Building a mobile robot with optical tracking and basic SLAM

Anders Marklund
Building a mobile robot with optical tracking and basic SLAM

Anders Marklund
Master’s Thesis
January 2009

Examiner: Kalevi Hyyppä
Luleå University of Technology

Supervisor: Per Johansson
Rubico AB
Preface

Today everybody have a cellphone, a DVD player, at least one computer and many more electronic devices at home to make our life a little easier and more comfortable. The demand for better and smaller embedded devices keeps increasing as technology moves forward.

Rubico AB is a member of The Collaborative, Analog Devices Third Party Program for Embedded Processor and DSP applications, and they focus on consultant services for embedded systems. Rubico have specialized on processors of the Blackfin architecture from Analog Devices and they have great experience with software development for uClinux\(^1\) and also contributes to the open source community.

My task at Rubico was to develop a demonstration product that would draw attention at fairs, student events and similar, and show an example of what the Blackfin DSP\(^2\) is capable of. The project began in June 2008 and ended in December same year. Everyone at Rubico have, in some way, assisted me in solving many problems and I would like to thank them all for making my time at Rubico a great conclusion to my studies!

Especially I’d like to thank:

- **Anders Larsson** and **Per Johansson** for assisting me with a little bit of everything.
- **Axel Alatalo** for hardware guidance and late night fault finding.

---

\(^1\)uClinux is a Linux distribution designed for microcontrollers.

\(^2\)Digital Signal Processor.

Anders Marklund, author
Abstract

This report is a master thesis in mechatronics and is about building a mobile robot with visual tracking capabilities. Later, the project was extended to also let the robot create a virtual map of the environment in which the robot was placed. The Robot's purpose is to demonstrate some of the capabilities of the Blackfin Digital Signal Processor and draw attention at fairs and events.

The report covers the selection of image sensor and other components necessary for visual tracking, and the ideas behind the software algorithms designed. It also brings up the selection of mapping sensors and software needed to make it work in parallel with the visual tracking ability.

Tests of the visual tracking shows that it works as intended. The mapping feature, however, is limited to a very small area of operation, if used indoors. Communication between the robot and a computer is obtained through a WLAN link and live pictures and the current map can be viewed from a custom made console. The user can also send commands to the robot through this console. To actually run the robot, a radio transmitter from a standard radio controlled vehicle is used.

Current issues and suggested improvements are also discussed in the report. Basic ideas and concepts of robot vision and mapping strategies are covered, but some general knowledge in electronics and programming in C/C++ is assumed from the reader.
# Contents

Introduction 4

1 Optical tracking 5  
  1.1 Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5  
  1.2 Robot base . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5  
  1.3 Image sensor . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6  
    1.3.1 CCD Sensors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6  
    1.3.2 CMOS sensors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7  
  1.4 Pan/tilt turret . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8  
  1.5 µC/camera module . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9  
    1.5.1 SR V-1 camera module . . . . . . . . . . . . . . . . . . . . . . . . . 10  
  1.6 Software . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10  
    1.6.1 Target identification . . . . . . . . . . . . . . . . . . . . . . . . . . 11  
    1.6.2 First step Color search . . . . . . . . . . . . . . . . . . . . . . . . 11  
    1.6.3 Laser pointer search . . . . . . . . . . . . . . . . . . . . . . . . . . 12  
    1.6.4 Color encoding and the SRV-1 . . . . . . . . . . . . . . . . . . . . 14  
    1.6.5 Pan/tilt control . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15  
  1.7 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15  

2 SLAM 17  
  2.1 Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17  
  2.2 Sensors . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17  
    2.2.1 Distance sensors . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17  
  2.3 Localization . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20  
    2.3.1 Heading . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20  
  2.4 Test platform . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21  
    2.4.1 Software drivers . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21  
    2.4.2 Sensor test results and problems . . . . . . . . . . . . . . . . . . . 21  
  2.5 Supply board . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24  
  2.6 Finalizing and mounting . . . . . . . . . . . . . . . . . . . . . . . . . . 24  
    2.6.1 Ultrasonic modules . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
## CONTENTS

2.6.2 Optical sensor ........................................... 25
2.6.3 Compass and supply boards ................................. 25
2.7 Mapping software ........................................... 25
2.8 Measurement rates ........................................... 27
2.9 Mapping structure and calculations ........................... 27
2.9.1 Sine lookup table ......................................... 28
2.9.2 Mapping calculations ....................................... 28
2.10 Mapping results and discussions .............................. 28

3 Overall results ................................................. 31
3.1 Optical tracking improvements ................................. 31
3.1.1 Servo control ............................................ 31
3.1.2 Object tracking ........................................... 31
3.1.3 Laser pointer tracking issues ............................... 32
3.2 Mapping improvements ........................................ 32
3.2.1 Vibration damping ......................................... 32
3.2.2 Map resolution ............................................ 32
3.2.3 Orthogonal measurements ................................. 33
3.2.4 Laser ranging .............................................. 33
3.2.5 Compass issues ........................................... 33
3.2.6 Network load .............................................. 33
3.2.7 Object marking strategies ................................. 34
Introduction

My first contact with Rubico was at the LARV\textsuperscript{3} event, where I met Anders Larsson and talked about the possibility of doing my masters thesis at Rubico. Rubico was looking for a product to show at student events that could demonstrate what Rubico is about. The demands from my supervisor at the university, was that a microcontroller must be involved and it should control some sort of electrical motor. A decision was made to build a radio controlled vehicle with a mounted camera, that automatically aims the camera towards a target, even if the vehicle is heading in a different direction.

