Word count 12,946

School of Engineering

University of Kent

Final Report for EENG6000 (Project)

A Single-Pixel Camera

By

Robert Whiting

RW372

Supervisor: Dr C Wang

A dissertation submitted as a part requirement for the degree of Bachelor of BEng (Computer Systems Engineering W. Year in Data Analytics)

Declaration

I certify that I have read and understood the entry in the School Student handbook on Plagiarism and Duplication of Material, and that all material in this assignment is my own work, except where I have indicated with appropriate references.

Signed: Robert Whiting

Date: 22/04/2023

Summary

Single-Pixel Camera (SPC) is a new and innovative technology in the field of imaging that offers a cost-effective solution compared to conventional/traditional multimillion-pixel cameras. SPCs have garnered attention for their ability to capture images outside the visible spectrum of light where detector technology can be difficult or expensive to produce. The ability to capture images outside the visible spectrum of light (where detector technology can be difficult or expensive to produce) is what makes SPCs a valuable tool for various applications. So, there's motivation for creating SPCs at a more cost-efficient price so that it can provide a solution for capturing images in various situations.

The primary goal of this project is to design and construct a budget prototype of a single-pixel camera that can perform tasks such as capturing an image using a single-pixel and reconstruct an image from raw measurement data through the use of proper decoding algorithms. This principle is founded on the concept that compressive sensing techniques, such as Lasso regression, can be used to recover the original image. To supplement this, a prototype simulation would also be created using the MATLAB Engine to showcase a reconstructed image and its similarity to the original using a single detector and a sequence of optimized patterns. This design process would highlight the potential challenges that could arise from data acquisition and algorithm performance, as well as how the single-pixel camera might perform in real-world scenarios before a physical concept is constructed.

Overall, the goal of this project is to showcase the potential of SPCs as a cost-effective solution for imaging compared to the conventional multimillion-pixel cameras as the system for the simulation would be able to return images up to a near perfect accuracy while the prototype produced pictures which while certainly would be low-res due to budget constraints, are ultimately good enough in its ability to pick out recognizable forms with a little imagination.

Table Of Contents

Nomenclature

Charge-coupled device **(CCD)**

Complementary metal-oxide-semiconductor **(CMOS)**

Single-pixel imaging **(SPI)**

Digital micromirror device **(DMD)**

Spatial light modulator **(SLM)**

Single point detector **(SPD)**

Light-emitting diodes **(LEDs)**

Single-Pixel camera **(SPC)**

Compressive sensing **(CS)**

Micro-electro-mechanical systems **(MEMS)**

Convolutional neural networks **(CNNs)**

Graphical User Interface **(GUI)**

MATLAB's GUIDE **(GUI Development Environment)**

Structural Similarity Index **(SSIM)**

1. Introduction

1.1 Background

In the modern age, it has been established that digital cameras use a pixelated array via millions of pixels to acquire images. These cameras rely on charge-coupled device (CCD) or complementary metal-oxide-semiconductor (CMOS) sensors to capture light and convert it into an electrical signal [1]. This process results in high-resolution images with a wide level of colors and details. However, the recent development of single-pixel imaging (SPI) devices has introduced a novel approach to image acquisition and reconstruction [2].

A single-pixel imaging (SPI) device reconstructs images by sampling a scene with a sequence of patterns and associating these patterns by corresponding the intensity measured to filter the scene along the transmitted intensity [3]. This is achieved using a digital micromirror device (DMD) or spatial light modulator (SLM) that generates the patterns, a single point detector (SPD) to measure the intensity of light reflected or transmission through the scene, and a reconstruction algorithm to create the final image based on the acquired data.

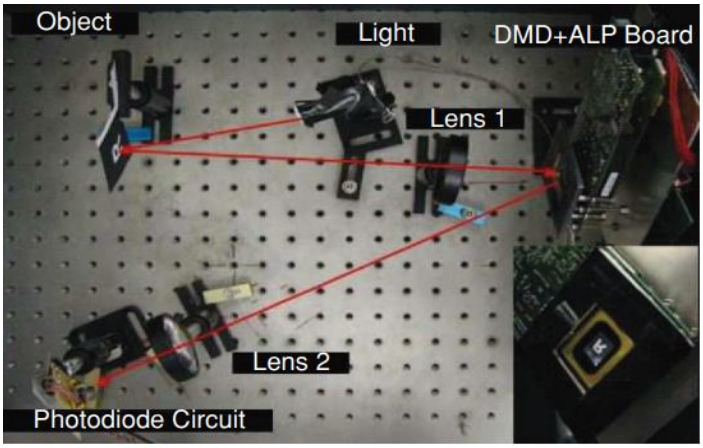


Figure 1: Schematic representation of a single pixel imaging setup, including the light source, spatial light modulator, single point detector, and the reconstruction process (source: Duarte et al., 2008)

Though not performing as well as digital cameras in conventional visible imaging, SPI brings significant performance advantages such as sensitivity at non-visible wavelengths, very precise timing resolution, and enhanced imaging capabilities in low-light conditions [2]. This makes SPI particularly suited for applications in fields such as remote sensing, scientific imaging, and biomedical imaging, where these unique characteristics are of great importance.

"Single-pixel imaging systems are particularly interesting for imaging at non-visible wavelengths, where detector arrays are either unavailable or very expensive, as well as for applications where very precise timing resolution is required." [2]

Single-pixel cameras also provide benefits in terms of reduced size, weight, and power consumption compared to their multi-pixel counterparts. These characteristics make SPI devices ideal for use in portable or remote equipment and in situations where access to power sources is limited [4]. Additionally, the single-pixel design makes the system less susceptible to noise and more robust in harsh environments, further expanding its potential applications. Another benefit of SPI technology is its compatibility with various types of light sources, such as lasers or light-emitting diodes (LEDs), and its ability to operate in different spectral regions, including ultraviolet, visible, and infrared. This versatility enables SPI devices to be adapted for a wide range of imaging tasks, from capturing images in non-visible wavelengths for scientific purposes to performing thermal imaging for industrial applications [2].

In conclusion, single-pixel imaging devices offer a unique and innovative approach to image acquisition and reconstruction. While they may not outperform conventional digital cameras in terms of visible image quality, their advantages in non-visible wavelengths, timing resolution, and robustness make them an attractive alternative for a variety of specialized

applications. As technology advances, it is expected that the performance of SPI devices will continue to improve, opening up new possibilities for imaging across various fields [4].

1.2 Motivation

Consumer usage of megapixel cameras has become nearly universal, with these devices being found in everything from mobile phones to advanced camera systems that outperform their predecessors. The widespread adoption of such cameras can largely be attributed to the use of silicon as the primary semiconductor material. Silicon has the unique property of effectively converting photons at visible wavelengths into electrons, enabling high-resolution imaging within the visible spectrum. However, when it comes to imaging outside the visible range of the electromagnetic spectrum, silicon-based detectors face limitations in terms of sensitivity, manufacturing complexity, and cost.

For instance, when capturing images in the infrared spectrum, a comparable resolution to that of a $500 digital camera for visible light can result in a staggering price of $50,000 [5]. This significant price difference highlights the need for alternative imaging technologies that can operate efficiently across a broader spectral range, without the limitations and expenses associated with conventional silicon-based cameras.

Therefore, single-pixel imaging (SPI) approach offers a promising solution to this challenge by providing a simpler and more cost-effective means of image acquisition. With its unique ability to operate efficiently across various spectral ranges, including non-visible wavelengths such as infrared and ultraviolet, SPI devices hold the potential to revolutionize imaging applications in numerous fields [2]. By leveraging the advantages of single-pixel cameras, researchers and industry professionals can access previously unattainable imaging capabilities at a fraction of the cost of traditional systems. Furthermore, the reduced size, weight, and power consumption of single-pixel cameras make them ideal for use in remote

sensing, aerospace, and other applications where portability and energy efficiency are crucial [4]. Additionally, the robustness of SPI systems in harsh environments and their compatibility with a variety of light sources further expand the potential applications for this technology, from environmental monitoring to biomedical imaging.

The motivation behind the development and adoption of single-pixel imaging technology lies in its ability to overcome the limitations and high costs associated with conventional silicon-based cameras when operating outside the visible spectrum. By providing a simpler, more affordable, and versatile imaging solution, SPI has the potential to transform the way we capture and analyse images across a wide range of applications and industries.

1.3 Aims & Objectives

This report will detail the design, simulation, and results of a single-pixel camera system. Section 2 covers the background knowledge required for understanding the project.

This is followed by a discussion of the single-pixel imaging concept and the associated theory in Section 3.

The design and construction of the single-pixel camera system, including the choice of components, are detailed in Section 4, with a description of the simulation process using relevant software tools and the design of the software the physical hardware.

Section 5 presents the results obtained from the single-pixel camera system and the simulation showcasing the imaging.

Lastly, the project management, self-reflection, and conclusion are detailed in Sections 6, 7, and 8, respectively.

This comprehensive report aims to demonstrate the potential of single-pixel cameras as a cost-effective and versatile alternative to that available in our current environment. As a result, this project attempted to achieve the following aims

- Introduce the project to explain what a Single-Pixel camera (SPC) is and how they function.

- Investigate the place that SPC currently hold in our day and age and What Role do they fulfill?

- Build a Prototype SPC

- To implement Image reconstruction from raw measurement data following a proper decoding algorithm.

- Potentially investigate Deep learning and the possibilities of it further improving image quality.

- Develop a SPC simulation to show the operation of a Single-Pixel Camera

1.4 Deliverables

- A project proposal that outlines the background, aims and objectives, technical requirements, and expected deliverables of the project.

- A final project report that summarizes the results of the project, including a description of the single-pixel camera system, simulation results, and analysis of the imaging data.

- A project poster that visually summarizes the key results and findings of the project.

- A functioning simulation of the single-pixel camera system, demonstrating its ability to capture and reconstruct images.

- A functioning single-pixel camera system that can be used to capture and process images.

- A viva voce examination to present and discuss the results of the project and the methodology used to achieve the objectives.

2. Literature Review

2.1 What's a Single-Pixel Camera and How do they function?

A single-pixel camera (SPC) is a novel imaging system that utilizes compressive sensing (CS) techniques to capture high-quality images using a single photodiode instead of an array of millions of photodetectors, as found in traditional cameras [5]. This technology is particularly useful for capturing images in non-optical wavelengths, such as infrared and ultraviolet, which are often expensive and challenging to produce using conventional imaging systems [6].

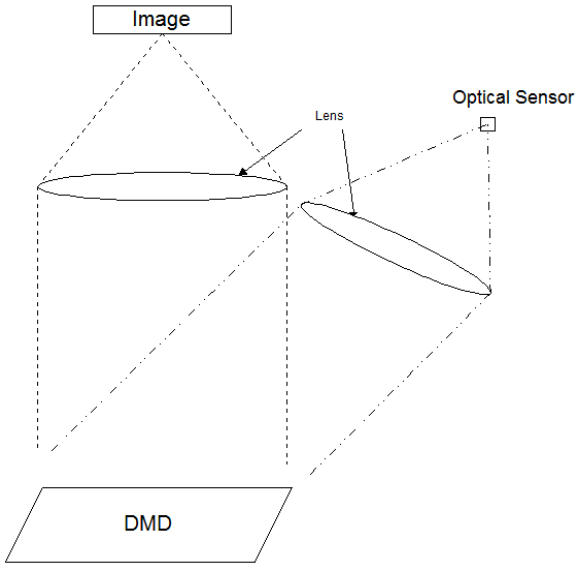


Figure 1: Single-Pixel Imaging (SPI)

The underlying principle of a single-pixel camera is based on the idea that many natural images are sparse or compressible in some domain, allowing for the recovery of images from a significantly reduced set of measurements [7]. This is achieved by reflecting the image off a digital micro-mirror device (DMD) onto a single photodiode through a lens, as shown in Figure 1 which due to the DMD being tilted ±15°, only some of the pixels focuses on the photodiode at any given time, while the rest is focused onto a light absorber.

By doing this consecutively and measuring the intensity of the light for each DMD arrangement, we can physically realize the sampling matrix of a CS system which any radiation reflected from this mirror (Ultraviolet and InfraRed) can be sampled, resulting in a more cost-efficient alternative[3].

Therefore To further understand the functioning of a single-pixel camera, it is essential to delve into the concept of compressive sensing. Compressive sensing is based on the idea that signals with sparse or compressible representations can be accurately reconstructed from a relatively small number of measurements [7]. In the context of single-pixel cameras, this means that the original high-resolution image can be recovered from a smaller number of intensity measurements obtained through the digital micro-mirror device.

Compressive sensing relies on two main principles: sparsity and incoherence.

- Sparsity refers to the fact that the signal can be represented with only a few non-zero elements in a suitable basis or domain.
- Incoherence, on the other hand, implies that the sensing basis is different from the sparsity basis, ensuring that the sparse structure is preserved when measurements are taken [8].

In single-pixel cameras, the DMD's sampling patterns provide the necessary incoherence, while the sparsity of natural images is exploited to recover the original image from the compressed measurements.

2.2 Existing Single-Pixel Camera Construction & Composition Techniques

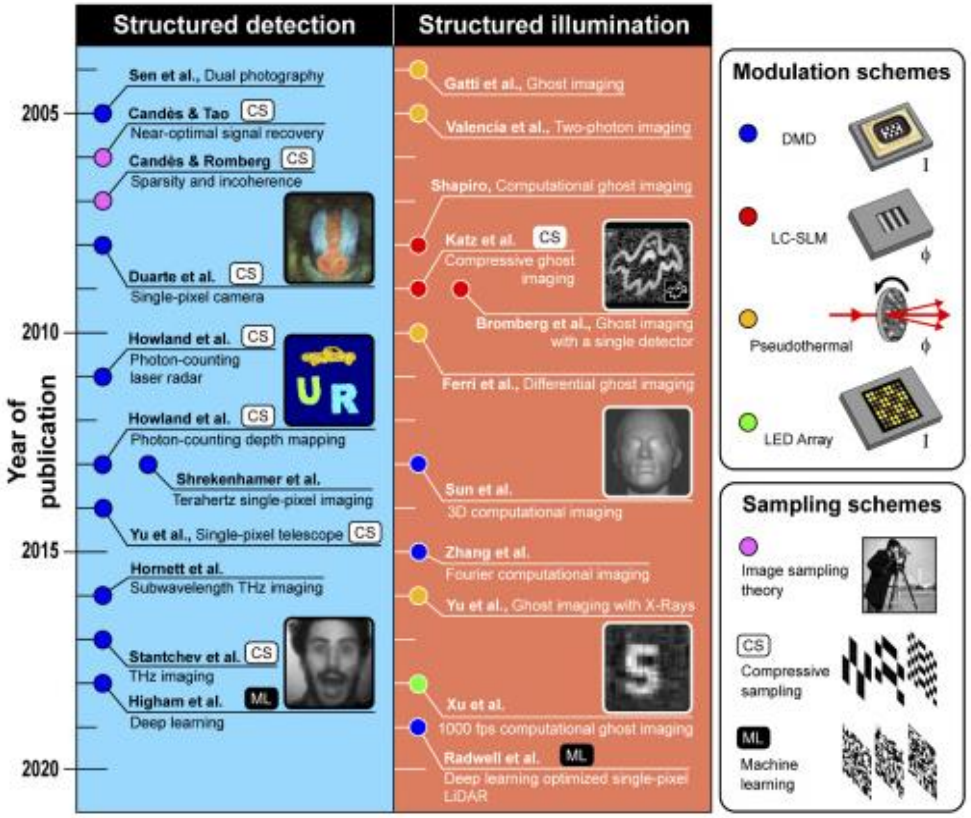


Figure 2: Timeline of SPI. Highlight periods of specific Modulation and Sampling schemes that show there has been a continuing trend of SPC being developed using an encoder via DMD. [5]

2.2.1 Digital Micro-Mirror Device (DMD)

The DMD is a micro-electro-mechanical systems (MEMS) device and a spatial light modulator (SPL) that consists of an array of hundreds of thousands to even millions of tiny micromirrors, each about 10 microns in size, which can be individually controlled to modulate the intensity of light reflected from the scene [3]. This allows for the selective sampling of the scene by directing light onto a photodiode or a light absorber [9]. The DMD generates random or structured sampling patterns, which can be binary or continuous, to

determine the encoding of the light intensity information onto a single photodiode. When combined with optics and light sources, a SPL enables users to program high-speed and very efficient patterns of light.

2.2.2 **Compressive Sampling** (Known as Compressed sensing, compressive sampling, CS, or sparse sampling)

Compressive sampling, also known as compressed sensing or sparse sampling, is a technique that allows the reconstruction of high-dimensional signals, such as images or audio, from a significantly reduced set of measurements [10], So even if you only had 5% of the pixels in an image, you could potentially reconstruct what was under that image using CS. The underlying assumption of CS is that the signal is sparse or compressible in some domain, which enables the recovery of the original signal even when a large portion of the measurements are missing or corrupted [8]. From this, a formula associated is:

$$X \text{ (Image or Signal)} = S \text{ (Sparse)} * \text{(Fourier Basis)} \Psi$$

The goal of CS is to recover the signal X from the observation using an optimization algorithm that leverages the sparsity of the signal [11]. For example, Figure 3 allows us to understand the idea of a really high dimensional signal (X) which could be reshaped into a tall million-pixel vector, and when we do Fourier transform this image, it can deal with throwing away most of those Fourier coefficients such as keeping only 5% or even 1% so that when Inverse Fourier is applied. It transforms that sparse signal (S) recovers a pretty faithful representation of the original image know as signal (X) in the formal.

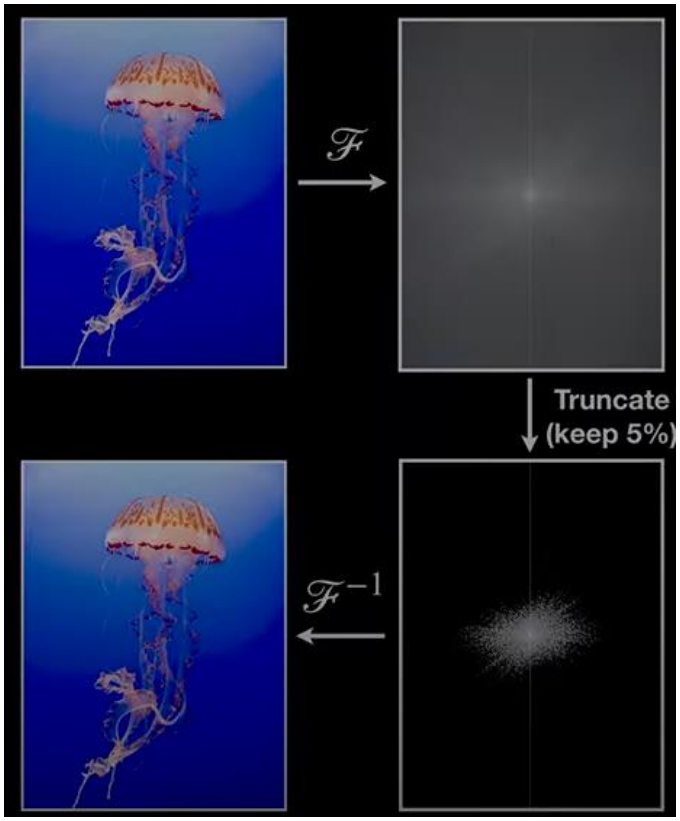


Figure 3: Showing Traditional Image compressing and decompressing using a basis function over a High Resolution-Image [5]

Additionally recent advancements in the field of CS have led to the development of new sampling schemes and reconstruction algorithms, which offer improved performance and robustness compared to traditional techniques. Some examples include the development of adaptive sampling strategies that adjust the sampling patterns based on the observed measurements, and the integration of deep learning techniques for improved image reconstruction.

2.3 What are typical applications of Single-Pixel Cameras

As there's inherent difference in comparison to digital cameras in areas such as visible imaging, SPC offer advantages in unconventional applications, some of these being.

2.3.1 Multi-Wavelength Imaging

Used for capturing images at various wavelengths, including visible, infrared, and ultraviolet light [12]. This capability enables the development of imaging systems that can provide valuable insights into the properties and behaviour of various materials and biological systems.

2.3.2 Terahertz Imaging

Is a non-destructive/non-ionizing technique used for imaging materials with unique transmission and reflection properties in the terahertz frequency range [13]. SPCs have been successfully used enabling the visualization of hidden structures and defects in materials, as well as the identification of concealed objects in security applications [14].

2.3.3 X-ray Imaging

Are widely used in various fields, such as medical imaging, non-destructive testing, and security screening allowing for the acquisition of high-quality images with reduced radiation doses and simplified hardware [15].

2.3.4 Three-Dimensional Imaging

Used where depth information is combined with intensity data to create a 3D representation of the scene [12]. By incorporating structured illumination patterns and computational reconstruction techniques, SPCs can achieve accurate depth measurements and 3D reconstructions at lower sampling rates and with reduced complexity compared to traditional imaging systems [16].

2.4 Future Directions and Opportunities

Recent advances in deep learning and computational techniques have opened new possibilities for improving the performance and capabilities of single-pixel cameras. Some of these opportunities include:

2.4.1 Deep Learning for Image Reconstruction

Deep learning techniques, such as convolutional neural networks (CNNs), have been successfully applied to improve the image reconstruction quality of single-pixel cameras [17]. By training neural networks to learn the underlying structure and sparsity patterns of natural images, it is possible to achieve faster and more accurate reconstructions, even for moderate-resolution images therefore dealing with the previously mentioned limitations.

2.4.2 Integration with Other Imaging Modalities

Combining single-pixel cameras with other imaging modalities, such as structured light or time-of-flight sensors, can enhance the capabilities of SPCs and enable new applications in areas like 3D imaging, multispectral imaging, and computational imaging [18].

2.4.3 Hardware Innovations

Advancements in DMD technology and the development of alternative light modulation techniques can significantly improve the performance of single-pixel cameras.

2.4.4 New Reconstruction Algorithms

The development of novel reconstruction algorithms, such as those based on sparse Bayesian learning or deep learning, can further improve the performance of single-pixel

cameras. These algorithms have the potential to provide more accurate and faster image reconstructions while reducing the computational complexity and resource requirements of the reconstruction process.

