The maximum/minimum value for each thruster is ±220. If the yaw controller gives the output 200,200 (port thruster, starboard thruster) and the surge controller gives the output -100,100, the resulting thruster values will be 100,220, instead of 100,300. In other words, as the controllers reach maximum thrust they are limited by each other. Another important factor is that changing the azimuth at the same time also changes the distance to the object in front of the ROV. This means that an unstable yaw controller will make the surge controller unstable. Time-delay in the system was also clearly effecting the stability of the system, especially on yaw. Cable drag was a big, but constant, disturbance on the yaw controller but did not effect the other controllers notably, as long as enough cable was supplied.

When testing the controller in open ocean at the fish farm, only the heave controller functioned correctly. The other two controllers were completely disabled by waves, current, and most importantly, by cable drag. The controller could have been fine-tuned to a more aggressive behavior in order to compensate for these disturbances, but the result would probably be far from stable. A future AUV will have several advantages over the current ROV when it comes to holding a steady position:

- An AUV has the advantage of being free from any tether cable. Thus, the cable drag will be eliminated completely.
- The ROV used in this project is of the class “micro-ROV”. It is small and has a weight of only 3.8kg. This makes it very vulnerable to both waves and cable drag.

The AUV will be a lot easier to keep stable if its weight is increased from that of the current ROV.

- The shape of the ROV is not optimal for this project. It is torpedo-shaped, making it ideal for surging fast through the water. A more cube-shaped chassis would probably be better for slow, steady maneuvering.
- The lack of sway capability is also a big drawback for the current ROV. It has no way of preventing current from pushing it sideways. One additional thruster on the AUV would cope with this problem.

8.3 Mesh Analysis

Many different methods for separating the net from the background was tried during development and testing, but the simple method of thresholding on the red color layer proved to be a superior technique. The amount of pixels that should be removed to create a good binary mask of the net varied with mesh size, thread diameter, algae growth, and other factors. Still, a rule of thumb is to start at approximately 50% filtering and adjust until a good result is achieved. The optimal result was always found in the interval 40 to 60%.

The use of sub blocks for the histograms turned out to be important in order to achieve good binary masks. An example of this can be seen in Figure 7.2, where the same threshold is used with and without the use of sub blocks. The use of sub blocks clearly creates a much more even distribution of the pixels in the image. Since the net is evenly distributed in the image, unless there are large obstacles in the way, this is a desirable effect.

Figure 7.2 also contains foreign objects. A rope goes diagonally through the image. There are also some air bubbles on the outside of the camera dome, creating bright spots in the image. The thresholding algorithm is not able to filter away any of these foreign objects. Still, the line search algorithms is able to find all horizontal lines in the search area, and all but one vertical line. The vertical line missed is on the left side of the search area. The ROV camera was not pointing straight towards the net when the image was grabbed, and meshes in the left part of the image are very close. To avoid detecting multiple lines from the same mesh thread, the algorithm makes sure there are at least 15 pixels between each line. This is a probable cause of the missed vertical line. As such, this effect would not be a problem if the ROV was pointed straight towards the cage net.

The search boarder was used because of the fisheye effect and the vignetting in the corners of the frames. Searching for lines without this boarder gave unreliable results. In a future AUV, a camera without these drawbacks would allow more net to be analyzed at one time. A camera with higher resolution and a better lens would allow the analysis to be performed at greater distances than the current maximum of approximately 60cm, thus covering even more net.

Chapter 9

Conclusion

The PRO3S was not made for high precision maneuvering in open ocean. Still, it has been a good tool for testing some of the key concepts of a future autonomous AUV for use in aquaculture. When constructing such an AUV, one can learn from the drawbacks of the PRO3S in order to make the AUV succeed where the ROV failed.

The laser module as a whole was a success. It has shown to be reliable and consistently capable of giving accurate measurements, both in fresh and in saline water. The results of the distance sensor were particularly impressive. This sensor was developed entirely from scratch for the specific purpose of measuring the distance to a fish cage and performed exceptionally well at this task. The distance measurements were not completely accurate but the sensor was very robust with regards to environment changes.

The concept of using a video camera to check for net integrity has proven largely promising. Tests at the fish farm also demonstrated the importance of a steady video feed aimed directly at the cage net. This will unquestionably be a lot easier to achieve with a future AUV. More work is needed in order to make a complete system detecting and logging net damage. The already existing algorithms can also be improved on. The software had no problems running on a 2011 model laptop computer, but more efficient algorithms may be needed for use with integrated hardware.