The idea was to buy a commercial radio controlled battle tank and modify it to carry an image sensor and a microprocessor on a printed circuit board. The target for the camera to follow might consist of the spot from a laser pointer or similar, but it should be customizable. The project was later expanded to add a mapping feature to the robot. Through the use of various types of sensors a virtual map of the robots surroundings should be created and displayed to the user.

\textsuperscript{3}Lule Arbetsmarknads Vecka, a local fair for students to meet with many companies.
Chapter 1

Optical tracking

1.1 Objectives

The first\(^1\) task was to build a mobile robot that automatically could track a target with an image sensor\(^2\), although the robot movements are manually controlled by a human. Tracking, is done by trying to keep the target in the center of the picture as observed from the image sensor.

The major components needed to accomplish this is a standard radio controlled vehicle, an image sensor, a microcontroller, and some sort of platform to direct the camera. Basic concepts about these components are covered in the following sections.

1.2 Robot base

A radio controlled battle tank was chosen to act as the base of the robot. The tank moves rather slowly which makes it easier to control in a crowded area. Also the tank has big flat surfaces that enables easy mounting of components. Figure 1.1 shows the chosen tank model. It has metal tracks and gears for good durability.

---

\(^1\)Due to reasons covered in future sections the project was expanded and other tasks added.

\(^2\)The terms camera or camera module in future sections, also refer to this component.
1.3 Image sensor

The two major types of image sensors on the market today are CCD and CMOS. Both of these sensors types gather photon energy by exposing a small pixel area to the light you wish to capture and converts this energy into a voltage. This voltage is then interpreted by some electronic circuits and picture data is obtained. The technique used to convert the photon energy into a voltage, and the way the voltage is treated before sent off chip, is where the sensors differ most. The different readout techniques contribute to significant implications for capabilities and limitations of the sensor. For more information see [1], the following sections only briefly describes the overall function of the two sensor types.

1.3.1 Charge coupled device sensors

When the exposure of a CCD sensor is complete, the charge in every pixel segment is transferred to a common buffer, then converted to a voltage and sent off chip for further processing as shown in figure 1.2. Usually the CCD image sensors require more external components than the CMOS sensors, however because of this they can be customized for a specific task to use the units full potential.
1.3.2 Complementary metal oxide semiconductor sensors

In the CMOS case, after the exposure, the photon energy is converted to a voltage directly at the pixel area and each pixel voltage is then read separately, see figure 1.3.

CMOS sensors are easier to implement out of the box when compared to an equal CCD sensor, because they require very few external components and most of its features are built into the chip. This makes it easier to implement, but it also limits the level of customization possible to a specific sensor. However both CMOS and CCD sensors are available with numerous different specifications and features, and a careful selection, depending on the application, will generally be enough to avoid any custom modifications.
1.4 Pan/tilt turret

To enable the camera to freely look in the direction of the target, it needs to be able to rotate both up/down and left/right with respect to the vehicle. The most common way to accomplish this is to use two electrical motors, where one motor creates the panning motion and the other creates the tilting motion. These type of mounts are commonly used with, for example, camera stands and surveillance equipment. Since this platform is to be rather light and small, two standard servos, instead of dc motors, mounted as shown in figure 1.4 would suffice for this application. However if standard/unmodified servos are used this will limit the rotation to about 180 degrees, since most servos can only rotate about 90 degrees in each direction. This hardware limitation also secures that the servos don’t wind up any cables. The downside of this kind of turret is that it may be somewhat wiggly if the mounted camera platform is too heavy. All weight will be balanced on the lower servo axis only, unless more support is provided by some external hardware.

One of the early ideas was to use the existing pan/tilt solution on the RC tank that controlled the cannon. However the rotation (pan) speed was way to slow and the tilt motion was made so that you had to tilt max up before it could begin to tilt down and same thing when going up. For these reasons this solution was abandoned.

Due to the simplicity, both in construction and control, the servo turret solution was chosen for the pan/tilt motion of this project.

Figure 1.3: CMOS photon conversion block scheme.
CHAPTER 1. OPTICAL TRACKING

1.5 Microcontroller/camera module

To be able to locate and interpret an object within an image, a high performance microcontroller was needed to be able to handle the image processing in real time.

Rubico has a close cooperation with Analog Devices and has specialized in their Blackfin processor which is designed for high speed digital signal processing. Some Blackfin models have integrated network support and since the staff at Rubico is experienced with this architecture it seems very well suited for this application.

