New Concepts for Infrared Tests in Production

ALEX DE VINCENZO
MARCUS CHRISTENSSON
BACHELOR´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

New Concepts for Infrared Tests in Production

LUND UNIVERSITY

AXIS® COMMUNICATIONS

Alex DeVincenzo and Marcus Christensson
Electrical engineering and automation – IEA15, EITL05
Lunds Tekniska Högskola
2018-06-08

Abstract

Currently, Axis Communications AB has an inefficient method of testing infrared light used by their security cameras. This method is inefficient because the same test apparatus cannot be used by different cameras. The apparatus is large, unwieldy, and unable to be easily transported, and that the camera tests itself via software picture analysis. Furthermore, different apparatus must be developed for different cameras, depending on the cameras field of view, maximum distance between the camera and the testing surface, etc. The process of testing different infrared lighting in cameras is a service that is used widespread within test locations worldwide.

In this thesis, a new method of testing IR light in security cameras is theorized, built and analyzed. The test apparatus encompasses both hardware and software, and a large majority of the solution was built, programmed, and tested in a lab environment. This new test method should universally usable, it should be able to test different models of Axis security cameras without the need to recalibrate the hardware. It should also be easy to use, and the test method software should be sequentially compilable in TestStand, a testing software used by Axis.

The thesis was written and completed by both Alex DeVincenzo and Marcus Christensson, henceforth known in the report as Alex and Marcus, respectively. If not otherwise stated, the work was equally shared.

The result of the thesis was that a proof-of-concept for an infrared test box was built and programmed. Actual test data from an Axis camera was gathered and tested, which resulted in usable test results.

Keywords

- Infrared
- Sensor
- Test
- Camera
- Software

Sammanfattning

Axis Communications AB har en ineffektiv metod för att testa infrarött ljus som används i deras av säkerhetskameror. Denna metod är ineffektiv eftersom som samma testsystem inte kan användas för olika kameror. Testapparaten är stor, svårhanterlig och svår att transportera, dessutom används kamerans egen mjukvara för att testa IR-ljuset genom bildanalys. Nya testapparater måste utvecklas för olika kameror, beroende på kamerans synfält, kamerans närgräns för fokus o.s.v. Processen att testa IR-ljus på kameror är utbrett genom hela organisationen, inte bara i Lund. Således kostar en ineffektiv metod både pengar och tid för företaget och det vore fördelaktigt att ha en effektivare metod.

I denna rapport kommer en ny metod för IR-ljus test att undersökas teoretiskt, byggas och analyseras. Testapparaten omfattar både mjuk- och hårdvara, är programmerad, testad och delvis byggd på Axis. Denna nya testmetod skall vara universellt användbar och den skall kunna testa olika sorters kameror utan att behöva kalibrera om hårdvaran. Den skall vara lätt att använda, och testmjukvaran skall vara kompatibel med TestStand, ett program som används vid testning hos Axis.

Resultatet av examensarbetet var att ett ”proof-of-concept” för en infraröd testlåda byggdes och programmerades. Testdata från en Axis kamera samlades in och testades, vilket resulterade i användbara testresultat.

Nyckelord

- Infrarött
- Sensor
- Test
- Kamera
- Mjukvara

Foreword

First of all, we would like to thank Axis Communications for allowing us to carry out our thesis work with their company. It's been an extremely educational, exciting experience and we look forward to working with them in the future. We've been able to come up with our own solutions and test different ideas, with valuable input from our supervisor, Henrik Karlsson, whenever we've needed it. The degree of freedom we received during our time here was highly appreciated and allowed us to develop our problem-solving capabilities. We would also like to thank all the other employees we've interacted with during our time at Axis. They've been nothing but helpful, enjoyable to interact with, made our time here more pleasant, and made us feel like a part of the team.

Contents

1. Introduction

1.1 Background

This thesis work was provided by Axis Communications AB, located in Lund. Axis Communications AB (Axis) was founded in 1984 by Martin Gren, Keith Bloodworth and Mikael Karlsson in Lund, Sweden. It started out as a company developing and selling protocol converters and printer interfaces. In 1996 Axis launched the first network camera on the market: Axis 200. Since then, the focus has changed to almost exclusively developing, marketing and selling network cameras. Axis currently employs approximately 2000 people and has offices in over 40 countries, although much of company activity is based in Lund.

The testing method of the infrared lighting is currently done via a large black box. The size of the box varies depending on which camera is tested. The IR light is sent from the camera onto a white sheet on the bottom of the box and a picture is taken. This picture is analyzed by software to determine if the camera and the IR light are directed the right way relative each other and the mechanical housing of the camera. The problem with this method is the size of the boxes due to the focus distance of different cameras, and that it must be designed for every type of camera. The goal of this thesis is to develop a method of testing which is generalized and usable for different types of Axis network cameras, smaller, and easier to implement.

1.2 Purpose

The purpose of this thesis is to design, produce, and put into use a new, more generalized and effective method of testing IR lighting integrated within Axis network security cameras which is smaller and easier to implement. The current method must be designed for each type of camera, it is camera-specific. The current design is also large, unwieldy, and takes up a significant amount of floor space which could be better utilized. This will cut back testing and labor costs for the company and will be more environmentally friendly in that less physical resources will be required in the testing process.

1.3 Project Goals

The goal with this thesis is to design a minimal test method, thereby freeing up resources and labor costs for the company. Both hardware and software components will be developed. As stated before, the current design is unwieldy and resource-intensive, requiring more labor hours and materials to analyze IR lighting within different network security cameras. The current analysis method is to use software to determine the intensity and focus of IR light after a test picture is taken. The design should be relatively small and able to test different types of cameras. The test should also be able to detect whether the IR lighting is well aligned with respect to the camera, and relative the cameras chassis. Software will be written for both the controller of the different sensors, and a user interface which takes input and displays test results.

1.4 Problem Description

The current problems are derived from the requirements list provided by Axis. The problems are as follows:

1. Is the solution generic, i.e. able to test different models of network security cameras?
2. Is the solution able to test lenses which spread light in different angles within a specific degree of accuracy? (10°-150°)
3. Is it safe for operators to handle, not exposing them to harmful IR light?
4. Does the hardware utilize “off the shelf” components?
5. Is the hardware 0.5x0.5x0.5 meters or less?
6. Can the module test IR light spread in relation to the optical module (are the camera and IR light diode well aligned?) and the cameras chassis?
7. Is the module easy to build?

1.5 Motivation of the Thesis

This thesis was chosen because it encompasses a large part of different aspects of the electrical engineering education at LTH. It has both electrical (choosing and calibrating sensors) and software (programming of the hardware to process different sensor input) components to it. Axis was chosen because they have a good reputation as a company, and is a growing company with promising career opportunities. Another motivation is that optimization of different processes, both regarding energy and physical resources, is an important environmental responsibility that every engineer has, and should take seriously.

1.6 Limitations

A majority of the programming was done in Microsoft Visual Studio 2017 in C#, since that is the requested programming language. C# is not a required course in the education, so the language was studied during the thesis during programming. Trial and error was used here, although it was not hard to learn in that C# is similar to Java. Another limitation is the learning process of programming in TestStand, a commonly used sequential testing platform. The construction of the test box itself was done off-site via a mechanical construction company, so yet another limitation was the dependency which was established between the thesis work on-site and the construction company, i.e. delivery wait times and adherence to specifications.

2. Technical Background

2.1 Introduction

In this chapter, the hardware and software which were used during the thesis work will be explained. Different controllers, sensors, developer environments and programming interfaces will be explained.

2.2 Hardware

2.2.1 Raspberry Pi

The [0] Raspberry Pi is a small programmable computer which has a free downloadable operating system. The model used was the Raspberry Pi 3 Model B. This model is, at the time of writing, the most powerful Raspberry Pi on the market, with a quad-core 1.2GHz 64-bit CPU, 1GB of RAM, a 40-pin GPIO and 4 USB 2 ports.

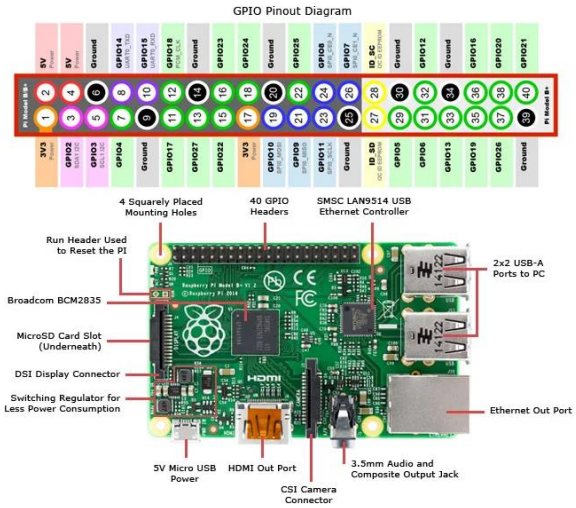

Figure 1: [1] Jameco Electronics image of a GPIO Pinout Diagram of the Raspberry Pi 3 Model B and an overview of the card's layout.

2.2.2 Arduino Uno

Figure 2: [3] Arduino, The Arduino Uno.

As per the Arduino [2] website, The Arduino Uno is "an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing."

The Arduino Uno has an ATmega328P microcontroller as well as 14 digital I/O pins, 6 of which provide PWM output, 6 analog input pins, and a 32KB flash memory provided by the microcontroller.

2.2.3 TSL260R Infrared Light Sensor

The TSL260R is a three-pin light to voltage optical sensor which has both a photodiode and a transimpedance amplifier within a single IC. Output voltage from the TSL260R is directly proportional to irradiance applied to the photodiode. Relevant aspects, such as output voltage, angular displacement, relative responsivity, and supply current vs output voltage is included below.

Benefits	Features
• Enables Extremely Fast Response to Change	• Single Photo-Diode and Trans Impedance Architecture
• Enables Fast Response to Visible Light in Range of 400nm to 700nm Wavelengths	• 260µs Output Rise-Time Response (TSL260R)
• Provides for High Sensitivity to Detect a Small Change in Light	• High Irradiance Responsivity 111mV/(µW/cm^2) at λp = 940nm (TSL260R)
• Provides Additional Sensitivity Advantages	• 2x Gain Lens

Figure 3: [4] Digi-Key Electronics, Benefits and Features of the TSL260R.

