



**Stellenbosch**

UNIVERSITY  
IYUNIVESITHI  
UNIVERSITEIT

---

forward together  
sonke siya phambili  
saam vorentoe

# **Development of a Stereoscopic Camera System for 3D Deformation Analysis**

Mechatronic Project 478  
Final Report

F du Plessis  
25866095

Supervisor: Prof RP Theart

October 2025

Department of Mechanical and Mechatronic Engineering  
Stellenbosch University  
Tel: +27 21 808 4204 | [www.eng.sun.ac.za](http://www.eng.sun.ac.za)  
Private Bag X1, Matieland, 7602, South Africa

Copyright © 2025 Stellenbosch University  
All rights reserved

# Executive Summary

<b>Title of Project</b>
Development of a stereoscopic camera system for 3D deformation analysis.
<b>Objectives</b>
<p>Objectives will focus on delivering a cost-effective final camera system that can be used for further research and development into the field of 3D digital image correlation (DIC):</p> <ol style="list-style-type: none"><li>1. To design and construct a cost-effective stereoscopic camera system suitable for researching and developing an in-house 3D DIC algorithm.</li><li>2. To develop a calibration procedure that is robust, user-friendly, and time-efficient, thus aiding in the overall experience when using the camera system.</li><li>3. To create an intuitive user interface that facilitates access to and execution of all relevant camera system functions.</li></ol>
<b>What is current practice and what are its limitations?</b>
The current systems available for conducting 3D DIC are from various corporate companies and are expensive. The systems are also closed-source in nature and do not provide the user with the ability to modify the software to cater to their specific needs. The software used has a poor user interface and a steep learning curve. This makes it overly complicated for most tasks that these systems encounter.
<b>What is new in this project?</b>
The project aims to develop a fully functional, open-source stereoscopic camera system optimised for Digital Image Correlation (DIC). The project offers a practical and scalable platform for experimentation, further research, and development. The project will feature stereo camera calibration, with an additional focus on using only one photo pair to complete the calibration procedure.
<b>If the project is successful, how will it make a difference?</b>
The system will provide the Mechanical and Mechatronic Department of Stellenbosch University with a portable and cost-effective stereoscopic camera system. The implementation of a user-friendly user interface will mitigate the steep learning curve, thereby ensuring that users can start capturing samples for material analysis quickly and easily. The camera system will allow the Mechanical and Mechatronic Department to develop and test their own fully open-source 3D DIC setup.

<b>What are the risks to the project being a success? Why is it expected to be successful?</b>
The risks are a limited budget and time frame. The project's complexity is high and is associated with restricted access to the workshop of the Mechanical and Mechatronic Engineering Department. The project is expected to be successful due to the implementation of rigorous cost and time management. Additionally, careful scheduling of manufacturing will significantly contribute to the project's success.
<b>What contributions have/will other students made/make?</b>
N/A
<b>Which aspects of the project will carry on after completion and why?</b>
The project establishes a robust foundation for the Mechanical and Mechatronic Department to develop an in-house 3D DIC algorithm. It aims to deliver a fully functional, cost-effective, open-source 3D DIC system, designed to be accessible and adaptable for research and educational purposes. The calibration procedure developed can be applied to any other stereoscopic camera systems used at Stellenbosch University.
<b>What arrangements have been/will be made to expedite continuation?</b>
The system's code and hardware designs will be uploaded to GitHub, ensuring it remains fully open-source and accessible for future development. Additionally, the hardware will utilise off-the-shelf components to streamline assembly and enhance scalability.

# Plagiarism declaration

I have read and understand the Stellenbosch University Policy on Plagiarism and the definitions of plagiarism and self-plagiarism contained in the Policy [Plagiarism: The use of the ideas or material of others without acknowledgement, or the re-use of one's own previously evaluated or published material without acknowledgement or indication thereof (self-plagiarism or text-recycling)].

I also understand that direct translations are plagiarism, unless accompanied by an appropriate acknowledgement of the source. I also know that verbatim copy that has not been explicitly indicated as such, is plagiarism.

I know that plagiarism is a punishable offence and may be referred to the University's Central Disciplinary Committee (CDC) who has the authority to expel me for such an offence.

I know that plagiarism is harmful for the academic environment and that it has a negative impact on any profession.

Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully (acknowledged); further, all verbatim copies have been expressly indicated as such (e.g. through quotation marks) and the sources are cited fully.

I declare that, except where a source has been cited, the work contained in this assignment is my own work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

I declare that have not allowed, and will not allow, anyone to use my work (in paper, graphics, electronic, verbal or any other format) with the intention of passing it off as his/her own work.

I know that a mark of zero may be awarded to assignments with plagiarism and also that no opportunity be given to submit an improved assignment.

Signature



Name: Franco du Plessis

Student no: 25866095

Date: 09 March 2025

# ECSA Exit Level Outcomes

ECSA Outcomes Assessed in this Module	
ECSA Outcomes	Chapter Addressed
GA 1. Problem solving: Demonstrate competence to identify, assess, formulate and solve convergent and divergent engineering problems creatively and innovatively.	Chapter 1, 3, 4, 5
GA 2. Application of scientific and engineering knowledge: Demonstrate competence to apply knowledge of mathematics, basic science and engineering sciences from first principles to solve engineering problems.	Chapter 2, 3, 4, 5
GA 3. Engineering Design: Demonstrate competence to perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes.	Chapter 3, 4
GA 5. Engineering methods, skills and tools, including Information Technology: Demonstrate competence to use appropriate engineering methods, skills and tools, including those based on information technology.	Chapter 2, 3, 4, 5
GA 6. Professional and technical communication: Demonstrate competence to communicate effectively, both orally and in writing, with engineering audiences and the community at large.	Entire report
GA 8. Individual, Team and Multidisciplinary Working: Demonstrate competence to work effectively as an individual, in teams and in multi-disciplinary environments	Entire report
GA 9. Independent Learning Ability: Demonstrate competence to engage in independent learning through well-developed learning skills.	Entire report

# Table of contents

	Page
<b>Executive Summary .....</b>	<b>i</b>
<b>Plagiarism declaration .....</b>	<b>iii</b>
<b>ECSA Exit Level Outcomes.....</b>	<b>iv</b>
<b>Table of contents .....</b>	<b>v</b>
<b>List of figures .....</b>	<b>viii</b>
<b>List of tables.....</b>	<b>x</b>
<b>1 Introduction .....</b>	<b>1</b>
1.1 Background .....	1
1.2 Problem Statement .....	2
1.3 Aim.....	2
1.4 Objectives .....	3
1.5 Limits and Exclusions.....	3
1.6 Motivation .....	4
1.7 Structure of Report.....	4
1.8 AI Usage in Report.....	5
<b>2 Literature review .....</b>	<b>6</b>
2.1 Stereo Vision Background .....	6
2.2 Digital Image Correlation.....	6
2.3 Stereo Vision Geometry.....	8
2.4 Binocular Disparity .....	9
2.5 Camera Calibration.....	9
2.5.1 Intrinsic Parameters .....	10
2.5.2 Extrinsic Parameters.....	10
2.6 Design Choice for Computer Platform .....	11
2.6.1 Single Board Computers .....	11
2.7 Previous Work Done.....	12
2.8 Chapter Summary.....	12
<b>3 Hardware Design: .....</b>	<b>13</b>
3.1 Selecting a Single Board Computer .....	13
3.2 Selecting Cameras .....	15
3.3 Storage Expansion .....	17
3.4 Stereo-Rig Frame.....	17
3.4.1 Mounting Rail.....	17
3.4.2 Camera Base .....	17
3.4.3 Connector Between Base and Mounting Rail .....	18