There were especially two camera modules of interest, the SRV-1 from Surveyor [2] (see figure 1.6) and a digital surveillance kit from Analog Devices [8] designed by Avnet [3] (see figure 1.5). Both these modules uses a Blackfin processor and although the module from Avnet has a higher performance processor and a better image sensor chip, the SRV-1 module from Surveyor was chosen. This is because all of the documentation, circuits and firmware (except for the wifi module) are open source and actively updated. There’s also an active online forum with a lot of other people working on similar projects and sharing their ideas and solutions. The SRV-1 module is also smaller and comes with a complete wireless network solution for communication and firmware uploads.
1.5.1 SRV-1 camera module

The SRV-1 camera modules main components consists of:

- 500MHz Blackfin processor, BF537
- 32MB SDRAM, 4MB SPI flash memory
- Omnivision OV9655 1.3 Mega pixel Sensor
- WiFi communication via Lantronix Matchport WLAN 802.11g radio

The circuit boards are small; only 50mm X 60mm and weighs about 140g. There is a 32 pin header for external connections to the serial peripheral interface, two wire interface, general purpose input/output ports and more, to connect additional hardware if necessary.

1.6 Software

To accomplish this task the appropriate hardware is needed and also something to tell the hardware what to do. The main objects for the software is to receive an image from the camera, identify a certain target within the image and send an appropriate signal to the servos to direct the camera towards the target.

To get started there are some basic functions already implemented by Surveyor upon delivery of the camera module, such as receiving an image and sending it through the wireless network. A computer can then view the
image through a console which can be downloaded from Surveyors website. In Surveyors default firmware there was a simple object finding algorithm that searches for pixels matching a specified color interval. But it wasn’t good enough for the purpose of this project. A control algorithm for the servos needs to be implemented and combined with the search algorithm.

1.6.1 Target identification

There is no general best way to find an object within a picture. It is very dependent of what kind of object you’re trying to find, in what environment it is located, if there are other similar objects nearby, how much light there is and so on. The image sensor has many handy features and among those is the automatic gain control. This will automatically adjust the gain if the lighting in the environment is changed, but a picture in the same environment but with two different lightings will still look different. A picture in a dark environment will have more intense (hi gain) but shady colors, while a picture in a bright environment will have clear and softer colors (low gain).

1.6.2 First step Color search

As a first step an algorithm that searches for connected pixels with a specified color was written. This may seem like an easy task, but a simple thing such as, the different lighting in the morning and the evening can give different readouts from the camera. To compensate for noise and different lighting within environments, it is necessary to search for a small interval of colors similar to the desired color instead of one exact color.

If other objects are located near the target with the same, or similar colors, multiple objects may be detected. This could be resolved by choosing a more narrow color interval, but that might also make the object undetectable if the lighting is changed.

To be able to separate the target from any other similar object nearby and filter out the noise that might appear, the image is divided into a number of smaller fields as shown in figure 1.7b. Every field is then searched for the color interval and if a field contains enough matching pixels, the field is marked as part of an object. This might not necessarily be the target though. All the fields are sequentially searched and all adjacent fields that are marked, is assumed to be part of the same object. For every object found, the amount of matching pixels they contain, is divided by the number of fields they include, and the object with the highest matching pixel density is selected as the target. This will filter out noisy areas of the picture that might be similar in color and also other objects of the same color that are
1.6. SOFTWARE

CHAPTER 1. OPTICAL TRACKING

1.6.3 Laser pointer search

A laser pointer gives a very intense red\(^3\) color, which gives a good contrast to other colors in the environment. However, there are many complications to keep in mind such as reflections and the reflective capability of different

---

\(^3\)Red is the most commonly used, but other colors are of course available.
The search algorithm previously discussed was supposed to make it easy to track the intense laser light. But the laser was too intense for the image sensor and the pixel sensors were saturated. This resulted in the red laser dot showing up as a white dot with a faint bright pink halo around it. This made it very difficult to track the laser by color, especially on bright materials such as a white wall for example. Instead, the brightness of the laser was used. The color encoding of the picture makes it possible to only inspect one of the image components, the luma component. This halves the amount of data to inspect for every pixel. To test this, a static threshold was set, and if any pixel had a higher luma value than the threshold, it was assumed to be part of the target. In most cases, this very simple criteria works really well, because the light intensity in a normally lit room usually doesn’t reach the threshold value, if set high enough. Other light sources such as lamps and reflections however, will also saturate the pixels and should be avoided if possible.

\[\text{Luma, } Y'\text{, is a color component related to, but not equal to, luminance. For more information see chapter 1.6.4}\]
1.6. Software

1.6.4 Color encoding and the SRV-1

The camera of the SRV-1 module can be configured to give pictures in both RGB and Y'CbCr1 format. This report will only cover the basics about Y’CbCr encoding and assumes knowledge about the common RGB color format. There is a common misconception between Y’CbCr and the similar format Y’UV. This format, however, is mainly used with analog video applications and not, as in this project, digital image processing.