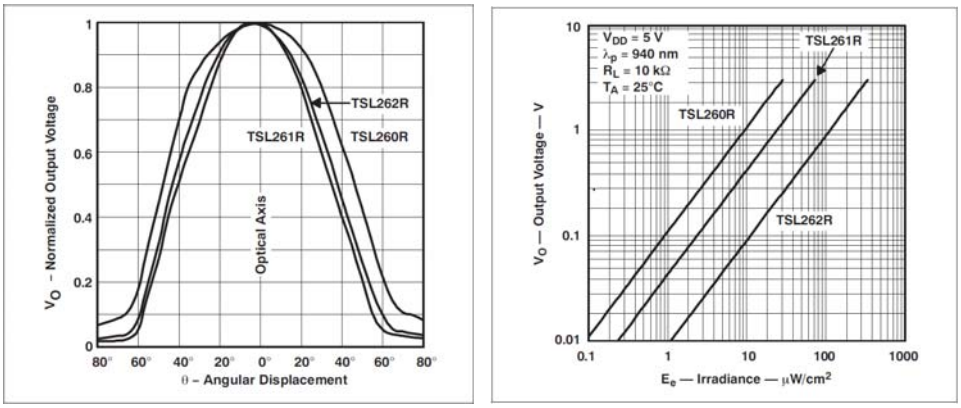

Figure 4: [5] [6] Digi-Key Electronics, Normalized Output Voltage vs Angular Displacement and Output Voltage vs Irradiance, respectively.

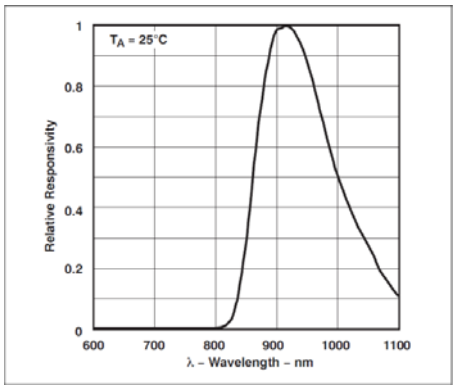

Figure 5: [7] Digi-key Electronics, Photodiode Spectral Responsivity.

2.2.4 Stepping Motor SST43D2085

The SST43D2085 is a 2-phase stepping motor with a stepping angle of 1.8°, and the possibility of further dividing the individual steps into smaller steps (microstepping).

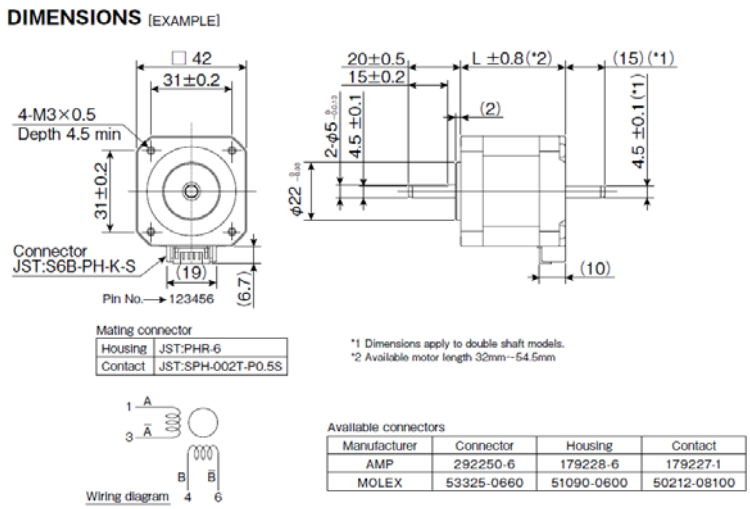

Figure 6:[8] ShianoKenshi's Dimensions and wiring diagram of the stepping motor.

2.2.5 Big Easy Driver

The Big Easy Driver is a microstepping driver which works with the Arduino, is designed to operate with bipolar stepper motors, and has a built-in translator and overcurrent protector. It's a programmable driver which sends input to the motor to control movement of the axel.

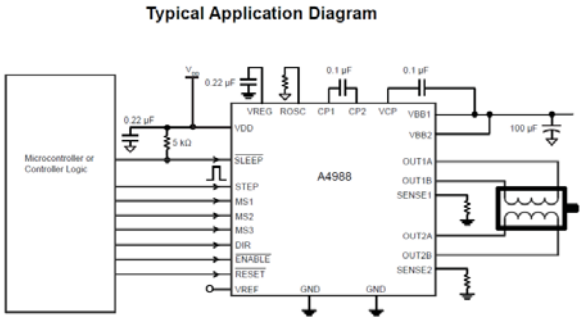

Figure 7: [9] Allegri MicroSystem's Block diagram of the big easy driver.

2.2.6 MCP3008 AD Converter

The MCP3008 is a 2.7V 8-channel 10-bit ADC by Adafruit which uses a SPI serial interface. The Arduino Uno itself is only equipped with 6 analog GPIO pins, and many more were needed for the testing apparatus, so the use of the MCP3008 was a cheap and effective way to increase the amount of analog input channels to the Arduino.

2.2.7 Chassis

A major point of discussion in this thesis was the dimensions of the testing apparatus. As such, a third-party company, Laetus AB, was utilized to construct the apparatus to meet the requirements as closely as possible. The purpose of the chassis is to house both the measurement software, and the unit under testing. Important aspects which were considered will be described in section 3.2.2 and 4.2.7, respectively.

2.2.8 Photoconductive Cell NSL19M51

The NSL19M51 is an unencapsulated photoconductive cell with a dark resistance of 20MΩ. It's power loss is 50Mw at 25°C with a nominal voltage of 100V. Its spectral top resides within 550nm. The output values vary between 0 and 1, 1 being complete darkness and 0 being flushed with light. The electrical characteristics, absolute maximum ratings, and physical description follows, as taken from the datasheet provided by the manufacturer.

Absolute Maximum Ratings

Operating & Storage Temp	-60°C to +75°C
Power Dissipation @ 25°C (1)	50 mW
Voltage (peak AC or DC)	100 V

Electrical Characteristics (T_A=25°C, source at 2854°K)(2)

Symbol	Parameter	Min	Typ	Max.	Units	Test Conditions
R_L	Light Resistance	20		100	KΩ	10 lux
			5		KΩ	100 lux
R_D	Dark Resistance	20			MΩ	10 sec after removal of light
λ_P	Peak spectral wavelength		550		nm	
Γ	Gamma		0.7			1 - 10 Lux
Γ	Gamma		0.7			10 - 100 Lux

Specifications subject to change without notice

Notes: (1) derate linearly to 0 at 75°C
(2) cells to be light adapted at 30 to 50 Ftc for 16 hrs minimum prior to electrical tests

Figure 8: [10] Farnell Element14's Absolute Max ratings and Electrical Characteristics.

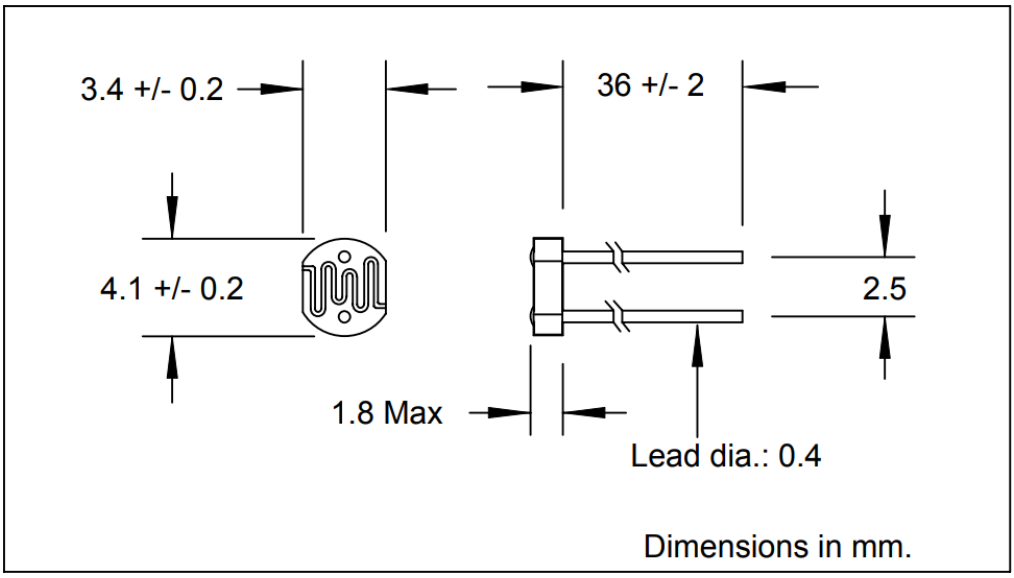

Figure 9: [11] Farnell Element14's NSL19M51 Dimensions.

2.2.9 Hermetic Photoconductive Cell NSL06S53

The NSL06S53 is a type 5 CdS hermetic photoconductive cell with a dark resistance of 20MΩ. It's power loss is 50mW at 25°C with a nominal voltage of 100V. Its spectral top resides within 550nm. The output values vary between 0 and 1, 1 being complete darkness and 0 being flushed with light. The electrical characteristics, absolute maximum ratings, and physical description follows, as taken from the datasheet provided by the manufacturer.

ABSOLUTE MAXIMUM RATING

(TA)= 23°C UNLESS OTHERWISE NOTED

SYMBOL	PARAMETER	MIN	MAX	UNITS
V_P	Voltage (peak AC or DC)		100	V
P_d	Power Dissipation @ 25°C (1)		50	mW
T_{Op}	Operating Temperature	-60	+75	°C
T_{Stg}	Storage Temperature	-60	+75	°C
T_S	Soldering Temperature (2)		+260	°C

Note:
(1) Derate linearly to 0 at 75°C
(2) >0.05" from case for <10 sec.
(3) Cells light adapted at 30 to 50 Ftc for 16 hrs minimum prior to electrical tests.
(4) Print "**NSL-06S53**" and date code "**YYWW**".

RELIABILITY

Contact API for recommendations on specific test conditions and procedures.

ELECTRO-OPTICAL CHARACTERISTICS

(TA)= 23°C, UNLESS OTHERWISE NOTED

SYMBOL	CHARACTERISTIC	TEST CONDITIONS	MIN	TYP	MAX	UNITS
R_L	Light Resistance	10 lux (3)	20		100	KΩ
		100 lux		5		KΩ
R_D	Dark Resistance	5 sec after removal of test light.	20			MΩ
λ_P	Spectral Peak			550		nm
Γ	Gamma	1 – 10 Lux		0.7		
Γ	Gamma	10 – 100 Lux		0.7		

Figure 10: [12] Farnell Element14's Absolute Maximum Ratings and Electrical Characteristics.

