SCHOOL OF INFORMATICS AND ENGINEERING

DIGITAL INSPECTION AND MEASUREMENTS TOOL

by

Justas Bartnykas

A report submitted in partial fulfillment of the requirements for the degree

BACHELOR OF ENGINEERING (HONOURS)
IN
COMPUTER ENGINEERING
IN
MOBILE SYSTEMS

SUPERVISOR: BENJAMIN TOLAND
SUBMISSION DATE: **X / X / 2019**

Software block

QtCreator (Free and open to use)
OpenCV (Free – OpenSource library)
CMAKE (Free and open to use Compiler) (For OpenCV)
Arduino IDE (Free and open to use)
Fusion 360 (Free with a limited terms license)
Cura 4.0 (Free open to use with multiple printers)
MOBA/PUTTY (Free and open to use) for Arduino debugging
Edraw (Not free, used for diagrams)
GitHub Desktop (Free and open to use for code timeline tracking) (Provide a link)
(How do you pull the project?)
Photoshop (Not free, used for diagrams etc) (Also used for the grid)
KiCAD (Free and open to use) Used for the PCB.
DoxyGen?
Ben's PCB Maker Software

Processes block

Lens correction (What is it? How does it work? How do you set it up? How do you train it? How do you use it afterwards? Why is it needed?

Setting up / Compiling OpenCV / Importing into QtCreator
Using the actual program. How does a user facilitate the actual program to be useful?

`Serial communications / What is it? Why is it? Why do I need it? What does it do?

Software processes

Camera setup interface
Setup the Camera (Camera init)
Different interfaces
Camera calibration
Measurements interface

Hardware block

Tevo Tornado 3D printer
Dino stand
Logitech C922 WebCam
28BYJ-48 Stepper motors
3D printed PLA Parts (Including the models)
Arduino ProMicro (Used for driving motors)
Stepper transistors
Buffer IC? (Might be dropped)
PCB (Ben's PCB Maker)
Ring light (Including the Serial Software controls)

Important notes

Ben wants to see real-world tested results. He wants some data that would not just be a part of the theory, but the theory should merge with the worked out examples of what I've actually done. For example he wants a measurement with and without the lens correction and he wants to see the difference and the importance of it there.
Crazy P says that Darlington pairs will never be fully saturated and therefore MOSFET's would be better.

ABSTRACT

The Digital Inspection and Measurements tool is a low cost DIY project that is both easy to build and simple to use. This is a mid-level replacement for any professional high-end gear that would be used in the industry.

// Finish me //

ACKNOWLEDGMENTS

Special thanks to my project supervisor Mr. Benjamin Toland for helping and guiding me throughout the duration of the project. His support and guidance was nothing short of perfection and has uplifted my spirits and motivation to successfully complete the project.

Special thanks to Mr. Sean Clancy for supporting me and for cheering me up throughout the toughest times of the year. His friendship and company has kept me energized to complete the year to my best ability.

Thanks to the TuDublin Blanchardstown facilities, lecturers, equipment and staff that made this opportunity available to me to attend and to complete my level 8 degree.

DECLARATION

The report submitted is the results of the student's own work-placement project work and has not been submitted for any other award. Where use has been made of the work of other people it has been fully acknowledged and referenced.

Student Name

_____

Table of Contents

LIST OF ABBREVIATIONS

IDE – Integrated Development Environment.
OOP – Object Oriented Programming.
DIY – The activity of decorating, building, and making repairs at home by oneself rather than employing a professional.
3D – Three Dimensional.
2D – Two Dimensional.
PCB – Printed Circuit Board.
QtCreator – GUI Based Software development IDE.
OpenCV – Open Computer Vision (Library).
GUI – Graphical User Interface.
USB – Universal Serial Bus (Universal computer communications interface).
WebCam – A video camera connected to a computer, allowing its images to be seen by Internet
users.
FPS – Frames per second (In terms of video frames). The more frames the smoother the video.