3.5	Lighting and Control .....	18
3.6	Component Enclosures .....	18
3.7	Constructed Hardware System .....	21
3.8	Chapter Summary .....	21
<b>4</b>	<b>Software Design:.....</b>	<b>22</b>
4.1	Image Capture .....	22
4.2	Camera Calibration .....	23
4.2.1	Convert Images to Grayscale .....	24
4.2.2	Chessboard Corner Detection .....	24
4.2.3	Monocular Camera Calibration .....	27
4.2.4	Stereo Camera Calibration .....	28
4.3	Rectify Stereo Image Pair .....	29
4.4	Illustration of Disparity Map .....	30
4.5	Single Image Camera Calibration .....	31
4.5.1	Preprocess Images .....	32
4.5.2	Circle Identifier .....	34
4.5.3	Assigning Dots into Columns .....	35
4.5.4	Sort Columns into Two Grids .....	37
4.5.5	Calibrating Cameras .....	37
4.6	Chapter Summary .....	38
<b>5</b>	<b>Result and Analysis .....</b>	<b>39</b>
5.1	Calibration Target Preparation .....	39
5.2	Camera System Setup .....	40
5.3	Image Capture .....	41
5.4	Camera Calibration .....	41
5.5	Disparity Map of Scene .....	44
5.6	Single Image Calibration .....	46
5.7	Chapter Summary .....	49
	<b>Conclusion .....</b>	<b>50</b>
	<b>References .....</b>	<b>51</b>
	<b>Appendix A Resource Use and End-of-Life Strategy .....</b>	<b>55</b>
	<b>Appendix B Techno-Economical Analysis .....</b>	<b>56</b>
B.1	Budget .....	56
B.2	Planning .....	59
B.3	Technical Impact .....	60
B.4	Return on Investment .....	60
B.5	Potential for Commercialisation .....	60
	<b>Appendix C Risk Analysis and Safety Procedures .....</b>	<b>61</b>
C.1	Safety Risks .....	61

C.2	Safety Report .....	61
<b>Appendix D</b>	<b>Wiring Diagram of Relay Control .....</b>	<b>68</b>
<b>Appendix E</b>	<b>Chessboard Pattern Calibration Report .....</b>	<b>69</b>
<b>Appendix F</b>	<b>Single-Image Calibration Report .....</b>	<b>70</b>



# List of figures

	Page
Figure 1: Tensile Test Specimen Before and After Conducting a Tensile Test. ....	6
Figure 2: The Setup for 2D and 3D DIC is Illustrated with Grayscale Observed Image on the Left and the Post-processed Results of the DIC on the Right. ....	7
Figure 3: Stereo Geometry .....	8
Figure 4: LaVision StrainMaster. ....	12
Figure 5: 3D Printed Enclosures. ....	20
Figure 6: Electronics Enclosure. ....	20
Figure 7: Camera System Used in this Project. ....	21
Figure 8: Flow of Software Implementation. ....	22
Figure 9: Stereo Camera Calibration Flow. ....	23
Figure 10: OpenCV Function Implemented for Grayscale Conversion. ....	24
Figure 11: Location of Inner Corners of the Chessboard Target. ....	24
Figure 12: Chessboard Corner Refinement Implementation on Corner. ....	26
Figure 13: Process of Handling Autofocus during Image Capturing. ....	27
Figure 14: Image Rectification Process. ....	30
Figure 15: Single-Image Calibration Places within the Existing Flow. ....	32
Figure 16: Flow of Circle Identification Function. ....	34
Figure 17: LaVision's 3D Calibration Plate. ....	36
Figure 18: Column Clustering Function Flow. ....	36
Figure 19: Chessboard Square Measured ( <i>left</i> ), Chessboard Fastened to Wooden Slab ( <i>right</i> ). ....	40
Figure 20: Original Image ( <i>left</i> ), Grayscale image ( <i>right</i> ). ....	42
Figure 21: Initial Corner Detection ( <i>left</i> ), Refined Corners ( <i>right</i> ). ....	42
Figure 22: Images Captured to Create a Disparity Map. ....	44
Figure 23: Rectified Image Pair with Epipolar Lines. ....	44
Figure 24: Rectified Image Pair with Epipolar lines .....	44
Figure 25: Grayscale Disparity Map. ....	45
Figure 26: Jet Colour Gradient. ....	45
Figure 27: Jet Gradient Disparity Map. ....	46

Figure 28: Single Image of 3D Calibration Plate ( <i>left</i> ), Grayscale Conversion of Image ( <i>right</i> ). .....	46
Figure 29: Gaussian Blur Applied ( <i>left</i> ), Adaptive Thresholding Applied ( <i>right</i> ). .	47
Figure 30: Morphological Opening ( <i>left</i> ) and Closing ( <i>right</i> ) Applied. ....	47
Figure 31: Detected Circles Organised into Two Grids.....	48
Figure 32: Gantt Chart of Completed Project. ....	59
Figure 33: Relay Control Circuit. ....	68
Figure 34: Chessboard Stereo Calibration Report. ....	69
Figure 35: Single-Image Calibration Report. ....	70

# List of tables

	Page
Table 1: Single-Board Computer Specification Comparisons .....	14
Table 2: Flags Provided by the Chessboard Corner Detection Function.....	25
Table 3: Blurring Methods Provided by the OpenCV Library. ....	32
Table 4: Parameters Passed to HoughCircles Function.....	35
Table 5: Chessboard Pattern Specifications. ....	39
Table 6: Initial Project Budget.....	57
Table 7: Final Cost Breakdown.....	58

# 1 Introduction

## 1.1 Background

Tensile testing is a fundamental and widely adopted method for evaluating the mechanical properties of materials under uniaxial tensile load. It provides critical insights into how a material behaves when subjected to stretching forces, including properties such as yield strength, ultimate tensile strength, and elongation (Saba et al., 2018). Although the behaviour of materials can be described mathematically to some extent, variations in composition, structure, and processing often lead to complex responses that cannot be fully captured through theoretical models alone. As such, tensile testing serves as an essential experimental approach for accurately characterising these properties and validating material performance in real-world applications.

This method utilises machines like the MTS Criterion C44 to apply a force to the sample, which can be monitored using the built-in load meter. During tensile testing, various methods can be used to collect point data that will be used to determine the material characteristics (AbouelNour et al., 2024). One of the most common methods employed uses a strain gauge that is physically attached to the specimen while performing a tensile test (Elmardi & Khayal, 2024).

Although strain gauges are a cost-effective solution to determining the properties of a material, they are only capable of measuring single-plane properties. Additionally, they must be physically attached to the point of measurement on the specimen to collect data ("Strain gauges", n.d.). This is undesirable when working with specimens whose surfaces are irregular or fragile, as it can lead to adhesion problems. Strain gauges are also sensitive to temperature changes and have a limited measurement range due to linear elongation (Shanmukha, 2017).

A solution to these limitations is to employ a method that does not require physical connections to the point of measurement, can accurately collect data of the specimen in multiple planes and remains a low-cost solution.

One possible solution is to use 3D Digital Image Correlation (DIC) analysis. DIC is a non-contact, image-based methodology used for measuring surface geometry, deformation, and strain, first established in the 1980s (Holmes et al, 2023). The sample is covered in a unique speckle pattern that is vital for the software to track any changes to the sample. DIC records a sequence of high-resolution images as the sample deforms, using mathematical algorithms to track surface displacement and extract deformation data. To achieve 3D analysis of a sample, the digital images are captured using a stereoscopic camera setup

(Zaremba & Nitkiewicz, 2024). This is achieved by using two cameras slightly angled toward each other, mimicking human vision. This setup creates a 3D representation of the sample's surface, enabling full-field deformation analysis.

This analysis method requires three components: a stereoscopic camera setup, DIC software and a reliable calibration method. Currently, commercially available DIC systems, such as those from Correlated Solutions and LaVision, are prohibitively expensive, with costs exceeding R1 000 000. While the M&M Engineering Department at Stellenbosch University has two of these systems, access to these systems is limited. Concerns about potential damage to the equipment are a key reason for its infrequent use in undergraduate studies. The problematic and confusing user interface (UI) further emphasises the need for a new system. This highlights the need for a cost-effective, in-house developed alternative.

An open-source DIC system offers greater opportunities for researching innovative methods for conducting DIC and analysing speckle patterns. Full accessibility to the system is essential, enabling researchers to modify all relevant components as needed.

## **1.2 Problem Statement**

The problem is that there is no accessible open-source stereoscopic camera system that allows accurate, low-cost 3D DIC analysis within the research and teaching environment.