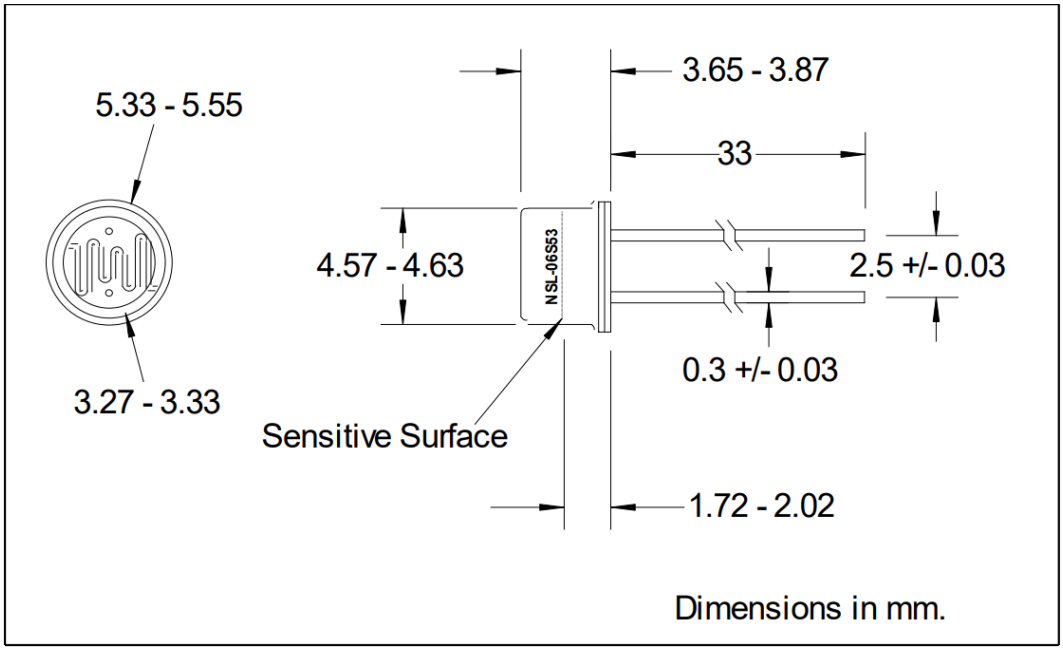

Figure 11:[13] Farnell Element14's NSL06S53 dimensions.

2.3 Software

2.3.1 Arduino

The Arduino software was written in Arduino's own IDE, which is freeware that can be downloaded from their website. The Arduino software is an open-source program which is described on the website that [14] it "makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software."

2.3.2 Visual Studio

Visual Studio is an IDE developed by Microsoft [15] and "is a creative launching pad that you can use to view and edit nearly any kind of code, and then debug, build, and publish apps for Android, iOS, Windows, the web, and the cloud." The programs were developed in C#, wherein the code was, in part, exported as a DLL file to be used by TestStand by the user, and in part be utilized as a UI to create camera profile data.

2.3.3 TestStand

TestStand is a program developed by National Instruments [16] that is "a ready-to-run test management software that is designed to help you develop automated test and validation systems faster." TestStand is used to create and execute software used for the testing process of different types of integrated hardware systems, streamlining the process of needing to test every separate part of the machine independently of each other. TestStand can integrate other programming languages, making use of libraries written, for example, in java or c#, which is what was done during this thesis.

3. Methods

3.1 Methods introduction

This chapter will explain the different stages in which work was carried out, tools used, and what was done within the stages. It will further explain methods of communication within Axis, and will include motivations as to why certain work methods were chosen over others.

3.2 Research Phase

Before the construction of the IR test apparatus, different aspects of the design were researched online and/or designed before ordering. The different phases of research will be explained below.

3.2.1 Sensors

Several different sensors were considered and tested before deciding on the type of sensor to be used in the IR test apparatus. Among these were the TSL260R IR light to voltage sensor, the NSL19M51 sensor, and the NSL06S53 sensor. To begin with, the sensors needed to be able to pick up on wavelengths between 850-940nm, because the IR diodes in Axis cameras can vary between these values. Before purchasing and testing these sensors, the datasheets were evaluated to ensure that their specifications matched the needs of the thesis work.

First, the TSL260R will be evaluated. Thereafter, the NSL19M51 LDR will be evaluated, and lastly, the NSL06S53 LDR. The method in which all three of these were tested will first be explained.

A rudimentary light-shielded testing box was built. It consisted merely of two cardboard boxes which were “light isolated” using white tack. A hole was then cut in the first box to insert the camera head containing different LED diodes, and a breadboard was mounted on the back of the second box. Sensors were attached to the breadboard, and wires were lead out of the box to the data collecting apparatus. The current to the separate three IR diodes within the camera were then varied between 100 to 3000mA, and the output from the sensors were noted and plotted into a graph. Considerations to be made were interference, such as if sensor 1 is in the first LEDs focus area, how much does sensor 2, which is in the second LEDs focus area, pick up on it? Ideally this should be as low as possible. These data points were then graphed and analyzed.

The TSL260R was thought to be a suitable sensor for the IR testing apparatus due to its irradiance responsivity (see figure 3, outlined in section 2.2.3). A very similar sensor, TSL262R, was to be tested due to its lower irradiance responsivity, although it wasn’t available for purchase. Other factors which were considered before the purchase of said sensors were the price of the sensors and ease of use. Because the sensor converts light intensity (irradiance) proportionally into voltage, it was thought that this sensor would be easy to work with. The relative responsivity was also taken into consideration, and thought to be optimal in that its peak responsivity was in the areas in which Axis’ IR diodes operate.

The NSL19M51 was considered for the IR testing apparatus due to its ease of use, in that it produces an analog resistance value based on the light intensity and that length of the diodes legs can be chosen by the operator. The NSL06S53 is a very similar photoconductive cell and was considered since its sensitive area is shielded, possibly producing more accurate values than the unshielded variant, the NSL19M51.

3.2.2 IR Test Apparatus Dimensions / Design

Figure 12: Chassis concept design by Alex DeVincenzo and Marcus Christensson.

The first aspect to be decided was how the sensors would be laid out within the chassis to best collect test data. To decide this, the requirements analysis was used which was provided by Axis (outlined in section 1.4). To calculate the angle of spread of the different IR diodes, a camera was placed 30cm away from the wall, and made to project an area of IR light onto the wall. The infrared light area was then traced, and trigonometry was used to calculate the vertical area of spread of the different areas of concentration of IR light. This was used to determine how closely spaced the light sensors would be placed within the chassis. A major positive aspect of the test design is that the resolution of the infrared light spread is higher compared to the previous test. In the previous test, the corners of the test box were much further away from the camera (approximately one meter) and the middle of the testing area was 30cm. With the current design, the distance between the camera and the measuring points are uniform, meaning that the resolution of the test is much higher than before.

The material of the chassis was also discussed with the external contractor and it was agreed upon that the material would not reflect infrared light, thereby ensuring minimal interference.

To develop the testing apparatus in the most efficient way, as stated before, an external contractor (Lucas Hermansson, Laetus AB) was employed to build the chassis for the sensors and the hardware. The requirements and design of the chassis were forwarded to said contractor, in which they attempted to adhere to them as strictly as physically possible.

3.2.3 Other Electronical Components

In this category, such hardware as motors, drivers and converters will be discussed. When considering which other hardware/software to use, the following points were considered:

- Relevance to the test construction (will the amount of needed current be available for the motor? Will the amount of wiring needed for certain sensors hinder the functionality of the chassis? Etc.)
- Cost
- Availability
- Ease of use
- Reliability
- Ease of implementation

Regarding the motor, ease of use was the most important aspect. Two different types of motors were considered, a four-phase and two-phase stepper motor, both with a 1.8° stepping angle. To make the decision, meetings with various employees at Axis were attended and the datasheets of the different motors were studied.

To connect the motor and control the motor, a motor driver circuit was needed. Several different solutions to this problem were considered, with help of Axis employees. One of the solutions proposed was a Stegia step motor driver, another was a motor driver developed at Axis, and the third proposed solution was the Big Easy Driver, a programmable motor driver circuit compatible with the Arduino. Trial and error combined with ease of implementation and ease of use were used to make this choice.

To connect all the sensors to the controller board, ADCs were needed. After the choice of microcontroller, the biggest factor to consider was which ADC would be appropriate for the chosen microcontroller, i.e. which would be easiest and most practical to implement. This was done via internet research, which consisted of checking to see which converter was the most popular with users, had the best reviews and software support available online.

4. Analysis

4.1 Introduction

Here, results from different tests, problems and their solutions, and different design choices will be described.

4.2 Hardware

4.2.1 Raspberry Pi

After working with the Raspberry Pi for approximately two weeks and consulting with others within the organization, it was decided that work with the Raspberry Pi would be discontinued, because the memory card oftentimes becomes corrupt. Thereafter it was decided to program the apparatus via an Arduino Uno.

4.2.2 Arduino Uno

Working with the Arduino was straightforward. One problem which occurred was that because the light sensors gave an analog output, and a majority of the Arduino's GPIO ports are digital, there would not be enough analog GPIOs on the Arduino to be able to accommodate all the sensors. The solution to this problem was the utilization of four ADC circuits. This way, eight different analog signals could be sent to the Arduino using only one digital input and three parallel inputs.

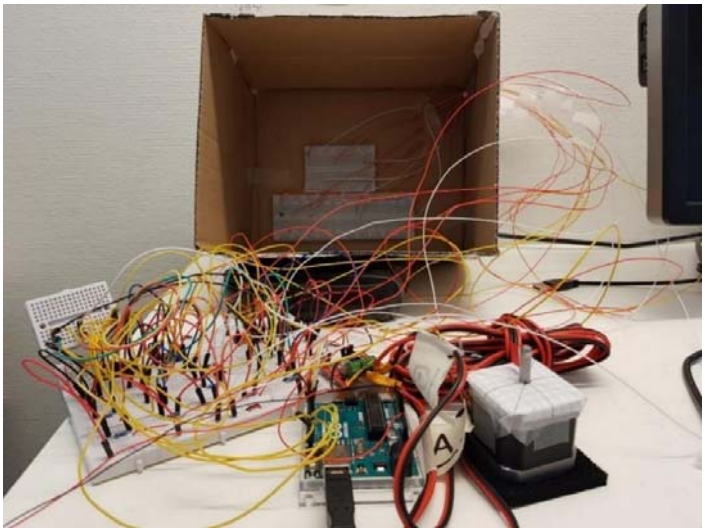

Figure 13:An inefficient method of connecting the Arduino to the sensors. Front, left: The breadboard with resistors, ADCs, and the Big Easy Driver. Front, middle: The Arduino. Front, right: The stepper motor. Back, within the box: Two breadboards connecting light sensors to the Arduino.

4.2.3 Wiring

During the testing of the different light sensors, it was noticed that using breadboards wasn't the most efficient use of space, and that the number of sensors which needed to be connected (50) would make it highly inefficient to continue in this manner.

It was decided that during production, a PCB would be designed and utilized instead of a breadboard, to make the design more efficient and compact.