Table of Figures

Chapter 1 Introduction

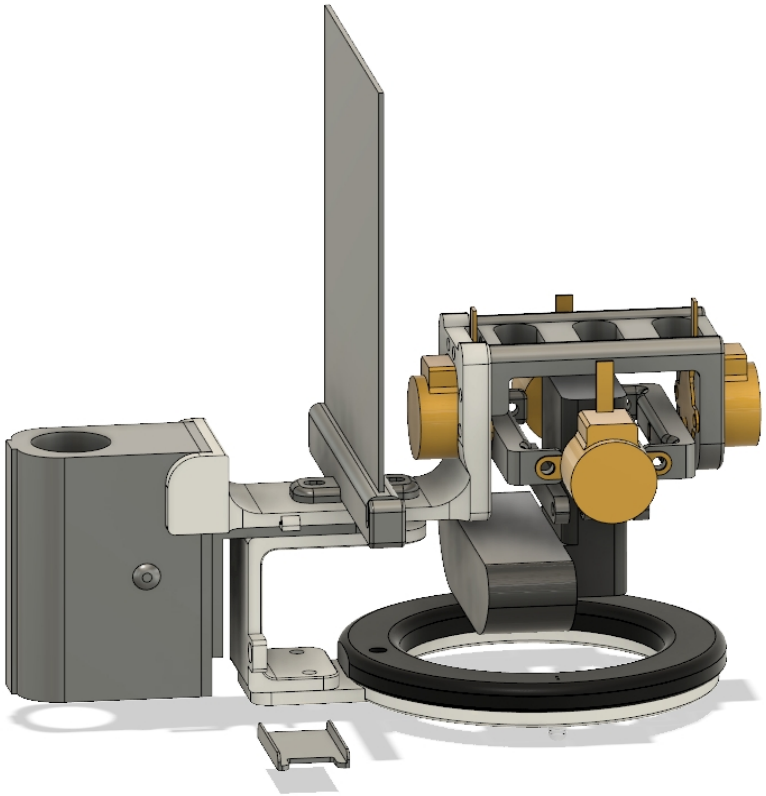

Figure 1: Project design overview diagram

The Digital Inspection and Measurements tool is based on high-end Digital inspection tools such as the Omni (By Ash Technologies) where the student has done his work placement over the summer. More can be found on the Ash Website in the Appendix [1] - (Ash Technologies 1994-2018).

This particular implementation is based on a simpler mid-end DIY tool that any hobbyist or enthusiast could build at home relatively cheaply and still have it to function as a useful tool to use on the daily or weekly basis. These types of tools are unique for being able to facilitate Software in order to perform Real-World operations such as placing a virtual ruler(s) for measuring a part or any object(s).

The tool is based on two main principles which are the Inspection mode and the Measurements mode. On the Inspection mode the user can perform a variety of different real-time adjustments such as changing the contrast/focus or the zoom etc. settings for the camera in order to have a closer and more detailed look (Especially on any small parts/writing). Whereas on the Measurements mode the user is able to place a scale between a known distance (Such as the grid attached to the base plate or any other known distance) and then place additional 20 rulers that will show the distances measured by the tool.

On high-end equipment such as the Omni this process is very highly calibrated resulting in a very wide customization and very accurate results, however this Project can still accurately measure within 0.5mm of accuracy or better.

1.1 Background

This report documents the research and development undertaken while completing a project to design and construct a Digital inspection and Measurements tool. It documents the hardware, software, 3D models, OpenCV software library and the QtCreator IDE which is a free – open source Software package based on the C++ programming language [2] that supports GUI applications. The report aims to give a detailed breakdown of all the major aspects of the project and to explain the setup and the rationale behind the decisions and choices made in order to conduct the project.