The Y’CbCr format consists of a brightness component, Y’, and two color components, Cb and Cr. The Y’ is a non-linear component called luma that is related to, but not equal to, luminance. This component alone gives the picture in gray scale (in a simplified manner) while the other two components represent the colors as show in figure 1.10. For more information about the Y’CbCr format and other details about image and video encoding please refer to [4].

Figure 1.9: Block scheme for the color search algorithm.
1.7. Results

Figure 1.10: From the left: Original picture, Y’-component, Cb-component, Cr-component.

1.6.5 Pan/tilt control

The pan/tilt servos are controlled in a very simple fashion. A PWM\(^5\) signal with a pulse width of 1.5ms corresponds to a centered position of the servo. An increase or decrease of the pulse width will increase or decrease the servo angle respectively up to a maximum of about \(\pm90^\circ\) from center position.

From any of the previously mentioned searching algorithms the center of the target can be calculated and then the target offset with respect to the image center is calculated. This offset is measured in pixels, since there is no way to know how far away from the camera the target is without using other sensors or advanced image processing techniques. The target offset from the pictures center is then scaled and converted to a register value of the microcontroller. This will change the signal to the servos and make them turn so the target is moved towards the center of the picture.

1.7 Results and discussion

Tests of the two different search algorithms both showed satisfactory results, but these algorithms only work if there is already a target within the picture and if the target is lost, nothing happens until the target reenters the picture. A good complement, would be an algorithm that finds an object that disappeared based on the target's previous motion pattern.

Also, the algorithm controlling the servos could be improved in many ways to make the movements smoother and more accurate. In the present implementation, the servos are controlled only by a static scale factor depending on the pixel offset from the image center. A regulator that could adapt to the target motion would perform better and smoother. It might also provide some target localization if the target is lost.

The overall performance of the robot is good, live pictures sent over the

\(^5\)Pulse Width Modulated, a square wave signal with a variable pulse width and fixed period.
wireless network appear on the screen at about 13 fps. If you add image processing such as target identification, the frame rate goes down to about 3-5 fps. This is mainly the wireless transfers and JPEG encoding, handled by the default firmware from Surveyor, that chokes the performance. If live images are disabled so that no images are transferred or JPEG-encoded, the frame rate is close to 30 fps. This is the maximum specified frame rate for the image sensor at the resolution 320x240 pixels.

Figure 1.11 shows the robot at this point in the project. Only 1 month had passed so far but the project was intended to last for 20 weeks. Therefore it was extended with an additional task...

Figure 1.11: The robot with the camera module mounted on a servo pan/tilt turret. This is what it looked like after the first task.
Chapter 2

Simultaneous localization and tracking

2.1 Objectives

The second task was to let the robot construct a virtual map of its surroundings, using appropriate sensors, in addition to the camera. This is also a big research area and more commonly known as Simultaneous Localization And Mapping or, SLAM. To accomplish this task, external hardware was designed and interfaced to the SRV-1 board as discussed in the following chapters.

2.2 Sensors

To be able to sense the environment, it is necessary to measure the distance to objects in the surroundings of the robot. Also, the robots movements and heading must be monitored to be able to correlate the data from the distance measurements to a position within a coordinate system.

2.2.1 Distance sensors

There are many types of distance sensors and they all have different characteristics. This report will briefly cover some of the most common distance sensors.

Laser based range sensors have a very good range, accuracy and a very narrow measurement beam, which would be ideal for this application. However the price is too high.

Infrared sensors is another option that is cheaper than laser. Infrared sensors commonly uses a triangulation method to calculate the distance.
The triangulation approach is based on the length and angles of a triangle, where the corners are represented by the infrared transmitter, receiver and the detected infrared light. The angle $\beta$ and length $b$ is known and fixed. The length $a$, is optically measured by the IR\textsuperscript{1} receiver. The angle $\alpha$, and also the distance $d$, can then be calculated. However, since this method is based on optically measured distances, the accuracy becomes poor when $d$ grows (i.e. $\alpha$ becomes small). Most IR sensors can not measure accurate distances longer than 50cm. There are sensors that can measure further, but they are either expensive or have a minimum distance of about 1 meter before the sensor gives any useful readings.

Figure 2.1: IR receiver, transmitter and the projected light creates a triangle and the distance, $d$, can be calculated.