4.2.4 The PCB

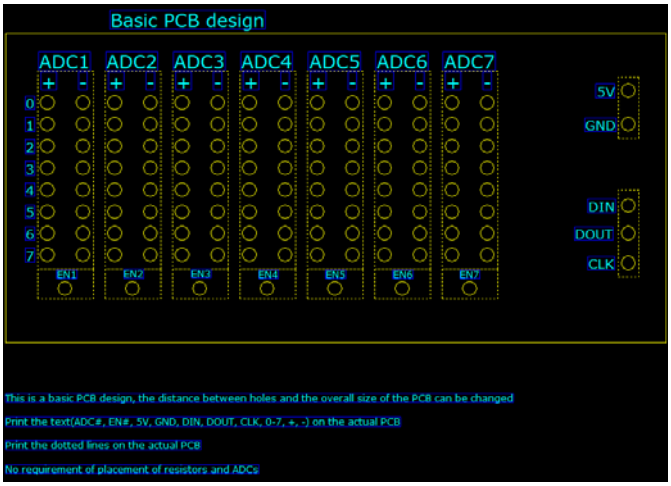

Figure 14: The basic design of the printed circuit board.

The printed circuit board was designed in PADS (personal automated design solutions) PCB design software. The PCB was designed by Marcus. The purpose of the PCB, as previously stated, was to reduce the amount of wiring thereby reducing the complication of the connection between the Arduino and the rest of the electrical components. A basic overview of the PCB can be seen in figure 15.

Each yellow rectangle above (ADC1, ADC2 etc.) represent an ADC, and the positive (left) side of the IC shows the 8 channels in which the sensors were to be connected.

When the sensor values are gathered from a ADC, the Arduino sets the Enable input to high, turning it on. The Arduino then sends the settings for the session with the DIN input. After that, the ADC sends to sensor values serially through the DOUT connector. All this is done at the rate set to the CLK input by the Arduino. Lastly, the Enable input is set to low when the session is complete. Because only one ADC is enabled at one time, every ADC can use the same DIN, DOUT and CLK connectors. However, every ADC needs a separate Enable connector.

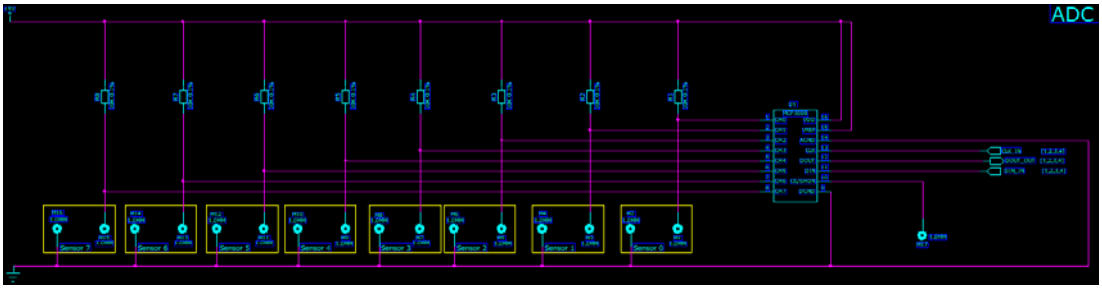

Figure 15: The circuit of an ADC

Axis has a library of components used for circuit drawing in PADS, but the ADC (MCP3008) was not included in this library. This meant that the ADC needed to be added (with a datasheet and a ROHS report).

Once the PCB design and circuitry is completed, it will be sent to a CAD-engineer on Axis to draw the PCB card. Lastly, this CAD design will be sent to a factory to produce the PCB.

4.2.5 Light Sensors

It was found that the TSL260R IR sensor was too sensitive for the needs of the IR test apparatus. When exposed to one of the three IR diodes with only 1000mW of energy, the sensor became saturated and thus unable to detect any further changes. The NSL19M51 and the NSL06S53 were tested in the same manner, and showed similar results relative each other. The difference in the two was that the hermetic, shielded sensor provided better isolation from outside light sources than the unshielded photoconductive cell. It was then decided that the hermetic photoconductive cell, NSL06S53, would be utilized because it suited the needs of the IR test fixture the best. Graphs of the two sensors behavior as a function of light intensity is included in sections 5.1.3 and 5.1.4, respectively.

4.2.6 Stepper Motor FL60STH45-2008AF / Choice of Motor and Driver

Although the stepper motor together with the big easy driver supported microstepping (the ability to divide a step (1.8°) further into smaller steps), it was decided that this functionality was not needed. Because the motor and the driver required more input current than the Arduino could provide, it was decided than an external current source would be used.

4.2.7 Chassis

The chassis arrived on the 8th of May 2018, which was approximately four weeks later than its projected arrival date. Because of the late arrival, an insufficient amount of work and testing was done on the construction. The chassis was delivered without housing for the sensors or screws to mount the motor, so an additional wait time was required to get the full solution functional. Furthermore, the mounting holes for the motor were incorrectly spaced, so an additional day passed before the motor could be mounted, which also halted development of the full solution.

4.3 Software

4.3.1 Arduino

One aspect which was decided was whether to use pre-written libraries for motor control or if the code was to be designed from scratch. It was decided that the motor control portion of the Arduino software would be coded from scratch, but that libraries would be used to communicate with the ADCs. The SPI library was used, which is included in the Arduino's own IDE. To simplify the code which was run in the main loop, methods to control the motor were to be defined. This code can be found in appendix 9.1.

4.3.2 Visual Studio

To bridge the Arduino communication and data over to TestStand, the testing software used by the department which will be utilizing the IR test apparatus, libraries would need to be coded in Visual Studio as DLL files for the two IDEs to communicate. The most common language used within the department was said to be C#, so the library files were to be coded in this language.

Because the test required measuring sensor data, comparing it with a set of data, and returning a results matrix, it was decided that a windows form application would be developed to best display and input camera profile data.

4.3.3 TestStand

A majority of the TestStand coding was done by Marcus. The main sequence in the test runs a series of different test and modules. If any of these separate modules fail, the whole test will fail.

Figure 16: An example of a failing test.

The test sequence used different modules to call the methods written in the DLL.

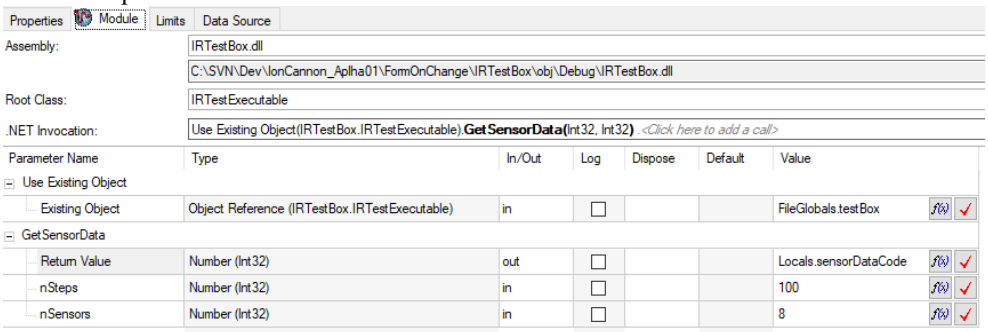

Figure 17: A module from TestStand.

The DLL is called in the Assembly tab and the object is chosen in the .NET Invocation tab. Parameters can be set by writing the value or variable under the Value tab (nSteps and nSensors). The returned value can be stored in a variable (Locals.sensorDataCode in this case) and used or presented in a different module.

A subsequence is a collection of modules that makes it easier to read and understand the main sequence. An example:

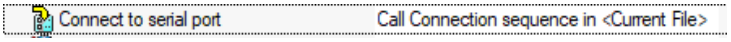

Figure 18: A subsequence in a main sequence.

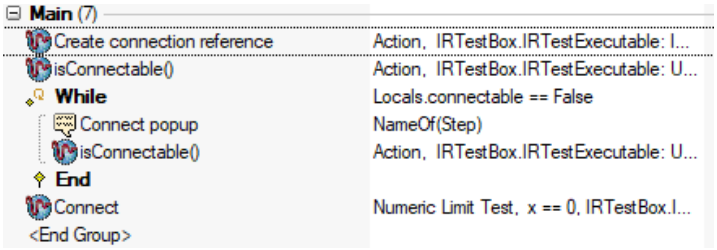

Figure 19: The subsequence expanded.

All that can be seen (unless you double-click it) in the main sequence is figure 19 but, everything in figure 20 is what is executed.

4.3.4 PADS

PADS (personal automated design solutions) is a software mainly used to draw and simulate electrical circuits and construct CAD schematics for PCBs. Axis has a library of components used for circuit drawing in PADS, but the ADC (MCP3008) was not included in this library. This meant that the ADC needed to be added (with a datasheet and a ROHS report). Adding components can be done by using the "Add Part" button and searching in the Axis library. Here's an example:

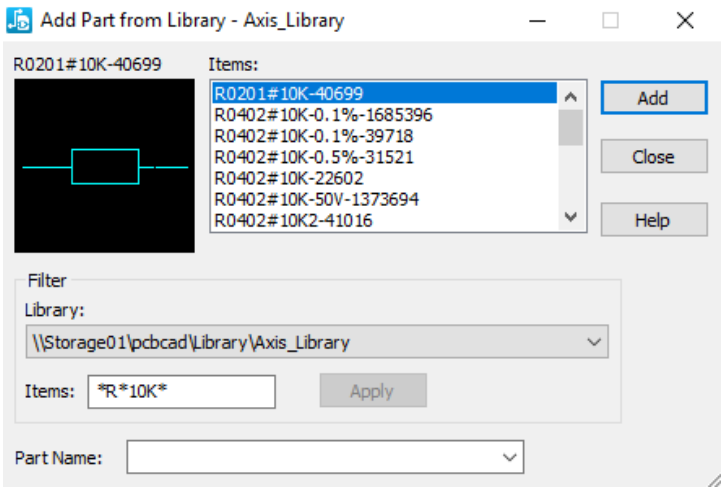

Figure 20: Add Part window

The circuit can be split into different sheets to fit the whole circuit in the same schematic file. These sheets can be connect with "off-page". Here's an example:

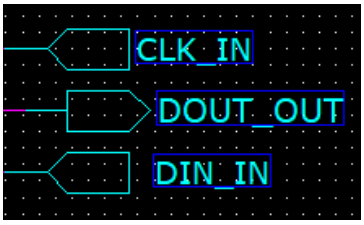

Figure 21: Off-page symbols.

The "arrows" show which direction the signal is expected to run. The DOUT_OUT sends a signal off the sheet while the CLK_IN and DIN_IN expects a signal from different sheet.

5. Results

Here, the results from the different hardware and software development will be presented.