1.2 Scope and Objectives

The aim of this project was to design and construct a digital software-driven tool that would have a wide variety of uses and would also be easy to build by any hobbyist or Engineering enthusiasts. The objective is to implement an expensive high-end piece of gear as a relatively cheap mid-end tool that would still be relatively useful and convenient to use. By researching and developing this project the intention was to gain experience and practical knowledge of real world, software based system applications.

The operator has the ability to control the operation of the two modes (Dynamic / Static) as well as the ability to switch between the two modes as fits. This allows the operator to adjust the camera in the Inspection (Dynamic) mode that is to adjust the brightness/contrast/focus etc… of the camera as well as to place rulers used for measuring in the Measurements (Static) mode. The main GUI application processes the visual information it receives from the web camera and via the help of the OpenCV library as well as native QtCreator libraries it displays the processed imagery onto the screen for the operator to see. This footage can then be saved or simply used on-the-go for inspection and quality control that is very useful not only on the industry, but also in a day to day home use.

1.3 Project Overview

This section helps give an understanding of how the project was broken down into individual components and how the project tasks were sequenced. An in-depth discussion and of each topic can be found provided in the chapter on Implementation.

1.3.1 Hardware components setup

The Project hardware components setup consists of:

- A camera. In this case the **C922 Pro Stream WebCam** was used, but any good quality camera / WebCam could also be used.
- A stand for the camera. In this case the **Dino-Lite Desktop Stand, For Digital Microscope Camera (MS35B)** was used, but any other purchased or custom-made stand could also be used as long as it's flexible enough to mount the camera and to adjust it afterwards.
- ***4x*** Stepper motors. In this case the **28BYJ-48 5V 4-Phase 5-Wire stepper motors** were used (However, this turned out to be not as practical as will be discussed later in the report).
- ***4x*** Stepper motor driver IC's. In this case the **L293x Quadruple Half-H Drivers** were used, but, depending on the stepper motors chosen this can be easily substituted for an alternative method of driving the motors (Such a buffers or MOS based motor drivers).
- ***2x*** PNP BJT Transistors. In this case the **2N4403 PNP Silicon General Purpose**
- **Transistors** were used.
- ***2x*** 1KΩ resistors. In this case **Resistor 1K Ohm 1/4 Watt PTH** resistors were used.
- ***12x*** Ultra-bright LED's. In this case the **C503D-WAN 9000K, 5mm White LED (Round Through Hole package)** LED's were used, but any alternative bright white color LED's would also work.
- ***12x*** 220Ω resistors. In this case **Resistor 220 Ohm 1/4 Watt PTH** resistors were used.
- ***12x*** heat-shrink tubes. The heat shrink was used for the ring light resistors.
- Some hot glue. This was used for keeping the ring light wiring in place.
- Male and female headers to mount on the PCB board (Used for connecting the motors and the ring light).

- An embedded micro-controller used for communicating with the main GUI application (In order to control the stepper motors as well as the ring light). In this case the **Arduino Pro-Micro** was used.
- A custom-made **Strip** or **PCB board** used for the Ring light & stepper motor supply circuitry.
- 3D Printed (**PLA plastic**) parts for holding the stepper motors, camera and the PCB in place.
- A **2D laser-printed grid** laminated in a plastic. In this case an A4 sized paper was used, laminated and later trimmed to size.
- **Screws and nuts**. In this case 18 screws were used in total followed by 18 nuts. All of which are M4 size and differ in length. The particular length varies, but, particular fitting lengths should be used accordingly to particular implementation.
- A **Micro USB Type B Plug**. In this case a 1 meter long cable was used for interfacing the Arduino Pro-Micro with the host machine.
- A Host machine. In this case an **ASUS STRIX series gaming laptop** was used, any alternative PC or laptop could also be used.

1.3.2 Hardware 3D models / 3D printed parts

The 3D printed parts of the Project were printed using the Tevo Tornado 3D printer as can be seen below:

Figure 2: Tevo Tornado 3D printer

Any generic 3D printer could have also been used, alternatively laser cut parts and or metal parts could have been used. 3D printing the parts is the easiest and most available method.