2.5 Performance Evaluation and Comparison with Traditional Cameras

Evaluating the performance of single-pixel cameras (SPCs) is crucial to understand their strengths, weaknesses, and limitations compared to conventional cameras. In this section, areas such as image quality, resolution, sensitivity, imaging speed, and comparison of SPCs to traditional cameras based on these factors are discussed.

2.5.1 Image Resolution

SPCs can achieve high-resolution images using digital micro-mirror devices (DMDs) and compressive sensing techniques [3]. However, their resolution is often limited by the number of micromirrors in the DMD and the reconstruction algorithms used. In contrast, traditional cameras use an array of millions of photodetectors, enabling higher spatial resolution [3]. Recent advances in compressed sensing techniques and optimization algorithms have enabled SPCs to achieve comparable resolutions to traditional cameras, even with reduced measurement sets [21].

2.5.2 Cost and Sensitivity

SPCs offer a potentially more cost-effective alternative to conventional cameras, particularly for non-visible wavelengths, as they require only a single photodetector and a DMD, which can be less expensive and easier to manufacture [5]. Additionally, SPCs can provide increased sensitivity in non-visible wavelength ranges, such as infrared and

ultraviolet, where traditional cameras may struggle due to the limitations of silicon-based photodetectors [23].

2.5.3 Imaging Speed and Motion Sensitivity

A primary limitation of SPCs is the need for sequential measurements, resulting in longer acquisition times compared to traditional cameras [24]. This sensitivity to motion may restrict the applicability of SPCs in dynamic environments or for imaging moving objects [22]. However, ongoing research in compressed sensing and deep learning techniques has the potential to significantly improve the imaging speed and real-time performance of SPCs [25].

2.5.4 Reconstruction Speed and Computational Complexity

Reconstruction speed, crucial for real-time or near-real-time applications, is influenced by the complexity of the reconstruction algorithm and the computational resources available [21]. The reconstruction of images from SPC measurements often requires computationally intensive algorithms, which may limit their real-time performance. Reducing the computational complexity of image reconstruction techniques, for example, by employing deep learning or other advanced optimization methods, is essential for improving the usability of SPCs in various applications [6].

2.5.5 Robustness and Adaptability in Various Conditions

An important performance evaluation is their robustness and adaptability in different imaging conditions. Traditional cameras may be affected by changes in lighting or environmental conditions, leading to reduced image quality. However, SPCs can offer improved performance in challenging imaging conditions due to their ability to operate efficiently across a broader spectral range [23]. Moreover, SPCs can be adapted for use in

low-light scenarios or where high dynamic range imaging is required. By employing advanced reconstruction techniques and optimizing the compressive measurements, SPCs can enhance their performance in these challenging conditions [20].

2.6 Case Studies

The illustrating of SPC's in real-world scenarios can be demonstrated in various applications, showcasing their potential for various imaging tasks which some can be identified from the following:

2.6.1 Hyperspectral Imaging for Agriculture

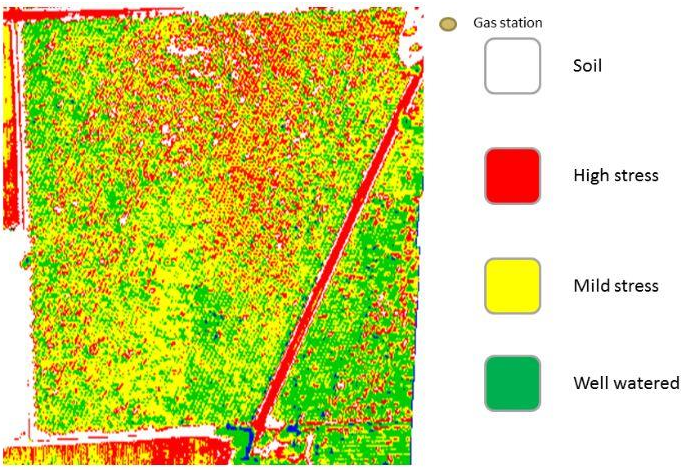


Figure 4: Hyperspectral Imaging for to demonstrate a stress map of a 160 acre Walnut orchard [34]

One promising application of single-pixel cameras is in hyperspectral imaging for precision agriculture. Hyperspectral imaging systems can collect detailed spectral information for each pixel in an image, allowing for the identification and analysis of various plant and

soil properties. However, conventional hyperspectral imaging systems are often expensive and complex, making their widespread adoption in agriculture challenging.

In a study by Huang et al. [26], a single-pixel hyperspectral camera was developed for agricultural applications. The system utilized a digital micro-mirror device and a single InGaAs photodiode to capture hyperspectral images in the near-infrared range, which is particularly useful for monitoring plant health. The researchers demonstrated that their single-pixel camera could effectively discriminate between different plant species and identify various stress factors, such as water deficiency and nutrient imbalances. This case study highlights the potential of single-pixel cameras for affordable and efficient hyperspectral imaging.

2.6.2 Terahertz Imaging for Cultural Heritage Preservation

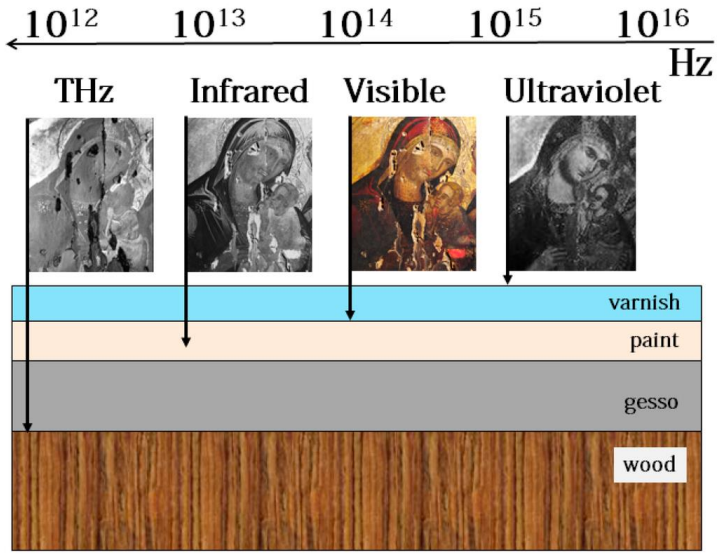


Figure 5: Terahertz Imaging being applied in application on Studying Paintings [35]

Terahertz imaging is an emerging technology for the non-destructive analysis of cultural heritage objects, such as paintings, manuscripts, and sculptures. Terahertz radiation can penetrate various materials, allowing for the visualization of hidden layers and structures beneath the surface. However, conventional terahertz imaging systems are often expensive and bulky, limiting their use in cultural heritage applications.

A recent study by Ruggiero et al. [27] employed a single-pixel terahertz camera for the inspection of a 17th-century painting. The camera utilized a single terahertz detector and a digital micro-mirror device for compressive sensing, enabling the acquisition of high-resolution terahertz images with a compact and cost-effective setup. The researchers demonstrated that their single-pixel camera could effectively reveal hidden features and provide valuable insights into the painting's composition and conservation state. This case study demonstrates the potential of single-pixel cameras for non-destructive imaging in cultural heritage preservation.

2.7 Search Strategy

The search strategy employed in this literature review was aimed at identifying relevant and up-to-date research articles, reviews, and technical reports related to single-pixel cameras, their design, applications, and challenges. I focused on locating peer-reviewed articles, conference proceedings, and book chapters published within the last decade to ensure that the most recent advancements in the field were covered. Additionally, I reviewed the reference lists of relevant articles to identify any additional sources that may have been missed during the initial search. The search was carried out using the following steps:

2.7.1 Database Selection

Several academic databases and search engines were selected for the literature search, including:

- IEEE Xplore
- ScienceDirect
- Google Scholar
- Scopus
- Web of Science

These databases and search engines were chosen due to their extensive coverage of scientific and technical literature in the fields of optics, photonics, and imaging, as well as their accessibility and user-friendly interfaces.

2.7.2 Keyword Identification

To maximize the effectiveness of the search, a list of relevant keywords and phrases was developed, including:

- Single-pixel camera
- Compressive sensing
- Digital micro-mirror device
- Compressed sensing
- Sparse sampling
- Infrared imaging
- Terahertz imaging

- X-ray imaging

These keywords served as the foundation for the literature search and helped to narrow down the scope of the research.

2.7.3 Search Filters and Sorting

To focus the search on the most relevant and recent publications, several filters were applied, such as:

1. Publication date: Prioritizing articles published within the last ten years to ensure the inclusion of recent advancements and developments.
2. Document type: Focusing on research articles, reviews, and technical reports to obtain in-depth information on the topic.
3. Language: Limiting the search to English-language publications for accessibility and consistency.

Additionally, search results were sorted by relevance and citation count to identify the most impactful and authoritative sources. Once the search results were obtained, I critically evaluated the sources based on their relevance, credibility, and quality. This involved assessing the authors' expertise, the study design and methodology, the findings and conclusions, and the overall contribution to the field. I prioritized sources that provided in-depth information and unique insights into the design, functioning, and applications of single-pixel cameras.

2.7.4 Organization and Synthesis of Information

After selecting the most relevant and high-quality sources, I organized the information into different sections and sub-sections, focusing on the background, construction, applications, limitations, and future challenges of single-pixel cameras. This involved synthesizing the information from various sources, identifying common themes and trends, and presenting the findings in a coherent and logical manner.

2.7.5 Citation Management and Referencing

Throughout the literature review process, I kept track of the sources using a citation management tool, which helped to ensure that all sources were accurately cited and referenced in the final document. I followed the provided citation format and included a comprehensive list of sources at the end of the review.

2.7.6 Continuous Updating and Revision

Finally, I continuously updated and revised the literature review to ensure that it remained current and relevant. This involved revisiting the databases and journals periodically to identify any new publications or updates in the field of single-pixel cameras. Any new findings or developments were incorporated into the review, and the existing sections were revised as necessary to maintain the accuracy and comprehensiveness of the information presented.

3. System Description

This chapter presents an overview of the single-pixel camera system, its components, and the development of a simulation that illustrates the image reconstruction process during its operation. Additionally the adjustments made to the original single-pixel camera design, which initially relied on a conventional system featuring a digital micro-mirror device (DMD) and a single photodetector will be discussed as due to unforeseen circumstances, the required equipment did not arrive on schedule, therefore to still produce a SPC before the designated deadline and perform the internal deadlines such as Final report and producing a physical concept, changes had to be made prompting a transformation of the system into constructing a single-pixel camera that uses the Adafruit RGB Colour Sensor (TCS34725) without the DMD and photoresistor. These modifications, including the integration of the sensor and 2 stepper motors which will be discussed in detail to ensure a comprehensive understanding of the updated single-pixel camera design.

Alongside the system to be designed, there'd be a simulation that serves as a visual demonstration of the image reconstruction process, highlighting the functionality of the single-pixel camera and its components. It emphasizes the crucial steps involved in single-pixel imaging and the influence of various parameters on the final image quality. Furthermore, the simulation provides a platform for testing and optimizing system performance, enabling performance evaluation and comparison with traditional cameras as demonstrated through its ability to compare the reconstructed image to the original.

Overall this chapter will explain the system components and the simulation which will offer valuable insights into the principles of single-pixel imaging and the capabilities of single-pixel cameras through the incorporation of the Adafruit RGB Colour Sensor that now allows the camera to capture colour images, broadening its potential applications and rendering it a more versatile imaging solution.

3.1 System Overview

3.1.1 System Overview for the Original design

The single-pixel camera design relies on only a single photodetector to collect light information from the scene being imaged. The initial photodetector to be used would've combined with a digital micro-mirror device (DMD), which then modulates the incoming light and creates a series of measurements that are used to reconstruct the image.

During the imaging process, the DMD would rapidly switches the mirrors between two states, either directing light towards the photodetector or reflecting it away. This creates a set of measurements that correspond to the intensity of light at various points in the scene. This information from the measurements is then processed using mathematical algorithms to reconstruct the image. This process is known as compressive sensing, as it compresses the information from the image into a smaller set of linear measurements. This algorithm is then used to reconstruct the image considering the underlying structure and sparsity patterns of natural images, enabling high-quality image reconstructions with a reduced number of measurements.

3.1.2 System Overview for the adjusted design

The updated single-pixel camera design features the Adafruit RGB Colour Sensor (TCS34725) which is a colour recognition sensor that provides the single-pixel camera with the ability to distinguish between different colours which range from the following red, green, and blue (RGB) light levels, resulting in the camera to capture colour images. For this system, motors would then ensure a precise and accurate movement, making them ideal for image acquisition.

This integration transformed the single-pixel camera into the right direction as it provided a versatile imaging solution that can handle a wide range of imaging tasks and

broadens the potential applications of the single-pixel camera so that it's suitable for tasks that require specific colour information, such as colour identification, colour recognition, and colour analysis. With the change of now being able to acquire coloured images, the system is able to maintain the advantages, such as low cost, compact design, and high flexibility while still employing techniques that'd have been used in the original such as, CS and the ability to reconstruct the image from raw data.

3.1.3 Simulation

The simulation provides a visual demonstration of the image reconstruction process and highlights the similarity of the single-pixel camera image to its original. Its design emphasizes the importance of each step in the process, including the linear measurements/reconstruction of the image, and how they contribute to the final reconstructed image. The simulation also allows for the testing and adjustment of various parameters, such as the number of masks and block length to optimize the system performance. This enables a performance evaluation of the single-pixel camera system compared to traditional cameras, which can be performed in real-time without the need for physical experimentation. Additionally, the simulation provides a flexible platform for further research and development, allowing for the exploration of new algorithms and techniques for image reconstruction. The simulation thus serves as an invaluable tool for advancing the capabilities and improving the performance of single-pixel cameras.

3.2 Theories Applied

The theory applied is still based on the principles of compressive sensing through the changes, which involves compressing a large amount of information from an image into a smaller set of linear measurements. These measurements are then processed using

mathematical algorithms to reconstruct the original image. In the original single-pixel camera, the DMD modulates the incoming light, directing it towards the photodetector or reflecting it away, creating a set of measurements that correspond to the intensity of light at various points in the scene. These measurements are then used to reconstruct the image using compressive sensing algorithms. The updated single-pixel camera design still utilizes these several key theories and principles to achieve its image reconstruction capabilities.

With the new adjustments brings new theories, the colour sensor for example is based on the theory of specific colour recognition, which involves the identification and analysis of different colours in a scene, while the use of motors in the updated design is based on the theory of motor control and precision movement to enable precise and accurate movement which are used to control the movement of the camera, enabling it to capture images from different angles and perspectives.

3.2.1 Simulation theory

The simulation of the single-pixel camera system is also built on the principles of compressive sensing as it’s the foundation of single-pixel imaging. To help in this process, various research studies have been crucial in the development of the simulation. Some of these sources include:

- Baraniuk, R. G. (2007). Compressive sensing. IEEE Signal Processing Magazine, 24(4), 118-121.
- [10] Candès, E. J., & Wakin, M. B. (2008). An introduction to compressive sampling. IEEE Signal Processing Magazine, 25(2), 21-30.
- Dr. Y.C. Shen's research in the field of terahertz pulsed spectroscopic imaging using optimized binary masks

These findings would provide insights into the optimization of binary masks for compressive sensing, which used, give the foundation of the simulation.

3.3 Hardware

Both designs concepts would've required a list of components but there're overlap as both designs require a computer and power supply to operate.

3.3.1 Original design hardware components

The original single-pixel camera design consisted of two main hardware components: the digital micro-mirror device (DMD) and the photodetector.

3.3.1.1 Digital Micro-Mirror Device (DMD)

The DMD controls the direction of incoming light which during imaging, rapidly switches between two states, directing light towards the photodetector or reflecting it away, creating a set of measurements that correspond to the light intensity at various points in the scene. These measurements are important as they're needed to reconstruct the image.

3.3.1.2 Photodetector

The photodetector used in the original design is a PIN diode detector. This device is used to collect light information from the scene being imaged. The photodetector is positioned behind the DMD and is responsible for detecting the light that is directed towards it. The light information collected by the photodetector is then used to generate the necessary measurements for image reconstruction.

3.3.2 Updated design hardware components

The updated single-pixel camera design features the integration of several new hardware components alongside the originals which were highlighted in the project proposal.

3.3.2.1 Adafruit RGB Colour Sensor (TCS34725)

The TCS34725 sensor is a device from Adafruit that converts light into digital data. It is able to detect the colour of light, either from ambient light or light reflecting off objects, and provides the red, green, and blue (RGB) values as well as the infrared (IR) intensity of that light. With this information, we can identify the colour of objects, measure the level of ambient light, and even make colour adjustments. The TCS34725 communicates with our microcontroller using I2C, which makes it simple to use with various microcontroller platforms like Arduino.

3.3.2.2 Stepper Motors and Drivers

The stepper motors with corresponding drivers play a crucial role in the single-pixel camera design. These motors are used to control the movement of the camera, allowing it to acquire images from different angles and perspectives. The stepper motors are driven by specialized drivers, which control the precise and accurate movement of the motors. The combination of stepper motors and drivers ensures that the camera can precisely control its position and orientation, allowing it to capture high-quality images. The drivers are responsible for providing the necessary power and control signals to the stepper motors, ensuring that they operate smoothly and reliably. The use of stepper motors and drivers provides the updated single-pixel camera with a high degree of flexibility and versatility, making it a suitable imaging solution for a wide range of applications.

3.3.2.3 Arduino Uno

The Arduino Uno is a microcontroller platform that is widely used in various electronics projects. In the context of the single-pixel camera (SPC), the Arduino Uno serves as the main control unit for the system. It is responsible for controlling the movement of the stepper motors, which are used to adjust the position of the TCS34725 RGB colour sensor. The Arduino Uno also communicates with the RGB colour sensor over I2C to receive the raw colour data and process it into a usable form. Additionally, the Arduino Uno is used to control the timing and sequence of the image acquisition process. It enables the user to program and control the SPC system according to their specific needs and requirements. The flexibility and ease of use of the Arduino Uno made it an ideal choice for the SPC system, allowing for quick and easy system adjustments and modifications.

3.4 Software

3.4.1 Original design software components

The original single-pixel camera design relied on software to process the measurements collected by the photodetector and DMD. The software was responsible for implementing the compressive sensing algorithms that reconstructed the image from the measurements. This software was designed to run on a computer or embedded system, and it was responsible for performing the following tasks:

- Collecting and storing the raw measurements from the photodetector
- Applying compressive sensing algorithms to the measurements
- Reconstructing the image from the processed measurements
- Displaying the reconstructed image on a screen or storing it for later analysis

3.4.2 Updated design software components

The updated single-pixel camera design also relies on software to process the RGB colour data collected by the colour sensor. The software is responsible for performing the following tasks:

- Collecting and storing the raw RGB colour data from the sensor
- Processing the RGB colour data into a usable form via CS so that it can be reconstructed
- Controlling the movement of the stepper motors
- Controlling the timing and sequence of the image acquisition process
- Storing the processed images for later analysis

The software used in the updated design runs on the Arduino Uno microcontroller. The use of this microcontroller enables the updated design to be highly compact and portable, making it ideal for a wide range of applications. Additionally, the use of a microcontroller allows for real-time control and processing of the RGB colour data, enabling the updated design to quickly respond to changes in the environment.

3.4.2.1 Image Processing Algorithms

The updated single-pixel camera design utilizes various image processing algorithms to improve the quality of the final reconstructed image. These algorithms are used to process the raw colour data collected by the TCS34725 RGB colour sensor and convert it into a usable form. These image processing algorithms are identified by the following:

Colour Correction Algorithm –

The first part is correcting the colour of the image. This is achieved by using a colour correction algorithm, which adjusts the red, green, and blue (RGB) values of the image to ensure that the colours are accurate and consistent. This algorithm can also be demonstrated via the simulation as it considers the spectral sensitivity and corrects for any discrepancies in the colour values. This step is crucial for ensuring that the final image has accurate and consistent colours.

Noise Reduction Algorithm –

The second step is to reduce noise in the image. This is achieved by using a noise reduction algorithm, which filters out any unwanted noise or artifacts from the image. This algorithm works by removing pixels that are significantly different from their neighbouring pixels, thus reducing the amount of noise in the image.

3.4.2.2 Image Reconstruction Algorithms

These algorithms are responsible for converting the raw colour data into a usable form, and for reconstructing the final image. The image reconstruction process in the single-pixel camera system is based on compressive sensing algorithms that use binary masks to optimize the system's performance.

The binary masks used in the compressive sensing algorithms play a crucial role in the image reconstruction process as they’re used to determine the pattern of linear measurements to reconstruct the image. Once the binary masks have been optimized, the compressive sensing algorithms are used to process the raw colour data collected by the TCS34725 RGB colour sensor. These algorithms consider the underlying structure and sparsity patterns of natural images, enabling high-quality image reconstructions with a

reduced number of measurements. The algorithms used in the single-pixel camera system are designed to be efficient and computationally feasible, enabling real-time image processing and reconstruction.

4. System Implementation

In this chapter, a brief overview of the updated single-pixel camera concept and its implementation process into a functioning system will be discussed. The implementation involves key steps beginning with the assembly of the hardware components, the installation and configuration of the software, and the integration of all the components into a single system. This chapter will provide a detailed description of the implementation process and the challenges that were encountered during the implementation stage. The hardware and software work that was undertaken, including the mechanical, electrical, electronic, computing, interfacing, and other tasks, will also be described in detail.

4.1 Software Implementation

4.1.1 Simulation Hardware

The hardware used in the simulation includes a computer and power supply. The computer is used to run the simulation software and display the results. The power supply is used to provide power to the computer.