Ultrasonic sensors have good range, both minimum and maximum, and come at a low cost. The ultrasonic beam width however, is usually quite wide and therefore may detect too much. If multiple sensors are used, echoes might be a problem if simultaneous measurements are taken. A good ultrasonic sensor is the module from Maxbotix [5] called EZ1 (see figure 2.3). It has a range of 6-254 inches (15-645 cm) and beam characteristics as shown in figure 2.2. The module have a number of different interfaces available, analog output, pulse width output, and a serial output. Some other ultrasonic sensor modules were also inspected but the EZ1 seemed to have the

\textsuperscript{1}Infrared.
best performance for this application. Four of these sensors will be used on
the robot, one on every side. For easy and precise measurements, the pulse
width output will be used. The pulse width is then timed using one of the
internal timers and some of the general purpose IO ports of the microcon-
troller. Sample results for measured beam patterns are shown in figure 2.2
on a 12-inch grid. The detection pattern is shown for:

A 0.25-inch diameter dowel, note the narrow beam for close small objects,

B 1-inch diameter dowel, note the long narrow detection pattern,

C 3.25-inch diameter rod, note the long controlled detection pattern,

D 11-inch wide board moved left to right with the board parallel to the
front sensor face and the sensor stationary. This shows the sensors
range capability.

Note: The displayed beam width of (D) is
a function of the specular nature of sonar
and the shape of the board (i.e.
flat mirror like) and should
never be confused with
actual sensor
beam width.

Figure 2.2: Approximate beam characteristics for the EZ1 ultrasonic distance
sensor at different supply voltages.
2.3 Localization and dead reckoning

To be able to correlate the measurements from the ultrasonic sensors, information about how the robot is moving and the current heading, is needed. To measure the movements of the robot, one possibility is to measure the movements of the tracks on the tank. The problem with this method is that the tracks can slip and thereby register a movement that didn’t actually occur. A better solution would be an optical sensor placed under the tank which measures the ground movements with respect to the tank. One device that does just this, is the sensor built into an optical mouse.

Avago technologies [6], manufactures sensors for optical mice and their chip, the ADNS-3080, was selected for this project. This chip is used with, for example, computer mice designed for high speed gaming where fast and accurate movements are required. This device communicates through a common serial peripheral interface and measures the traveled distance in counts per inches or: cpi. The resolution can be changed between 400cpi and 1600cpi depending on the application and it can be configured for a measurement rate of above 6.5kHz.

2.3.1 Heading

Now both the distance to objects around the robot and the robots traveled distance is known, the only thing left is to figure out in what direction it is currently moving. A yaw-rate gyro measures the angular velocity around a vertical axis and could be used for this application. But gyros tend to drift and to get an absolute angle instead of the velocity, continuous integration of the robots rotation is needed. Because of the drifting problem this could easily become distorted and inaccurate. Another option would be the Honeywell [7] electronic two axis compass, HMC6352, with I2C² interface which can deliver the current heading at request. It also features a user calibration

²The same as TWI, or Two Wire Interface.
function to calibrate the compass for a specific environment.

The compass solution was chosen because of its easy implementation, it requires less overhead when operating and since it is less sensitive to vibrations.

2.4 Test platform

To be able evaluate if it would be possible to get the above mentioned sensors to work together as intended, two separate circuit boards were designed and connected to a Blackfin 537 STAMP-board\(^3\). The advantage of using the STAMP board is that it is very easy to connect to all the different peripheral interfaces available from the microprocessor, and it’s supported by VisualDSP++\(^4\). VDSP makes it easier to debug software because it enables the user to halt the running code and inspect the register settings and variable values, in the middle of a running program. This is a valuable feature when trying to learn a new peripheral interface of the microcontroller for example.

2.4.1 Software drivers

Two versions of peripheral drivers for all the sensors were written. The first version was a basic polled version, a chance to better understand how to work with the different interfaces, while the second version was interrupt driven and a lot more efficient. The second version also allowed for more parallelism in the sense that all sensors could be triggered to take a measurement at the same time since they are all on different peripheral interfaces.

2.4.2 Sensor test results and problems

During the tests of the ultrasonic modules some problems with oscillations in the output signal appeared. The readings could vary more than one meter when measuring the distance to a wall without moving the sensor. This proved to be due to bad decoupling and an extra capacitor, to supply the current needed to emit the ultrasonic sound wave, solved the problem.

These ultrasonic modules have a feature to automatically send a trigger signal to another module of the same type and can therefore easily be chained. This means that a microcontroller triggers a sensor, and when that sensor is

\(^3\)A development board from Analog Devices with the same processor as the SRV-1.

\(^4\)VDSP, a development and debugging environment from Analog Devices.
done, it triggers the next sensor and so on until all sensors are done. This is the safe way of utilizing multiple sensors, another way would be to trigger all sensors at the same time (in parallel). This could be more complicated and can cause faulty readings due to echoes and wave canceling phenomena. But it would also result in a major performance boost since it would only take a fourth of the time to complete all measurements. Tests of the parallel approach showed that the indoor environment caused too much echoes that interfered with the measurements and the sequential triggering had to be used.

The optical sensor also proved to be reliable, during the initial tests. For these tests a 3mm plastic board was used to separate the sensor from the surface. The optimal distance would have been 2.4mm (with the original lens), according to the data sheet and because of this, the surface was a little out of focus. This caused the readout to be reduced to about 80% of the real distance traveled, when used on a patterned surface. A white paper gave no readings at all unless pressed a little closer to the lens, which proved that the problem was focus related. A paper pile of about 2.4mm as a distance spacer, gave very promising results, even on white paper.