The plastic used was a white Tevo branded PLA plastic that is very easy to work with, but is strong and durable making it suitable for the Project requirements. The process of modeling and slicing the parts can be seen below in the chapter on Implementation.

1.3.3 Software IDE / Compiling libraries

The main Project application is programmed with the QtCreator IDE. The application is GUI based and it allows the user to adjust the settings in real-time via sliders, buttons and check-boxes etc...

Figure 3: QtCreator Software IDE icon

Additionally the Project uses the OpenCV library that had to be compiled for the particular host machine. This was done via the Graphical CMake IDE.

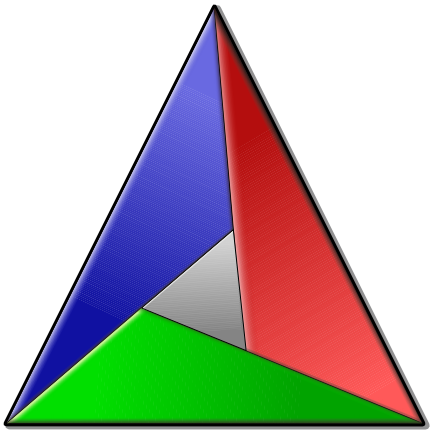

Figure 4: CMake Software IDE icon

There are different versions of OpenCV that can be compiled, however for this particular Project OpenCV-343 was used.

1.3.4 Software setup

The Software block diagram can be seen below.

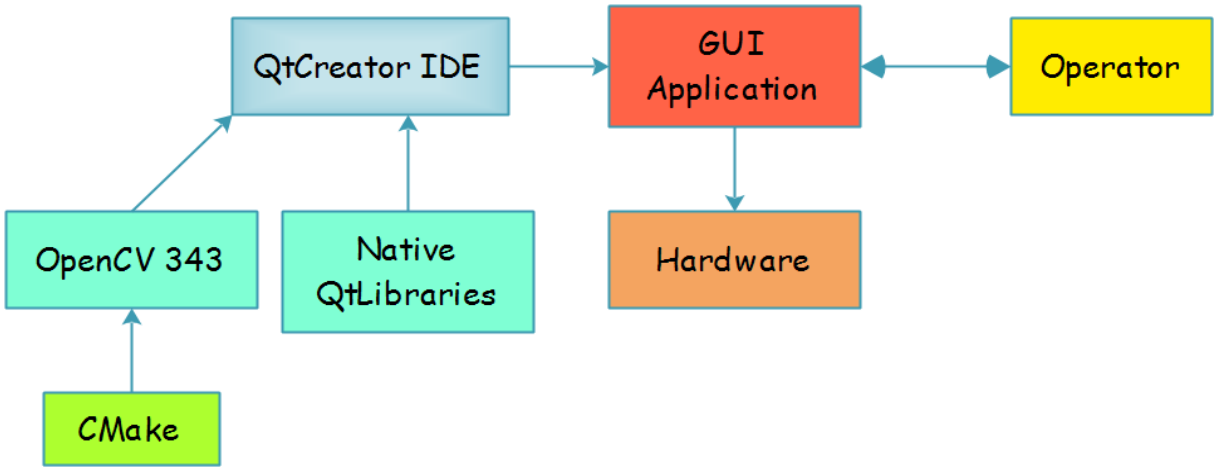

Figure 5: Software setup block diagram

The Software setup consists of the GUI Application that is implemented through the native QtCreator libraries (Such as timers etc) and the help of OpenCV 343. OpenCV is used for manipulating the images as well as setting up the camera object and saving the images dialog while the native libraries are used for the logistics of the GUI (Buttons, sliders, timers etc...).