4.1.2 Simulation Software

The simulation software is designed to run on a computer via MATLAB. The choice of MATLAB as the programming language for this project was driven by several factors, including my familiarity with the language and the abundant resources available for MATLAB development. My experience alongside the vast online community of MATLAB users and developers, would make it easy to find solutions to any technical issues. Additionally, the availability of pre-built add-ons and toolboxes in MATLAB greatly facilitated the construction of the system, as they provided a wide range of functions and

algorithms that could be integrated into the software with minimal effort. Some of the toolboxes would be:

- **Image Processing Toolbox**: This toolbox provides a range of functions for image acquisition, processing, analysis, and visualization.
- **Signal Processing Toolbox**: This toolbox provides functions for signal analysis, filtering, and feature extraction.
- **Computer Vision System Toolbox**: This toolbox provides functions for image and video processing, including object detection and tracking, image enhancement, and camera calibration.
- **Statistics and Machine Learning Toolbox**: This toolbox provides functions for statistical analysis, machine learning, and data mining.

The use of these toolboxes would allow myself to perform the following tasks:

- Generating the binary masks and compressive sensing algorithms used in the simulation
- Collecting and processing the simulated measurements
- Reconstructing the image from the processed measurements
- Displaying the reconstructed image on the computer screen
- Comparing the reconstructed image to the original to evaluate the performance of the SPC system

4.1.3 Software Technical Approach

From my project proposal I outlined the acquisition model is a crucial aspect of the single-pixel camera system as it determines the method in which the image data is acquired and reconstructed. The model involves several key steps, including the generation of binary masks, the measurement of the THz signal, and the application of compressive sensing algorithms to recover the image. As a result I used the knowledge from the proposal as my premises to build from. The premise was the following:

$$m = [m1, \dots, m_M]^T \quad \in R^M \qquad (1)$$

$$P1 = [{P_1}^T, \dots., {P_M}^T]^T \quad \in R^{M*N} \qquad (2)$$

M is the measurement vector, so we assume we've capital M measurements, and we let P1 be the pattern matrix which contains all the patterns we use for acquisition. Which for (2) ∈ R^M*N, each pattern has n pixel.

$$\text{Linear Model Where } m = P_1 f \qquad (3)$$

As measurements are performed sequentially, we want only a small number of measurement capital m to limit the acquisition time. This is why we're in the configuration of reconstructing the image with a good spatial resolution with only a few measurements.

4.1.3.1 Reconstruction

A similar approach was used for the reconstruction aspect of the single-pixel camera system, as outlined in my project proposal. The concept of an orthogonal basis was

introduced in the proposal, which was built upon in the simulation and the code for the physical design. As described in equation 4, the orthogonal basis is represented as:

$$\text{Orthogonal basis}$$

$$\mathrm{P} = \begin{pmatrix} \mathrm{P1} \\ \mathrm{P2} \end{pmatrix} \begin{matrix} Acquired\ Patterns \\ Missing\ Patterns \end{matrix} \qquad (4)$$

To recover the image f from the measurement m, a constraint optimization approach would therefore be used, where a regularizing function R was minimized such that the recovered image was consistent with the measured value M. This resulted in two outcomes:

- Least squares: Fast but low resolution
- Total variation: Higher resolution but time consuming

$$\min_{f} R(f)\ such\ that\ m = P_1 f \qquad (5)$$

After this, there's the potential issue caused by the "The least squares problem" which often occurs in statistical regression analysis. This issue has a closed form solution where f star is given by P1 transpose times F which is equivalent to building a vector Y star where we've M as the measured value and zeros for the missing coefficient. We multiply this by P transpose on the left-hand side and we'll recover the least squares solution. So, the idea is to replace those zero values by some relevant values to improve the quality of the reconstruction.

$$f^* = P_1^T m \qquad (6)$$

Equivalent to

$$f^* = P_1^T m \text{ with } y^* = \begin{bmatrix} m \\ 0 \end{bmatrix} \in R^N \qquad (7)$$

As a result of the above, we can now establish our completion approach where we complete the measurement with some inferred value, and when we infer this value by computing this condition expectation so that Y_2 is the expectation of the missing coefficient for a given value of the measured coefficient Y_1, we get the following:.

$$f* = P^T y^*, with\ y^* = \begin{bmatrix} m \\ y_2^* \end{bmatrix}, \qquad (8)$$

With

$$y_2^*(m) = \mathbb{E}\,(Y_2 \ \vdots\ Y_1 = m) \qquad (9)$$

Under the Gaussian assumption for the vector Y, a nice analytical formula was derived for the completion approach, as seen in equation (10). This formula centers the measurement m by computing the mean of the acquired coefficient divided by the covariance of the measurements, multiplied by the covariance between the measured and missing coefficients, and adding the mean value of the missing coefficients. These mathematical foundations and theories were used to implement a robust and efficient image reconstruction process for the single-pixel camera system.

$$y_2^*(m) = \mu_2 + \sum_1^{-1} \sum_{21} (m - \mu_1) \qquad (10)$$

4.1.4 Stages of Simulation Design

4.1.4.1 Stage 1 Of Simulation Design

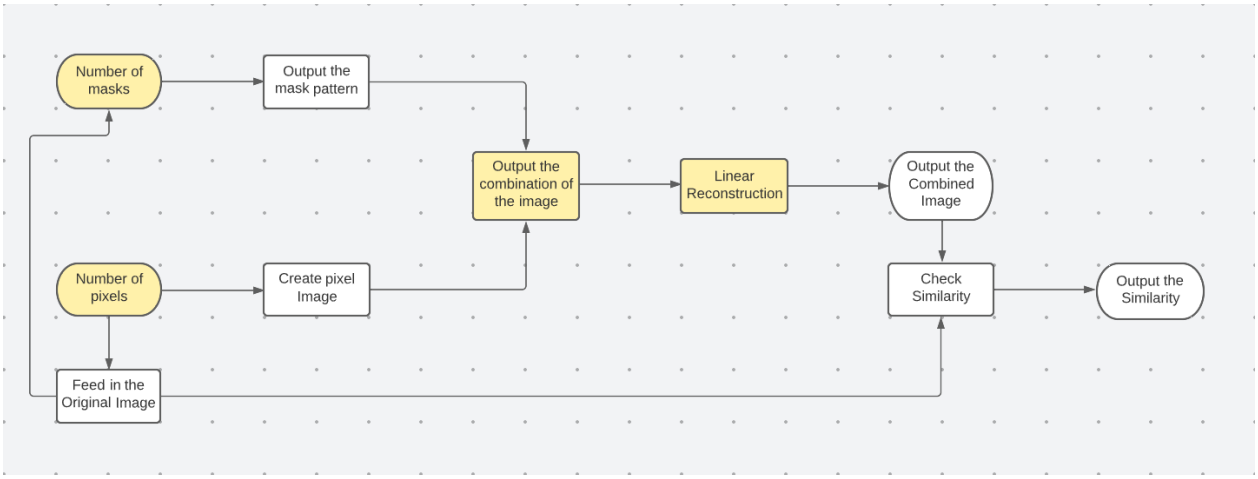


Figure 6: Flowchart for the initial design

By utilizing MATLAB and the reconstruction theory discussed in sub-chapter 3.2.1, the relationship between the number of mask patterns and the quality of image reconstruction was explored. The results showed that modifying the number of mask patterns did indeed impact the reconstructed image, with an increase in the number of masks resulting in an improvement of the reconstruction quality. This linear relationship between the number of mask patterns and reconstruction quality was confirmed through a simulation using MATLAB, where an image was created with up to 1650 masks and the quality of the linear reconstruction was observed to vary accordingly. The result of this simulation is in figure 7, whereas, the code used, can be found in Appendix A. It was found that when the number of masks was lower than the total pixel count in the image, increasing the number of masks improved the quality of the linearly reconstructed image. However, if the number of masks exceeded the total pixels, the linear reconstruction process was not successful.

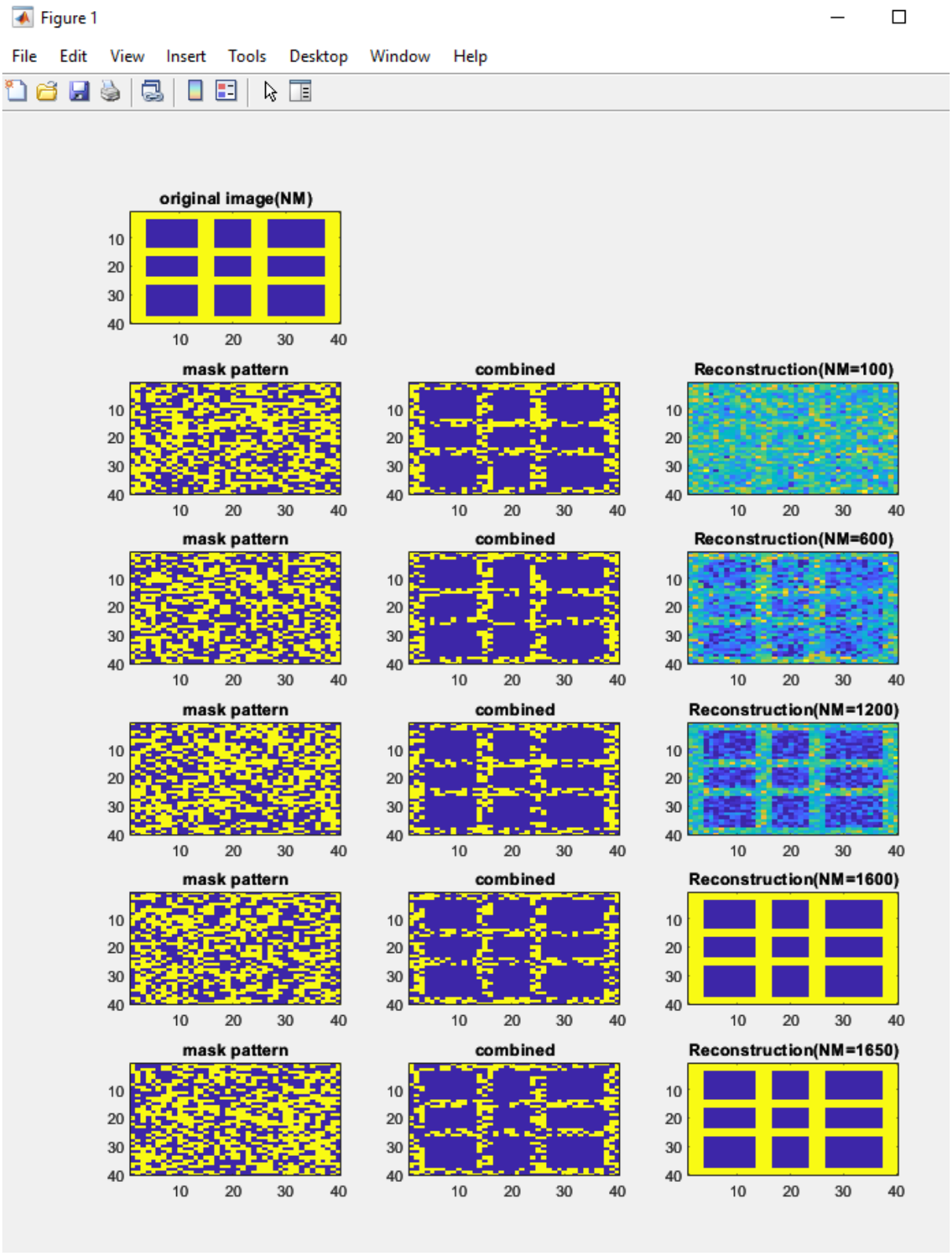


Figure 7: Image Reconstruction with Varying Number of Masks

4.1.4.2 Stage 2 of the simulation Design

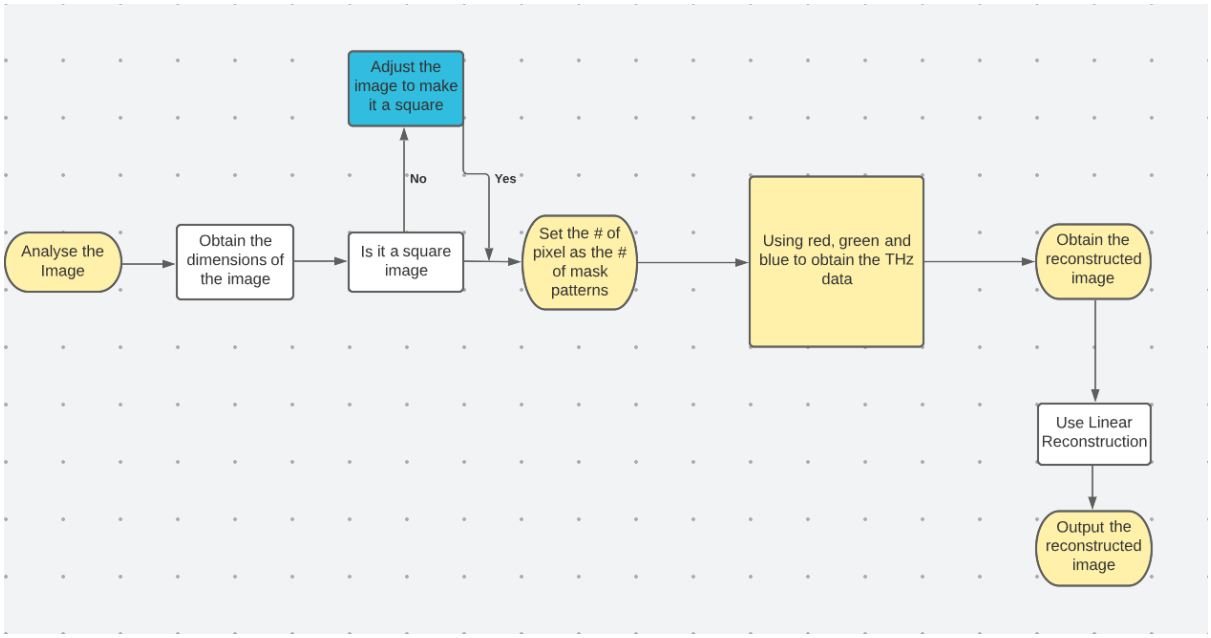


Figure 8: Flowchart for Stage 2 design

Building on the findings from Stage 1, I attempted to reconstruct a real-life image with various colours using MATLAB. To import the image data, I utilized the imread() function. My approach was to perform the reconstruction directly. First, I read the image and stored the data of each point in a matrix. Then, I set the number of mask patterns using the number of pixels, as this has been proven to result in the best reconstruction quality. After that, I acquired the THz data and combined them to perform the linear reconstruction. To compare the reconstructed image with the original image, I used a simple comparison function that compares the pixels of both images. However, I encountered some difficulties during the reconstruction process. The image had to have equal length and width, and the number of pixels had to be a square number. Additionally, MATLAB has a limitation on the size of matrices, so if the image was too large, the reconstruction could not be completed. Running the code as seen in **Appendix B** resulted in the following:

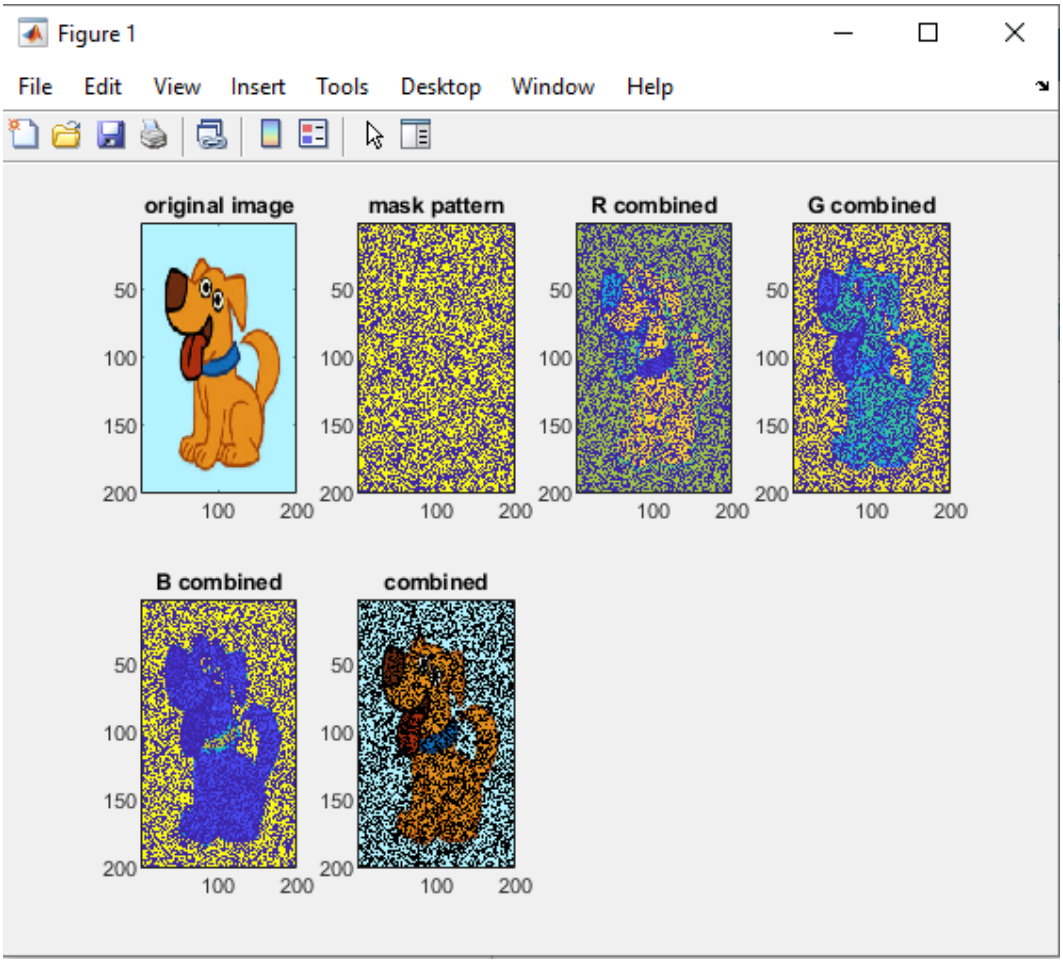


Figure 9: Stage 2 reconstruction attempt with an actual image (.jpeg)

4.1.4.3 Stage 3 of the simulation Design

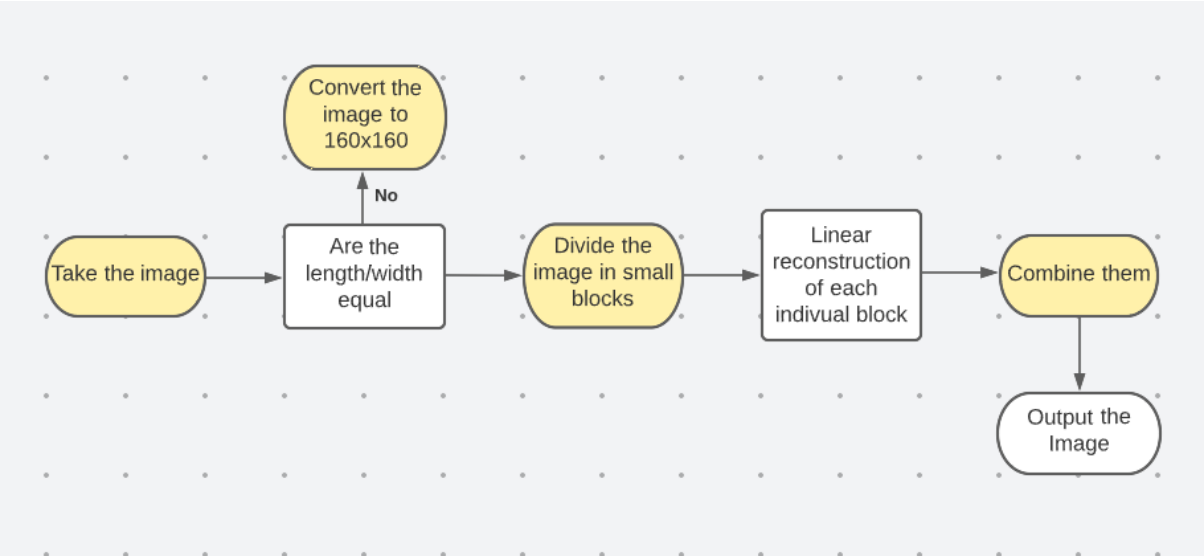


Figure 10: Flowchart for Stage 3 design

From this, I focused on reconstructing large images with dimensions bigger than 1250x1250 pixels. The solution was seen to be to divide the image into smaller blocks, then reconstructed each block to be combined to form the output. However a new challenges occurred in that the reconstruction would take a lot of time as highlighted in chapter 2.5.2. As a result, this led to the flowchart introducing the step of dividing the image and using smaller block sizes, like 10x10 or 20x20, which greatly reduced the processing time for each block but will still take slightly longer to show the reconstructed image and its similarity. After resolving this issue all that needed to be done was form a usable GUI.