5.1 Hardware

5.1.1 Stepper Motor FL60STH45-2008AF & Chassis

Before mounting the motor, a rudimentary test was done to ensure that it rotated 180°, stopped for 5 seconds, and returned to its home position. This was done by connecting a small red flag to the motor axel and watching to ensure the correct positions were attained. However, once the motor was mounted to the chassis and connected to the metal arch which housed the light sensors, its behavior became erratic. It would, at times, behave as it should, but at times behave randomly. The axel was then detached from the metal arch, but the motor remained mounted to the chassis, and the flag test was then carried out once more.

The flag test confirmed once again that the motor was correctly programmed. It rotated 180°, and returned to its home position after several tests. It was then determined that the issue was mechanical. Lubrication was applied to the axel fasteners where needed, and the test was run again after connection to the axel, with similar results. It was then noticed that inclusion of the screw which fastens the motor axel to the rotating metal band is what caused the mechanical issues. When the motor was removed from the chassis, it was noticed that there was a small flat section on the motor axel which needed to be positioned right relative to the screw. Once this was noticed, the issue was resolved.

5.1.2 Photoconductive Cell NSL19M51

The values of the sensitivity test are displayed by the following graph. Only LED2 was activated, as this is the most intense IR diode in the camera which was tested, and thought to give more accurate measurements than the other 2 less focused diodes.

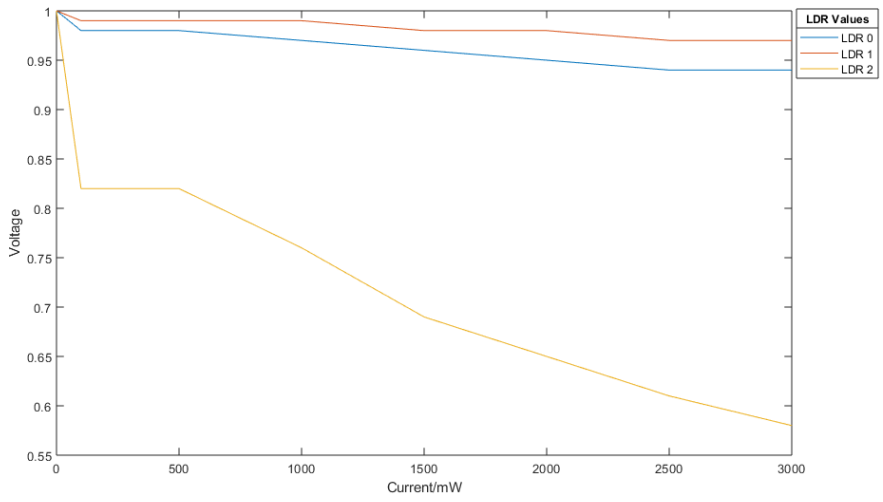

Figure 22: Output voltage vs current to diode 2.

Diode 2 was activated, and LDR 2 was placed in the area in which the most IR light was concentrated. As seen on the graph, LDR 2 responded the most, but LDR 0 and 1 were marginally affected. It should be noted that LDR 0 was closer to LDR 2 than LDR 1 was.

5.1.3 Hermetic Photocell NSL06S53

The same test which the NSL19M51 was implemented on the NSL06S53 sensors. The values are displayed in the following graph.

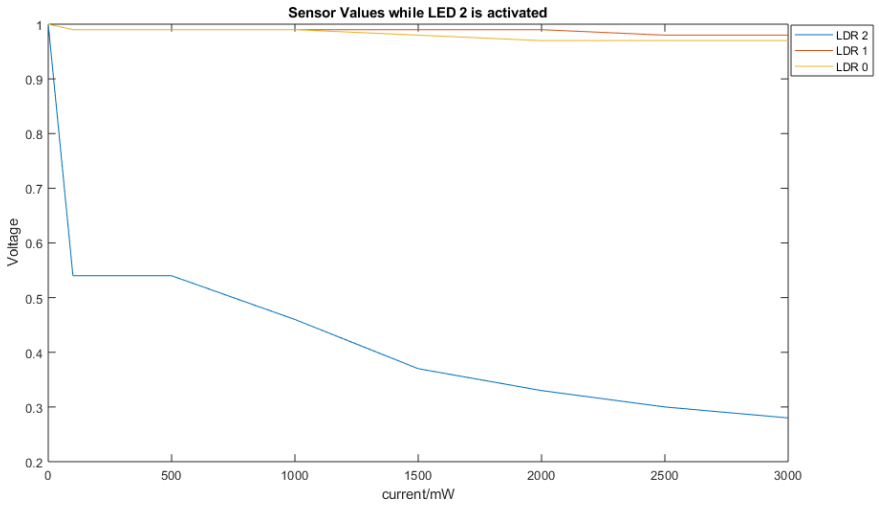

Figure 23: Sensor values of the NSL06S53 while varying current to diode 2.

As the graphs show, the shielded sensors are both more sensitive to light intensity, and they are more isolated from each other. For these reasons, it was decided to use the NSL06S53 sensors in construction of the IR test apparatus.

5.1.4 The PCB

The actual PCB was not built in time by the end of this bachelor thesis, but will be built during the summer.

5.1.5 Arduino Uno

During the end of the thesis work, the Arduino Uno's GPIO pins were not enough for all the functionality which is required for the infrared test box. Four ADCs, as well as the big easy driver/motor were connected, and the Arduinos GPIO ports were filled. Further input/output sensors, such as positional sensors to detect where the metal frame holding the sensors is, and a switch to detect if the chassis door is open, are also needed. Because of this, it is decided that in future development, an Arduino Mega will be used.

5.1.6 Chassis

As stated in 4.2.7, the chassis was delivered approximately one month late. Because of this in combination with the fact that there were missing components and components which did not work as they were thought to work, a large amount of work needs to be done with the chassis to get it test-site ready, such as installing the infrared protection switch on the door, mounting and programming the position detectors for the metal frame, mounting the Arduino to the side of the fixture, and drilling a hole for the cables. However, a rudimentary proof of concept was

constructed which gathers sensor data while rotating the metal band with sensors mounted to it 180°.

Below, a size comparison of one of the old IR test chassis and the current IR test chassis will be shown, as well as an image showing the test being carried out from the camera's point of view within the closed chassis.

Figure 24: A front and side view of one of the old IR Test chassis.

Figure 25: Complete view of the new chassis.

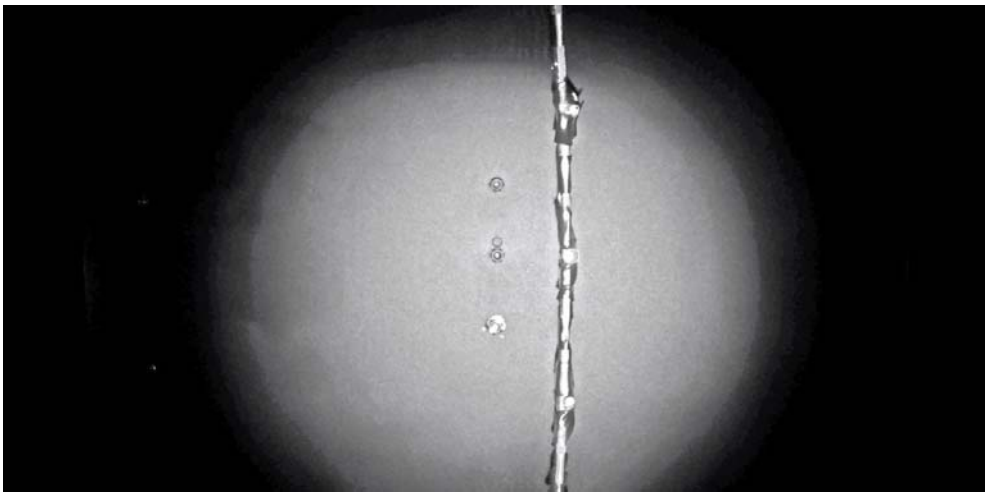

Figure 26: A still picture from a video of a camera recording the testing process inside of the chassis. The vertical object is the sensor band, and the white area is the area of the chassis door which is illuminated with infrared light.

5.2 Software

5.2.1 Arduino

For code simplification, the following methods were coded: methods defined to control motor movement, reset driver pins, reset the axel position, and to read a single value from one channel of an ADC (a light sensor value). During testing, an issue regarding synchronization of Visual Studio code and the Arduino IDE was noticed. The Arduino would begin measurements as soon as it was supplied with current, which was problematic as the data gathering software would need to be started, manually, at the same time. A solution to this issue was to make the Arduino wait until it got a verification signal sent to it via the serial port from the Visual Studio code.

An additional complication which arose was the use of delays within the Arduino code. Because the sensor data and the motor needed to be monitored and controlled continuously, the use of delays was impractical because it paused the systems functionality. A solution to this issue was the use of timers instead of delays, where appropriate. Screenshots of relevant code can be found in appendix 9.1.

5.2.2 Visual Studio

A majority of the C# code was programmed by Alex. A backend UI was coded to consist of functions to save, load, create and display sensor data and camera profile data. The UI shows results of calculations of test data between the sensor data and profile data, and where the data has been saved/loaded on the user's computer. This part of the program is only to be used by the PTG department to create camera profiles. Later, in production, only the TestStand code and the DLL file will be used.

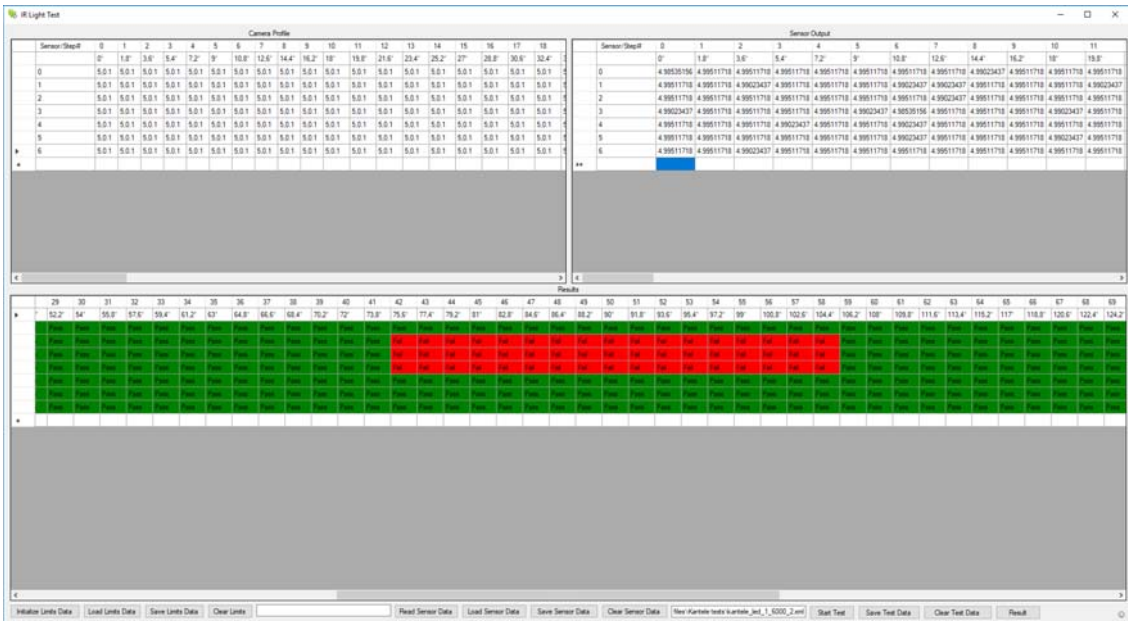

Figure 27: The backend UI of the IR Light Test program before camera profile calibration.