An autonomous AUV for use in aquaculture is still far away, but hopefully this project will work as a good foundation upon which to build further research. By using the same ROV, a new master thesis can most certainly be written on the same subject. However, at some point the ROV's weaknesses will create the demand for a more specialized vessel in order to facilitate further research.

Chapter 10

Further Work

Hopefully, the findings of this project will be a contribution towards the development of an AUV for future use in aquaculture. In the following sections, ideas and suggestions for the continued work towards the AUV will be discussed.

10.1 AUV construction

As mentioned earlier, the ROV is not optimal for cage net inspection. An AUV will be completely free from any tether cables disturbing movements and as such, it will have a natural advantage over the ROV. Considerations that could be made when constructing the AUV include the following:

- The ROV's small weight makes it very unstable with regards to waves and water movements. Making a bigger and heavier AUV would reduce this problem.

- The ROV is torpedo shaped, made for surging fast through the water. An AUV for use in aquaculture would probably not need to travel at high velocity. The hull should instead be made with primary focus on stability and maneuverability.

- An extra thruster, providing sway capability, would give the AUV an important advantage in being able to both move sideways and maintain its position in areas with sideways currents.

- A better camera with higher resolution would allow for analysis to be performed at greater distances. Using a digital camera would remove image noise from the feed.

10.2 Laser Module

The laser module met all expectations during use in this project. The module could easily be used in a future AUV without any modifications. The concept of using two laser lines to measure distance to a cage net has been successful. If more research in the area of rangefinding is desirable, an alternative technology to look into is the use of sound waves, e.g. sonar. However, there seems to be no need to move away from the current rangefinding concept. If a new module is made, some suggestions for improvements that could be included are the following:

- Laser generators with higher power output than 5mW could be used to increase the maximum working distance of the distance sensor.
- The distance between the laser generators could also be increased in order to achieve higher precision.
- More sensors could be included, e.g. a GPS system for deciding position when the AUV is at the water surface.

10.3 Software

The next logical step in development of the software would be to implement damage detection. Damage detection was discussed briefly at the end of section 5.3, but never implemented in the software. The reason for this is that there were no footage with damage present to test such an algorithm. The software records the position of all horizontal and vertical lines found, and so, in theory, the implementation of a a damage detection algorithm should be rather simple.

A broken mesh would lead to a missing line at the breakage point. By calculating the distance between all lines, horizontally and vertically, a broken mesh could be detected by looking for large deviations in line distance. Alternatively, the software could be calibrated before each search. By keeping a constant distance to the cage net, the distance between all lines should also be constant. If deviations occur, this would indicate mesh damage or possibly larger holes. Of course, the deviation could also be due to disturbances in the video feed. The software should therefore only log net damage if several consecutive frames indicate damage in the same spot.

A search pattern should also be developed. For an AUV working with a circular cage, the easiest pattern would be to circle the cage, starting at the top. The AUV would keep a constant distance by use of the laser distance sensor and move sideways. An algorithm would also be needed to make sure the AUV was always pointed straight towards the cage, for example by looking at the distance between meshes in the video feed. If the AUV points slightly to one side, the meshes will appear closer on that side. The compass heading would indicate when the AUV had made a complete circle around the cage. The AUV could then move a little deeper, and do another circle. This would be repeated until the AUV reached the cage floor.

If the software at any point detects net damage, the AUV should stop its search pattern, maintaining its position until the software confirms the damage. The depth and compass heading should then be stored together with video feed of the damage area. Compass heading together with depth would indicate the approximate spot of damage for a circular cage.

For a square of rectangular cage, a similar search pattern could be used. The AUV would know from the video feed when it hits corners, turn 90° and continue travelling sideways.

The reason for searching horizontally instead of vertically is that vertical position is easily measured by the depth sensor. Horizontal position is much harder to track. By moving horizontally around the whole cage, knowing the exact horizontal position at all times is not necessary.

Bibliography

M.-C. Amann, T. Bosch, M. Lescure, R. Myllylä, and M. Rioux. Laser ranging: a critical review of usual techniques for distance measurement. *Optical Engineering*, 40:10–19, 2001.

Arbeidstilsynet. Arbeidsmiljø og sikkerhet i havbruk. Website, 2011. `http://www.norsksertifisering.no/PDF/Havbruk%20regelverk.pdf`.