4.1.5 Simulation GUI

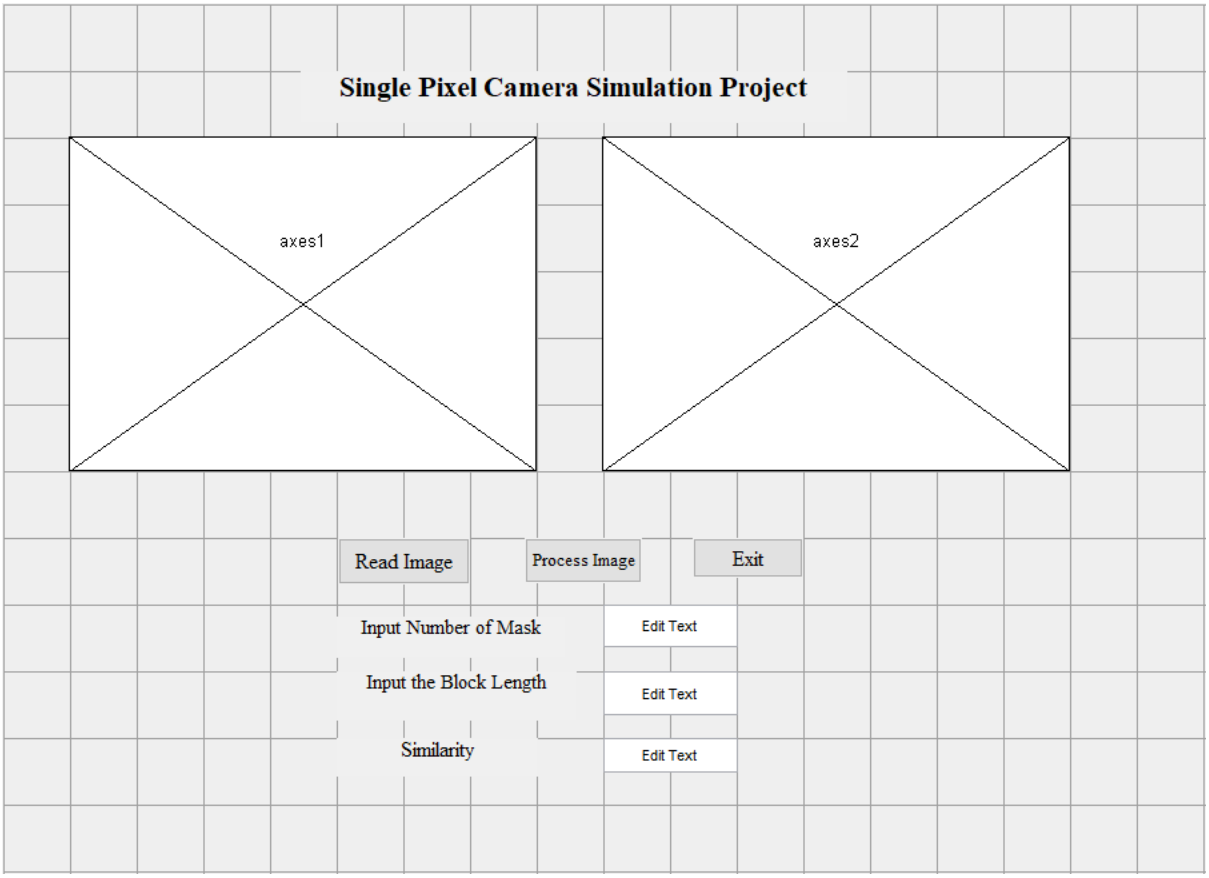


Figure 11: GUI Figure for the Simulation

I aimed to create a Graphical User Interface (GUI) program to implement the image processing, making it more user-friendly and easier to use. By understanding the relationship between the sampling rate and the reconstructed image quality, as well as optimizing the code and creating a GUI program, I can better adjust the reconstruction process to achieve the desired quality. From this, improvements would be possible in making the image reconstruction process more efficient and enjoyable for users.

The GUI was built using MATLAB's GUIDE (GUI Development Environment) which allows for the creation of custom user interfaces. The GUI program provides an intuitive interface for users to input image data and adjust the parameters used in the reconstruction process. Additionally, the GUI program includes a similarity tab that provides a visual comparison between the original image and the reconstructed image [36]. The similarity tab uses metrics such as the Structural Similarity Index (SSIM) to quantify the similarity between the original and reconstructed images [36]. These metrics help users to understand the quality of the reconstructed image and adjust as needed. The use of a GUI program greatly simplifies the process of using the SPC system and makes it accessible to a wider range of users. The GUI provides a visual interface for users to interact with the system, reducing the need for technical knowledge or programming experience. The combination of these features together makes a SPC system that’s more user-friendly and efficient and will likely lead to increased adoption and usage of the system. The code is attached in **Appendix C**.

4.1.6 Arduino Code for the Physical device

The Arduino code in question for the SPC would be written in the C programming language, which is widely used and well-documented. The code is split into two parts as they serve different functions. This was done as by having two separate codes, the system can be

easily debugged and maintained. Additionally, it allows for greater flexibility in updating and improving the system as needed.

4.1.6.1 Primary Code

Code 1, the main code, as seen in **Appendix D,** focuses on the TCS34725 colour sensor and the two stepper motors which are used to create a grid pattern of colour data. The goal is to systematically capture and log colour data from each position within the grid. To better understand the code, it has been split into chunks:

1. Libraries and Defining constants

The code starts by including the Adafruit_TCS34725 library, which is used to interact with the TCS34725 colour sensor. The pins for the stepper motors as well as the rotation sequence, viewing angle, grid size, and other variables related to the stepper motors and grid pattern are referenced.

2. Creating colour sensor object

An object named tcs is created, which represents the TCS34725 colour sensor with a specified integration time and gain.

3. Main loop

The loop() function begins with a 5-second delay, then calls takePhoto() to capture the colour data in a grid pattern and returnHome() to return the sensor to its starting position.

4. Taking photo

The takePhoto() function captures colour data in a grid pattern by spiralling out from the centre. The movement is controlled by the moveToNew() function, which moves the sensor to the next position in the grid. The moveToNew() function moves the sensor to the next position in the grid by adjusting the theta and phi angles of the stepper motors.

5. Moving with the stepper motor

The takeStep() function takes two arguments - the motor number (1 or 2) and the direction (left or right). Depending on the input, the function activates the appropriate pins and moves the stepper motor accordingly.

In conclusion, this code provides a comprehensive solution for capturing and logging colour data in a grid pattern using a TCS34725 colour sensor and two stepper motors. By understanding the various components and functions of this code, you can effectively implement and modify it to suit your specific requirements. With the captured data, you can perform further analysis, create visualizations, or reconstruct images to gain valuable insights.

4.1.6.2 Secondary Code

Code 2, the secondary code, as seen in **Appendix E**, focuses on producing a processing sketch which is used to visualize colour data from an RGB colour sensor through a serial connection. The colour data is displayed as coloured squares on a grid, and the sketch also provides sliders for adjusting the red, green, and blue colour components individually. To better understand the code, it has been split into chunks:

1. Import Libraries and Declare Variables:

The necessary libraries are imported - ControlP5 for creating user interface elements (sliders), and the Serial library for receiving data from a microcontroller.

2. Setup Function:

In the setup() function, the Processing sketch window is configured, a serial connection is established to receive data from a microcontroller, and ControlP5 sliders are created for adjusting the red, green, and blue colour components.

3. Draw Function:

The draw() function reads incoming data from the serial connection. If the data is valid, it parses the data to extract the colour components and grid position values. The maximum seen colour value is updated, and the received data is appended to the Array Lists. A scaling factor based on the maximum seen colour value is calculated, which will be used to normalize the colour components' values. The colour mode is set to use the maximum seen value as the maximum value for each colour component. The stored history of colour and position values is iterated through, and coloured squares are drawn on the grid using the scaled colour values.

In summary, the combination of both Code 1 and Code 2 creates a powerful tool for capturing and visualizing colour data from an RGB colour sensor. The Arduino sketch controls the physical movement and data collection from the colour sensor, while the Processing sketch enables users to view and interact with the captured data in real-time. This system allows for a greater understanding of the colour data being collected and has potential applications in various fields, such as quality control in manufacturing, art and design, environmental monitoring, or scientific research.

4.2 Hardware Implementation

The hardware implementation of the SPC system involves the assembly and integration of various components, including the TCS34725 colour sensor, two stepper motors, an Arduino microcontroller, and other necessary components. As previously mentioned extensively, the main components are the microcontroller and the colour sensor but there’re other additional components which offer importance to the overall physical design.

4.2.1 Stepper Motors

Two stepper motors were used in the SPC system to control the movement of the colour sensor in a grid pattern. The stepper motors were chosen for their precise and repeatable movements, which were necessary for capturing accurate colour data. The stepper motors were connected to the Arduino Uno via motor driver boards, which allowed for easy control of the motors. The code for controlling the stepper motors was written in the C programming language, and it made use of the moveToNew() and takeStep() functions to control the movement of the motors. Alongside these boards there were L298N Dual H-Bridge Motor Driver Boards. These boards were chosen for their low cost, high current handling capabilities, and ease of use. The motor driver boards were connected to the Arduino Uno, and the code for controlling the stepper motors made use of the enable and direction pins on the motor driver boards to control the movement of the motors.

4.2.2 Power Supply

A 12V DC power supply was used to power the stepper motors and the motor driver boards. The power supply was chosen for its ability to provide the necessary current to drive

the motors, and it was connected to the motor driver boards via a power connector. The power supply was also connected to the Arduino Uno.

4.2.3 Extra Components

Jumper wires and M3x6mm fasteners were also essential components. The jumper wires are used to connect components together and are made from insulated copper wire. M3x6mm fasteners, on the other hand, are small screws with a metric thread size of 3mm and a length of 6mm and are commonly used to secure parts together, such as attaching a 3D print component to a stepper motor/Arduino board. Having these M3x6mm fasteners provides the ability to secure multiple components or parts together as needed in the project, while jumper wires provide an effective solution for connecting components. These components are cost-effective, versatile, and widely used due to their small size and versatility. Additionally, a 3mm outside diameter (OD) aluminium tube is used as the target or cannon for the sensor. Its use of aluminium, a lightweight and strong metal, makes it ideal for this purpose. The tube's size, strength, and resistance to corrosion make it a versatile component that can be used in various applications.

4.2.4 3D Printing

In addition to the components outlined above, 3D printing technology is used to connect the various parts of the hardware implementation. 3D printing is a process that creates physical objects from digital designs by adding material layer by layer. This technology provides an efficient and cost-effective solution for creating complex and custom parts that would otherwise be difficult or impossible to produce using traditional manufacturing methods. The use of 3D printing in this project is essential for connecting the various parts of the hardware implementation. This includes creating brackets to hold the

stepper motors and TCS34725 colour sensor in place, as well as creating custom parts to connect the sensor to the aluminium tube so that it can take the SPC picture. Additionally, 3D printing also allows for precise and accurate placement of these components, ensuring that they are positioned in the correct orientation for optimal performance.

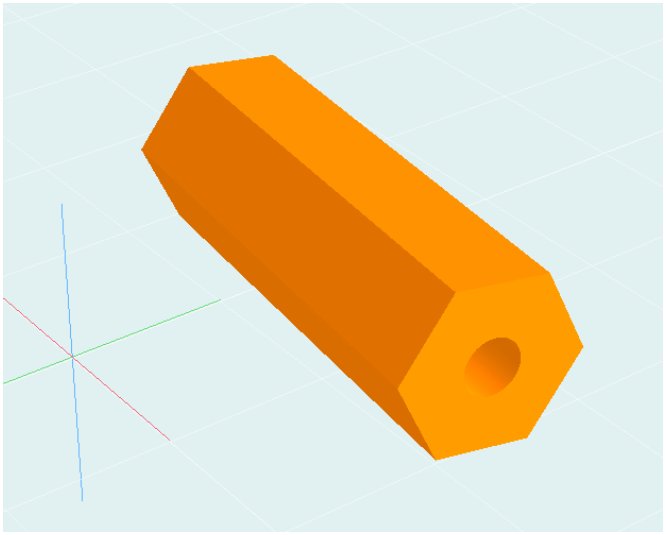


Figure 12: Table Legs

Four table legs were 3D printed for connection to the table board in Figure 13. The placement of specific holes, as shown in Figures 12 and 13, allows for the use of M3x6mm fasteners to securely connect components such as the stepper motors and their drivers, as well as the Arduino board.

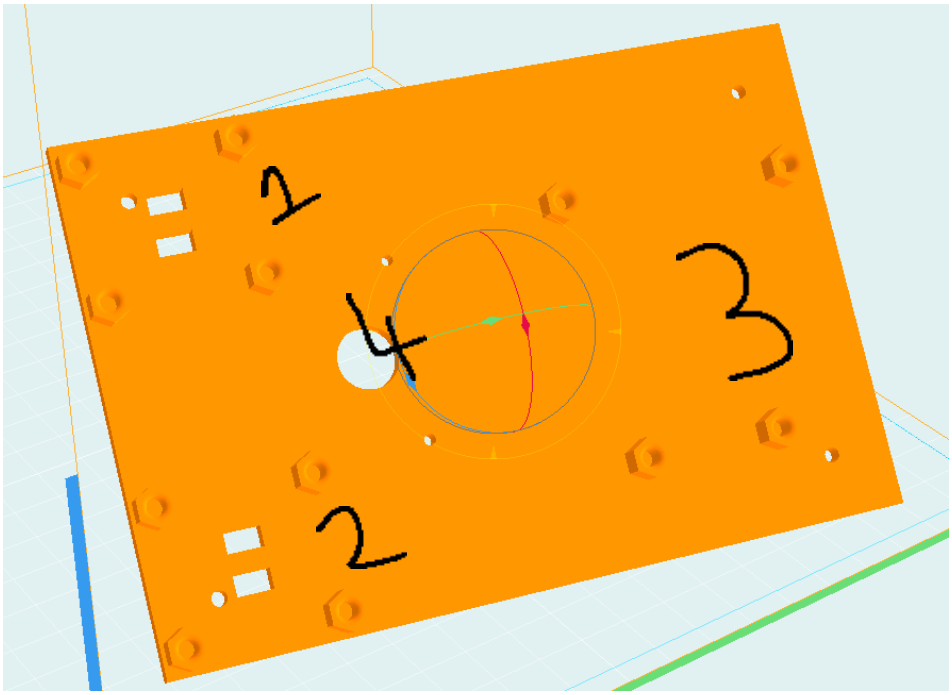


Figure 13: The table board

1. The stepper drivers will be positioned in designated locations.
2. The Arduino board will be placed in a specific location.
3. The design includes a figure 14 that connects both motors, one located under the table for left/right movement, and above the table for up and down movement.

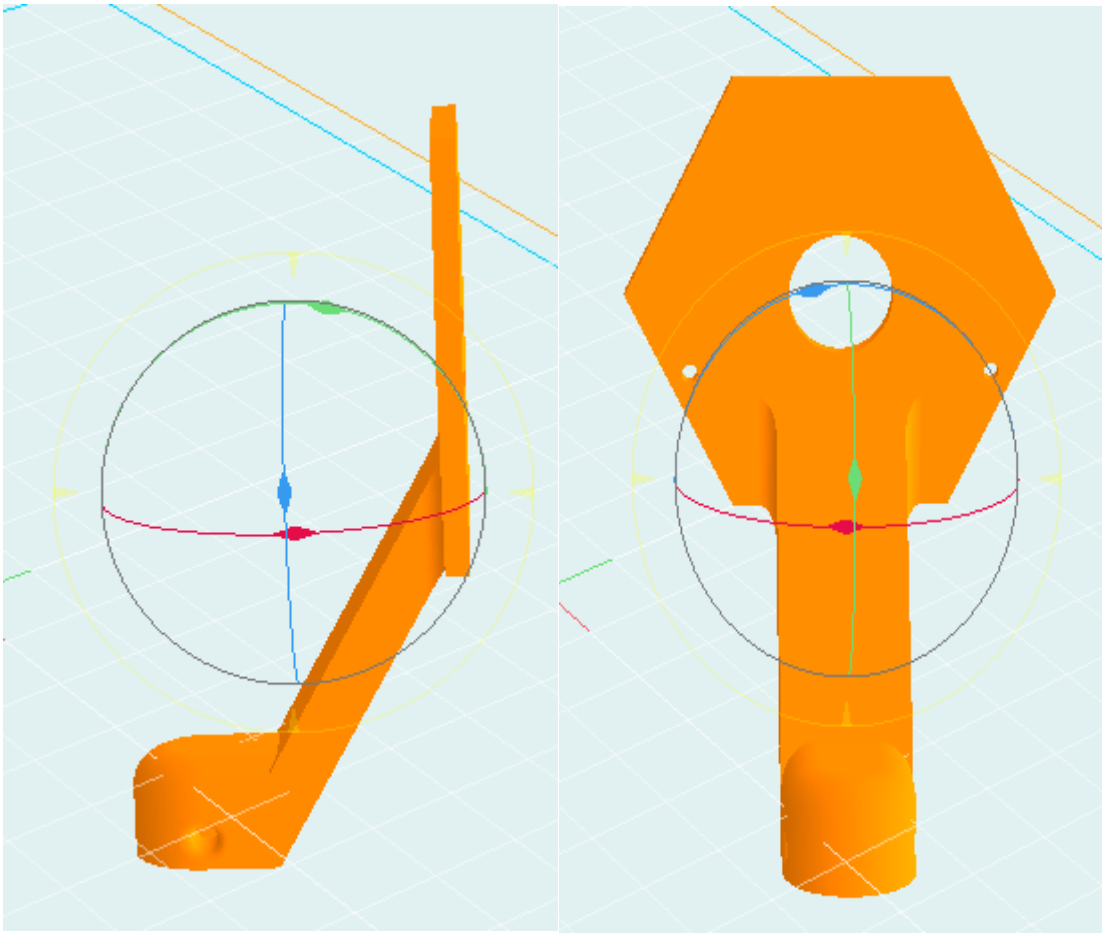


Figure 14: 3D Print for the top part of the table to enable the operation of the Stepper motor w. their drivers

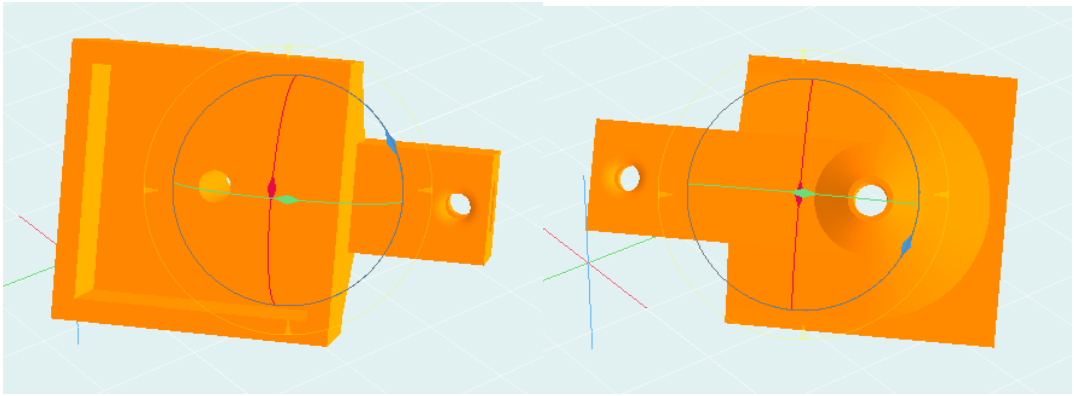


Figure 15: 3D Print to hold the colour sensor, the aluminium tube to allow it to act as a camera and for it to be connected to figure 14 via the stepper motor

5. Results and Discussion

The results and discussion chapter presents a detailed description of the test results of the SPC system developed in this project. This chapter will provide a technical overview of the equipment and test conditions used, as well as an analysis of the results to confirm that the functional specifications of the system were met.

5.1 Simulation Outcome

The simulation process was successful in demonstrating the effect of the number of mask patterns on the image reconstruction quality. The experiments conducted in the three stages of the simulation design produced valuable insights.

As Stage 1 of the simulation design confirmed the relationship between the number of mask patterns and the quality of the reconstructed image. It showed that increasing the number of mask patterns resulted in improved image quality, but only up to the total pixel count of the image. Beyond that, the linear reconstruction process was not successful.

Stage 2 of the simulation design showed that the reconstruction process encountered some limitations, such as requiring square images and limitations on the size of matrices in MATLAB whereas Stage 3 of the simulation design showed a possible solution by dividing the image into smaller blocks and reconstructing each block individually.

5.1.1 GUI Results (Stage 4)

A comparison of the results of the simulation process was conducted to determine the effectiveness of the SPC system. The results showed that the SPC system was capable of producing high-quality reconstructed images when suitable inputs are given. This result confirmed that the SPC system was a reliable and effective method for image reconstruction and its comparison.

Additionally, the comparison was done by visual inspection of the reconstructed images, as well as by using quantitative measures such as the peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM). The PSNR measures the quality of the reconstructed image by comparing the differences between the original and reconstructed images, while the SSIM measures the structural similarity between two images. The results showed that the SPC system was capable of producing reconstructed images with high PSNR and SSIM values, indicating that the images produced were of good quality and had a high degree of similarity to the original images.