## **1.3 Aim**

The project will focus on designing and constructing a stereoscopic camera setup for conducting 3D DIC analysis. This setup will serve as a cost-effective alternative to the existing commercial systems while maintaining usability and accuracy.

The project aims to provide the Mechanical and Mechatronic Engineering Department with an additional DIC setup accessible to students from undergraduate to PhD levels. The scope includes developing the stereoscopic camera setup and designing a highly accurate and intuitive calibration procedure. This calibration process will be guided by intuitive and user-friendly interfaces, ensuring ease of use. Additionally, this system will be able to extract disparity maps of the analysed image scene. However, the development of the 3D DIC algorithm itself falls outside the scope of this project.

Unlike current commercial systems, this setup will feature an intuitive interface that simplifies operation and reduces the need for extensive training. As an in-house developed system, it offers flexibility for continuous improvement while

reducing long-term costs associated with purchasing and maintaining commercial equipment. The open-source nature of the project enables anyone worldwide to access, build upon, and enhance the system, fostering collaborative innovation.

## **1.4 Objectives**

The project will be defined by three main objectives, each with its own sub-objectives.

### **1. Physical System:**

- a. Design and construct a stereoscopic camera system that fits within an area of approximately  $1 \text{ m}^3$  to reduce the system footprint while operating.

### **2. Calibration Procedure:**

- a. Develop and implement a stereo camera calibration procedure that creates an accurate representation of changes in depth within an image scene.
- b. Create the calibration procedure in such a manner that it can be completed within approximately 120 seconds from initiation to completion.

### **3. Control Software:**

- a. Develop optimised, user-friendly interfaces for navigating or conducting tasks when using the camera system. These interfaces will control all functions, including calibrating the system, capturing and storing images, and creating disparity maps of image scenes for analysis and interpretation.

## **1.5 Limits and Exclusions**

The project has defined limits and exclusions to establish clear boundaries within which it will be conducted. The camera system will focus on tracking samples with sizes closely aligned with existing tensile testing standards ("Standard & specimens for metals tensile test", n.d.). As a result, the system will be limited to tracking samples between 50 mm and 150 mm in both length and width. Although the system is capable of monitoring samples outside of this range, the project scope will be limited to the dimensions as mentioned above. This is because most tensile testing done to calculate material properties is conducted on samples that follow the ASTM standard, which are approximately the same size ("Standard & specimens for metals tensile test", n.d.) The project will not utilise

high-speed cameras, and therefore, the project scope will exclude high-speed testing. The cameras will operate at a maximum frame rate of 45 frames per second (fps). The cameras are capable of operating at a maximum frame rate of 56 fps at a resolution of  $2304 \times 1296$  pixels ("Camera - Raspberry Pi Documentation", n.d.). The limit of 45 fps allows adequate room for processing in the background while capturing images.

The project must be completed within the academic year and will operate within a maximum hardware budget of R5 500. The development of a 3D DIC algorithm is excluded to ensure project completion within the given timeframe. The project scope will not include the connection made between the tensile tester and the camera system.

## **1.6 Motivation**

The motivation to conduct this project stems from the need to develop an in-house 3D DIC setup for the Mechanical and Mechatronic Engineering Department at Stellenbosch University. The system significantly reduces the initial capital cost compared to current commercial systems. Transitioning from these expensive commercial systems will free the Mechanical and Mechatronic Department from being restricted to using only specific replacement parts, which are often tied to the commercial system. Additionally, this reduces downtime caused by waiting for internationally shipped parts, which also incur import taxes. Beyond cost reduction, this project offers flexibility and customisation, enabling continuous improvements and adaptation to future developments in 3D DIC technology. Furthermore, the project allows the department to replicate the system, facilitating scalability and broader use across different research groups. Ultimately, the project establishes a solid foundation for future advancements in the 3D DIC field, particularly in terms of the potential development of an in-house, custom-designed 3D DIC algorithm.

## **1.7 Structure of Report**

The report will start with a literature study talking about the project background, relevant components and previous work done in Chapter 2. The hardware design will be discussed in Chapter 3, with the analysis and motivation for design choices. Chapter 4 will analyse the software design and flow of information needed to achieve the full functionality required. The results will be examined in Chapter 5, along with the testing procedures followed to obtain them.

## **1.8 AI Usage in Report**

This project was completed with assistance from AI. The use of AI was responsible, and everything in this report is the author's own work, unless otherwise indicated. AI was employed to correct spelling errors, improve grammar, and enhance the flow of paragraphs. It was also utilised to aid in understanding and analysing open-source code and its interactions.



## 2 Literature review

This chapter reviews the background, literature and previous work regarding stereoscopic camera systems. The elements discussed in this chapter will focus on the aspects related to the relevant implementation of 3D DIC.

### 2.1 Stereo Vision Background

Stereoscopic camera systems utilise two or more spatially separated image sensors to capture the same scene from different viewpoints. This arrangement mimics human binocular vision and enables depth perception through the principle of triangulation (Perumal et al., 2021). The fundamental physical assumption is that a single point in the real world will project to distinct pixel locations in each camera's image plane. The difference in these pixel positions, referred to as binocular disparity (as discussed in Section 2.4), is used to infer depth (Anzai & DeAngelis, 2010). This process, which reconstructs three-dimensional geometry from two-dimensional images, is known as stereo vision (as discussed in Section 2.3), and forms the basis of spatial measurements in three-dimensional Digital Image Correlation systems (as discussed in Section 2.2).

### 2.2 Digital Image Correlation

Digital Image Correlation (DIC) is a technique that utilises digital images to measure the deformation of objects under stress. This non-contact method is widely used in fields such as materials science and engineering because it provides detailed data without requiring physical contact with the object. It is accomplished by continuously analysing the differences between two sequentially captured images. In practical engineering applications, DIC is employed to measure the deformation of specimens, such as the tensile test specimen seen in Figure 1, during tensile testing. This is done for various reasons such as conducting quality control on manufactured samples, predicting the performance of materials to aid in material selection, and to compare materials during research and development.



**Figure 1: Tensile Test Specimen Before and After Conducting a Tensile Test.**

*Source: "Tensile test brittle deformation fracture", 2021*

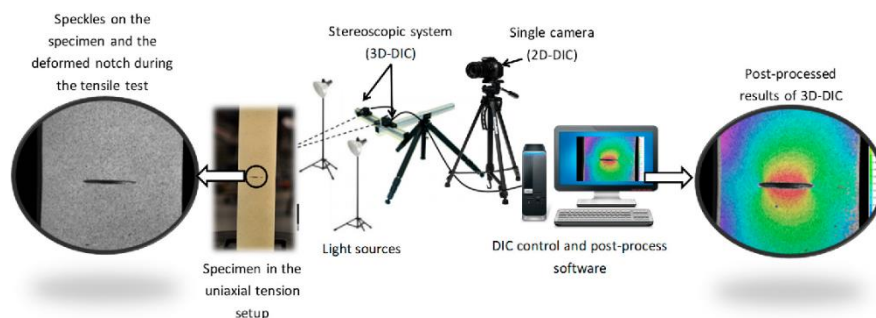
The DIC system captures images of the specimen's surface at various stages as a controlled tensile force is applied. By analysing these images, DIC provides a full-field strain map that highlights how the material deforms, necks, or localises strain before failure. This detailed spatial data enables engineers to accurately

determine critical mechanical properties, such as Young's modulus, yield strength, and ultimate tensile strength. The DIC system requires data from the tensile tester, like tensile force, to conduct the deformation analysis. This creates a requirement for the camera system to be able to receive data from the tensile tester. The non-contact measurement capability of DIC ensures that the specimen remains unaltered during testing, thereby delivering reliable results necessary for material characterisation, product development, and safety assessment in real-world engineering scenarios.