1.3.5 Software GUI layout / Mode interfaces

The Project GUI layout can be seen below:

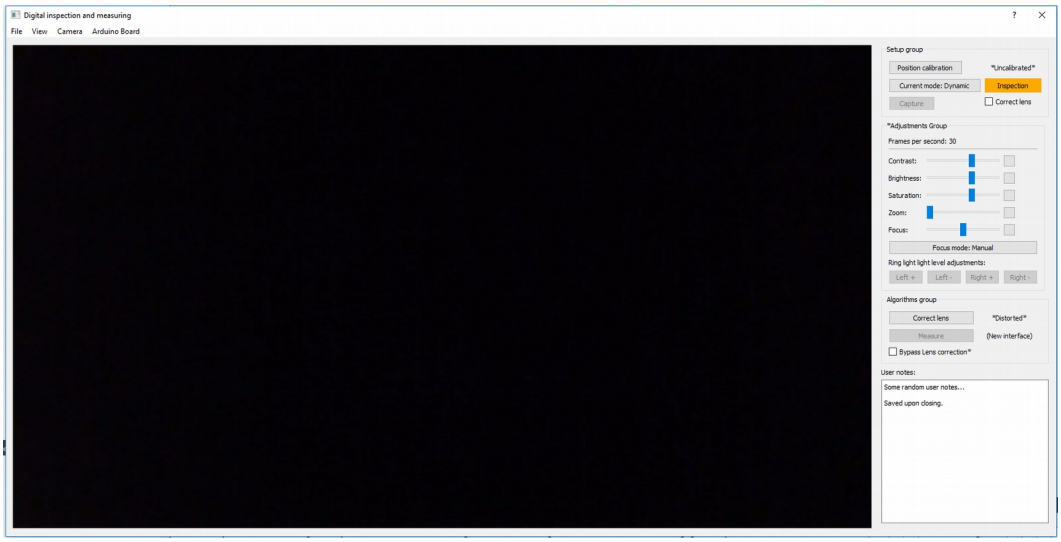

Figure 6: Project GUI layout design

As can be seen the layout is based on having the video feed on the left side (Black box) and the controls panel on the right side.

As can be seen currently the Dynamic mode is being used (Allowing the operator to adjust the camera hardware).

OpenCV is used to overlay graphics onto frames. This can be done both dynamically and statically. An illustration of this can be seen below:

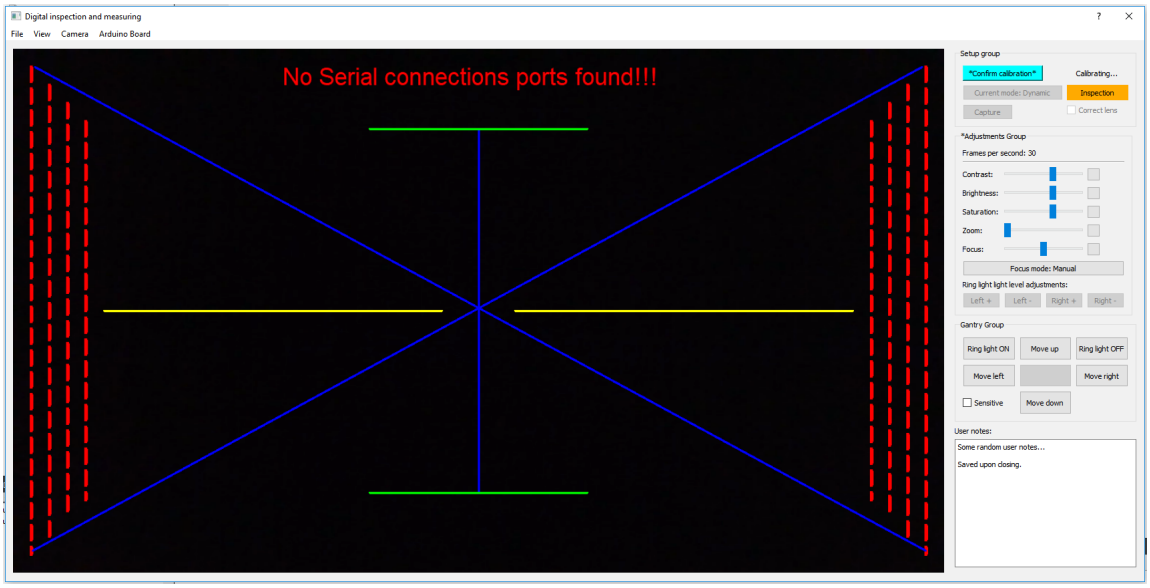

Figure 7: GUI Camera Calibration layout

1.3.6 Software design & Software testing (OOP, modularity)

The Software is designed based on the concepts of OOP. This makes the Project Software very modular and re-usable additionally, it makes the Software easy to debug and to modify as separate chunks of code can be found in their own dedicated classes.

An example of this can be seen below:

```
if (_camObj.cameraInit(_camPort, _fps, _res_Width, _res_Height) == -1)
{
    if (_camObj.cameraInit(0, _fps, _res_Width, _res_Height) == -1)
        Errors::fatalError("The camera cannot be initialized.");
}
```

Figure 8: Example of Camera initialization code (MainWindow Class calls a Camera class method)

In the above example, a method of: '**cameraInit**(const int camera, const int fps, const int &width, const int &height)' is being called from the MainWindow Class, where the method is located in a separate **Camera** class.

The **Camera** class diagram can be seen below:

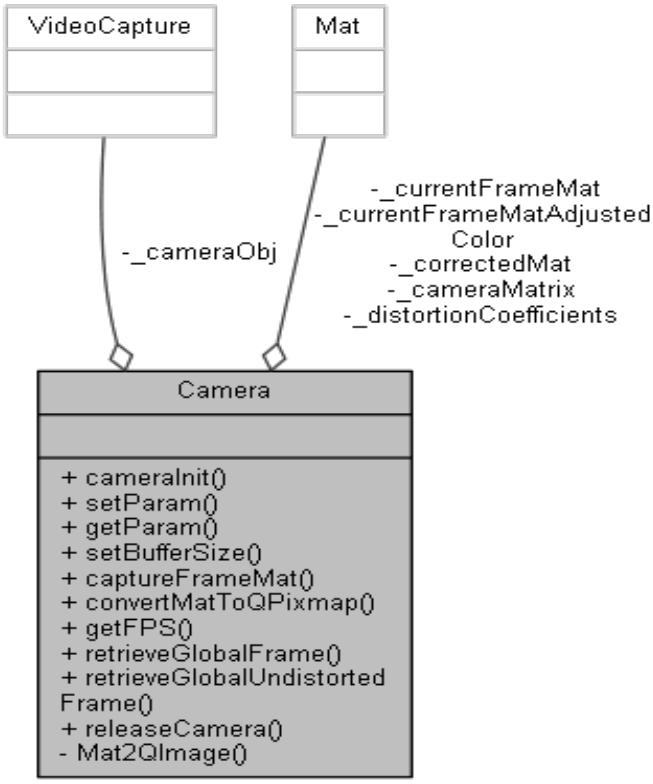

Figure 9: Camera class diagram (DoxyGen)

This also allows for easy testing and debugging as the different components are broken down into their own individual class files for example **Camera** class, **Errors** class, **Options** class etc...

The QtCreator offers a built in debugger that was also used to step through the application step-by-step for in-depth debugging.

1.3.7 Camera lens distortion / Measurements

The Camera used for the project was the Logitech C922 which is a professional streaming camera, however, it like most of Digital cameras produces a lens distortion meaning that the images it captures are distorted which makes the Measuring process impossible to do accurately.

In order to fix this issue there are two ways to do it, a Hardware based lens distortion (Using a lens to UN-distort the image) which can be tricky and expensive, and would only work for the particular camera as different cameras have a different distortion effect. And the more common, cheaper and more accessible Software lens distortion.