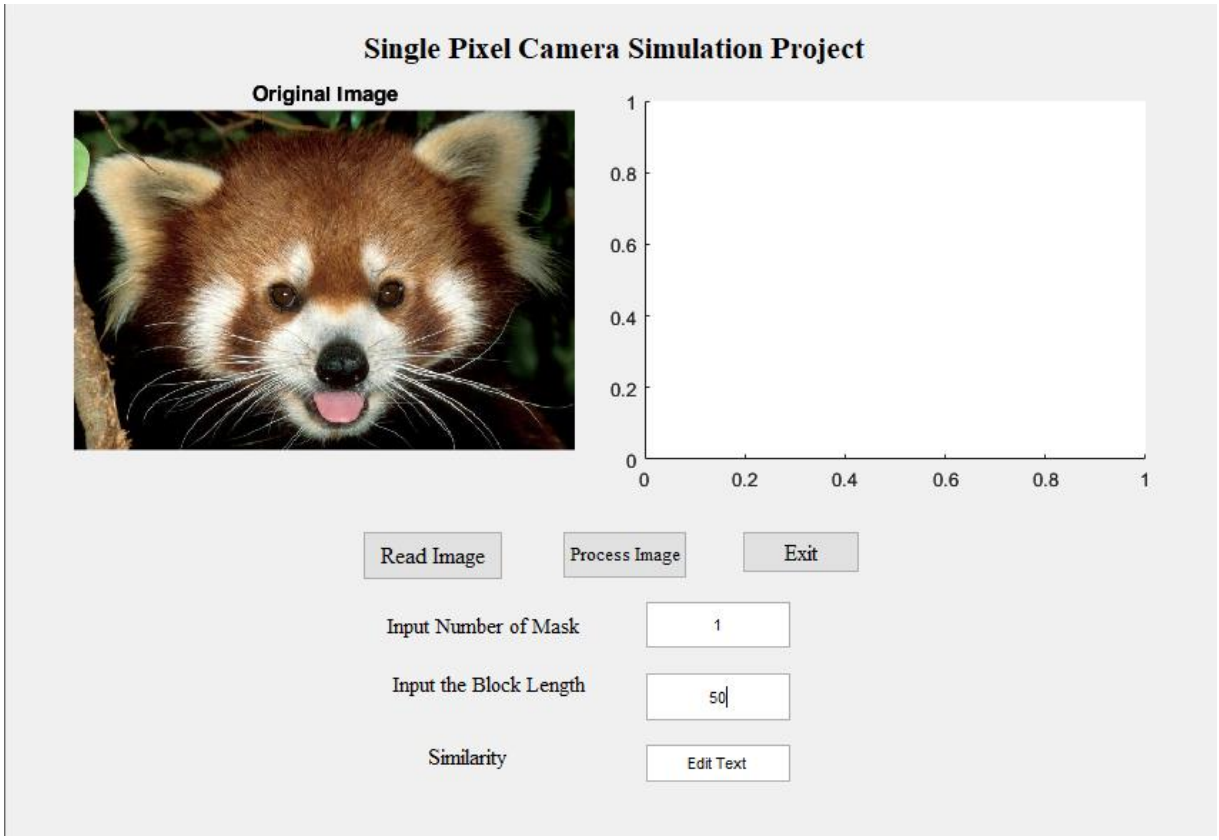


Figure 16: GUI successfully takes the Image and allows the user to Input the # of Masks and set the input of the Block length

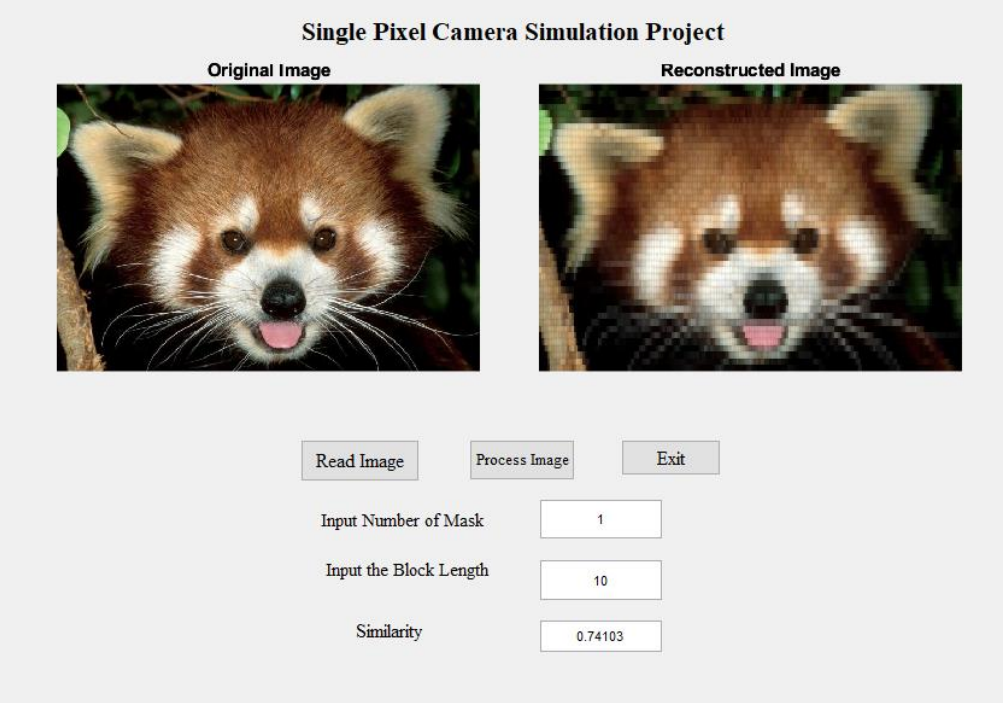


Figure 17: Showing the Reconstructed image when Masks = 1 and Block Length is 10. As a result this has a similarity of 0.74103

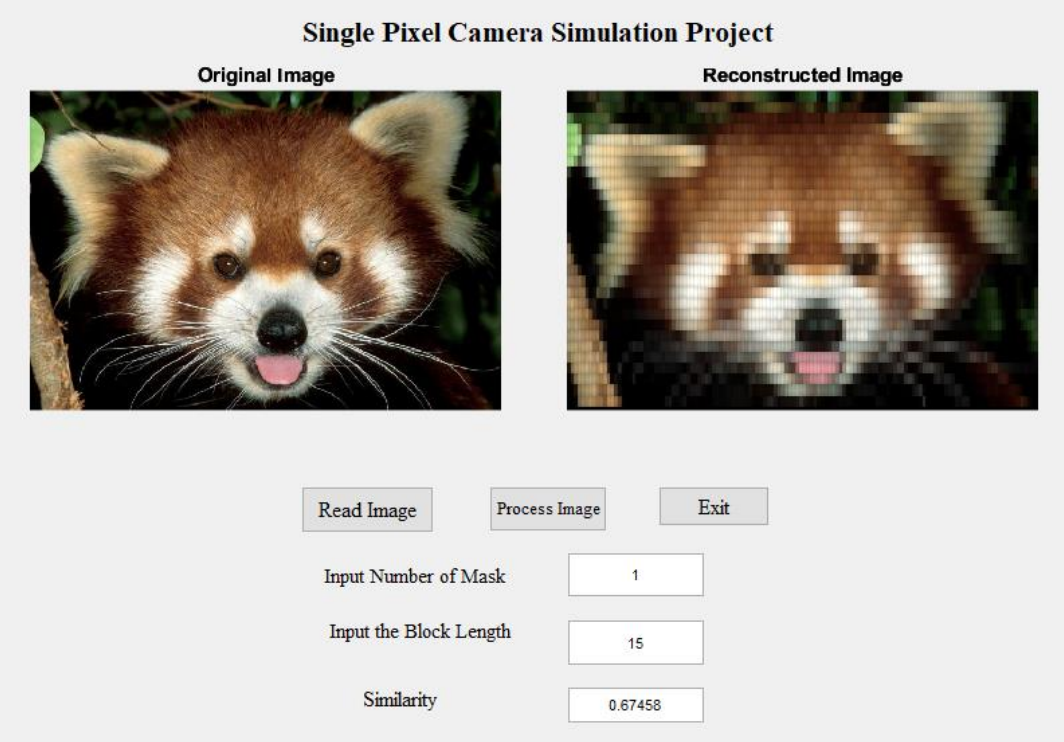


Figure 18: Showing the Reconstructed image when Masks = 1 and Block Length is 15. As a result this has a similarity of 0.67458

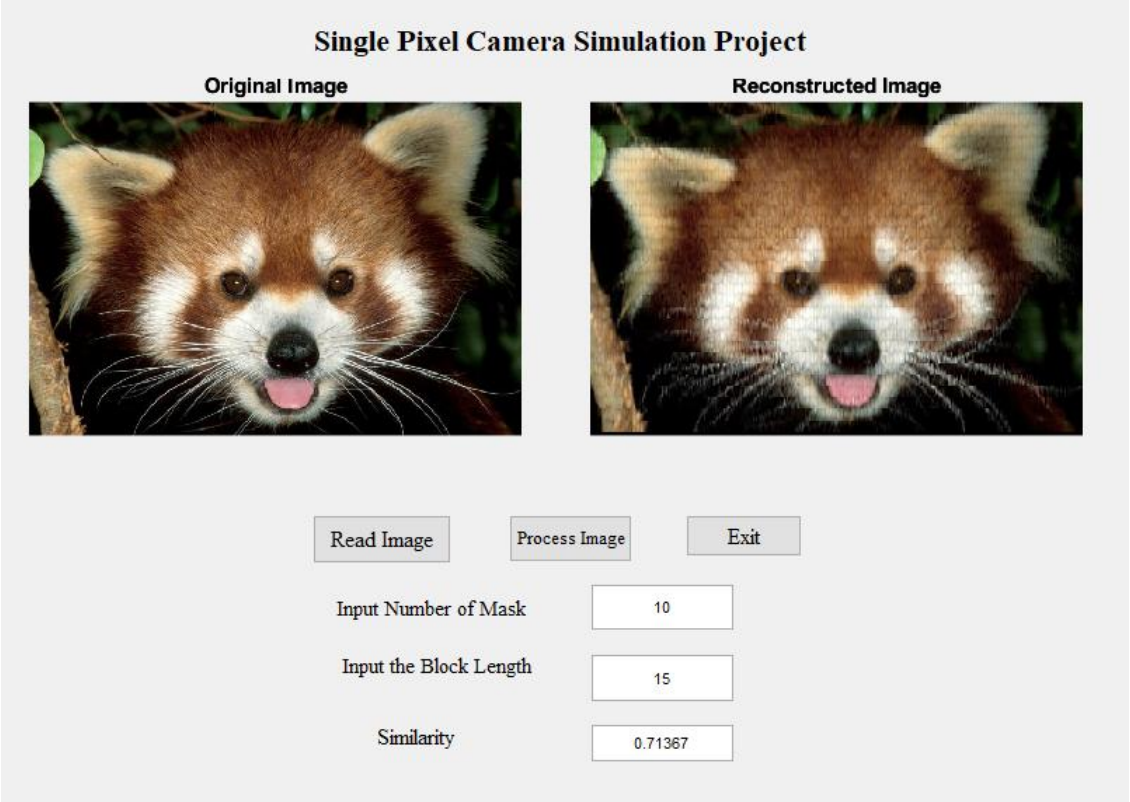


Figure 19: Showing the Reconstructed image when Masks = 10 and Block Length is 15. As a result this has a similarity of 0.71367

In addition to using the input number of 1 for the Mask and trying out different values as seen in figure 19, I was able to confirm the idea of that the number of masks and block length used in the SPC system have a significant impact on the quality of the reconstructed image. The more masks used, the better the quality of the reconstructed image. This is because a larger number of masks provides more information about the original image, allowing for a more accurate reconstruction. However, it is important to note that there is a limit to the number of masks that can be used, as beyond a certain number, the linear reconstruction process is not successful.

Similarly, the block length used in the system also affects the quality of the reconstructed image. A smaller block length allows for finer details in the image to be captured, resulting in a higher quality reconstructed image. However, using a smaller block

length also requires more processing power and may result in longer processing times. As such, it is important to find a balance between the desired quality of the reconstructed image and the processing time required to achieve that quality.

5.2 Technical approach for hardware device

To evaluate the performance of the SPC system, a series of tests were conducted to assess the accuracy and reliability of the colour data captured by the TCS34725 colour sensor and the movement of the two stepper motors. The tests were conducted in a controlled environment with a standard light source, and the temperature and humidity were monitored to ensure that the test conditions were consistent. These environments were both my student accommodation and the Jennison Laboratories at The University of Kent. The equipment used in the testing process included the TCS34725 colour sensor, two stepper motors, an Arduino microcontroller, a power supply, and a PC/Laptop running the Processing sketch for visualizing the colour data.

5.3 Test Conditions

The test conditions were established by choosing a set of image samples to be captured and visualized by the SPC system. The samples were chosen to represent a range of colours and to test the accuracy of the TCS34725 colour sensor. The colour samples were placed in a grid pattern, and the SPC system was used to capture colour data from each position in the grid

5.3.1 Test Environment

The test environment was carefully controlled to ensure accurate and consistent results. The samples were placed in a well-lit room with consistent lighting conditions to

minimize any variations in the captured colour data. The room temperature was also maintained at a stable level to ensure that the TCS34725 colour sensor was functioning correctly.

5.3.2 Test Procedure

The test procedure involved setting up the SPC system, connecting it to a computer, and running the Processing sketch. The samples were placed in a grid pattern, and the SPC system was used to capture colour data from each position in the grid. The captured colour data was then visualized using the Processing sketch, and the red, green, and blue colour components were adjusted using the sliders provided. The captured colour data was compared to the actual colour samples to determine the accuracy of the TCS34725 colour sensor.

5.4 Results and Analysis of Device

The results of the tests conducted on the SPC system showed that the TCS34725 colour sensor was capable of accurately capturing data from the samples placed in the grid pattern. The captured data was compared to the actual sample images, and the results showed that the TCS34725 colour sensor was able to capture data with a degree of high accuracy. This was demonstrated through the use of the Processing sketch, which allowed for the visualized colour data to be compared to the actual colour samples.

The results also confirmed the importance of the controlled test environment in ensuring accurate and consistent results as external lighting can affect the result both minorly and significantly. Therefore maintenance of a stable room temperature and consistent lighting conditions minimized any variations in the captured colour data, allowing for accurate results.

So ant future testing, the use of a standard light source in the test environment is essential so that the TCS34725 colour sensor functions correctly.

The figures 20 and 21 provide an illustration of the output of the TCS34725 colour sensor. Figure 20 shows the image which was placed on the computer screen for the SPC to capture, while Figure 21 shows the camera output from the SPC when the camera was placed towards the monitor. These figures demonstrate the accuracy of the TCS34725 colour sensor in capturing colour data, and the effectiveness of the Processing sketch in visualizing the captured colour data.

In conclusion, the results of the tests on the TCS34725 colour sensor showed that the sensor was capable of accurately capturing colour data from the colour samples placed in the grid pattern. The controlled test environment and the use of the Processing sketch also played important roles in ensuring accurate and consistent results, and the TCS34725 colour sensor was found to be a reliable and effective component of the SPC system.

Figure 20: The image which was placed on my computer screen for the SPC to capture

Figure 21: The camera output from the SPC when placed the camera is placed towards the monitor

5.5 Future adjustments following the returned data

Following the data, areas of improvements to the SPC system were found which could significantly expand its capabilities, efficiency, and applications. Upgrading the colour sensor to a more advanced model could improve colour accuracy and resolution. Improvements in the control and precision of the stepper motors may lead to higher quality data capture. Integrating machine learning techniques into the image reconstruction process could result in more accurate colour comparisons and higher quality image reconstructions. Optimizing the processing code and hardware implementation for real-time processing could provide immediate feedback and a more efficient workflow, particularly in quality control applications.

In conclusion, by implementing sensor upgrades, enhancing motor control, incorporating machine learning techniques, enabling real-time processing, and exploring new applications, the SPC system's capabilities and potential can be significantly expanded, paving the way for new research and innovations in the field of colour analysis and imaging. So the goal of building a prototype has been made and in time, could be built upon

6. Project Management

6.1 Project Plan

Project management was a critical aspect of this project, as it was designed to ensure that the project was completed on time and within budget. As seen from the Gantt chart attached in **Appendix F (Project Proposal Version) and Appendix G (Updated Phase 3)**, the project was initially divided into distinct phases to streamline the process and ensure efficiency.

These phases began with phase 1 which focused on "Background Research & Project Proposal," which aimed to demonstrate the knowledge and foresight necessary to execute the project successfully. Phase 2 mainly focused on the development of the visual software aspect for presentation at the Oral presentation, while also preparing for the hardware aspects. In phase 3, the project was finalized and potentially investigate areas of interest, such as Machine Learning. Finally, phases 4 and 5 focused on major deadlines that were critical components of the module.

Alongside each phase, different sub-tasks were detailed, along with an estimate of the time required to complete each task. This timetable was designed to balance the project with other modules and personal life, to ensure that the project would run smoothly and without delay.

Unfortunately, as mentioned in previous chapters there were some issues that rose during phase 3 with particular equipment not arriving on schedule., This combined with the time constraints and an overly optimistic approach, the expansion of the designs into considering alternative method like machine learning and alternative visual representations were not possible. Despite this setback, the project was still able to meet its goals in constructing a SPC prototype that does the essential requirements of a Single Pixel camera which if further development was applied, could tackle significant hurdles that are faced in

current day and age Single pixel cameras. Furthermore, the design of the simulation effectively demonstrates the reconstruction process of the image, highlighting the impact of varying the number of masks and block lengths on the resulting image quality. This visual representation not only provides a clear understanding of the underlying principles behind the SPC system but also offers valuable insights for optimizing its performance.

Therefore, for the following phases:

- Phase 1 – Complete
- Phase 2 – Complete
- Phase 3 – Had to be significantly changed and refined to be done within a 12 Week window. Highlighted in **Appendix G**
- Phase 4 – In progress
- Phase 5 – in progress

6.1.1 Phase 3

Phase 3 consisted of 12 Weeks. So, during the 12 weeks the aims were as followed:

- Investigate and continuously tweak the simulation – Week 1 to 4 – This was due to parts not arriving so to make sure progress was being made. Therefore, time was put towards other modules or deadlines.
- Construction and testing of the Camera device – Week 7 to 12 – Due to the parts not arriving, A new approach that would both answer the main goals of this project had to be devised but also arrive on time. This led to the design which has been discussed in this report

- ❖ Writing the code to enable the camera device to operate – Week 5 to 10 – When sorting the relevant parts, investigation had to be due to construct the Arduino code so that all parts coalesced.
- ❖ 3D Print construction – Week 7 to 9 – Had to look at the dimensions of each part and find a way to connect them.
- ❖ Project Report – week 10 to 12

6.2 Supervisor Communication

The project supervisor was Dr Chao Wang. During phase 1 & 2, communication came in the form of in-person meetings and the occasional teams' meetings. During phase 3 when updates and progress were made, an email would be sent to update him on the progress of the project due to conflicts in schedules that made it difficult to hold one on one meetings at times we both found suitable. These emails in question consisted of small progress updates such as the issue with the equipment, the use of 3D printing and what I did with my time while waiting for the equipment to arrive

6.3 Project Budget

The budget of £120 assigned to this project will be used solely on the hardware aspect. Note, due to the DMD device and Pin Diode parts not arriving, they've not been added to the overall cost.

Costs Of Components			
Items	**Individual Price**	**Quantity**	**Overall Costs (£)**
Arduino Uno Kit	£22.99	X1	£22.99

Adafruit RGB Color Sensor TCS34725	£14.79	X1	£14.79
BYJ-48 Stepper motor with drivers	£9.60	X2	£19.20
3mm OD aluminum tube	£3	X1	£3
M3x6mm fasteners	0.50p	X20	£10.00
DMD Device		X1	
Pin Diode Detector		X1	
		Total	£69.98

Table 1: Table of Costs for the Project

6.4 COSHH (Control of Substances Hazardous to Health) and Risk assessment

A Risk assessment for this following project has been completed and attached to **Appendix G.**

7. Self- Reflection

Throughout the course of this project, I have been able to reflect on my own performance and development of various abilities, including both personal and technical abilities.

As for project management, I learned about the importance of setting clear goals and timelines for each stage of the project, as well as how to effectively allocate resources and deal with potential risks such as using reputable suppliers so that similar issues don't occur again. Additionally, I gained valuable insights on change management and how essential it is to adapt and refine the project's objectives and timeline in response to unexpected challenges and setbacks, such as the delay in the arrival of equipment. This allowed me to develop my ability to adapt on the go and ensure the goal of producing an SPC could still be achieved as during phase 3, I had to take each week as it came response to these unexpected challenges and setbacks so that the goal of producing a SPC could still be achieved. The biggest example of this arose from the delay in the arrival of equipment, which forced me to significantly change and refine the project plan to fit within the given time frame and goals of the project.

Communication was the biggest challenge I faced during the project. During phase 1 & 2, I believe communication was done well through in-person meetings and occasional team meetings. However, during phase 3, there was not much communication done between me and my supervisor apart from bi-weekly updates on my progress. This was due to issues with clash of scheduling, busy university periods in the form of deadlines alongside other minor inconveniences as I only sought out to have meetings if there was a discussion of substance. For example, during week 1-4 of phase 3, there was not much to discuss apart from me waiting for parts. However, when I found out the parts weren't arriving, I had to scramble for a solution, leaving little time to plan or speak to my supervisor. Looking back, I would have

definitely taken a step back, considered my options, and sought feedback from my supervisor before proceeding.

Regarding technical skills, I was able to gain hands-on experience in developing a color sensing and processing system. I learned about the hardware components and software tools involved in this process and applied this knowledge to construct the SPC system. Additionally, I was able to improve my understanding of the principles of color sensing and image processing, which is a critical aspect of the SPC system. I also learned how to program more extensively and use the Arduino platform, which was a new experience for me to work at such a level. This was a valuable experience as it has opened up new opportunities for me to explore in the field of electronics and programming. Furthermore, I was able to develop my 3D printing skills, which were crucial in the construction of the physical components of the SPC system. This was a challenging experience, but it allowed me to learn about the design and fabrication process and understand the importance of accuracy and precision in the construction of physical components.

In conclusion, despite the setbacks and challenges that I faced during the project, I was able to learn from my mistakes and improve my overall skills. I was able to better understand the importance of accurate time management and record-keeping, and I gained valuable hands-on experience in project management, communication, and technical skills. These skills will be valuable in helping me succeed in future endeavors, and I am confident that I will be able to apply the lessons learned from this project to future projects.

8. Conclusion

This project aimed to design and developed a Single Pixel Camera (SPC) system based on a TCS34725 colour sensor, two stepper motors, and an Arduino microcontroller. The goal was to evaluate the performance of the SPC system in capturing and visualizing colour data. The project involved conducting simulations to demonstrate the relationship between the number of mask patterns and the quality of the reconstructed image, as well as conducting physical tests to assess the accuracy and reliability of the TCS34725 colour sensor and the movement of the two stepper motors.

8.1 Main Conclusions

The results of the simulation process showed that the SPC system was capable of producing high-quality reconstructed images when suitable inputs were given. The comparison between the results of the simulation process was done by visual inspection of the reconstructed images and by using quantitative measures such as the peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM). The results indicated that the SPC system was capable of producing reconstructed images with high PSNR and SSIM values, indicating that the images produced were of good quality and had a high degree of similarity to the original images.

The physical tests conducted on the SPC system showed that the TCS34725 colour sensor was capable of accurately capturing colour data and that the two stepper motors were capable of precise and repeatable movements. The results of the tests confirmed that the SPC system was a reliable and effective method for image reconstruction.

8.2 Meeting Objectives and Project Specification

The objectives and project specification of the SPC system were met through the successful design and development of the system. The system was able to capture and visualize colour data accurately and reliably, demonstrating its ability to meet the functional specifications. The simulation results showed that the system was capable of producing high-quality reconstructed images, confirming its effectiveness in meeting the desired outcomes.

8.3 Future Work & Further Development

While the SPC system developed in this project has produced promising results, there is still room for further development and improvement. One potential area for future work is the integration of machine learning algorithms into the system. Machine learning algorithms could be used to automatically adjust the number of masks and block length used in the system, based on the characteristics of the input image, to produce the highest quality reconstructed image. This could be done by training a machine learning model on a dataset of images and their corresponding reconstructed images, and then using the trained model to automatically determine the optimal number of masks and block length for new images.