The compass returned completely useless results at the initial tests but that was before it was calibrated. To calibrate the compass, it should be rotated two rounds in about 20s with a constant speed and completely level. For every degree the compass tilts, it may add as much as two degrees error to the measured heading. For the evaluation boards mentioned earlier, a temporary calibration device based on a disco ball motor was built, see figure 2.4a. Later, a more solid calibration tool was built to be able to calibrate the compass when it was mounted on the tank. To set a good rotation speed a hi torque servo was used as the primary motor and together with the base plate shown in figure 2.4b and a piece of plywood the calibration tool was complete.

The servo used for the second calibration tool was a Hitec HS-635HB servo, which unmodified could only rotate about 180 degrees. A servo is normally controlled by sending a pulse that corresponds to a certain angle of the servo. With some careful modifications the servo can be converted into a continuously rotating motor where the pulse width controls the velocity instead of a position. There are mainly two steps in modifying this servo for continuous rotation\textsuperscript{5}, physically removing a stop from one of the gears, and removing (or replacing) the potentiometer that measures the position of the servo. The stop shown in figure 2.5a is removed with a cutter or

\textsuperscript{5}This may vary from model to model, and some servos require more advanced modifications.
knife and then the potentiometer (under the circuit board shown in figure 2.5b) is pulled out of the casing, still connected to the circuit board. The potentiometer can now be used to trim the center position by sending a 1.5ms pulse and manually adjusting the potentiometer until the servo stops.

Another complication regarding the compass, was the electromagnetic interference from the motors powering the tank. Tests showed that a distance of at least six centimeters from the tracks was necessary to reduce the interference to a minimal level when the motors were running. The interference from the motors was also transported through the metallic gears and led out into the tracks. Therefore, if the compass was placed six centimeters in front of the tank (about 30cm from the motors) or six centimeters from the side (about 10cm from the motors), gave the same results.
2.5 Supply board

The Tank came with a 7.2V Ni-Cd battery. This type of battery can reach almost 9V when fully charged. The SRV-1 module have a voltage regulator which accepts voltages between 4.75-18V so the battery can be directly connected to the power terminals of the SRV-1 module. The servos, however, could not operate at a higher voltage than 6V for normal operation. To protect them, another voltage regulator was needed. For this purpose a new circuit board with a voltage regulator and connectors for all the sensors, was designed. From this board a single ribbon cable from the supply board to the SRV-1 module was used to reduce the amount of loose cables connected to the rotating camera platform. Also some control LEDs, a simple low battery detection circuit and some switches to control the triggering of the ultrasonic sensors (sequential or parallel triggering) were included on the board.

2.6 Finalizing and mounting

The separate tests of all the sensors showed good results and it was time to put everything together and mount it on the tank.

2.6.1 Ultrasonic modules

The ultrasonic modules were attached to a piece of plastic with a 90° angle, which in turn was screwed onto the tank.
2.6.2 Optical sensor

The big downside with this kind of optical sensor is that it must stay at the same distance from the surface at all times even if the tank is moving and bouncing a little. The idea was to mount the sensor directly under the tank and because of this, some kind of suspension was necessary to keep the sensor on the ground. For this, a new circuit board with ribbon cable connectors in the rear and front was designed and connectors was also installed in the rear and front of the tank. This circuit board was then hanged under the tank and two ribbon cables kept it in place. This allowed the sensor to stay on the ground even if the tank was bouncing. The ribbon cables also minimized the sideways motion of the sensor board.

2.6.3 Compass and supply boards

The compass need to have some distance to the motors and the supply board is preferably mounted close to the pan/tilt turret. The supply board, the compass and the rear ultrasonic module were mounted on a plastic box, standing on the rear of the tank. The supply board was mounted on the bottom inside the box for easy access to cables and switches, the compass was mounted on top of the box for an extended distance to the motors, and the ultrasonic module was mounted on the rear side of the box facing backwards. See figure 2.6.

2.7 Mapping software

The main idea is to take periodic measurements from all the sensors and construct a map based on that information. Kalman filters and Monte Carlo localization is often mentioned in these research areas and my intention was to try and implement these filters but there wasn’t enough time. The software written, takes periodic measurements from all the sensors and constructs a simple map by writing values to a large two dimensional array. The array is then sent through the wireless network to a computer that displays the map to the user. To display the map to the user a simple SDL-based console was developed. The console was based on a previous console for Surveyors dual camera module. It was modified to support only one camera and some features to send commands to the robot and displaying the custom map, was added.
(a) All sensors mounted.

(b) The optical sensor mounted underneath the tank.

Figure 2.6: Robot overview
2.8 Measurement rates

The ultrasonic range sensors are externally triggered by the microcontroller and it takes 49ms for every sensor to complete a measurement. Four of these sensors are used in series, which makes it 196ms (about 5Hz) for the entire sensor chain to complete the measurements.