J. G. Balchen. Possible roles of remotely operated underwater vehicles (rov) and robotics in mariculture of the future. *Modeling, identification and control*, 12:207–217, 1991.

J. G. Balchen, T. Andresen, and B. Foss. *Reguleringsteknikk.* Institutt for teknisk kybernetikk, NTNU, fifth edition, 2003.

A. Bradski, G. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library.* O'Reilly Media, first edition, 2008.

J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

A. Carlsen. Navigational assistance for mini-rov. MSc. Thesis, Norwegian University of Science and Technology, 2010.

R. Dorsch, G. Häusler, and J. M. Herrmann. Laser triangulation: fundamental uncertainty in distance measurement. *Applied Optics*, 33:1306–1314, 1994.

FAO. The state of world fisheries and aquaculture. Technical report, Fisheries and Aquaculture Department, Food and Agriculture Organization of the United Nations, 2006. `http://www.fao.org/docrep/009/A0699e/A0699e00.htm`.

A. R. Frost, A. P. McMaster, K. G. Saunders, and S. R. Lee. The development of a remotely operated vehicle (rov) for aquaculture. *Aquacultural Engineering*, 15:461–483, 1996.

O. Jensen, T. Dempster, E. Thorstad, I. Uglem, and A. Fredheim. Escapes of fishes from norwegian sea-cage aquaculture: causes, consequences and prevention. *Aquaculture Enviroment Interactions*, 1:71–83, 2010.

R. A. Klepaker, K. Vestgård, J. Hallset, and J. G. Balchen. The application of a free-swimming rov in aquaculture. *Modeling, identification and control*, 8:19–26, 1987.

H. Kondo, K. Nakane, E. Shimizy, E. Shimizu, J.-K. Choi, K. Nagahashi, M. Matsushima, Y. Nishida, T. Arismoto, Y. Miyamoto, K. Amakasu, M. Endo, and R. Matsui. Design and concept of a biointeractive autonomous underwater vehicle (ba-1). *OCEANS 2010 IEEE - Sydney*, pages 1–7, 2010.

J. W. Nilsson and S. A. Riedel. *Electric Circuits*. Prentice Hall, seventh edition, 2005.

K. Pazul. Controller area network (can) basics. 2002.

J. Skjaeveland. Utvikling av akustisk målemetode for oppdrettsmerders form. MSc. Thesis, Norwegian University of Science and Technology, 2009.

R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, first edition, 2010.