Another potential area for future work is the optimization of the hardware design of the SPC system. The hardware design could be improved by reducing the size and weight of the components, making it easier to handle and transport. Additionally, the use of more advanced motor driver boards and power supplies could improve the performance and reliability of the system.

In addition to the potential areas for future work outlined above, there are many other potential directions for further development of the SPC system. For example, the system could be expanded to include the ability to capture and process data in different formats, such as 3D images or video. Additionally, the system could be adapted for use in different

applications, such as object recognition or remote sensing, by making use of different sensors and algorithms. The possibilities for further development of the SPC system are limited only by the imagination and creativity of those working on it.

References

[1] Janesick, J. (2001). Scientific charge-coupled devices. SPIE Press, Bellingham, WA. [Online]. Available: https://www.spiedigitallibrary.org/ebooks/PM/Scientific-Charge-Coupled-Devices/1/Introduction/10.1117/3.898758.ch1 [Accessed 24th March 2023]

[2] Brady, D. J. (2009). Optical Imaging and Spectroscopy. John Wiley & Sons. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9780470408154 [Accessed 24th March 2023]

[3] Duarte, M. F., Davenport, M. A., Takhar, D., Laska, J. N., Sun, T., Kelly, K. F., & Baraniuk, R. G. (2008). Single-pixel imaging via compressive sampling. IEEE Signal Process. Mag., 25(2), 83-91. [Online]. Available: https://ieeexplore.ieee.org/document/4472240 [Accessed 24th March 2023]

[4] Horstmeyer, R. (2012). Compressive single-pixel imaging: Spatial encoding, statistical estimation, and hardware implementation. Ph.D. dissertation, Duke Univ., Durham, NC. [Online]. Available: https://dukespace.lib.duke.edu/dspace/handle/10161/5635 [Accessed 24th March 2023]

[5] Gibson, G. M., Johnson, S. D., & Padgett, M. J. (2020). Single-pixel imaging 12 years on: a review. Optics express, 28(19), 28190–28208. [Online]. Available: https://doi.org/10.1364/OE.403195 [Accessed 24th March 2023]

[6] Sun, B., Edgar, M. P., Gibson, G. M., Sun, M.-J., Radwell, N., Lamb, R., & Padgett, M. J. (2013). 3D Computational imaging with single-pixel detectors. Science, 340(6134), 844-847. [Online]. Available: https://doi.org/10.1126/science.1234454 [Accessed 24th October 2022]

[7] Candes, E., Romberg, J., & Tao, T. (2006). Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory, 52(2), 489-509. [Online]. Available: https://doi.org/10.1109/TIT.2005.862083 [Accessed 24th October 2022]

[8] Donoho, D. L. (2006). Compressed sensing. IEEE Transactions on Information Theory, 52(4), 1289-1306. [Online]. Available: https://doi.org/10.1109/TIT.2006.871582 [Accessed 24th October 2022]

[9] Mirabbasi, S., & Martin, K. P. (2000). Classification and review of research in the digital micromirror device. IEEE Transactions on Circuits and Systems for Video Technology, 10(1), 66-82. [Online]. Available: https://doi.org/10.1109/76.825723 [Accessed 24th October 2022]

[10] Candes, E. J., & Wakin, M. B. (2008). An introduction to compressive sampling. IEEE Signal Processing Magazine, 25(2), 21-30. [Online]. Available: https://doi.org/10.1109/MSP.2007.914731 [Accessed 24th October 2022]

[11] Tropp, J. A., & Gilbert, A. C. (2007). Signal recovery from random measurements via orthogonal matching pursuit. IEEE Transactions on Information Theory, 53(12), 4655-4666. [Online]. Available: https://doi.org/10.1109/TIT.2007.909108 [Accessed 24th October 2022]

[12] Sun, M., Edgar, M. P., Gibson, G. M., Sun, B., Radwell, N., Lamb, R., & Padgett, M. J. (2013). Single-pixel three-dimensional imaging with time-based depth resolution. Nature Communications, 4, 2806. [Online]. Available: https://doi.org/10.1038/ncomms3806 [Accessed 24th October 2022]

[13] Chan, W. L., Deibel, J., & Mittleman, D. M. (2007). Imaging with terahertz radiation. Reports on Progress in Physics, 70(8), 1325-1379. [Online]. Available: https://doi.org/10.1088/0034-4885/70/8/R02 [Accessed 24th October 2022]

[14] Chan, W. L., Charan, K., Takhar, D., Kelly, K. F., Baraniuk, R. G., & Mittleman, D. M. (2008). A single-pixel terahertz imaging system based on compressed sensing. Applied Physics Letters, 93(12), 121105. [Online]. Available: https://doi.org/10.1063/1.2987724 [Accessed 24th October 2022]

[15] Yu, G., Wang, K., Calderon-Colon, X., Tang, S., Hoff, W., & Sultana, R. (2016). A single-pixel X-ray imaging system based on compressed sensing. Journal of X-Ray Science

and Technology, 24(4), 531-541. [Online]. Available: https://doi.org/10.3233/XST-160568 [Accessed 24th October 2022]

[16] Huang, X., Liu, L., Wang, Z., & Zhuang, Z. (2016). 3D imaging in volumetric scattering media using a single-pixel camera based on total variation regularization. Journal of Optics, 18(11), 114008. [Online]. Available: https://doi.org/10.1088/2040-8978/18/11/114008 [Accessed 24th October 2022]

[17] Xin, J., Xu, R., & Chen, Q. (2018). All-optical machine learning using diffractive deep neural networks. Science, 361(6406), 1004-1008. [Online]. Available: https://doi.org/10.1126/science.aat8084 [Accessed 24th October 2022]

[18] Colaço, A., Faria, D. R., & Lobo, J. (2018). The advantages of an RGB-D camera in single-pixel imaging. Sensors, 18(9), 2884. [Online]. Available: https://doi.org/10.3390/s18092884 [Accessed 24th October 2022]

[19] Sun, M., Edgar, M. P., Phillips, D. B., Gibson, G. M., & Padgett, M. J. (2015). Efficient single pixel imaging in the temporal and spectral domains. Optics Express, 23(20), 26608-26617. [Online]. Available: https://doi.org/10.1364/OE.23.026608 [Accessed 24th October 2022]

[20] Bioucas-Dias, J. M., & Figueiredo, M. A. (2007). A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image deblurring. IEEE Transactions on Image Processing, 16(12), 2992-3004. [Online]. Available: https://doi.org/10.1109/TIP.2007.909319 [Accessed 24th October 2022]

[21] Eldar, Y. C., & Kutyniok, G., eds. (2012). Compressed Sensing: Theory and Applications. Cambridge University Press. [Online]. Available: https://doi.org/10.1017/CBO9780511794308 [Accessed 24th October 2022]

[22] Renna, F., Silva, J., Carreira, M. J., & Rodrigues, M. R. (2012). Motion-adaptive spatio-temporal regularization for accelerated dynamic MRI. Magnetic Resonance Imaging, 30(10),

1409-1420. [Online]. Available: https://doi.org/10.1016/j.mri.2012.04.024 [Accessed 24th October 2022]

[23] Katz, O., Heidmann, P., Fink, M., & Gigan, S. (2014). Non-invasive single-shot imaging through scattering layers and around corners via speckle correlations. Nature Photonics, 8(10), 784-790. [Online]. Available: https://doi.org/10.1038/nphoton.2014.189 [Accessed 25th March 2023]

[24] Sun, B., Edgar, M. P., Phillips, D. B., Gibson, G. M., & Padgett, M. J. (2013). Improving the signal-to-noise ratio of single-pixel imaging using digital microscanning. Optics express, 21(10), 12539-12545. [Online]. Available: https://doi.org/10.1364/OE.21.012539 [Accessed 25th March 2023]

[25] Xin, H., Chen, J., & Wei, D. (2019). Deep learning-based single-pixel imaging via compressive sensing. Optics Communications, 448, 84-92. [Online]. Available: https://doi.org/10.1016/j.optcom.2019.05.020 [Accessed 25th March 2023]

[26] Huang, W., Liao, S., Li, Z., & Feng, Q. (2018). Single-pixel hyperspectral imaging with machine-learning classification for agriculture. Optics express, 26(12), 15399-15409. [Online]. Available: https://doi.org/10.1364/OE.26.015399 [Accessed 25th March 2023]

[27] Ruggiero, M., Locquet, A., & Citrin, D. S. (2020). Terahertz imaging with a single-pixel detector based on compressive sensing for art conservation. Optics express, 28(6), 8216-8226. [Online]. Available: https://doi.org/10.1364/OE.389106 [Accessed 25th March 2023]

[28] Yu, X., Chen, W., Zhang, F., & Li, H. (2018). High-speed compressive single-pixel imaging with multi-scale measurements. Optics Express, 26(4), 4492-4502. [Online]. Available: https://doi.org/10.1364/OE.26.004492 [Accessed 25th March 2023]

[29] Dong, Q., Heidarinejad, M., Zhu, Q., & Dahl Knudsen, E. (2018). DeepInverse: Deep learning for compressive imaging reconstruction via sparse prior and nonlinear mappings.

arXiv preprint arXiv:1803.04768. [Online]. Available: https://arxiv.org/abs/1803.04768 [Accessed 25th March 2023]

[30] Poudel, P., Horisaki, R., & Kawabe, T. (2017). Spectral imaging with compressive single-pixel detection. Optics express, 25(22), 26953-26959. [Online]. Available: https://doi.org/10.1364/OE.25.026953 [Accessed 25th March 2023]

[31] Arce, G. R., Brady, D. J., Carin, L., Arguello, H., & Kittle, D. S. (2014). Compressive coded aperture spectral imaging: An introduction. IEEE Signal Processing Magazine, 31(1), 105-115. [Online]. Available: https://doi.org/10.1109/MSP.2013.2278977 [Accessed 25th March 2023]

[32] Liu, H., Guo, S., Huang, H., Xiao, Z., & Cao, Y. (2018). Real-time infrared ultra-spectral signature classification via deep learning and compressed sensing. Optics express, 26(2), 1593-1603. [Online]. Available: https://doi.org/10.1364/OE.26.001593 [Accessed 25th March 2023]

[33] Zhang, Y., Li, C., Li, Y., & Xu, W. (2017). Single-pixel imaging based on compressive sensing and optical encryption. Journal of Optics, 19(4), 045702. [Online]. Available: https://doi.org/10.1088/2040-8986/aa5a0d [Accessed 25th March 2023]

[34] A. (2020, May 29). Solutions for Water Stress in Agriculture. Analytik Ltd. [Online] https://analytik.co.uk/water-stress-solutions-for-agriculture/ [Accessed 25th March 2023]

[35] Cosentino A. Terahertz and Cultural Heritage Science: Examination of Art and Archaeology. Technologies. 2016; 4(1):6. [Online] https://doi.org/10.3390/technologies4010006 [Accessed 25th March 2023]

[36] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4), 600-612. [Online] https://www.cns.nyu.edu/pub/lcv/wang03-preprint.pdf [Accessed 25th March 2023]

Appendix

Appendix A

Part1.m

```
% test_linear_rec.m
clear all;

% Sample pattern
NP = 40; % Number of pixels
image_pattern = create_sample_pattern(NP);
figure(1), subplot(6, 3, 1), imagesc(image_pattern), title('Original image(NM)')

% Simulate mask set data
num_masks = [100, 600, 1200, 1600, 1650];
for i = 1:length(num_masks)
    NM = num_masks(i);
    test(image_pattern, NM, NP, i);
end

% Functions
function [pattern] = create_sample_pattern(NP)
    pattern = zeros(NP);
    pattern(:, 1:3) = 1;
    pattern(:, 14:16) = 1;
    pattern(:, 24:26) = 1;
    pattern(:, 38:40) = 1;
    pattern(1:3, :) = 1;
    pattern(14:16, :) = 1;
    pattern(24:26, :) = 1;
    pattern(38:40, :) = 1;
end

function test(image_pattern, num_masks, num_pixels, idx)
    [mask_data, thz_data] = generate_data(image_pattern, num_masks, num_pixels);
    reconstructed_image = linear_rec(thz_data, mask_data);
    plot_results(image_pattern, idx, mask_data, thz_data, reconstructed_image);
end

function [mask_data, thz_data] = generate_data(image_pattern, num_masks, num_pixels)
    mask_data = randi([0, 1], num_masks, num_pixels * num_pixels);
    thz_data = mask_data * image_pattern(:);
end

function plot_results(image_pattern, idx, mask_data, thz_data, reconstructed_image)
    subplot(6, 3, 3 * idx + 1), imagesc(reshape(mask_data(end, :), size(image_pattern))), title('Mask pattern')
    subplot(6, 3, 3 * idx + 2), imagesc(image_pattern .* reshape(mask_data(end, :), size(image_pattern))), title('Combined')
    subplot(6, 3, 3 * idx + 3), imagesc(reconstructed_image), title(['Reconstruction(NM=', num2str(size(mask_data, 1)), ')']);
end

function [newimg] = linear_rec(THzData, MaskData)
    % LINEAR_REC Reconstructs an image from THz measurement data and a set of binary masks using linear reconstruction

    % Get dimensions of the data
    [NM, NP2] = size(MaskData);
    NP = sqrt(NP2);

    % Compute the pseudoinverse of MaskData
    MPinv = pinv(MaskData);

    % Reconstruct the image using linear reconstruction
    newimg = reshape(MPinv * THzData, NP, NP, []);
end
```

Appendix B

```
function [newimg] = linear_rec(THzData, MaskData)
% LINEAR_REC Reconstructs an image from THz measurement data and a set of binary masks using linear reconstruction

% Get dimensions of the data
[NM, NP2] = size(MaskData);
NP = sqrt(NP2);

% Check if the dimensions of the data are compatible
if length(THzData) ~= NM
    error('The length of THzData must equal the number of masks.');
end

% Compute the pseudoinverse of MaskData
MPinv = pinv(MaskData);

% Reconstruct the image using linear reconstruction
newimg = reshape(MPinv * THzData, NP, NP, []);
end
```

```
function test(ima, NM, NP, x)
% Generates random binary masks, applies them to the input image, and reconstructs the image using linear reconstruction

% Display the mask pattern
subplot(6, 3, 3*x+1), imga = imagesc(ima); title('mask pattern')

% Display the combined pattern
subplot(6, 3, 3*x+2), imgb = imagesc(ima); title('combined')

% Create the binary masks and apply them to the input image
MaskData = zeros(NM, NP*NP);
for i = 1:NM
    temp = rand(NP) > 0.5;
    MaskData(i, :) = temp(:);
    tempa = reshape(temp, NP, NP);
    set(imga, 'CData', tempa);
    tempb = tempa .* ima;
    set(imgb, 'CData', tempb);
end
```

```
o_ima=imread('5.jpg');

figure(1),subplot(2,3,1),imagesc(o_ima), title('original image')
% simulate mask set data

[a,b] = size(o_ima);
if a>b
   c=b;
else c=a;
end


H=im2double(o_ima);
ima=imresize(H,[c c]);
[m,n,l]=size(ima);



NP=c;
figure(1),subplot(2,4,1),imagesc(ima), title('original image')
% simulate mask set data
NM=200; % nunmber of masks
MaskData=zeros(NM,c*c);
subplot(2,4,2),im1=imagesc(ima);title('mask pattern')
subplot(2,4,3),im2=imagesc(ima);title('R combined')
subplot(2,4,4),im3=imagesc(ima);title('G combined')
subplot(2,4,5),im4=imagesc(ima);title('B combined')
```

```
for i=1:NM
    temp=rand(c); temp=temp>0.5;
    MaskData(i,:)= temp(:);
    temp1=reshape(temp,c,c);%mask pattern
    set(im1,'CData',temp1);
    temp2=temp.*double(ima(:,:,1));%combined patten
    set(im2,'CData',temp2);
    temp3=temp.*double(ima(:,:,2));
    set(im3,'CData',temp3);
    temp4=temp.*double(ima(:,:,3));
    set(im4,'CData',temp4);


end
    temp5(:,:,1)=temp2;
    temp5(:,:,2)=temp3;
    temp5(:,:,3)=temp4;
    temp6=double(temp5);
    subplot(2,4,6),imagesc(temp6),title('combined');

    temp7=ima(:,:,1);
    THzData=double(MaskData)*double(temp7(:));
    newimg(:,:,1)=linear_rec(THzData, MaskData);
    temp7=ima(:,:,2);
    THzData=double(MaskData)*double(temp7(:));
    newimg(:,:,2)=linear_rec(THzData, MaskData);
    temp7=ima(:,:,3);
    THzData=double(MaskData)*double(temp7(:));
    newimg(:,:,3)=linear_rec(THzData, MaskData);


figure(1);subplot(2,4,7),imshow(newimg,[]);% display the reconstructed image
title('Reconstruction');
```

Appendix C

```
function [Im]=ya(Img, f, g ,x)

a = f-mod(f,x);
b = g-mod(g,x);
H=im2double(Img);

Im=imresize(H,[a b]);
end
```

```
function simliarity( Im, new_IM )
%UNTITLED4 Summary of this function goes here
%   Detailed explanation goes here
I1=Im;
I2=new_IM;
ssimval = ssim(I1,I2);


    fprintf('The ssim value is %0.4f.\n',ssimval);

    title(sprintf('Reconstruction - simliarity is %0.4f',ssimval));

end
```

```
function test(ima, NM, NP, x ) %[ ima, imga, imgb, imgc ] =
subplot(6,3,3*x+1),imga=imagesc(ima);title('mask pattern')
subplot(6,3,3*x+2),imgb=imagesc(ima);title('combined')
a=3*x+1;
b=3*x+2;
c=3*x+3;
MaskData=zeros(NM,NP*NP);

for i=1:NM
    temp=rand(NP); temp=temp>0.5;
    MaskData(i,:)= temp(:);
    %pause(0.1)
    tempa=reshape(temp,NP,NP);%mask pattern
    set(imga,'CData',tempa);
    tempb=tempa.*ima; %combined patten
    set(imgb,'CData',tempb);
end
THzData=MaskData*ima(:); % simulate terahertz measurement data
% reconstruction
newimg=linear_rec(THzData, MaskData); % call the reconstruciton function
% reifne and plot
newimg=squeeze(newimg);
newimg=newimg.*(newimg>0);
subplot(6,3,3*x+3),imgc=imagesc(newimg);title(['Reconstruction(NM=' , num2str(NM), ')']);


end
```

```
function [ newimg ] = linear_reconstruction( o_ima, NM )
[a,b] = size( o_ima ); %get the size of the original image
if a>b
   c=b;
else c=a; % make the length and width equal
end

H = im2double(o_ima); %change the data type of the image
ima = imresize(H,[c c]); %change the size of the image

NP=c; %use the adjusted value
MaskData=zeros(NM,NP*NP); %generate the MaskData


for i=1:NM
    temp=rand(NP);
    temp=temp>0.5; %ensure the value exceeds 0.5
    MaskData(i,:)= temp(:);
end


    temp7=ima(:,:,1);     % decorate the dimensions
    THzData=double(MaskData)*double(temp7(:)); %obtain the R dimension THzData
    newimg(:,:,1)=linear_rec(THzData, MaskData);  %reconstruct the R dimension
    temp7=ima(:,:,2);
    THzData=double(MaskData)*double(temp7(:));  %obtain the G dimension THzData
    newimg(:,:,2)=linear_rec(THzData, MaskData); %reconstruct the G dimension
    temp7=ima(:,:,3);
    THzData=double(MaskData)*double(temp7(:));   %obtain the B dimension THzData
    newimg(:,:,3)=linear_rec(THzData, MaskData); %reconstruct the B dimension

end
```

```
function [newimg] = linear_rec(THzData, MaskData)
% LINEAR_REC Reconstructs an image from THz measurement data and a set of binary masks using linear reconstruction

% Get dimensions of the data
[NM, NP2] = size(MaskData);
NP = sqrt(NP2);

% Compute the pseudoinverse of MaskData
MPinv = pinv(MaskData);

% Reconstruct the image using linear reconstruction
newimg = reshape(MPinv * THzData, NP, NP, []);
end
```

```
function varargout = gui(varargin)

% Initialize GUI state
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @gui_OpeningFcn, ...
                   'gui_OutputFcn',  @gui_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);

% Assign callback function if input argument is a character string
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

% Return output arguments if requested
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% GUI opening function
function gui_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);

% GUI output function
function varargout = gui_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

% Callback for Read Image button
function read_image_Callback(hObject, eventdata, handles)
global im
[fn,pn,~]=uigetfile('*.jpg','Choose Image');
im = imread([pn fn]);
axes(handles.axes1);
imshow(im);title('Original Image');
handles.image = im;
guidata(hObject, handles);
```

```
% Callback for Process Image button
function process_image_Callback(hObject, eventdata, handles)
global im
X = str2double(get(handles.Input2,'String'));
[f,g] = size(im);
[Im]=ya(im, f, g, X);
NM = str2double(get(handles.Input1,'String'));
hold on
L = size(Im);
height = X;
width = X;
max_row = floor(L(1) / height);
max_col = floor(L(2) / width);
seg = cell(max_row,max_col);
m_image = cell(max_row,max_col);

% Split input image into segments
for row = 1:max_row
    for col = 1:max_col
        seg(row,col)= { Im( (row-1)*height+1 : row*height, (col-1)*width+1 : col*width, : ) };
    end
end

% Apply linear reconstruction on each segment
for i=1:max_row*max_col
    [m_image{i}] = linear_reconstruction(seg{i}, NM);
end

% Combine reconstructed segments into a new image
new_image = zeros(f, g, 3);
for row = 1:max_row
    for col = 1:max_col
        new_image( (row-1)*height+1 : row*height, (col-1)*width+1 : col*width, : ) = m_image{row,col};
    end
end

hold off
```

```
% Resize and display the new image
H=im2double(new_image);
new_IM=imresize(H,[f g/3]);
M_H = im2double(im);
im = imresize(M_H,[f,g/3]);
global similarity
similarity = ssim(im, new_IM);
axes(handles.axes2);
imshow(new_IM);title('Reconstructed Image');
set(handles.edit_simi,'string',num2str(similarity));
guidata(hObject, handles);

% Callback for Input1 edit text field
function Input1_Callback(hObject, eventdata, handles)
input = get(handles.Input1,'String'); % Get the input from the edit text field
input = str2double(input); % Convert the input string to a number
guidata(hObject, handles);

% Input1 edit text field object creation function
function Input1_CreateFcn(hObject, eventdata, handles)
% Set the background color of Input1 edit text field to white if the platform is a PC
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% Callback for Input2 edit text field
function Input2_Callback(hObject, eventdata, handles)
input = get(handles.Input2,'String'); % Get the input from the edit text field
input = str2double(input); % Convert the input string to a number
guidata(hObject, handles);

% Input2 edit text field object creation function
function Input2_CreateFcn(hObject, eventdata, handles)
% Set the background color of Input2 edit text field to white if the platform is a PC
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% Callback for edit_simi edit text field
function edit_simi_Callback(hObject, eventdata, handles)
guidata(hObject, handles);

% edit_simi edit text field object creation function
function edit_simi_CreateFcn(hObject, eventdata, handles)
% Set the background color of edit_simi edit text field to white if the platform is a PC
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

guidata(hObject, handles);
```