A secondary class called IRTestBox, which was exported as a DLL file, was written and used within the UI. This was used in combination with TestStand as a part of the full testing solution.

To create a camera profile, a camera which was known to be working was mounted to the test box. The infrared lighting was turned on and the same test was executed four times. LED1 in the camera was turned on, and given 6000mW of energy. The mean values of the relevant positions were calculated, as well as the deviations, which were rounded up and set as the tolerance for the different relevant positions. Thereafter, the camera profile data was used to carry out 9 different tests within the UI. Five of the tests carried out after the camera profile was established was carried out in the same manner as the first five tests used to gather data and create the profile LED1 activated with 6000mW on five separate occasions.

These tests passed, as expected. Thereafter, LED1 was shut off and LED0 was turned on with 6000mW and the test was executed. This test failed. The same test was done with LED2, with similar failing results. Thereafter, all the LEDs were turned off and the test was executed, with failing results as expected. Finally, all the LEDs were turned on and the test was executed, with failing results as expected. This camera profile was later used in TestStand.

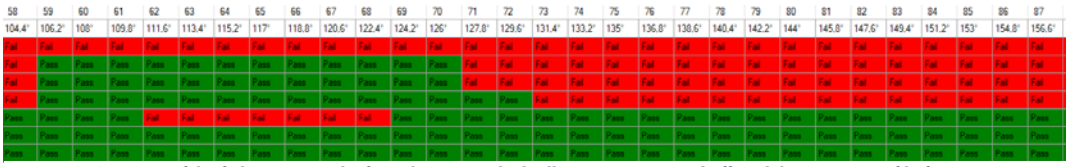

Figure 28: A snippet of the failing test results from the test in which all LEDs were turned off, and the camera profile for LED1 at 6000mW was used.

5.2.3 TestStand

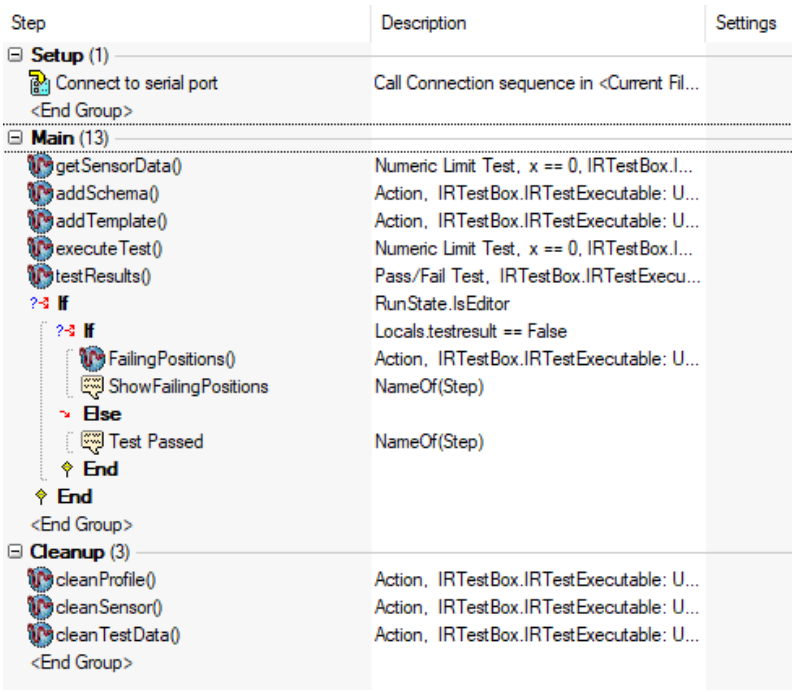

Figure 29: The TestStand main sequence.

The sequence was started by running the Setup, which includes the “Connect to serial port” subsequence (explained in 4.3.3), once the USB cable from the Arduino was connected to the test station (computer running the test), the test can start.

The step getSensorData() started rotating the step motor and gathers data from the sensors after each step. After that, a template was loaded with the addSchema() and addTemplate() steps. The executeTest() step compared the sensor data and the template data and stored the pass/fail table.

The testResults() step analysed the table to see if any position had failed and returns a true if every position had passed. The FailingPositions() and ShowFailingPositions steps are only for developmental purposes and will not be executed in the actual factory. This was done by adding the RunState.isEditor variable.

Lastly, the Cleanup was executed to release any used memory and prepare for future testing.

Some steps have a “Numeric Limit Test” to ensure that the step was executed correctly. For example, the getSensorData() step doesn’t test anything but is needed for the testResults() step. If the data was not gathered successfully, the DLL method will return something other than 0 and fail the x=0 condition for the test. This results in it being easier to troubleshoot a sequence that crashes during the test. Different numbers returned by the DLL method means different errors.

5.2.4 Software Cohesion

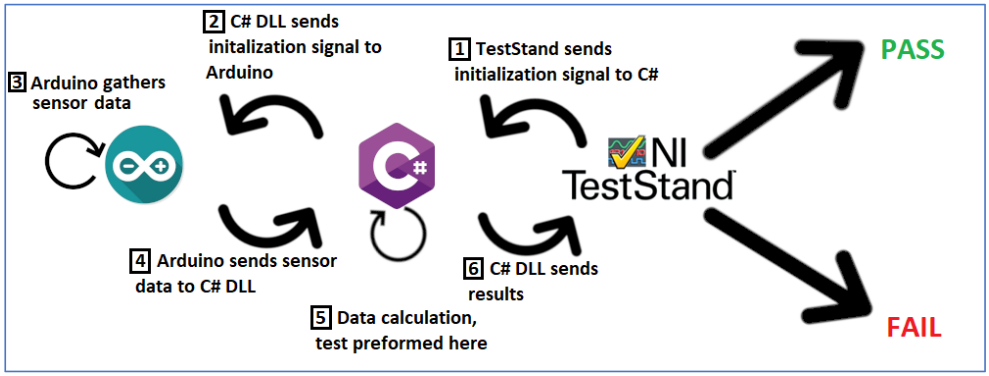

Figure 30: A visual representation of the communication between different programs.

As illustrated above, there are several facets to the programming of the infrared test apparatus. To begin with, the Arduino needed to be programmed to gather and send sensor data, and control motor functions. After this was programmed, a C# DLL was programmed to store and calculate test results, as well as create user camera profile data. (In this graphic, C# represents only the DLL and not the UI.) The last step is that a TestStand sequence was programmed, which utilizes both the C# DLL and the Arduino code, and returns a pass or fail result based on test parameters. Communication between the Arduino and different programs were done via serial communication.

6. Conclusion

6.1 Overview

In conclusion, the following points were addressed:

1. The solution is generic, i.e. able to test different models of network security cameras due to the nature of the metal band which houses the sensors. The distance between the sensors and the camera is always the same, giving it better resolution than the previous test methods.
2. The solution can test lenses which spread light in different angles within a specific degree of accuracy. (10°-150°) It's able to measure light distribution 180° from the front of the camera housing.
3. The solution is not yet able to protect users from IR light, but a switch is added to the chassis and will be developed before being sent to test sites. This switch will turn off all electrical apparatus if the door is opened, thus protecting operators from being exposed to infrared light.
4. The hardware utilizes "off the shelf" components.
5. The hardware is not 0.5x0.5x0.5 meters or less – the chassis is 0.6x0.32x0.6 meters. This aspect was deemed less important than the ability to test the cameras within 180° in the future.
6. The module can test IR light spread in relation to the optical module (if the camera and IR light diode are well aligned) and the cameras chassis, since it is the sensors which are testing the infrared light distribution and not the cameras own lens.
7. The module isn't overly complicated to build.

A full hardware and software solution was designed, programmed, and built. The chassis, hardware, and software together encompass a solution which solves most of the problems outlined in section 1.4, the problem description. Those which were not addressed will be worked in the future. The result of the work done on this thesis will be used within Axis Communications AB to test different cameras infrared lighting in a more effective way, which was the broad goal of the thesis.

6.2 Reflection over Ethical Aspects

The positives of this thesis are that less material (wood, metal etc.) is needed to test different cameras, which leads to both financial and environmental gains. The solution explored by this thesis also costs the company less time, in that testing is streamlined (one test apparatus for all cameras, which means less physical setup required for testing) so that each test can be completed in a quicker, more efficient manner.

However, because of the additional electrical components (the motor, 50 sensors, Arduino etc.) the infrared test box consumes much more power and electrical resources. Nevertheless, if ethically sourced, the additional electrical resources don't necessarily need to be a negative aspect of the thesis work.

This is an area which Axis Communications is already proactive. According to their 2014 sustainability report [17], "In 2013, we at Axis began our work to ensure that no rare earth minerals used in our products originate from conflict areas, in support of the electronics industry's initiative to prevent trade in these minerals."

Not only does Axis ethically source their materials by choosing not to buy from conflict zones, they also strive to avoid using materials which have a negative impact on the

environment. “In order to ensure an effective and comprehensive overview of materials used in Axis’ products, the company made a significant effort in 2014 to improve and systematize the overview of product components and materials. This effort has enabled Axis to more easily adjust according to new legislation and research, but also to implement its own, more comprehensive phase-out of different harmful materials.” (Axis Communications AB, “Axis Sustainability Report 2014”, page 25, accessed 3 May 2018)

6.3 Future Development

The potential for development with this thesis is very promising. Several different components will be improved upon. Because the PCB was not ordered in sufficient time to be implemented, this is one of the areas which will be improved. Also, due to the late delivery, the chassis itself was insufficiently tested/developed, thus was not sufficiently probed for further weaknesses and key functionality (such as shutting off the hardware if the door is open, adding moveable sensors to the sensor band) were not able to be added. This will be done in the future. Another component which can be researched is whether the encasing on the sensors is adequate. If not, further development will be pursued in this respect. The chassis’ light isolating properties are also an aspect that needs to be investigated.

7. Terminology

GPIO – General Purpose Input/Output. Programmable pins that can send data to the board or send data from the board.

CPU – Central Processing Unit.

RAM – Random Access Memory.

LDR - Light Dependent Resistor.

LED – Light Emitting Diode.