DIC is primarily divided into two forms, 2D and 3D, both of which use a random speckle pattern during image analysis. The sample being analysed is covered in a random speckle pattern after the necessary surface preparations have been completed. The application of the speckle pattern is done using various methods like spray painting, spin coating and lithography, to name a few (Dong & Pan, 2017). The sample surface and background must have a high contrast to ensure optimal dot tracking conditions. This can be achieved by using a white background and black dots, for example. Consistent speckle size, typically 3-5 pixels in size, enhances spatial resolution, enabling more precise displacement measurements, especially when the sample is covered in an approximately equal distribution of white and black areas ("Fundamentals of Speckling for DIC", 2023).

In DIC, the displacement of each speckle, represented by an individual point in the pattern, is measured as the distance it moves between consecutively captured images. The DIC software analyses this movement to determine the vector quantities of displacement. It tracks small regions of the image across successive frames by matching patterns to those in the reference image. This technique is applied across the entire image to monitor the displacement of all speckles during the deformation process. From this, normal, shear, and principal strains are determined. For 3D DIC, which measures depth and out-of-plane motion, two or more cameras are required to capture stereoscopic data ("Digital Image Correlation", n.d.).

Two-dimensional DIC is limited to measuring in-plane deformations exclusively due to its single-camera setup. This approach is more straightforward and requires fewer computational resources compared to three-dimensional DIC. However, the technique is incapable of capturing out-of-plane displacements, which may result



**Figure 2: The Setup for 2D and 3D DIC is Illustrated with Grayscale Observed Image on the Left and the Post-processed Results of the DIC on the Right.**

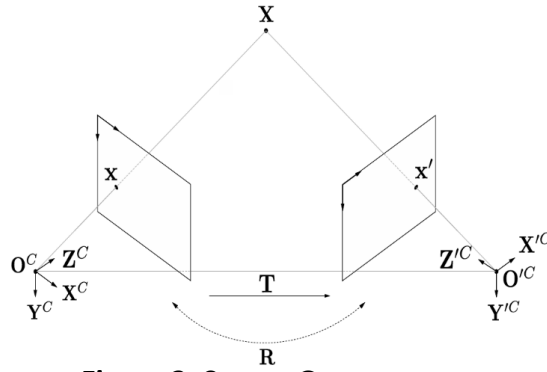
*Source: "DIC and test set up", 2018*

in inaccuracies if the sample surface undergoes rotation or bending out of the plane (Sutton et al., 2008).

Three-dimensional DIC employs a multi-camera system to capture three-dimensional deformation, including out-of-plane movements, thereby facilitating more precise sample analysis. The intricacy of the setup necessitates meticulous and accurate calibration to attain optimal accuracy (Tiwari et al, 2007). The setup of 2D and 3D DIC can be seen in Figure 2, with an illustration of the elongation measured throughout the sample.

## 2.3 Stereo Vision Geometry

In a stereoscopic camera setup, the image measurements of each camera are given with respect to its own image coordinate system. To ensure accurate 3D spatial information, these measurements must be generalised to a single standard coordinate frame. A general representation of stereo vision geometry is shown in Figure 3, which illustrates the notation used in this report. To simplify calibration, the standard reference frame used by the system will be the same as the coordinate frame of the left camera.



**Figure 3: Stereo Geometry**

Source: "Generic representation of the stereo geometry", 2017

The centre of the left and right cameras' optical sensors is defined as  $O^C$  and  $O'^C$  respectively. Furthermore, the left camera's coordinate system is indicated by  $X^C$ ,  $Y^C$  and  $Z^C$  and has a real point  $X$  projected onto the image plane as  $x$ . Similarly, the right camera's coordinate system is indicated by  $X'^C$ ,  $Y'^C$  and  $Z'^C$  and has a projected point  $x'$ . Relative translation and relative rotation are indicated by  $T$  and  $R$  respectively. The relative rotation is the transformation that allows the alignment of one camera with another, expressed as a 3x3 matrix. The relative translation indicates the positional difference between the optical centres of the cameras, expressed as a 3D vector. These two parameters represent the extrinsic camera parameters and will be determined using stereo calibration. Baseline ( $B$ ) is the distance between the centres of the two cameras' optical sensors. The distance between the camera's sensor and lens is known as the focal length ( $f$ ),

measured in millimeters. Focal length can also be expressed in pixel units as  $f_x$  and  $f_y$  and is used to scale the world coordinates into the image plane.

## 2.4 Binocular Disparity

Binocular disparity is the horizontal offset between projections of the same 3D point onto two horizontally separated camera sensors. When you view a scene with two cameras, each camera sees the world from a slightly different angle. The difference in pixel coordinates of a feature between the left and right images is its disparity (Young, 2020).

To calculate the disparity ( $d$ ) of pixels in an image, it must first be established which pixel in the image on the right matches a given pixel in the left image (“Depth Chapter 11”, n.d.). This can be completed using several methods, such as block matching (BM), semi-global block matching (SGBM), and feature-based matching (“OpenCV”, n.d.). In this report, SGBM will be used due to its increased accuracy and parameter tuning capabilities (“OpenCV”, n.d.). SGBM has a total of 11 parameters, compared to feature-based matching, which has five parameters, and block matching, which has two parameters. This is why SGBM was selected.

After the corresponding points have been identified, using SGBM, sub-pixel enhancement can be applied to boost depth precision (“OpenCV”, n.d.). The enhanced data can be used to calculate the disparity using Equation 2.1.

$$d = x_L - x_R \quad (2.1)$$

The left pixel’s horizontal location is represented by  $x_L$  and the pixel on the right by  $x_R$ . Thus, the disparity ( $d$ ) is the difference between the two projected points on the image frame. After the disparity is calculated, the triangulation equation can be applied to compute the real-world depth of each point (“Depth Chapter 11”, n.d.).

$$Z = \frac{f \cdot B}{d} \quad (2.2)$$

In Equation 2.2,  $Z$  refers to the depth along the camera’s optical axis measured in millimeters.

## 2.5 Camera Calibration

To achieve an accurate correlation between 3D real-world points and 2D image points on an image, we must apply camera calibration. This process is characterised by calculating the intrinsic and extrinsic camera parameters. Intrinsic parameters describe the internal optics of the camera, including focal length ( $f_x, f_y$ ) and principal point ( $c_x, c_y$ ), both of which is measured in pixels.

Extrinsic parameters, on the other hand, define the camera's position in the real world relative to a reference coordinate system, like its rotation and translation (Hartley & Zisserman, 2004).

Calibration is done to account for and correct the imperfections found in the cameras' lenses and sensors, ensuring accurate measurements. Accurate calibration ensures a highly accurate representation of real-world properties.

A calibration object is used to define real-world coordinates of 3D points, such as a checkerboard or calibration plate, with known dimensions. The cameras capture the images of the calibration plate and find the pixel coordinates  $(u, v)$  for each 3D point. The intrinsic and extrinsic camera parameters are calculated using a camera calibration algorithm.

### 2.5.1 Intrinsic Parameters

The focal length (measured in pixels) indicates the distance between the optical centre and the image plane, adjusted according to the pixel dimensions of the sensor. The principal point is the pixel coordinate where the camera's optical axis intersects the image plane, typically located near the centre of the image. The focal length and principal point are expressed in a matrix called the intrinsic matrix ( $K$ ), shown in Equation 2.3. The skew parameter ( $s$ ) is generally assumed to be zero for most cameras and can usually be ignored ("Pinhole Camera Model", n.d.).

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \approx \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

### 2.5.2 Extrinsic Parameters

Extrinsic parameters define the spatial relationship between the camera and the world coordinate system. They consist of a rotation matrix ( $R$ ) and a translation vector ( $T$ ), which together transform 3D points from the real-world coordinate system into the camera coordinate system. Rotation ( $R$ ) is a  $3 \times 3$  matrix that specifies the orientation of the camera with respect to the reference coordinate system. It determines how the camera is "pointing" in space. Translation ( $T$ ) is a  $3 \times 1$  vector that defines the position of the camera centre relative to the world origin. It describes the location of the camera (Anwar, 2022).