Appendix D

```
// Arduino SPC

#include "Adafruit_TCS34725.h"

// Setting up pins for motor stepper motors
const int yMotorPins[] = {11, 10, 9, 8};
const int xMotorPins[] = {7, 6, 5, 4};
const int photoPin = 3;

// Stepper motor rotation sequence
const int Seq[][4] = {{1, 0, 0, 0},
                      {1, 1, 0, 0},
                      {0, 1, 0, 0},
                      {0, 1, 1, 0},
                      {0, 0, 1, 0},
                      {0, 0, 1, 1},
                      {0, 0, 0, 1},
                      {1, 0, 0, 1}};
int nSteps[] = {0, 0};

// Setting up viewing angle and grid size
const float viewingAngle = 20.0;
const float photoSize = 40.0;
const float gridSize = tan((viewingAngle/360.0)*2*PI)/photoSize;
const float radPerStep = (2.0 * PI) / 4076.0;
float currentTheta = 0.0001;
float currentPhi = 0.0001;
float currentX = 0.0;
float currentY = 0.0;
float currentDX = sin(radPerStep) / cos(currentTheta);
float currentDY = sin(radPerStep) / cos(currentPhi);
float startPosition = 0.0;
float nextX;
float nextY;
int gridX = 0;
int gridY = 0;
int seqStepY = 0;
int seqStepX = 0;

// Color sensor Adafruit_TCS34725
Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_154MS, TCS34725_GAIN_60X);

void setup() {
  Serial.begin(9600);
  for (int nPin = 0; nPin < 4; nPin++) {
    pinMode(yMotorPins[nPin], OUTPUT);
    pinMode(xMotorPins[nPin], OUTPUT);
  }
}
```

```
void loop() {
  // Wait 5 seconds before starting
  delay(5000);
  takePhoto();
  returnHome();
  while (true) {
    delay(1000);
  }
}

int takePhoto() {
  float moveN = 1;

  // Spiral from center outwards
  for (int nRowsCols = 0; nRowsCols < photoSize / 2.0; nRowsCols++) {
    for (int x = 0; x < moveN; x++) {
      nextX = currentX + gridSize;
      gridX++;
      moveToNew();
    }
    for (int x = 0; x < moveN; x++) {
      nextY = currentY + gridSize;
      gridY++;
      moveToNew();
    }
    moveN++;
    for (int x = 0; x < moveN; x++) {
      nextX = currentX - gridSize;
      gridX--;
      moveToNew();
    }
    for (int x = 0; x < moveN; x++) {
      nextY = currentY - gridSize;
      gridY--;
      moveToNew();
    }
    moveN++;
  }
}
```

```
90
91  void moveToNew() {
92    while (abs(currentX - nextX) > currentDX || abs(currentY - nextY) > currentDY) {
93      currentDX = sin(radPerStep) / cos(currentTheta);
94      currentDY = sin(radPerStep) / cos(currentPhi);
95      currentX = tan(currentTheta);
96      currentY = tan(currentPhi) / cos(currentTheta);
97
98      if (abs(currentX - nextX) > currentDX) {
99        if (currentX < nextX) {
100         takeStep(1, "left");
101         currentTheta += radPerStep;
102       }
103       if (currentX > nextX) {
104         takeStep(1, "right");
105         currentTheta -= radPerStep;
106       }
107     }
108     if (abs(currentY - nextY) > currentDY) {
109       if (currentY < nextY) {
110         takeStep(2, "left");
111         currentPhi += radPerStep;
112       }
113       if (currentY > nextY) {
114         takeStep(2, "right");
115         currentPhi -= radPerStep;
116       }
117     }
118   }
119
120   // Log the data for the "pixel" to be used in image
121   // Format: x,y,r,g,b
122   Serial.print(gridX);
123   Serial.print(",");
124   Serial.print(gridY);
125   Serial.print(",");
126   uint16_t clr, red, green, blue;
127   tcs.getRawData(&red, &green, &blue, &clr);
128   Serial.print(red);
129   Serial.print(",");
130   Serial.print(green);
131   Serial.print(",");
132   Serial.println(blue);
133 }
```

```
134
135   // This function takes in a stepper motor number (1 or 2) and a direction (left or right)
136 void takeStep(int motor, const char* dir) {
137   if (motor == 1) {
138     for (int nPin = 0; nPin < 4; nPin++) {
139       int xPin = xMotorPins[nPin];
140       digitalWrite(xPin, Seq[seqStepX][nPin] ? HIGH : LOW);
141     }
142   } else if (motor == 2) {
143     for (int nPin = 0; nPin < 4; nPin++) {
144       int yPin = yMotorPins[nPin];
145       digitalWrite(yPin, Seq[seqStepY][nPin] ? HIGH : LOW);
146     }
147   }
148   delay(2);
149
150   if (strcmp(dir, "left") == 0) {
151     if (motor == 2) {
152       nSteps[0]++;
153       seqStepY = (seqStepY == 7) ? 0 : seqStepY + 1;
154     }
155     if (motor == 1) {
156       nSteps[1]++;
157       seqStepX = (seqStepX == 7) ? 0 : seqStepX + 1;
158     }
159   }
160   if (strcmp(dir, "right") == 0) {
161     if (motor == 2) {
162       nSteps[0]--;
163       seqStepY = (seqStepY == 0) ? 7 : seqStepY - 1;
164     }
165     if (motor == 1) {
166       nSteps[1]--;
167       seqStepX = (seqStepX == 0) ? 7 : seqStepX - 1;
168     }
169   }
170 }
171
172 void returnHome() {
173   while (abs(nSteps[0]) > 1 || abs(nSteps[1]) > 1) {
174     if (nSteps[0] > 1) { takeStep(2, "right"); }
175     if (nSteps[0] < 1) { takeStep(2, "left"); }
176     if (nSteps[1] > 1) { takeStep(1, "right"); }
177     if (nSteps[1] < 1) { takeStep(1, "left"); }
178   }
179 }
180
181 // This Arduino code controls a camera system that captures an image by moving a color sensor in a grid pattern to sample the colors of an object.
182   // The program uses Adafruit's TCS34725 color sensor library and controls two stepper motors to move the sensor in the X and Y directions.
183   // The code is organized into multiple functions, including setup and loop, which are responsible for initializing the hardware and controlling the main program flow.
184   // The takePhoto function handles moving the sensor in a spiral pattern, and moveToNew function calculates the necessary steps to reach the next position on the grid.
185   // The takeStep function controls the stepper motors, and the returnHome function moves the sensor back to the starting position.
186   // Throughout the process, the color sensor captures RGB values of each position, and the data is logged to the serial monitor to be used for image reconstruction.
```

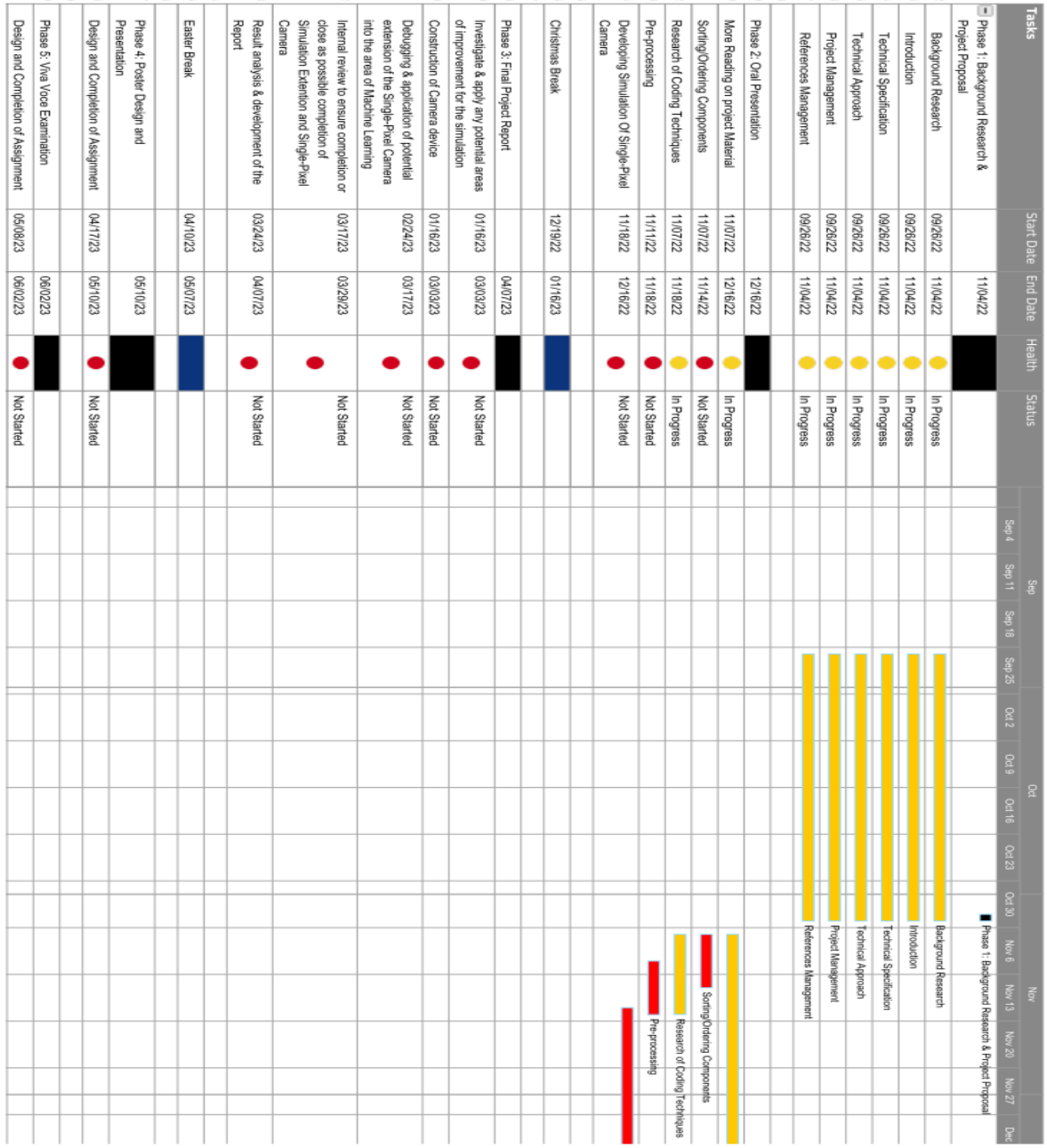
Appendix E

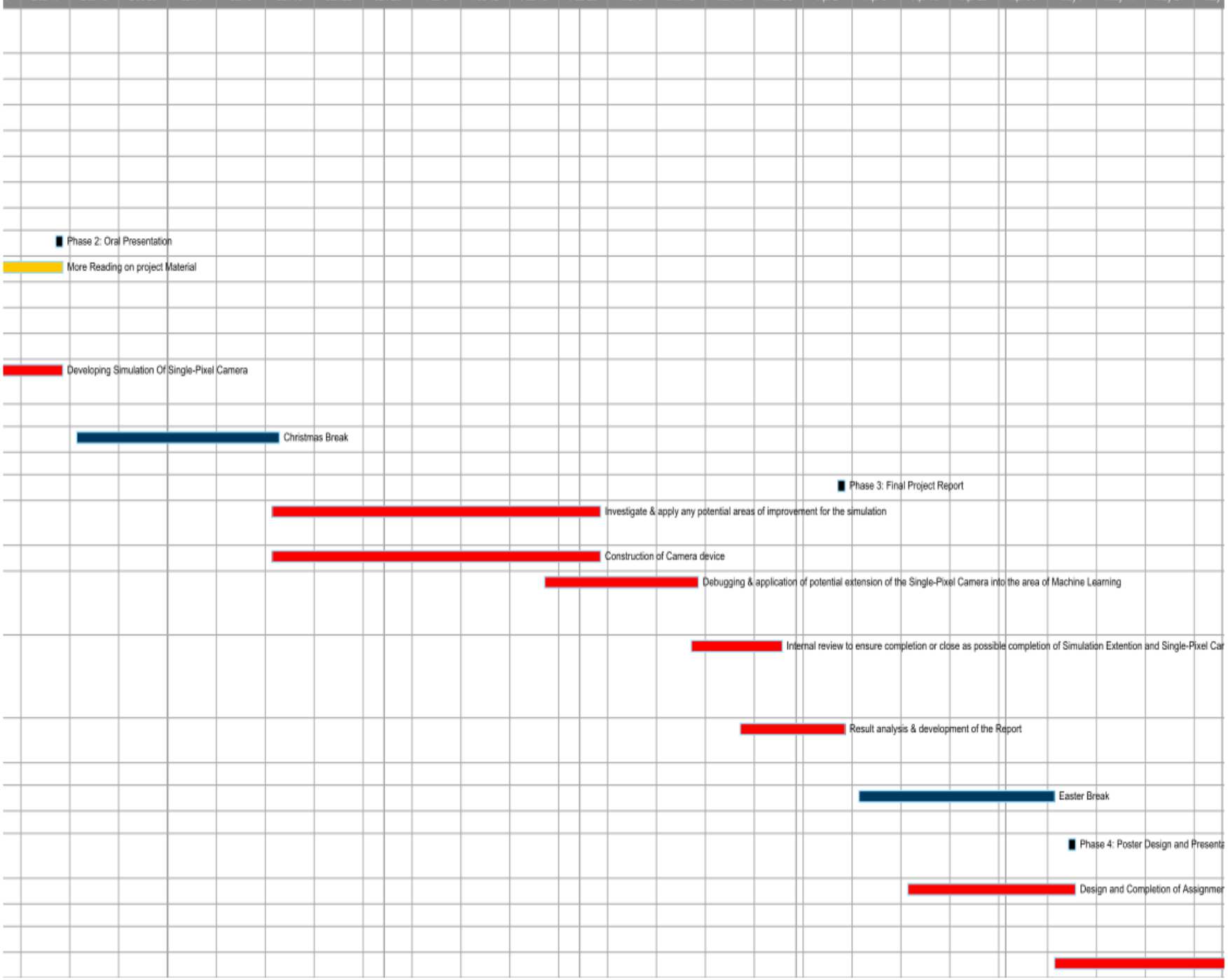
```
// Single RGB Color Sensor Plotter

import controlP5.*;
import processing.serial.*;

ControlP5 cp5;
int rBoost, gBoost, bBoost;
Serial myPort;
String val;
float gridX, gridY;
int photoSize = 900;
int origin = photoSize / 2;
int gridSize = photoSize / 100;
int xPos = origin, yPos = origin;
int red, green, blue;
int maxSeen = 1;
float multiplier = 1.0;
ArrayList<Integer> reds = new ArrayList<Integer>();
ArrayList<Integer> greens = new ArrayList<Integer>();
ArrayList<Integer> blues = new ArrayList<Integer>();
ArrayList<Integer> xPositions = new ArrayList<Integer>();
ArrayList<Integer> yPositions = new ArrayList<Integer>();

void setup() {
  size(900, 900);
  background(0);
  String portName = Serial.list()[0];
  myPort = new Serial(this, portName, 9600);
  myPort.bufferUntil('\n');

  cp5 = new ControlP5(this);
  cp5.addSlider("rBoost")
     .setPosition(10, 650)
     .setRange(0, 200)
     .setValue(100);
  cp5.addSlider("gBoost")
     .setPosition(10, 670)
     .setRange(0, 200)
     .setValue(100);
  cp5.addSlider("bBoost")
     .setPosition(10, 690)
     .setRange(0, 200)
     .setValue(100);
}
```

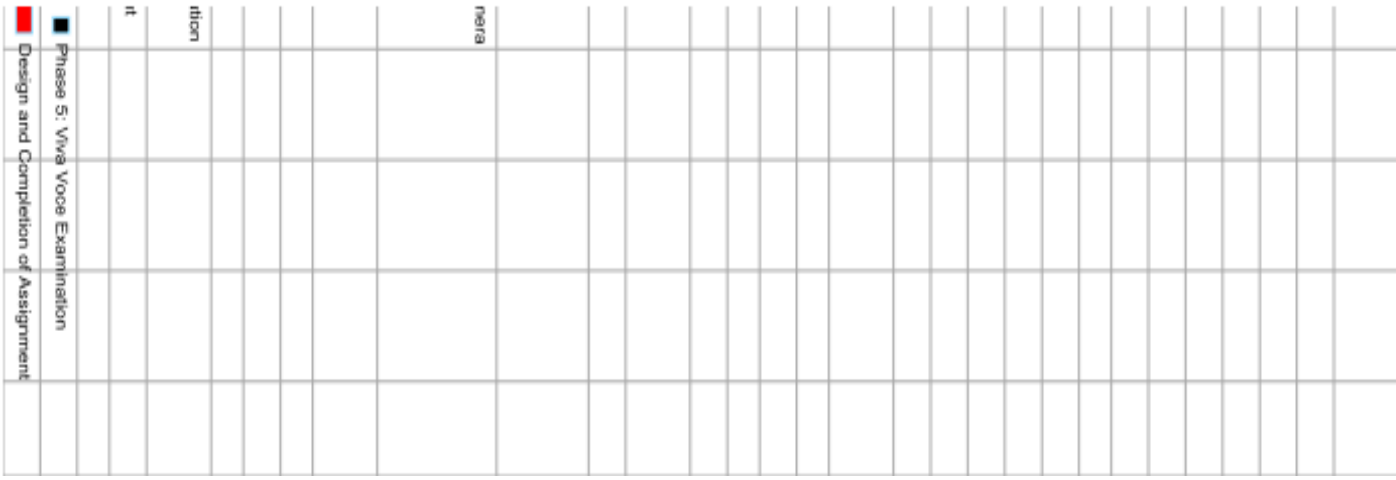
```
void draw() {
  val = myPort.readStringUntil('\n');
  if (val != null) {
    String[] list = split(val, ',');

    red = Integer.parseInt(list[2]);
    green = Integer.parseInt(list[3]);
    blue = Integer.parseInt(list[4].trim());

    maxSeen = max(max(red, green), blue, maxSeen);

    reds.add(red);
    greens.add(green);
    blues.add(blue);
    xPositions.add(xPos);
    yPositions.add(yPos);

    multiplier = 255.0 / maxSeen;

    println("scaling:", multiplier * maxSeen);
    println(red, " ", blue, " ", green);

    colorMode(RGB, maxSeen);
    for (int p = 0; p < xPositions.size(); p++) {
      fill(reds.get(p) * rBoost / 100.0, greens.get(p) * gBoost / 100.0, blues.get(p) * bBoost / 100.0);
      stroke(reds.get(p) * rBoost / 100.0, greens.get(p) * gBoost / 100.0, blues.get(p) * bBoost / 100.0);
      rect(xPositions.get(p), yPositions.get(p), gridSize, gridSize);
    }

    gridX = Integer.parseInt(list[0]);
    gridY = Integer.parseInt(list[1]);
    xPos = origin + (int)(gridX * gridSize);
    yPos = origin - (int)(gridY * gridSize);
  }
}

// This code is a Processing sketch that visualizes color data from an RGB color sensor.
// It receives color data from a serial port, typically connected to an Arduino or another microcontroller, and displays the data as colored squares on a grid.
// The sketch also provides sliders for adjusting the red, green, and blue color components individually.

// Here's a step-by-step explanation of what the code does:

// 1. Import necessary libraries: ControlP5 for creating user interface elements (sliders in this case) and the Serial library for receiving data from a microcontroller.
// 2. Declare global variables for storing color component values, grid positions, maximum seen value, and other related data. ArrayLists are used for storing the history of color and position values.
// 3. In the setup() function, set up the Processing sketch window, create a serial connection to receive data from a microcontroller, and configure ControlP5 sliders for adjusting the red, green, and blue color components.
// 4. In the draw() function, read incoming data from the serial connection. If the data is valid, parse the data to extract the color components and grid position values.
// 5. Update the maximum seen color value and append the received data to the ArrayLists.
// 6. Calculate a scaling factor based on the maximum seen color value, which will be used to normalize the color components' values.
// 7. Set the color mode to use the maximum seen value as the maximum value for each color component.
// 8. Iterate through the stored history of color and position values, and draw colored squares on the grid using the scaled color values.
// 9. Update the current grid position based on the received data.

// The result is a grid of colored squares that represent the color data received from the serial connection, allowing for a visual representation of the RGB color sensor's readings.
// The sliders enable users to boost or reduce the influence of individual color components in the visualization.
```

Appendix F

Gantt Chart




Phase 2: Oral Presentation
More Reading on project Material
Developing Simulation Of Single-Pixel Camera
Christmas Break
Phase 3: Final Project Report
Investigate & apply any potential areas of improvement for the simulation
Construction of Camera device
Debugging & application of potential extension of the Single-Pixel Camera into the area of Machine Learning
Internal review to ensure completion or close as possible completion of Simulation Extention and Single-Pixel Cam
Result analysis & development of the Report
Easter Break
Phase 4: Poster Design and Presenta
Design and Completion of Assignmen