The Software lens distortion algorithm was chosen and the OpenCV based project was implemented on a separate project to train the algorithm how to UN-distort the particular camera's lens.

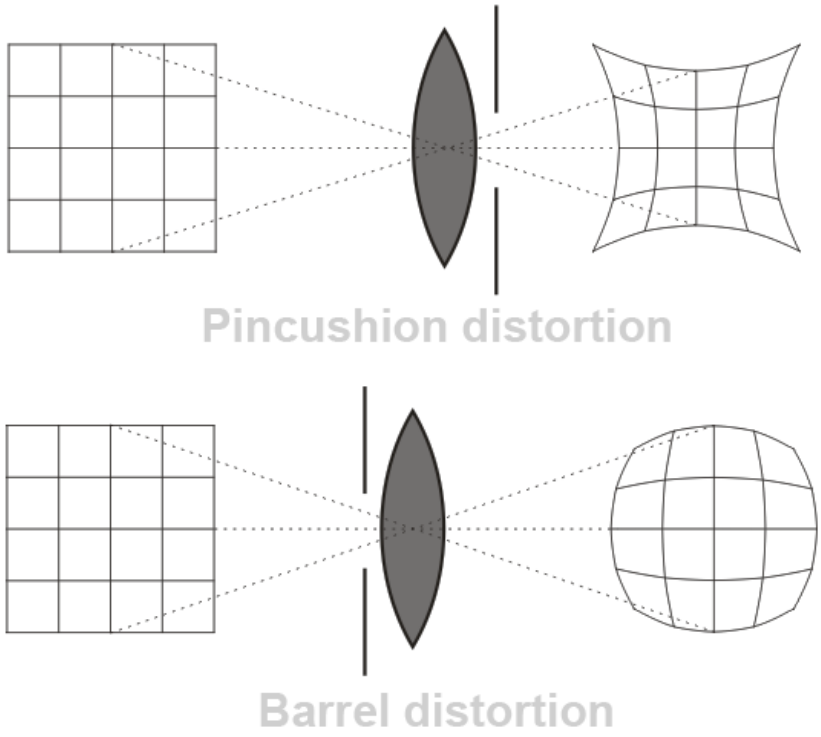

Figure 10: Visualizing the effect of Lens distortion [3]

The process of the OpenCV lens distortion can be found here []. This will be discussed in more detail in the chapter on Implementation.

Chapter 2 Literature Review

The area of Digital Inspection and Measurements is a relevant subject in the technology industry at present. There are many established and startup companies doing research and developing products with practical, real world, commercial applications.

Chapter 3 Implementation

3.1 Initial Considerations

A number of aspects of the project required consideration and research from the very start

3.2 Hardware

3.2.1 Beaglebone

3.2.9 3D Printing

3.3 Operating Systems

3.3.1 Embedded Linux

3.4 Software

3.4.1 Beaglebone Server Socket Software

3.4.2 Beaglebone GPIO Software

3.5 Networking

3.5.1 Beaglebone

Chapter 4 Testing

4.1 Software Testing

4.1.1 GPIO Stub Function

4.1.2 GPIO Harness Function

REFERENCES

1. Unknown, Ash Technologies (1994-2018), Ash Technologies website, [online] Available: https://ash-vision.com/about-us/company-overview/ [Accessed: 25 March 2019].
2. Unknown, QtCreator (1995-2018), The QtCompany website, [online] Available: https://www.qt.io/company [Accessed: 18 December 2018].
3. A. Brown, "The Haute Girl Explains Four Ways to Not Look Overweight in Photographs", *Fstoppers*, 2014. [Online]. Available: https://fstoppers.com/portraits/haute-girl-explains-four-ways-not-look-overweight-photographs-9323. [Accessed: 16- Apr- 2019].
4.