The combination of  $R$  and  $T$  allows us to map a point from world coordinates  $(X^W, Y^W, Z^W)$  into the camera coordinate system, using Equation 2.4.

$$\begin{bmatrix} X^C \\ Y^C \\ Z^C \end{bmatrix} = R \begin{bmatrix} X^W \\ Y^W \\ Z^W \end{bmatrix} + T \quad (2.4)$$

This step is pivotal as it aligns the world coordinate system, defined by the calibration object, with the camera's coordinate system, thereby enabling the correlation of real-world geometry with image measurements.

## **2.6 Design Choice for Computer Platform**

The camera system must be able to operate independently, capturing and processing images without requiring external assistance. To achieve this functionality, the camera system requires its own computing system. Multiple options can be taken to achieve this outcome, for instance: Microcontroller or Single-Board Computer (SBC). Each of these options has its own advantages and disadvantages. For this project, the SBC was the most suitable choice for achieving the desired outcomes due to the following reasons. The average processing power of microcontrollers is considerably less than that of an SBC. The CPU, RAM and storage capabilities of microcontrollers are also inferior to those found in most SBCs. These are crucial for the smooth operation of any stereoscopic camera system. Another benefit of the SBC is that it runs a whole operating system, which introduces the ability to run high-level programming languages like Python and C++ ("The Comparison between Microcontrollers and Single board Computer", 2024). The versatility and enhanced connectivity options established SBCs as the optimal choice for this project. In the future, the SBC will also connect to a laptop or computer to utilise its additional processing power for conducting 3D DIC. DIC.

### **2.6.1 Single Board Computers**

Single-board computers (SBCs) are self-contained computers built on a single circuit board. They are characterised as being compact, versatile and affordable. SBCs integrate all the essential components needed to function as a computer onto a single board. They have their own processor, memory, storage, and input-output interfaces, and only require a display, power connection, and navigation peripherals to function as a standalone computer. Most SBCs are not only limited to being used as a computer but also support expansion ports for adding features (Arun KL, 2023). This will benefit the system's future use in 3D DIC, where the tensile tester can transmit valuable data, like tensile force, to the SBC via the GPIO pins. The process of choosing the most suitable SBC will be discussed in Section 3.1.

## 2.7 Previous Work Done

The system developed in this project is inspired by LaVision's commercial digital image correlation (DIC) system, StrainMaster, which is used to measure sample deformation, as seen in Figure 4. The LaVision system supports both 2D and 3D DIC and is operated through its proprietary software, DaVis, which manages all system functionalities ("LaVision DaVis 11", n.d.). It features a stereo camera setup with a modular design, consisting of a rail mounted on a bipod, equipped with movable lights and cameras. The system can perform multiple calibration methods, including the use of a chessboard or a custom 3D calibration plate. The 3D calibration plate has a height difference that allows the system to calibrate a stereo camera pair using a single image by leveraging the disparity in depth between the two levels.



**Figure 4: LaVision StrainMaster.**

*Source: "StrainMaster-portable", n.d.*

## 2.8 Chapter Summary

This chapter provided the literature needed to comprehend the concepts used throughout the project. The next chapter will discuss the hardware design choices made during the process of creating the stereo camera system.

## 3 Hardware Design:

This chapter will discuss all the choices made during the process of designing and creating the hardware used in this project. The components that will be discussed are used to create a stereoscopic camera system. The section will include a discussion of the single-board computer, cameras, system frame, lighting, storage and component enclosures.

### 3.1 Selecting a Single Board Computer

The world of single-board computers (SBCs) is extensive and caters to all requirements. The four main criteria to consider when selecting one are: processing power, input/output (I/O) ports, expansion capabilities, and price range. Each criterion is chosen for specific reasons. Processing power is essential because the SBC must be capable of capturing, processing, and analysing data on the system. I/O ports are crucial, as they enable the user to connect peripherals to the SBC easily. Peripherals may include, but are not limited to, a computer screen for data display, as well as a USB wired or Bluetooth mouse and keyboard for navigating the central control software. A crucial aspect of the I/O ports is the presence of general-purpose input/output (GPIO) pins. These pins are fully programmable and can be used to send or receive electrical signals (Butler, 2022). For this project, the GPIO pins will control the lighting of the camera rig. In the future, the pins could be utilised to read data from external equipment such as load cells or extensometers, assisting in the development of the 3D DIC system.

Expansion capabilities, because the system will require specific additional expansion components to achieve the end goal of having a stereoscopic camera system. These expansions may include installing an SSD kit to enhance storage speed and increase overall capacity or adding cameras to establish the stereoscopic camera setup needed for this project. The system is being developed to a specific operational standard while keeping future possibilities in mind, especially in case someone else takes over and expands it further. Price range, because the project is focused on creating a budget-friendly stereo camera system, and the computing units are one of the most expensive components in the system.

In Table 1, a few of the options that were considered for the SBC will be listed along with their relevant specifications. The criteria mentioned above were used to select an SBC system that would achieve the project's outcome.

The three candidate SBCs differ notably in processing power and memory. The Raspberry Pi 5 (RPi 5) can be equipped with RAM ranging from 2 to 16 GB and features a 2.4 GHz quad-core CPU, providing ample headroom for running a high-resolution camera system. The additional memory enables smoother handling of



image buffers and video streams. The Raspberry Pi 4B, by contrast, features a range of 1 to 8 GB of RAM and a quad-core CPU at 1.5 GHz, providing adequate but more limited performance for demanding imaging tasks. The BeagleBoard BeagleY-AI integrates 4 GB of RAM and a 1.4 GHz quad-core processor, offering less raw performance and memory capacity than the RPi 5, which may limit scalability when managing larger datasets or higher frame rates.

This project requires an SBC that has two CSI ports for cameras to facilitate stereoscopic vision, an Ethernet port to connect to a wired internet connection if needed, and multiple USB ports to navigate the SBC. The presence of dual CSI ports is one of the most important properties that the SBC must have, a vital feature required to connect two cameras simultaneously for the stereo camera system, essential to achieving stereoscopic vision in this project. The USB ports are needed to ensure the system has a robust user navigation method, allowing for the physical connection of peripherals such as keyboards and mice, providing robust connectivity. Additionally, the SBC should support the possibility of remotely logging in and controlling the SBC via protocols such as SSH or VNC, thereby enhancing accessibility without requiring physical interaction. The Ethernet port is also necessary to ensure system robustness, providing a reliable fallback for network connectivity when Wi-Fi is unavailable. Furthermore, the presence of an HDMI port allows the user to connect a display to operate the SBC and conduct all the necessary steps, such as setup, configuration, and troubleshooting.

The SBC must feature adequate expansion capabilities, including the presence of GPIO pins and provisions for SSD upgrades. The presence of GPIO pins is a vital requirement for user customisation in the open-source system design, ensuring that the system can be altered to fit a specific scenario. The GPIO pins can also be used to control key system components from a central control software, like turning on lights for improved sample clarity. Furthermore, the SSD upgrade capability is essential to increase the speed at which photos and videos can be saved, while also boosting overall storage capacity. Both the Raspberry Pi 5 and the BeagleBoard BeagleY-AI feature a PCIe connector, allowing for the connection of a fast SSD.

**Table 1: Single-Board Computer Specification Comparisons**

Parameter	Raspberry Pi 5 <sup>1</sup>	Raspberry Pi 4B <sup>2</sup>	BeagleBoard BeagleY-AI <sup>3</sup>
Processing Power	Quad-core processor @ 2.4 GHz 8 GB LPDDR4 RAM	Quad-core processor @ 1.5 GHz 8 GB LPDDR4 RAM	Quad-core processor @ 1.4 GHz 4 GB LPDDR4 RAM
I/O Ports	4 x USB 2 x microHDMI 1 x Gigabit Ethernet	4 x USB 2 x microHDMI 1 x Gigabit Ethernet	4 x USB 1 x microHDMI 1 x Gigabit Ethernet
Expansion Capabilities	40-pin header 2 x MIPI CSI ports	40-pin header 1 x MIPI CSI port	40-pin header 2 x MIPI CSI ports

<sup>1</sup> "Raspberry Pi", 2025

<sup>2</sup> "Raspberry Pi", 2025

<sup>3</sup> "Mouser Electronics", 2025

Parameter	Raspberry Pi 5 <sup>1</sup>	Raspberry Pi 4B <sup>2</sup>	BeagleBoard BeagleY-AI <sup>3</sup>
	PCIe FFG Connector		PCIe FFG Connector
Price Range	± R1 680	± R1 580	± R1 455

Given the primary emphasis of the project on cost-effectiveness, the selected SBC must strike a balance between affordability and delivering all essential features without unnecessary extravagance. This ensures that resources are optimised, particularly in an open-source environment where scalability and value for money are paramount. The pricing of SBCs is intrinsically linked to their performance metrics, processing power, and integrated capabilities.