IC – Integrated Circuit.

PCB – Printed Circuit Board

UUT/DUT – Unit Under Test / Device Under Test

IDE – Integrated Development Environment

DLL – Dynamic Link Library

SPI – Serial Peripheral Interface

UI – User Interface

PWM – Pulse Width Modulation

PTG – Production Test Group

ROHS – Restriction of Hazardous Substances (in electronic equipment)

8. References

0. Raspberry Pi Foundation, 'About Us', [Online] Available: https://www.raspberrypi.org/about/ [accessed 21st May 2018]
1. Jameco Electronics, 'How to Navigate your Raspberry Pi 3 Model B', [Online] Available: https://www.jameco.com/Jameco/workshop/circuitnotes/raspberry-pi-circuit-note.html [accessed 15th February 2018].
2. Arduino, 'Arduino Uno Rev 3', [Online] Available: https://store.arduino.cc/arduino-uno-rev3 , [accessed 22nd February 2018].
3. Arduino, 'Introduction', [Online] Available: https://www.arduino.cc/en/Guide/Introduction , [accessed 22nd February 2018].
4. Digi-Key Electronics, 'ams TSL260R-LF', [Online] Available: http://ams.com/eng/content/download/250428/976477/142384 , [accessed 16th February 2018].
5. Digi-Key Electronics, 'ams TSL260R-LF', [Online] Available: http://ams.com/eng/content/download/250428/976477/142384 , [accessed 16th February 2018].
6. Digi-Key Electronics, 'ams TSL260R-LF', [Online] Available: http://ams.com/eng/content/download/250428/976477/142384 , [accessed 16th February 2018].
7. Digi-Key Electronics, 'ams TSL260R-LF', [Online] Available: http://ams.com/eng/content/download/250428/976477/142384 , [accessed 16th February 2018].
8. ShinanoKenshi, '2 Phase Hybrid Stepping Motor', [Online] Available: http://www.ap.shinanokenshi.com/products/hb_step/pdf/sst43d_bi.pdf , [accessed 27th February 2018].
9. Allegri MicroSystems LLC, 'DMOS Microstepping Driver with Translator and Overcurrent Protection', [Online] Available: https://cdn.sparkfun.com/datasheets/Robotics/A4988-Datasheet.pdf , [accessed 27th February 2018].
10. Farnell Element14, 'NSL 19M51. - LDR, 20 MΩ, 50 mW, 100 V', [Online] Available: http://www.farnell.com/datasheets/77395.pdf?_ga=2.225165594.1667130889.1518684579-1757521863.1518428390 , [accessed 15th February 2018].
11. Farnell Element14, 'NSL 19M51. - LDR, 20 MΩ, 50 mW, 100 V', [Online] Available: http://www.farnell.com/datasheets/77395.pdf?_ga=2.225165594.1667130889.1518684579-1757521863.1518428390 , [accessed 15th February 2018].
12. Farnell Element14, 'NSL 06S53. - LDR, 20 Mohm, 50 mW, 100 V', [Online] Available: http://www.farnell.com/datasheets/2013773.pdf?_ga=2.9910909.971296038.1518776547-1757521863.1518428390&_gac=1.54234842.1518779837.CjwKCAiAn5rUBRA3EiwAUCWb26XMeUJrK7GKBrnG2E1L431Tqta8pwMsl7VHS21mKSdyogWam0RuehoCaZAQAvD_BwE , [accessed 15th February 2018].
13. Farnell Element14, 'NSL 06S53. - LDR, 20 Mohm, 50 mW, 100 V', [Online] Available: http://www.farnell.com/datasheets/2013773.pdf?_ga=2.9910909.971296038.1518776547-1757521863.1518428390&_gac=1.54234842.1518779837.CjwKCAiAn5rUBRA3EiwAUCWb26XMeUJrK7GKBrnG2E1L431Tqta8pwMsl7VHS21mKSdyogWam0RuehoCaZAQAvD_BwE , [accessed 15th February 2018].
14. Arduino, 'Download the Arduino IDE', [Online] Available: https://www.arduino.cc/en/Main/Software , [accessed 06th April 2018].
15. Microsoft, 'Visual Studio IDE overview', [Online] Available: https://docs.microsoft.com/en-us/visualstudio/ide/visual-studio-ide , [accessed 06th April 2018].
16. National Instruments, 'What is TestStand?', [Online] Available: http://www.ni.com/teststand/whatis/ , [accessed 27th March 2018].
17. Axis Communications AB, "Axis Sustainability Report 2014", page 27, [Online] Available: https://www.axis.com/files/brochure/report_axis_sustainability_2014_1504.pdf , [accessed 3rd May 2018].

9. Appendix

9.1 Arduino Code

Only snippets of code will be shown here, as well as a description of the function of said code.

```
void moveForward(double posRad, int timeBetween) //pos = position in degrees.
{
  pos = posRad * (180 / pi); //Convert from radians to degrees
  stepsAmnt = pos / 1.8; //One full step is 1.8 degrees

  digitalWrite(dir, LOW); //Pull direction pin low to move "forward"
  unsigned long previousTime = 0; //Keeps track of time OF last LOW step
  unsigned long elapsedTime; //Keeps track of time SINCE last LOW step
  boolean firstStep = 0; //Used for the first step
  int cntr = 0; //Used to count the steps

  do //Loop the forward stepping pos degrees
  {
    if (firstStep == 0)  //To make sure the first step is executed.
    {
      elapsedTime = timeBetween + (timeBetween / 2);
      firstStep = 1;
    }
    else {
      elapsedTime =  (micros() - previousTime);
    }
    if (elapsedTime >= timeBetween && elapsedTime < (2 * timeBetween))
    {
      digitalWrite(stp, HIGH); //Trigger one step forward
    }
    if (elapsedTime >= (2 * timeBetween))
    {
      digitalWrite(stp, LOW); //Pull step pin low so it can be triggered again
      cntr++;
      time = 0; //Is this needed?
      previousTime = micros();
    }
  } while (cntr < stepsAmnt);
}
```

Figure 31: The function which moves the motor axel forward 180 degrees. To be used in the main loop.

```
void resetPosition(int timeBetween) //Reverse default microstep mode function (backwards pos degrees)
{
  int state = digitalRead(dir);
  if (state == HIGH)  //Changes the direction of the motor
  {
    digitalWrite(dir, LOW);
  }
  else if (state == LOW)
  {
    digitalWrite(dir, HIGH);
  }
  unsigned long previousTime = 0;
  unsigned long elapsedTime;
  boolean firstStep = 0;
  int cntr = 0;
  do //Loop the forward stepping pos degrees
  {
    if ( firstStep == 0)
    {
      elapsedTime = timeBetween + (timeBetween / 2); //To make sure the first step is executed
      firstStep = 1;
    }
    else {
      elapsedTime =  (micros() - previousTime); // Time since the last LOW step
    }
    if (elapsedTime >= timeBetween && elapsedTime < (2 * timeBetween))
    {
      digitalWrite(stp, HIGH); //Trigger one step forward
    }
    if (elapsedTime >= (2 * timeBetween))
    {
      digitalWrite(stp, LOW); //Pull step pin low so it can be triggered again
      cntr++;
      time = 0; //Is this needed?
      previousTime = micros();
    }
  } while (cntr < 100);
  pos = 0;
```

Figure 32: The function which resets the motor axel to its original position.

```
int adc_single_channel_read(byte readAddress, int adc)
{
  byte dataMSB =    0;
  byte dataLSB =    0;
  byte JUNK    = 0x00;
  SPI.beginTransaction (MCP3008);
  if (adc == 1) //Here we decide which converter to read from, i.e converter one
  {
    digitalWrite         (CS_MCP3008_1, LOW);
    SPI.transfer         (0x01);                              // Start Bit
    dataMSB =            SPI.transfer(readAddress << 4) & 0x03;  // Send readAddress and receive MSB data, masked to two bits
    dataLSB =            SPI.transfer(JUNK);                  // Push junk data and get LSB byte return
    digitalWrite         (CS_MCP3008_1, HIGH);
    SPI.endTransaction   ();
  } else if (adc == 2)  //Converter 2
  {
    digitalWrite         (CS_MCP3008_2, LOW);
    SPI.transfer         (0x01);                              // Start Bit
    dataMSB =            SPI.transfer(readAddress << 4) & 0x03;  // Send readAddress and receive MSB data, masked to two bits
    dataLSB =            SPI.transfer(JUNK);                  // Push junk data and get LSB byte return
    digitalWrite         (CS_MCP3008_2, HIGH);
    SPI.endTransaction   ();
  }
```

Figure 33: The function which is used to read data from the AD converters. There are 4 in total, but due to the repetitive nature of the code the last two have been omitted.

```
void readAndPrint()
{
  double vRef        = 5.01;
  int    adc_reading7_1 = 0;  //initialising value of channel 7 of AC 1
  int    adc_reading6_1 = 0;  //channel 6 of MCP 1
  int    adc_reading0_2 = 0;  //channel 0 of MCP 2
  int    adc_reading1_2 = 0;  //channel 5 of MCP 1
  int    adc_reading2_2 = 0;  //channel 0 of MCP 2
  int    adc_reading6_3 = 0;  //channel 6 of MCP 3
  int    adc_reading7_4 = 0;  //channel 7 of MCP 4
  int    adc_reading6_4 = 0;  //channel 6 of MCP 4

  adc_reading7_1 = adc_single_channel_read(adc_single_ch7_1, 1);  //reading sensor value
  adc_reading6_1 = adc_single_channel_read(adc_single_ch6_1, 1);  //reading sensor value
  adc_reading0_2 = adc_single_channel_read(adc_single_ch0_2, 2);  //reading sensor value
  adc_reading1_2 = adc_single_channel_read(adc_single_ch1_2, 2);  //reading sensor value
  adc_reading2_2 = adc_single_channel_read(adc_single_ch2_2, 2);  //reading sensor value
  adc_reading6_3 = adc_single_channel_read(adc_single_ch6_3, 3);  //reading sensor value
  adc_reading7_4 = adc_single_channel_read(adc_single_ch7_4, 4);  //reading sensor value
  adc_reading6_4 = adc_single_channel_read(adc_single_ch6_4, 4);  //reading sensor value


  Serial.println((adc_reading7_1 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading6_1 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading0_2 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading1_2 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading2_2 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading6_3 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading7_4 * vRef) / 1024, 8);  //Convert from binary to decimal number
  Serial.println((adc_reading6_4 * vRef) / 1024, 8);  //Convert from binary to decimal number
}
```