The compass was setup to continuously measure the current heading at 20Hz and it stores the most recent value in a local buffer within the sensor chip. This value is then returned upon a request from the microcontroller.

The optical mouse sensor counts, with a resolution of 400 counts per inch, until the value is read by the microcontroller and then the count is cleared. The counts are stored in one byte, or 8 bits, using the two’s complement form. If the count should exceed 127 (or -128) before it is read, the value is buffered and the user must read from the sensor several times until all the buffers are empty. Only one byte is returned for each read. The tank moves at a maximum speed of about 0.5m/s, which makes the maximum count rate 7874 counts per second, as seen from the formulas below.

\[
\frac{50\text{cm/s}}{2.54\text{cm/in}} = 19.7\text{in/s} \quad (2.1)
\]

\[
19.7\text{in/s} \cdot 400\text{counts/in} = 7874\text{counts/s} \quad (2.2)
\]

To avoid loosing data, because the count buffers were full, the maximum amount of counts received per periodic measurement, must not exceed 127. If the sensor was read 100 times per second, 100Hz, the maximum amount of counts per readout would be 78.74. This will make sure the buffers remain empty and only one byte will have to be read every periodic measurement.

The maximum refresh rate for the periodic measurement were decided by the ultrasonic sensor chain because it had the lowest measurement rate, 5Hz. But since the optical sensor needs to be read at 100Hz, these values are locally buffered in the microcontroller, until the compass and the range sensors are read at 5Hz.

2.9 Mapping structure and calculations

The structure of the map is based on a 320x240, two-dimensional matrix where every element represents a 2x2in square of the real world. The map is constructed in a binary fashion, so that every map element can either be occupied or clear. To represent a clear path a zero is written to the corresponding array element, any non-zero value will represent an occupied map segment.
2.9.1 Sine lookup table

A sine lookup table is used to improve performance of the coordinate calculations. True sin and cos functions would require significantly more computations and the accuracy wouldn’t be significantly better than the lookup table for this application. The lookup table is made from 901 values based on equation 2.3:

\[ \text{sine\_lookup\_table}[\text{angle} \cdot 10] = 10000 \cdot \sin(\text{angle}) \]  \hspace{1cm} (2.3)

where \( \text{angle} \) is in the range of 0-90° with 0.1° steps. By multiplying with 10000, a very low roundoff error is achieved when working with integers. The table is constructed in such a way that, \( \text{angle} \cdot 10 \), represents the index of the corresponding sine value, this is because the heading from the compass is formatted this way. This table is then used to create approximate sin and cos functions for the entire 0.0-359.9° span.

2.9.2 Mapping calculations

When the mapping first is initialized, the tank is centered in the map and everything is cleared, no objects are located on the map at this moment. After the first periodic measurement, the new X and Y-coordinates for the tank can be calculated. The distance traveled is obtained from the optical mouse sensor data, the current heading is obtained from the compass. This information together with the approximate sin and cos functions gives the new coordinates. With the use of all sensors data, and the approximate sin and cos functions, the coordinates for the objects located orthogonally from all sides of the tank, can also be calculated. These coordinates represent the index values of the map array, where the non-zero values should be written to mark an occupied map segment.

2.10 Mapping results and discussions

The initial tests of all the mapping sensors working together were very promising. However, the office environment contained too much magnetic interference for the compass to handle and the results were less impressive when the area of operation was extended beyond a few meters. Also, the ultrasonic sensors were too sensitive for the vibrations and sound level caused by the electrical motors and the metallic tracks when the motors were running at

\footnotetext{Floating point calculations are available but more computational expensive, integer calculations are preferred in real time applications.}
full power. The ultrasonic measurements could return anything between six inches\(^7\) up to the actual distance. Unfortunately there was not enough time to resolve these problems completely within the specified project time frame. See section 3.2.1 for a more detailed discussion about these issues.

The results shown in figure 2.7a were obtained by pushing the tank by hand to avoid interferences caused by vibrations. In figure 2.7b the actual map is shown together with the results and the path used when mapping. The results are somewhat rotated compared to the original map and the end position is somewhat dislocated. The overall result is quite good considering the circumstances, but the corridor to the left is a little bent. This is because of the ultrasonic sensors measuring the nearest distance to the wall within the wide beam width. When the tank is rotating, the same distance will be measured but the heading will change causing a curved pattern within the map.

\(^7\)This is the minimum distance the sensor is able to measure.
Figure 2.7: Mapping results
Chapter 3

Overall results and possible improvements

During the construction and testing of this project, many things that could have been done in a different way, to achieve a better performance or more accurate results, were found. The most significant ones are discussed in the following sections.

3.1 Optical tracking improvements

3.1.1 Servo control

The algorithm to control the motion of the servos aiming the camera is currently very basic. The position error, in pixels, is multiplied with a constant and then translated into a pulse length value\(^1\). There are numerous ways to improve the performance of this p-regulator, not much time have been spent developing this part of the project.