Among the options considered, the Raspberry Pi 5 stands out as the most expensive variant, with prices around R1 680. It is closely followed in cost by the Raspberry Pi 4 Model B, which retails at approximately R1 580. In contrast, the BeagleBoard BeagleY-AI emerges as the most affordable option, available for around R1 445. However, despite its attractive pricing, the Raspberry Pi 4B falls short in key areas, notably lacking dual CSI ports. This limitation renders it unsuitable, as it would necessitate additional hardware workarounds that could compromise cost-effectiveness and system simplicity.

Ultimately, the slower processing speed of the BeagleBoard BeagleY-AI and the smaller community support did not justify the savings made when choosing this option, despite its dual CSI ports. Keeping in mind that the system must be future-proof, the Raspberry Pi 5 was selected as the chosen SBC for this project.

## 3.2 Selecting Cameras

Choosing the right cameras is essential for the project's success. Cameras serve as the primary connection between the real world and the data processed by the software via the single-board computer. The cameras that will be used in this project must integrate seamlessly with the chosen SBC. The Raspberry Pi 5 features four USB ports and two CSI ports, all of which can be used to connect cameras. The USB ports of the RPi 5 are split, two of them are USB 3.0 ports and the other two are USB 2.0 ports. This difference is an important characteristic to keep into account when deciding on the correct USB camera, because of the speed difference between the USB 3.0 and USB 2.0 ports. It is required that the cameras be exactly the same model; this helps the system to use the same method to communicate and control the cameras.

Although multiple suitable USB cameras are available, this project will restrict the system to use only cameras that connect to the RPi 5 through the CSI ports. This was decided because of the ability to synchronise the frames captured by the cameras using the CSI ports. It is crucial for the stereoscopic camera systems to capture synchronised frames during 3D DIC to ensure accurate calculations. The

higher bandwidth provided by the CSI ports along with the lower latency made it far superior than USB cameras.

Several cameras communicate using an FFC that connects to a CSI port. To make the camera system user-friendly, the camera must have autofocus. This requirement ensures that the user can achieve perfect image focus through software alone, without physically adjusting the camera. Cameras without autofocus require the user to manually adjust the lens until the desired focus is achieved, adding extra steps that can cause difficulties. Depending on the overall camera system setup, it can be challenging to monitor outputs while adjusting focus. It may also result in moving the system during adjustments, which could affect the accuracy of results in successive tests.

Raspberry Pi offers a variety of camera modules that can connect to the RPi 5 via the CSI ports. The Camera Module V2 was launched in 2016 and features an 8-megapixel camera, but it is limited to manual focus only. Its successor, released in 2023, is the Camera Module V3, which has a 12-megapixel camera and includes autofocus. The Camera Module V3 comes in four different models with various features, such as standard and wide-angle lenses, as well as infrared options. Raspberry Pi also provides two camera modules with lens mount options. The High-Quality Camera model offers two lens mounting options, the CS- and M12-mount variants. Both of these cameras have manually adjustable focus, which disqualifies them for this project.

ArduCam is another big provider in the camera scene for SBCs. They offer options like the ArduCam PTZ camera controller module, which, when paired with their 8MP IMX219 camera, provides a system with autofocus and a replaceable lens. This feature would be a bonus, enabling the cameras to zoom in on the object during testing. Unfortunately, this module does not work with all the camera libraries utilised in this project and falls outside of the budget with a price of R1 647 per camera ("RobotShop", n.d.). Another module that fits the requirements is the ArduCam 12MP 477P autofocus high-quality camera module. This module features autofocus and an M12 lens mount, but unfortunately, it falls outside of the budget with a price of R1 855 per camera ("12.3MP Motorized Focus Camera Module for Raspberry South Africa", n.d.).

This project will utilise the Raspberry Pi Camera Module V3 configured with the standard lens and an IR filter. This option falls precisely within the project's budget and includes all the required features. The standard option is chosen above the wide-angle option to ensure that the best resolution image of the sample is captured. This project will not benefit from the NOIR module due to it operating under the controlled illumination provided by the lights as discussed in Section 3.5. The Sony IMX708 sensor has a resolution of 4608 x 2592 pixels and a FOV of 75 degrees. The camera module is capable of capturing a smooth 56 fps at a

resolution of 2304 x 1296 pixels, which is good when the camera system is used to conduct 3D DIC.

### **3.3 Storage Expansion**

The camera system would benefit from additional storage to increase the amount of data that can be captured on the device, as well as an improvement in read and write speeds. The faster images and videos can be saved to the storage, the quicker the software can start analysing the saved data. The micro SD card was tested using the built-in SD card speed test and showed a sequential write speed of 72 037 KB/s. The price of a 512 GB U1 micro SD card is R1 099 and will, in theory, improve the write speed. The cost of a Raspberry Pi 512 GB SSD kit is R1 150, just over R50 more than the micro SD card, while increasing the sequential write speed to 859 488 KB/s. Each camera captures a video feed of 2304x1296 pixels at 45 fps. This corresponds to a total required writing speed of 393 660 KB/s per camera, or a combined writing speed of 787 320 KB/s. By expanding the storage of the Raspberry Pi 5 with a faster SSD kit, the project will not be limited by storage speed.

### **3.4 Stereo-Rig Frame**

The system's frame acts as the backbone to which all components are attached. It must have a rigid structure to ensure consistency during use, while also maintaining modularity to adapt to various situations the system might face. This section will be divided into three parts: the mounting rail, the base, and the connection between them.

#### **3.4.1 Mounting Rail**

To ensure the system has a solid backbone, all components must be connected to a single rigid piece. This piece must have enough flexibility to allow all components to be connected and adjusted to suit the intended application. An aluminium extrusion was selected for its rigidity, lightweight nature, and widespread availability. The specific dimensions chosen for this project are 20x20x1 000 mm. This helps keep costs and weight low while providing the necessary rigidity for the mounting rail. Many products are available for 20x20 aluminium extrusions, making future upgrades or extensions to the mounting rail easier.

#### **3.4.2 Camera Base**

The camera system needs a secure base for attachment. The base should be adjustable enough to adapt the system for different scenarios. The project will use an adjustable tripod as a stable foundation that can be configured for various situations. The tripod features a built-in bubble level to assist the user in levelling

the camera system. When folded, the tripod measures 350 mm long, making it easy to store, and it can extend to a height of 1 020 mm.

### **3.4.3 Connector Between Base and Mounting Rail**

The system must include a connecting piece between the aluminium extrusion and the tripod base. This ensures that the stereo-rig frame functions as a single solid unit. The connector is designed using Autodesk Inventor Professional 2025 and is 3D printed with Polylactic Acid (PLA) for this project. However, it is not limited to being made from PLA alone. The connector can be machined from metal if increased rigidity or durability is needed. It features three holes: two for mounting to the aluminium extrusion with T-bolts, and a third for attaching to the tripod. The tripod connection hole will incorporate a threaded brass insert to guarantee a secure and robust connection. The threaded brass insert can be added with the use of a hot soldering to melt the plastic. The connection piece can also be used to attach the aluminium extrusion to a different base or even the tensile tester itself with custom extensions.

## **3.5 Lighting and Control**

When capturing photos and analysing samples, consistent lighting is essential for reliable and accurate results. The system should include its own illumination, providing sufficient light regardless of ambient conditions. To prevent damage to the Raspberry Pi, the lighting control circuit must be isolated from it, yet still manageable via the Pi's GPIO pins.