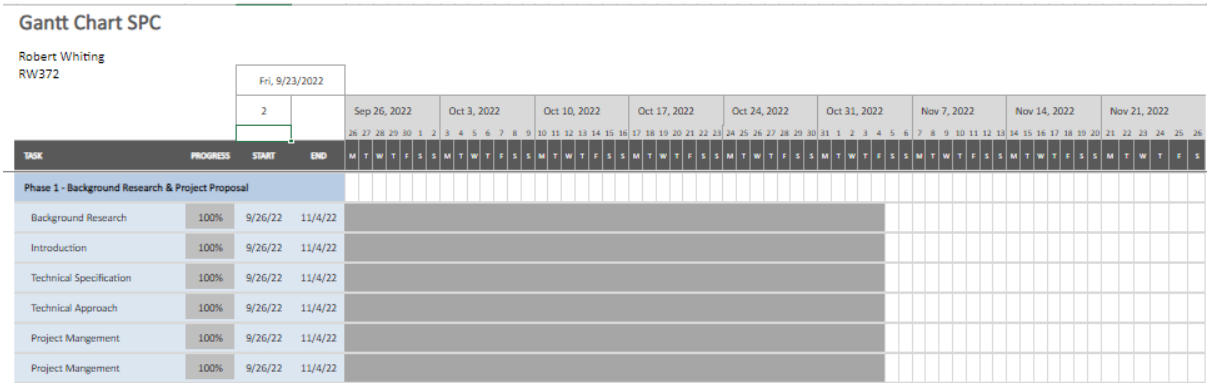
Tasks	Start Date	End Date	Health	Status
			Legend	
				Not Started
				In Progress
				Complete
				Holidays
				Deadline

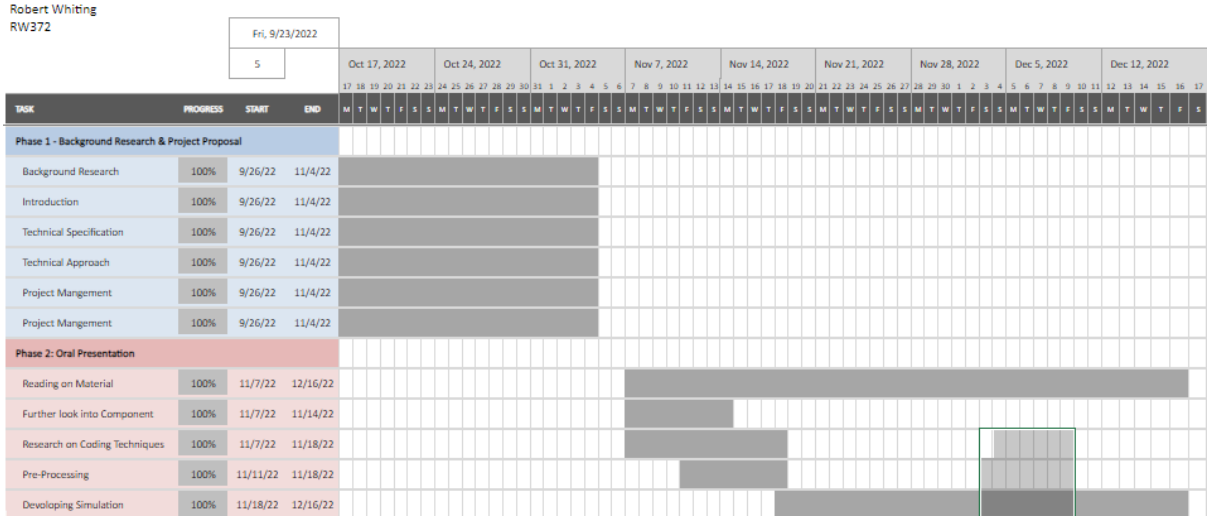
Sep | Sep 4 | Sep 11 | Sep 18 | Sep 25 | Oct | Oct 2 | Oct 9 | Oct 16 | Oct 23 | Oct 30 | Nov | Nov 6 | Nov 13 | Nov 20 | Nov 27

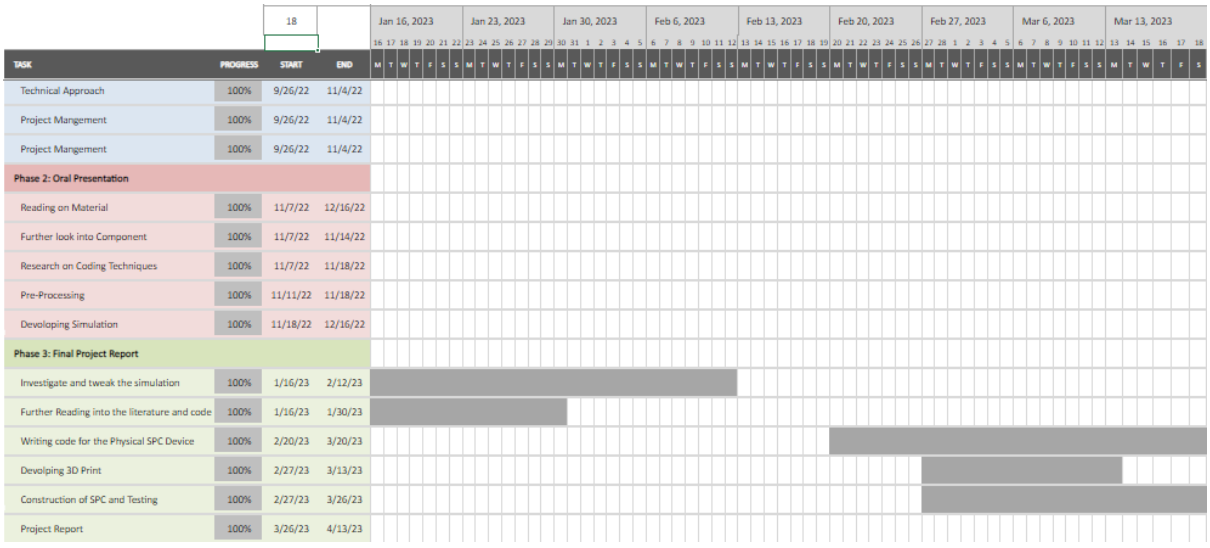
Dec | Dec 11 | Dec 18 | Dec 25 | Jan | Jan 1 | Jan 8 | Jan 15 | Jan 22 | Jan 29 | Feb | Feb 5 | Feb 12 | Feb 19 | Feb 26 | Mar | Mar 5 | Mar 12 | Mar 19 | Mar 26 | Apr | Apr 2 | Apr 9 | Apr 16 | Apr 23 | May | Apr 30 | May 7 | May 14 | May 21

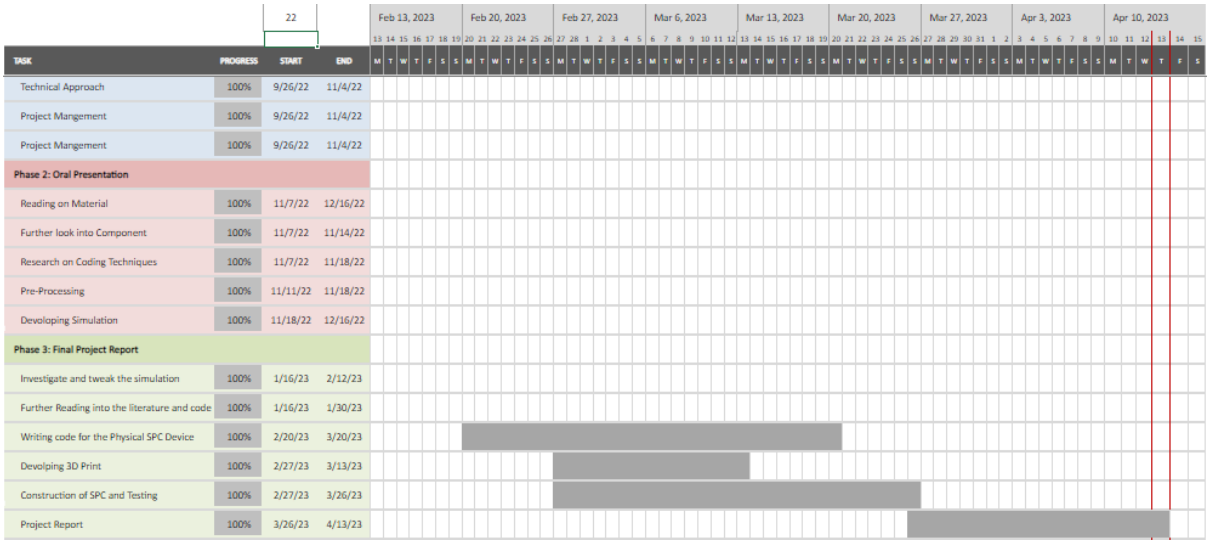
Jun | Jun 4 | Jun 11 | Jun 18 | Jun 25

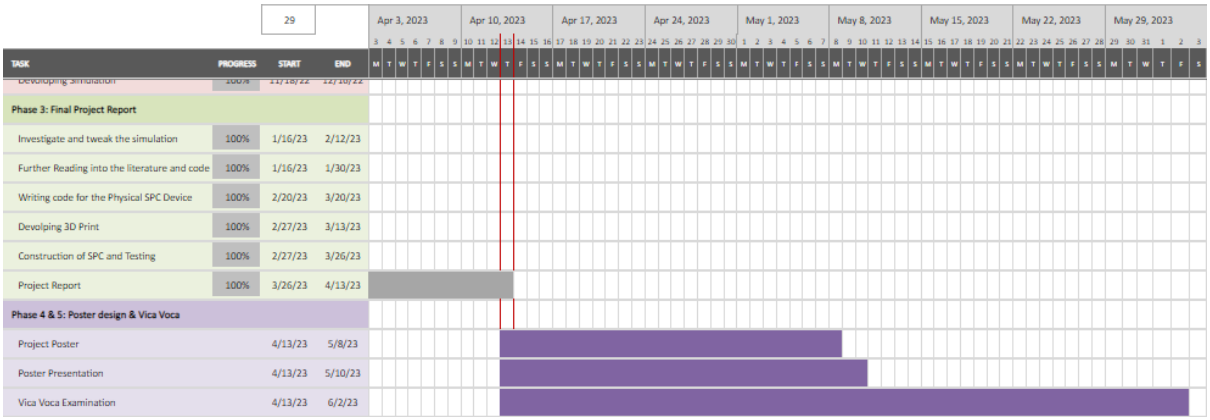
Appendix G


Gantt Chart SPC
Robert Whiting
RW372
Fri, 9/23/2022
2
Sep 26, 2022
Oct 3, 2022
Oct 10, 2022
Oct 17, 2022
Oct 24, 2022
Oct 31, 2022
Nov 7, 2022
Nov 14, 2022
Nov 21, 2022
TASK
PROGRESS
START
END
Phase 1 - Background Research & Project Proposal
Background Research 100% 9/26/22 11/4/22
Introduction 100% 9/26/22 11/4/22
Technical Specification 100% 9/26/22 11/4/22
Technical Approach 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22


Robert Whiting
RW372
Fri, 9/23/2022
5
Oct 17, 2022
Oct 24, 2022
Oct 31, 2022
Nov 7, 2022
Nov 14, 2022
Nov 21, 2022
Nov 28, 2022
Dec 5, 2022
Dec 12, 2022
TASK
PROGRESS
START
END
Phase 1 - Background Research & Project Proposal
Background Research 100% 9/26/22 11/4/22
Introduction 100% 9/26/22 11/4/22
Technical Specification 100% 9/26/22 11/4/22
Technical Approach 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Phase 2: Oral Presentation
Reading on Material 100% 11/7/22 12/16/22
Further look into Component 100% 11/7/22 11/14/22
Research on Coding Techniques 100% 11/7/22 11/18/22
Pre-Processing 100% 11/11/22 11/18/22
Devoloping Simulation 100% 11/18/22 12/16/22


18
Jan 16, 2023
Jan 23, 2023
Jan 30, 2023
Feb 6, 2023
Feb 13, 2023
Feb 20, 2023
Feb 27, 2023
Mar 6, 2023
Mar 13, 2023
TASK
PROGRESS
START
END
Technical Approach 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Phase 2: Oral Presentation
Reading on Material 100% 11/7/22 12/16/22
Further look into Component 100% 11/7/22 11/14/22
Research on Coding Techniques 100% 11/7/22 11/18/22
Pre-Processing 100% 11/11/22 11/18/22
Devoloping Simulation 100% 11/18/22 12/16/22
Phase 3: Final Project Report
Investigate and tweak the simulation 100% 1/16/23 2/12/23
Further Reading into the literature and code 100% 1/16/23 1/30/23
Writing code for the Physical SPC Device 100% 2/20/23 3/20/23
Devolping 3D Print 100% 2/27/23 3/13/23
Construction of SPC and Testing 100% 2/27/23 3/26/23
Project Report 100% 3/26/23 4/13/23


22
Feb 13, 2023
Feb 20, 2023
Feb 27, 2023
Mar 6, 2023
Mar 13, 2023
Mar 20, 2023
Mar 27, 2023
Apr 3, 2023
Apr 10, 2023
TASK
PROGRESS
START
END
Technical Approach 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Project Mangement 100% 9/26/22 11/4/22
Phase 2: Oral Presentation
Reading on Material 100% 11/7/22 12/16/22
Further look into Component 100% 11/7/22 11/14/22
Research on Coding Techniques 100% 11/7/22 11/18/22
Pre-Processing 100% 11/11/22 11/18/22
Devoloping Simulation 100% 11/18/22 12/16/22
Phase 3: Final Project Report
Investigate and tweak the simulation 100% 1/16/23 2/12/23
Further Reading into the literature and code 100% 1/16/23 1/30/23
Writing code for the Physical SPC Device 100% 2/20/23 3/20/23
Devolping 3D Print 100% 2/27/23 3/13/23
Construction of SPC and Testing 100% 2/27/23 3/26/23
Project Report 100% 3/26/23 4/13/23


29
Apr 3, 2023
Apr 10, 2023
Apr 17, 2023
Apr 24, 2023
May 1, 2023
May 8, 2023
May 15, 2023
May 22, 2023
May 29, 2023
TASK
PROGRESS
START
END
Phase 3: Final Project Report
Investigate and tweak the simulation 100% 1/16/23 2/12/23
Further Reading into the literature and code 100% 1/16/23 1/30/23
Writing code for the Physical SPC Device 100% 2/20/23 3/20/23
Devolping 3D Print 100% 2/27/23 3/13/23
Construction of SPC and Testing 100% 2/27/23 3/26/23
Project Report 100% 3/26/23 4/13/23
Phase 4 & 5: Poster design & Vica Voca
Project Poster 4/13/23 5/8/23
Poster Presentation 4/13/23 5/10/23
Vica Voca Examination 4/13/23 6/2/23

Appendix H

Risk Assessment

University of Kent | Division of Computing, Engineering and Mathematical Sciences

RISK ASSESSMENT FORM

EENG6000 Project - Single-Pixel Camera Risk Assessment

Overview of Activity: 3rd Year Project - To build a prototype signal pixel camera using an encoder (digital micro-mirror device, DMD) and a PIN diode detector. Work is to be completed both at home and in both the Engineering and Computer labs.

Name: Robert Whiting (rw372)

Supervisor\Line Manager: Doctor C Wang

Start date of activity: 01/11/2022

Location of your activity: Engineering Lab; Engineering WorkShop; JCS123

General activity type: Project (Student)

Pending approval

Activity Description

Assembly of electronic circuits & Use of Equipment
Covid-19 - Covid-19 associated risks
Data Loss
Prolonged work at a computer station - For example: Software & CAD design
Soldering (Electronics)
Tripping/falling
Workshop Equipment & Tools
Additional - To build a prototype signal pixel camera using an encoder (digital micro-mirror device, DMD) and a PIN diode detector. This work will involve the use of areas and equipment provided my the Engineering department located in the Jennison Building on Kent Campus

Activity: Assembly of electronic circuits & Use of Equipment								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		
Cuts and Abrasions from using hand tools	X				X		Myself	Awareness of using tools & equipment correctly via induction training and support from EDA staff.
Flying debris / trimming component leads	X				X		Everyone	PPE (protective eye glasses) are provided
Flying debris / trimming component leads	X				X		Everyone	Safe working: make sure bystanders aren't affected, aim to the floor or away from people when trimming components, cut one leg at the time.
Hygiene / Dirt	X				X		Everyone	PPE (gloves) are provided.
Hygiene / Dirt	X				X		Everyone	Wash hands immediately after activity.
Hygiene / Dirt	X			X			Everyone	Clean work area once finished - Use dust pans provided
Electric Shock / Malfunction	X				X		Myself	Equipment must be PAT tested

Activity: Covid-19								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		
Contact with shared surfaces with risk of contamination, i.e. doors.	X				X		Everyone	Sanitising wipes or spray should be used to clean down surfaces after use. Use hand washing facilities if nearby. In the case of doors, Where possible leave doors open to reduce hand contact. This must not apply to any designated fire door unless it is automated.
Contact with equipment with risk of contamination (e.g. electronic components, chargers, tools)	X				X		Everyone	With equipment, avoid sharing if possible. If not possible and there are hand washing facilities nearby, staff should use these to wash their hands before and after using the equipment. If there are no nearby hand washing facilities, then either sanitising wipes with which to clean the equipment or hand sanitiser should be provided.
Poor ventilation with risk of contamination with aerosol particles		X			X		Everyone	Where possible, room door will remain open for the duration of stay of the occupants. Windows must be opened if safe and only form of ventilation in multi-occupancy. Where possible doors can be kept open. Any poor ventilation rooms where this can not occur must have occupancy reduced.
Activity which requires collaboration between occupants			X		X		Everyone	Activity should only occur if essential.
Contact with shared surfaces with risk of contamination, i.e. kitchen surfaces	X				X		Everyone	Areas must be kept tidy with surfaces kept clear as much as possible. Where items used by individuals must be cleaned and removed immediately.
Contact with items with risk of contamination: passing person-to-person / deliveries	X				X		Everyone	Where items need to be passed between staff, they should not be handed person to person, but left at a designated hand over point. Similarly, deliveries should not be handed person to person, but instead be left in designated secure areas for pick up.
Contact with others: Work Meetings			X		X		Myself & Supervisor	Hybrid Working will necessitate that most formal work meetings should preferably continue to be held by MS Teams. This is also a useful COVID control.
Uni Kent - COVID-19 risk assessment (general)	X			X			Everyone	tinyurl.com/4rzrdefb . Please refer to this to identify additional activities that you may wish to add to your specific situation.
Contact with others: working in shared occupancy locations			X		X		Everyone	Consider staggered working and if possible that people are not directly across. Face coverings are strongly encouraged.

Activity: Data Loss								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		
Data loss / corruption due to different causes		X			X		Myself	Backup the work in as many ways as possible (USB drives, external drives, CD/DVD, cloud, etc.)

Activity: Prolonged work at a computer station								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		

Illness/discomfort/injury caused by prolonged work at a computer station caused by prolonged work at a computer station.	X				X		Myself	Awareness of DSE via induction training.
Illness/discomfort/injury caused by prolonged work at a computer stationcaused by prolonged work at a computer station.	X				X		Myself	Follow the recommendations in HSE DSE guide.
Illness/discomfort/injury caused by prolonged work at a computer station e.g. Eye strains, Carpal tunnel syndrome, Back pain, Blood clots in leg, Headaches.		X			X		Myself	Take breaks often away from the PC. Keep yourself hydrated

Activity: Soldering (Electronics)								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		
Inhalation of solder/flux fumes	X				X		Myself	Fume extraction is in place and in working order on the soldering iron tips.
Inhalation of solder/flux fumes	X				X		Myself	PPE (Face masks) are provided.
Burns to skin	X				X		Myself	Safe use of soldering iron via induction training.
Absorption of lead/flux residue through skin	X				X		Myself	PPE (gloves) are provided.
Absorption of lead/flux residue through skin	X				X		Myself	Wash hands immediately after activity.
Burns to skin	X				X		Myself	Don't touch the board, leave it cool down

Activity: Tripping/falling								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		
Tripping/falling		X			X		Everyone	Keep personal belongings under the desk/workbench
Tripping/falling		X			X		Everyone	Keep chairs/stools tucked in under the desk/workbench
Tripping/falling		X			X		Everyone	Running is not permitted in labs/workshops
Tripping/falling		X			X		Everyone	Secure cables on floors by using cord covers
Tripping/falling	X				X		Everyone	Keep gangways / pathways clear of obstacles. Be aware of the surroundings
Activity: Workshop Equipment & Tools								
Hazard	**Likelihood** (Chance of something happening)			**Consequence** (Impact of event happening)			**Who the hazard affects**	**Reduction of risk**
	Unlikely	Likely	Definite	Low	Moderate	High		
Pillar Drills	X				X		Myself	Induction training on how to use the equipment
Pillar Drills	X				X		Myself	Long hair and loose clothing to be tied back
Pillar Drills	X				X		Myself	Use PPE: heavy duty gloves and goggles
Pillar Drills	X				X		Myself	Clamp item being drilled to avoid it spinning
Pillar Drills	X				X		Myself	Ensure drill guard is in place and is used
Hand Tools, fabrication	X				X		Myself	Induction training on how to use the equipment
Hand Tools, fabrication	X				X		Myself	Use PPE: heavy duty gloves and goggles
Laser cutter	X				X		Myself	Only trained Technicians, staff or approved competent person may use the laser cutter.
Laser cutter	X				X		Myself	Ensure water cooler, laser head air feed and fume extraction is turned on
Laser cutter	X				X		Myself	Induction training on how to use the equipment
Laser cutter	X				X		Myself	Laser cutter is not suitable for these materials: MDF PVC, Polycarbonate/Lexan, ABS, HDPE, Polystyrene Foam, Polypropylene Foam, Epoxy, Fiberglass, Coated Carbon Fibre, Food
Mills, lathes	X				X		Myself	Only trained Technicians, staff or approved competent person may use the equipment
Mills, lathes	X				X		Myself	Induction training on how to use the equipment
Mills, lathes	X				X		Myself	Long hair and loose clothing to be tied back
Welding, brazing	X				X		Myself	Only trained Technicians, staff or approved competent person may use the equipment
Welding, brazing	X				X		Myself	Induction training on how to use the equipment