3.1.2 Object tracking

The object tracking algorithms are currently quite "dumb" since they only work if there’s actually an object in the picture. If the object disappears from the picture, the robot will simply wait until the object appears again. By storing the most recent locations of the object one would be able to guess where the object may be located on the next measurement. It would then be possible for the robot to try to localize the target, based on it’s previous

\(^1\)This value corresponds to a specific pulse width, and the rest is handled by the internal timer of the microcontroller.
motion pattern, if it is lost. Kalman filtering is a very common aid in tracking moving targets and it could probably be useful in this application as well.

3.1.3 Laser pointer tracking issues

The algorithm for tracking a laser pointer is even dumber. It was first written as just a quick test, but it worked surprisingly well and was left in that state. The algorithm simply searches for any light source with a higher $Y'$-value than a specified static threshold, then assumes that it is the laser pointer. This goes for every light source in the environment, such as lamps, displays, windows, reflections or whatever light that might be intense enough to exceed the threshold. To improve the accuracy, a pixel limit could be set to filter out any object that is too large. Another approach would be to set a shape restraint to only target small round objects. The image sensor sees the laser pointer as a small white dot with a faint pink halo around its perimeter, this could be used to filter out the laser pointer as well.

3.2 Mapping improvements

3.2.1 Vibration damping

The dc motors together with the metallic tracks caused a lot of vibrations and a very high sound level. This interfered severely with the distance measurements. This made it impossible to run the tank at full speed remotely and use the mapping feature at the same time. The vibrations can be damped by using silicon grommets when mounting the sensors to the tank. More mass could also be attached to the sensor circuit board to dampen the high frequency vibrations.

3.2.2 Map resolution

The map resolution or, the number of matrix elements per area unit, could be increased to achieve a more detailed map and the size of the map matrix could also be increased. The level of detail are also limited by the large beam width of the ultrasonic sensors. If a more detailed map is of interest, the ultrasonic sensors should be replaced with laser sensors for better accuracy. If the map size is increased, some adaptations in the SDL-console is necessary to be able the view the map correctly.
CHAPTER 3. OVERALL RESULTS

3.2. MAPPING IMPROVEMENTS

3.2.3 Orthogonal measurements

The wide beam width of the ultrasonic sensors will often detect objects that aren’t actually orthogonally to the tank. Since only a single sensor is used on every side, this is an acceptable approximation. If multiple ultrasonic sensors would have been used in an array, they can be triggered in such a way that the resulting beam width is very narrow due to wave cancellations, but that’s outside the scope of this thesis.

3.2.4 Laser ranging

Replacing the ultrasonic sensors with laser sensors would increase the accuracy of the map substantially and also the measurement rate could be increased. However, the price would be significantly higher as well.

3.2.5 Compass issues

Tests made in the late stage of the project showed that the compass only gives decent results within a very limited area of operation when used indoors. Due to the sensitivity of the compass, many iron or steel objects close to the compass will distort the magnetic field significantly and confuse the compass. There are also many other implications such as switched power supplies and many other sources of magnetic fields/disturbances that will affect the measurements indoors. One way to get around this might have been to use a yaw-rate gyro instead of a compass, since the drifting problem of a gyro might not have been as serious as previously predicted in chapter 2.3.1.

3.2.6 Network load

In the current configuration, the transmission of the map matrix is rather slow, due to the size of the map matrix. To reduce the amount of data that needs to be sent across the wireless network, the bit pattern of every element in the map array, could be used instead of the specific value, to represent a clear or occupied map segment. This would require a little more overhead when assigning the values of the respective map elements, but the amount of data would be reduced to an 1/8 or, 9.6kb instead of 76.8kb for a map of size 320x240 segments. Every matrix element currently consists of a char\(^2\), and the bit pattern could be used to represent 8 squares of the map, instead of one. For example, the value 25 stored in the map matrix would have a

\(^2\)“char” is an eight bit data type in the C programming language
3.2. MAPPING IMPROVEMENTS CHAPTER 3. OVERALL RESULTS

bit pattern\(^3\) of 00011001, this could be interpreted as map segments 0, 3 and 4 being occupied while segments 1, 2, 5, 6 and 7 are clear. This would also require some adaptations of the map-console.

### 3.2.7 Object marking strategies

The ultrasonic sensors can’t detect small objects far away, therefore one could assume that objects close up are smaller than those far away. This could be realized by marking objects that are close, by a single map element and objects that are far away by a short centered line orthogonal to the sensor direction. The length of the line should depend on the distance and waveform of the sensor.

To account for moving objects and keeping the map up to date, the map could be assumed to consist of a single solid object spanning the entire map upon initialization. Paths are then cleared from the tank up to the objects found by the sensors. This will let the robot remove previously detected objects from the map if the path is changed. For example, if a person is standing between the robot and a wall, the robot will detect the person and clear the path up to the coordinates of the person. If the person then walks away, the robot will clear the path to the wall, including any previous objects in between.

\(^3\)Count positions from right to left, starting at 0.
Bibliography