For this project, two 10 W LED flood lights, each producing 1 000 lumens, will be used. Their small size and affordability align well with the project's goals. Powered by 220 V, they require a relay circuit to be safely controlled by the system through the GPIO pins. The power extension cord providing power to the RPi 5 power supply has two 3-point plugs, allowing for an additional connection for the 220V lights. An off-the-shelf 2-relay module, known for its simplicity and compactness, was chosen. It operates at 5 V DC and can handle 10 A at 250 V AC per relay. An external 5V supply will power the relays to protect the GPIO pins. Control will be via two GPIO pins on the Raspberry Pi (GPIO 14 and 15), designated for the left and right lights. The control circuit is isolated from the high voltage side. Refer to Figure 33 for the connection diagram in Appendix D

## **3.6 Component Enclosures**

This section discusses the enclosures developed or chosen for the camera system components. The first enclosure addressed is the one designed for the Raspberry Pi 5. Although many enclosures are available for the RPi 5, none meet the specific requirements of this project. Commercial options, such as the official RPi 5 case or

the Argon NEO 5 case, each have limitations. The official RPi 5 case lacks the mounting options needed to attach it to the aluminium extrusion. Additionally, its lid cannot close once the SSD kit has been installed ("Raspberry Pi Case for Raspberry Pi 5", 2024). The Argon NEO does not facilitate routing the camera cables, rendering it unsuitable for this project. The Raspberry Pi community has created multiple CAD files suitable for 3D printing, enabling custom fitting for various projects, such as building a desktop PC or attaching to the back of a monitor. None of the cases that were found suited the intended use case for this project.

A RPi 5 case was designed in Autodesk Inventor Professional 2025 to meet specific requirements. These include allowing the SSD kit to be mounted onto the RPi 5 while remaining enclosed, providing sufficient space for routing camera cables from the RPi 5 and the cameras, incorporating appropriate mounting options to secure the case to the aluminium extrusion, and ensuring adequate ventilation and port access.

The case will be 3D printed due to its cost-effectiveness and easy accessibility. The decision to use 3D printing also aims to keep the project on an open-source track. This allows anyone with access to a 3D printer to produce the case, even if they are not working on this specific project. The process begins by converting a 3D model into G-code, a format readable by 3D printers. Filament is then heated to its melting point and laid onto the print bed according to the G-code instructions. This process is repeated layer by layer until the object is complete. The plastic selected for this project is Polylactic Acid (PLA) due to its affordability, ease of use, and material properties (Carlota, 2023). The case will not encounter significant forces and will stay below the glass transition temperature of 65 °C for PLA. Other common filaments are either difficult to print, like ABS, or very expensive, like PA12-CF Nylon.

The enclosure for the Raspberry Pi Camera Module V3 had the same problems as the RPi 5 case. The lack of commercially available options, joined with the existing CAD models, led to the design of a custom camera enclosure. This camera enclosure must fit the Raspberry Pi Camera Module V3 snugly, protect all the circuitry and lens mechanisms inside and allow for adjustable mountings to the aluminium extrusion. The enclosure was also designed using Autodesk Inventor Professional 2025 and 3D printed out of PLA for the same reasons as the RPi 5 case. The 3D-printed enclosures can be viewed in Figure 5.



**Figure 5: 3D Printed Enclosures.**

The final enclosure is for housing the electronics responsible for controlling the lights, as seen in Figure 6. Due to the circuit operating with 220 V and having live exposed contacts, the enclosure must be capable of reliably and safely isolating the electronics from everything in its surroundings. To mitigate the risk of a faulty 3D printing part posing a potential threat, the enclosure must be an off-the-shelf component rated to handle 220 V. To ensure that the enclosure meets these requirements, the Ingress Protection (IP) rating will be used as guidance. The IP rating guidelines indicate the level of protection of an electrical enclosure against the environment. The rating starts with IP and then leads with two numbers, the first indicating the protection against solids and the second against water. The enclosure must protect against, at a minimum, any solid objects entering the enclosure and against water splashing onto it. To satisfy these requirements, the enclosure must at least be rated to IP 44. The enclosure that was chosen is rated to IP 55, due to the extra protection against dust and water (Luke, 2013). This will ensure that the electronics remain completely protected against dust and water during operation. The CAD files can be downloaded by following this [link](#) to the GitHub page.



**Figure 6: Electronics Enclosure.**

*Source: "Junction Box Square", 2025*

### 3.7 Constructed Hardware System

Figure 7 illustrates the entire physical system after combining all of the hardware into one camera assembly.



**Figure 7: Camera System Used in this Project.**

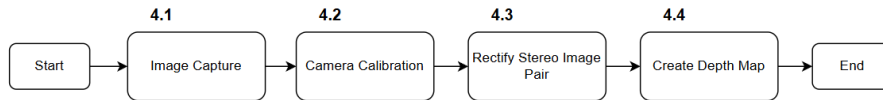
### 3.8 Chapter Summary

This chapter discussed the hardware design of the camera system created in this project. The choices that were made in picking components and materials were adequately provided and motivated. This ensures that the reason for the hardware is sufficient if someone would reference back to this report without any context. The next chapter will provide the software design choices made in this project.



## 4 Software Design:

This chapter covers all the software design choices and optimisations made during the project's development. Each aspect of the process will be examined individually, beginning with image capturing. The calibration process will follow, then image rectification and disparity mapping. The section will end with a discussion of how single-image calibration will be implemented. Figure 8 illustrates the flow of the software implemented in this project.



**Figure 8: Flow of Software Implementation.**

### 4.1 Image Capture

This code provides image acquisition functionality for a stereo camera calibration system built with a Raspberry Pi 5 and two Raspberry Pi Camera Module V3 devices. Its primary purpose is to capture synchronised, high-resolution stereo image pairs of a calibration target under controlled exposure and focus conditions. These images form the essential dataset for calibrating the geometric and optical parameters of the stereo camera setup, enabling the accurate construction of disparity maps.

The relatively new Raspberry Pi Camera Module V3 can be controlled by Python software libraries like `libcamera` or `Picamera2`. `Libcamera` is an open-source camera stack designed for various platforms, featuring a core user space library supported by Linux kernel application programming interfaces (APIs) and drivers. It simplifies the complexity of embedded camera hardware by offering an intuitive API and separating untrusted vendor code from the open-source core (“`libcamera`”, n.d.). `Picamera2` is built on top of the open source `libcamera` project by directly using the Python bindings supplied by `libcamera`. This is done to simplify the experience when using Raspberry Pi applications compared to `libcamera`’s own bindings (*The Picamera2 Library*, 2022). This project will use `Picamera2` to control the two Raspberry Pi Camera Module V3’s due to the added simplicity when working with all Raspberry Pi components.

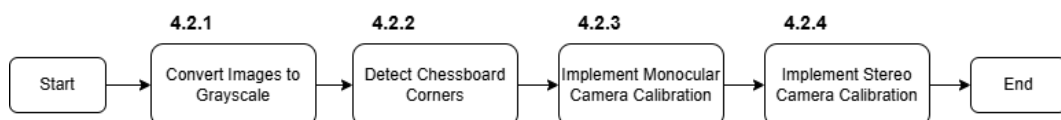
The process starts by initialising the two cameras and sets them up for high-resolution still captures. The cameras are set to capture at their maximum resolution of 4608 x 2592 pixels. This is done to maximise the resolution for not only the future steps during the calibration process, but also for the end goal of applying 3D DIC. The cameras have a 2-second delay before they can be used to allow the auto-exposure and white balance to stabilise. This is implemented to prevent early-frame inconsistencies during the stabilisation period.

Due to the camera system using two LED flood lights, the cameras experience a phenomenon known as the strobe effect. This effect is caused by the misalignment of how vast the camera shutter opens and closes, as well as how fast the LED is turning on and off (“Crushing Photography”, n.d.). The alternating cycle of the LED turning on and off is caused by it being powered by a 50 Hz alternating current, as found in South Africa. This means that the LEDs flicker a total of 50 times per second. To compensate for this effect, Picamera2 has built-in functions that are able to handle this effect. The first function is called `AeFlickerMode()` which must be set to 1. This indicates to the library that the code will utilise functions to combat the flickering. The next function is called `AeFlickerPeriod()` and indicates to the system what the desired flicker period is. Due to the LEDs flickering 100 times per second the period must be 5 000. This indicates that the flicker period is 10 ms. The final setting that must be changed is the cameras `ExposureTime`. This helps with aligning the cameras shutter speed with the LEDs flickering rate. The exposure time must be a multiple of the shutter speed (1/50s). Testing the camera rig indicated that the `ExposureTime` must be set to 10 000.