Figure 34: The function which retrieves and serially prints sensor values from the AD converters.

```
void setup() {
  SPI.begin      (); //Opens the SPI port
  Serial.begin   (57600); //Opens arduino for serial communication with computer
  //AD converter 1 is set
  pinMode        (CS_MCP3008_1, OUTPUT);
  digitalWrite   (CS_MCP3008_1, LOW);
  digitalWrite   (CS_MCP3008_1, HIGH);
  //AD Converter 2 is set
  pinMode        (CS_MCP3008_2, OUTPUT);
  digitalWrite   (CS_MCP3008_2, LOW);
  digitalWrite   (CS_MCP3008_2, HIGH);
  //AD converter 3 is set
  pinMode        (CS_MCP3008_3, OUTPUT);
  digitalWrite   (CS_MCP3008_3, LOW);
  digitalWrite   (CS_MCP3008_3, HIGH);
  //AD converter 4 is set
  pinMode        (CS_MCP3008_4, OUTPUT);
  digitalWrite   (CS_MCP3008_4, LOW);
  digitalWrite   (CS_MCP3008_4, HIGH);

  //assign output pins for stepper motor
  pinMode(stp, OUTPUT);
  pinMode(dir, OUTPUT);
  pinMode(MS1, OUTPUT);
  pinMode(MS2, OUTPUT);
  pinMode(EN, OUTPUT);
  //Serial.begin(57600);
  setStepToOne();
}
```

Figure 35: The setup stage of the Arduino code. setStepToOne() disables microstepping in the big easy driver.

```
void loop() {
  digitalWrite(EN, LOW);
  double currentPos = 0;
  if (Serial.available() == 1) {
    int stepSetting = Serial.read();
    double nbrOfSteps = 100;
    double steplings = pi / nbrOfSteps;
    switch (stepSetting)
    {
      case 48:
        break;
      case 49:
      case 50:
      case 51:
      case 52:
        do {
         //Sensor value code readings
          readAndPrint();
          moveForward(pi / nbrOfSteps , 5000);
          delay(1);
          currentPos = (currentPos + pos);
        } while (currentPos < 180);
        readAndPrint();
        resetPosition(1000);
        resetBEDPins();
        break;
    }
  }
}
```

Figure 36: The main loop of the Arduino program. The different cases handled microstepping, but in that it was disabled, they were made redundant. The code reads the values from the AD converters and moves the axel forward. After 180 degrees, it resets the axel.

9.2 Chassis Design

9.2.1 Initial Design

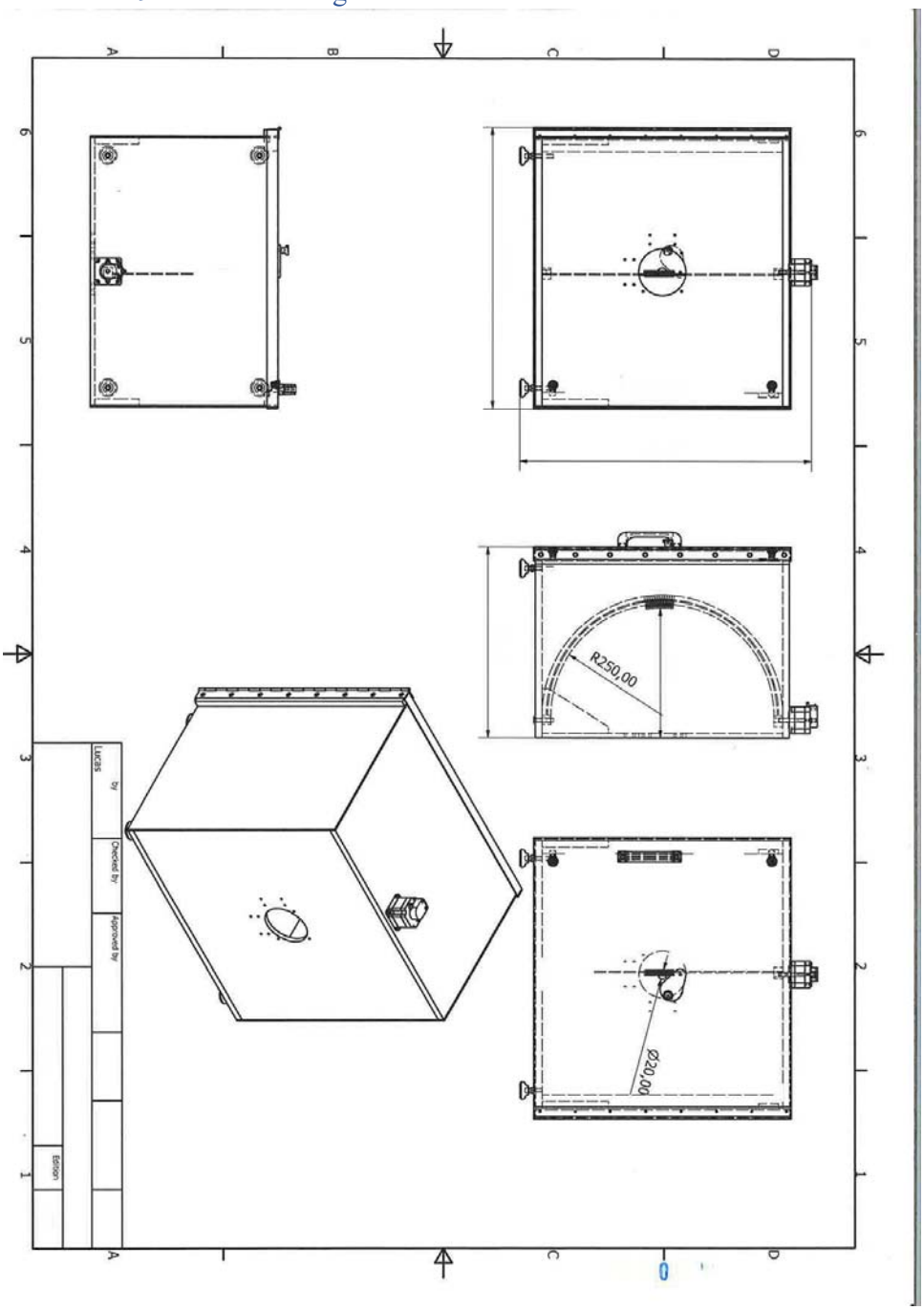

Figure 37: The initial design of the chassis.

9.2.2 Final Design

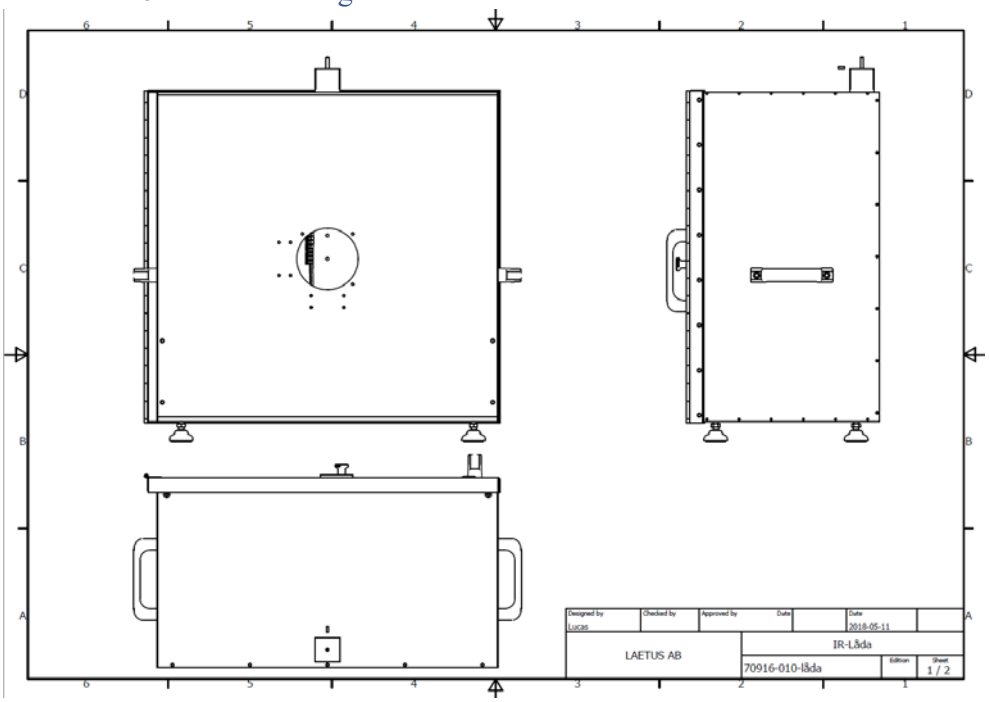

Figure 38: Back, overview, and side view of the final design.

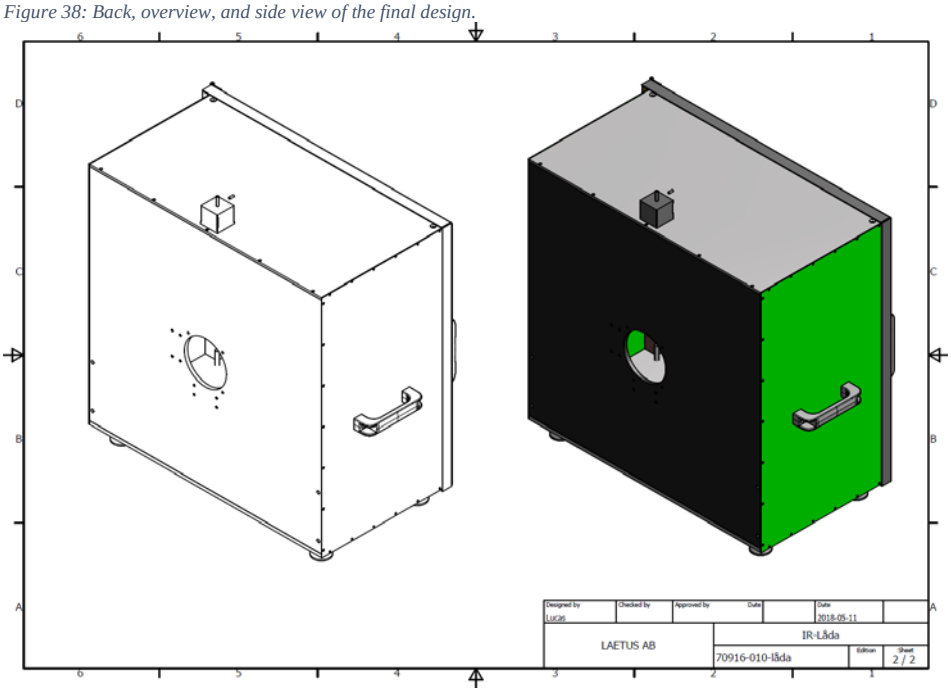

Figure 39: An overview of the final design.

Printed by Tryckeriet i E-huset, Lund 2018

LUND
UNIVERSITY

Series of Bachelor's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2018-634
http://www.eit.lth.se