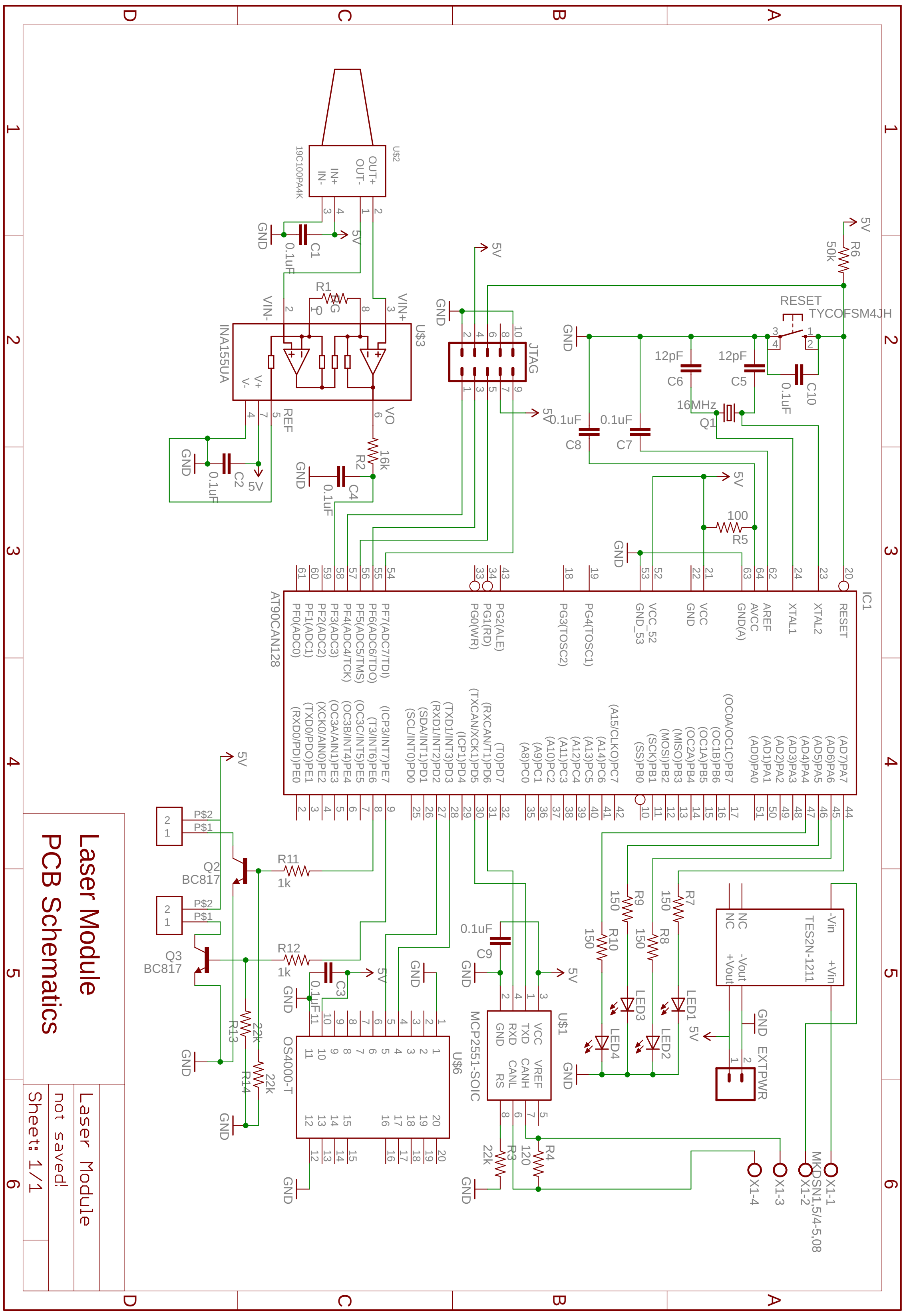
Videoray. Pro 3 s specifications. Website. `http://videoray.com/products/12-pro-3-s`.

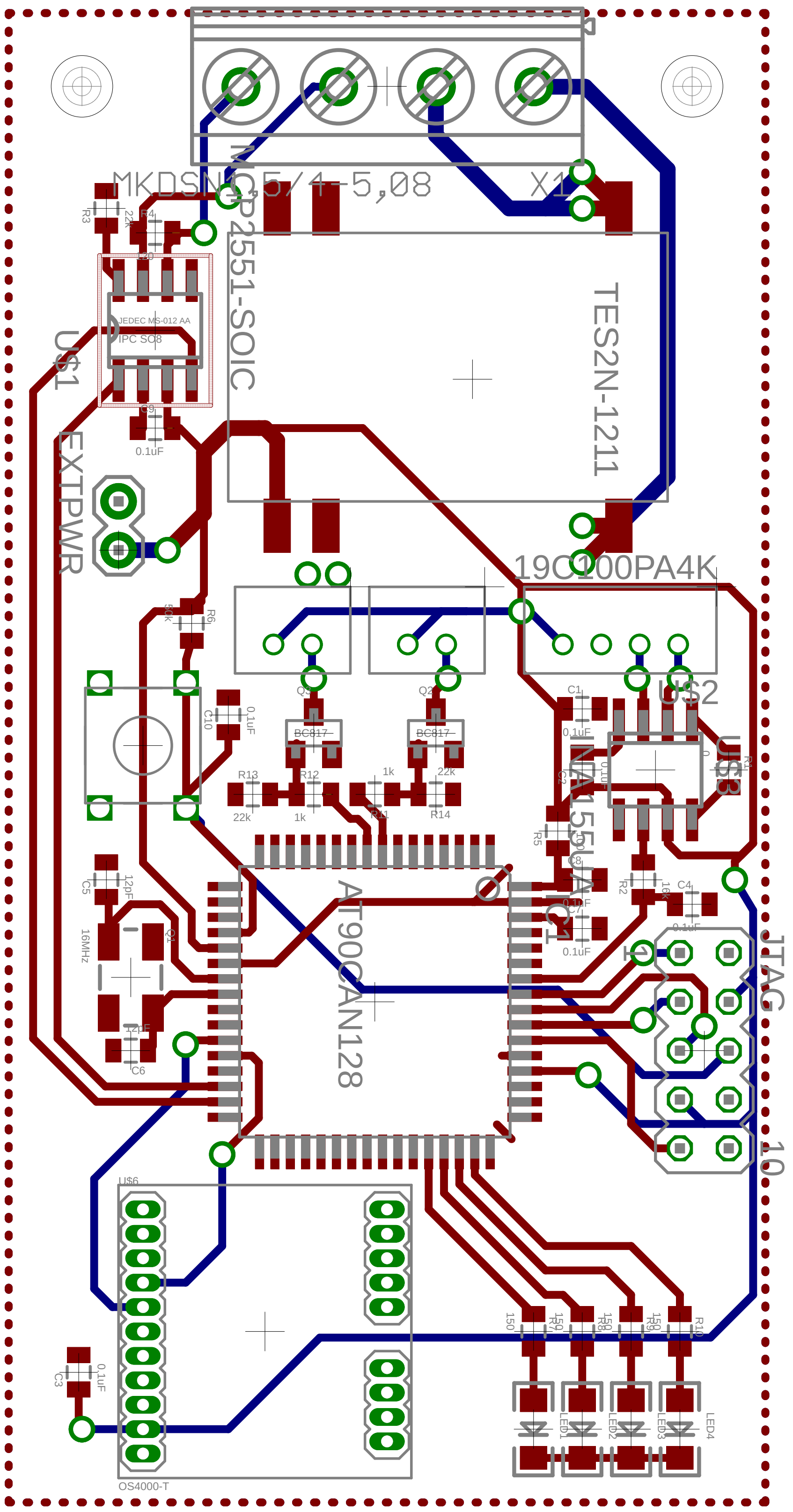
F. M. White. *Fluid Mechanics*. McGraw-Hill, fourth edition, 1998.

B. Worm. Impacts of biodiversity loss on ocean ecosystem services. *Science Magazine*, 314:787–790, 2006.

Appendix A

PCB Schematic and Board Design


Laser Module
PCB Schematics
Laser Module
not saved!
Sheet: 1/1
AT90CAN128
IC1
INA155UA
U$3
19C100PA4K
U$2
JTAG
MCP2551-SOIC
U$1
OS4000-T
U$6
TES2N-1211
EXTPWR
MKDSN1,5/4-5,08
RESET
TYCOFSM4JH
16MHz
Q1
BC817
Q2
Q3
LED1
LED2
LED3
LED4
5V
GND


MKDSN 1,5/4-5,08
X1
MCP2551-SOIC
JEDEC MS-012 AA
IPC SO8
TES2N-1211
US1
EXTPWR
19C100PA4K
US2
US3
INA155UA
IC1
AT90CAN128
JTAG
10
US6
OS4000-T
R3
R4
22k
R6
C10
0.1uF
C9
Q1
Q2
BC817
R13
R12
R11
R14
1k
22k
C1
C2
C4
C5
C6
C3
C8
R1
R2
R5
R7
R8
R9
R10
150
16MHz
12pF
LED1
LED2
LED3
LED4

Appendix B

CAN-message format

CAN-messages sent from the PC to the laser module contain only one byte providing information to enable/disable the lasers and status LED's on the PCB. The MCU has been configured to alternatively receive larger CAN-messages, also containing information to configure the compass:

0. Bit level Controls

   D7 D6 D5 D4 D3 D2 D1 D0

   - D0 = 0, Right laser disabled; D0=1, Right laser enabled

   - D1 = 0, Left laser disabled; D1=1, Left laser enabled

   - D2 = 0, Status LED's disables; D2=1, Status LED's enabled

   - D3 = 0, Message does not contain compass configuration; D3=1, Message contains information to configure compass

   - D4 = 0, Compass configuration does not contain numerical value; D4=1, Compass configuration contains numerical value

   - D5 Reserved

   - D6 Reserved

   - D7 Reserved

1. Compass configuration command as ASCII character
2. Compass configuration value as ASCII characters (LSB)
3. Compass configuration value as ASCII characters
4. Compass configuration value as ASCII characters
5. Compass configuration value as ASCII characters

6. Compass configuration value as ASCII characters (MSB)

CAN-messages sent from the laser module to the PC are marked by an identifier (ID), informing the PC what the message contains:

- 0x001: Message contains pressure reading
- 0x002: Message contains azimuth reading
- 0x003: Message contains pitch reading
- 0x004: Message contains roll reading
- 0x005: Message contains temperature reading

The data bytes in the messages contain the reading values. For pressure data, only two bytes are used:

Byte 0: D7 D6 D5 D4 D3 D2 D1 D0

- D0 = ADC-reading in binary (LSB)

- D1-D7 = ADC-reading in binary

Byte 1: D7 D6 D5 D4 D3 D2 D1 D0

- D0 = ADC-reading in binary

- D1 = ADC-reading in binary (MSB)

- D2-D7 Reserved

For the remaining readings (compass readings), values are sent in four bytes as ASCII values:

0. Value as ASCII characters (LSB)
1. Value as ASCII characters
2. Value as ASCII characters
3. Value as ASCII characters (MSB)

Appendix C

VideoRay Communication Protocol

The Communication Protocol for VideoRay Pro III and Desktop Computer
Revised 11/04/06 by Marcus Kolb

Physical media: RS232, baud rate 9600, 8 bit, 1 stop, no parity.

- **Enabling computer control**

  When the VideoRay is powered on it waits 5 seconds for a byte to be received on the RS232 port. If it receives anything, it enters into computer control mode. Otherwise the vehicle will be directly controlled by the control panel.

- **Normal Communications Between VideoRay and PC**

  VideoRay waits for 8 bytes containing information for running the vehicle. Then, the VideoRay sends out 7 bytes containing a 3 byte identifier, compass and pressure data.

  The PC sends 8 control bytes and waits for 7 data bytes coming from VideoRay and if it receives them, it sends out the next 8 control bytes immediately. VideoRay will keep waiting for the entire 8 bytes until it receives all of them (VideoRay Pro works in polling mode).

  The total time used for exchanging information between VideoRay Pro and PC is about 15.6ms ((8+7)/(9600/(1+8+1)). This will not affect the control characteristics of the vehicle provided the PC does not make the VideoRay Pro keep waiting for too long.

- **Information of the 7 Bytes VideoRay Sends**

  The first 3 bytes of the 7 bytes contain an identifier then compass low byte, compass high byte, pressure low byte and pressure high byte.

  1. 0x40 (hex) (All VideoRay models)

2. 0x31 (hex) (All VideoRay Pro III)
3. 0x02 (hex) (data type for future use)
4. Low byte of Orientation
5. High byte of Orientation
6. Low byte of Depth
7. High byte of Depth

The relation between low byte, high byte and the real value is:

Real value = Low byte + 256 x High Byte, for instance:

Orientation* = Low byte of Orientation + 256 x High Byte of Orientation (0-359)

Depth = Low byte of Depth + 256 x High Byte of Depth (0-1023)

*When Orientation is calculated, the following conversion is needed for the real orientation:

Real Orientation = 360 - Orientation; // mirror the image of the orientation

if (Real Orientation < 90) Real Orientation = 270 + Real Orientation; // shift 90 degrees counterclockwise

else Real Orientation = Real Orientation - 90;

- **Information in the 8 Bytes PC Sends**

  1. 0x23 (hex) (for all VideoRay models)
  2. 0x31 (hex) (for Pro III)
  3. Current for the port thruster, minimum 0, maximum 220
  4. Current for the starboard thruster, minimum 0, maximum 220
  5. Current for the vertical thruster, minimum 0, maximum 220
  6. Current for the lights, minimum 0, maximum 200
  7. Bit level Controls for the manipulators and auto depth:

     D7 D6 D5 D4 D3 D2 D1 D0

     - D0 = 0, Manipulator 1 close; D0=1, Manipulator 1 open

     - D1 = 0, Manipulator 1 disable; D1=1, Manipulator 1 enable

     - D2 Reserved

     - D3 Reserved

     - D4 Reserved

- D5 Reserved

- D6 = 0, Front light / camera; D6=1, Rear light / camera

- D7 Reserved

8. Bit level controls of camera tilt and focus, the direction of the thrusters:

   D7 D6 D5 D4 D3 D2 D1 D0

   - D0 = 0, Tilt up; D0 = 1, Tilt down

   - D1 = 0, Tilt disable; D1 = 1, Tilt enable

   - D2 = 0, Focus near; D2 = 1, Focus far

   - D3 = 0, Focus disable; D3 = 1, Focus enable

   - D4 = 0, Port backward; D4 = 1, Port forward

   - D5 = 0, STBD backward; D5 = 1, STBD forward

   - D6 = 0, Vertical up; D6 = 1, Vertical down

   - D7 is reserved