After the cameras have been successfully initialised, they remain ready until all the photos have been successfully captured, or the program is terminated. The cameras capture images in the RGB (Red, Green, Blue) colour space format, while OpenCV operates by assuming the BGR (Blue, Green, Red) colour space format. The image arrays must thus be converted to account for this, using the function `cvtColor()`. The output of this function will be an image array that is saved in the BGR colour space format.

## 4.2 Camera Calibration

In this section, the four steps necessary to calibrate a stereoscopic camera system will be discussed and explained. For this project, we will focus on calibrating the camera system using a planar calibration target. Various types of calibration targets can be used, such as chessboard patterns, circle grids, ArUco markers, and Charuco targets. The project will focus on using a chessboard pattern to calibrate the camera system and will follow the flow as indicated in Figure 9.



**Figure 9: Stereo Camera Calibration Flow.**

### 4.2.1 Convert Images to Grayscale

It is a general practice to convert images to grayscale when calibrating cameras. This is done to simplify corner detection, reduce computational complexity and improve the robustness to colour and lighting variations. The chessboard corner detection algorithm relies on intensity gradients captured in grayscale, making colour irrelevant in this situation. This conversion is done for each image using the function `cvtColor()`, which takes the captured image in the BGR colour space and converts it into the grayscale colour space, as seen in Figure 10.

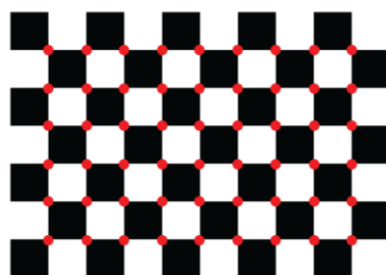
```
grayscale_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

**Figure 10: OpenCV Function Implemented for Grayscale Conversion.**

### 4.2.2 Chessboard Corner Detection

The inner corners of the chessboard pattern calibration target will be used to calibrate the camera system. To locate them, OpenCV provides a function called `findChessboardCorners()`. The input parameters of this function include an image, the pattern size of the calibration target, and various operational flags to modify the function's behaviour. The function then outputs an array containing the detected corners in a specific order (row by row, from left to right within each row).

The image used by the function to locate the corners must be an 8-bit image (grey or colour). The pattern size of the calibration target refers to the grid size of the inner corners of the chessboard. This parameter should be provided in the format (width x height) to be correctly input into the `findChessboardCorners()` function. To determine the number of inner corners of a chessboard calibration target, count the points where the black squares touch each other. Figure 11 illustrates the location of the inner corners of a chessboard grid. This chessboard has a 10 x 7 grid with the inner corners marked as red dots. These dots are located at the points where the black squares touch each other. The parameter `pattern size` for this calibration target will be given as 9 x 6. The easiest way to determine the pattern size parameter for a chessboard calibration target is to subtract one from the width and height.



**Figure 11: Location of Inner Corners of the Chessboard Target.**

*Source: "3nYBxKgncpGoxWARhUYIDSCyy", 2025*

The final parameter that must be specified is the flags, which indicate the behaviour of the function. The names of the flags and their effects on behaviour are provided in Table 2.

**Table 2: Flags Provided by the Chessboard Corner Detection Function.**

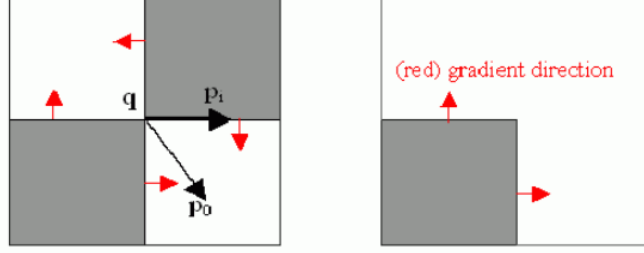
Flag	Behaviour Change
CALIB_CB_PLAIN	The image is taken as is, and all other flags are ignored.
CALIB_CB_FAST_CHECK	Run a fast check to determine whether chessboard corners are found.
CALIB_CB_NORMALIZE_IMAGE	Normalise the image gamma before applying thresholding.
CALIB_CB_ADAPTIVE_THRESH	Utilise adaptive thresholding to convert the image to black and white.
CALIB_CB_FILTER_QUADS	Use additional criteria to sort out false quads extracted at the contour retrieval stage.

The flags are an essential step to ensure that corner detection is reliable. The CALIB\_CB\_PLAIN parameter will not be used because it processes the image as is and does not aid in corner detection. It skips robust binarisation and dilation, reducing the function's tolerance to real-world issues. Binarisation is the process of converting a grayscale image into a binary image. Dilation is a morphological operation that closes small gaps that appear after binarisation, helping to detect small or blurred squares. The CALIB\_CB\_FAST\_CHECK parameter will be used to prevent the function from running unnecessarily long on images that do not contain a chessboard. The CALIB\_CB\_ADAPTIVE\_THRESH parameter will be used to assist in handling photos taken under uneven lighting conditions by implementing local thresholding. Local thresholding is a technique that computes a different threshold for each pixel based on its neighbouring pixels, making it effective for images with varying illumination.

The CALIB\_CB\_NORMALIZE\_IMAGE parameter will be implemented to enhance the contrast in low-contrast images so that black and white squares separate better before applying binarisation. The CALIB\_CB\_FILTER\_QUADS parameter will be used to reduce clutter from non-board rectangles visible in the image, which speeds up neighbour linking by filtering candidates to those that look square-like.

After the initial corners have been detected, the `cornerSubPix()` function will be used to refine them to sub-pixel accuracy. This is a crucial step for the high-accuracy task of camera calibration. The exact corner position must be known more precisely than whole pixels. This is achieved by iteratively adjusting each corner's position to ensure the measurements are not limited to integer-level accuracy.

The function starts by taking a grayscale image and an initial set of corner coordinates as inputs. The search window is defined as a square window of size  $(2 \times \text{winSize} + 1) \times (2 \times \text{winSize} + 1)$ . This search window is centred on a corner  $q$ , as seen in Figure 12. Within the window, the image intensity gradient ( $\mathbf{DI}_{p_i}$ ) is computed at each pixel ( $p_i$ ) (“OpenCV: Feature Detection”, n.d.). These gradients are used to relate changes in intensity to subpixel shifts in corner position.



**Figure 12: Chessboard Corner Refinement Implementation on Corner.**

Source: “cornerSubPix”, 2025

To locate the exact corner, each displacement vector  $(q - p_i)$  from the true corner ( $q$ ) to points ( $p_i$ ) within the window should be orthogonal to the local gradient at  $p_i$ . This means that the algorithm aims to satisfy the condition where the displacement vector  $(q - p_i)$  is orthogonal to the image intensity gradient ( $\mathbf{DI}_{p_i}$ ), resulting in zero. This relationship is illustrated in Equation 4.1, where the orthogonality error ( $\epsilon_i$ ) for a pixel  $p_i$  within the local window is represented. The algorithm utilises a least-squares method, aiming to minimise the orthogonality error.

$$\epsilon_i = \mathbf{DI}_{p_i}^T \cdot (q - p_i) \approx 0 \quad (4.1)$$

The updated corner position  $q$  is obtained by solving Equation 4.1, yielding a new sub-pixel estimate of the corner within the search window. The algorithm sets the new centre of the neighbouring window to  $q$  and continues iterating until it stays within the set threshold defined in the `criteria` input parameter. This project uses two parameters under the `criteria` input parameter. These parameters determine the epsilon or maximum iteration values that the algorithm will follow when deciding when to terminate. The epsilon value is set to 0.001, meaning the process will stop when the centre point  $q$  moves by no more than 0.001 pixels. This helps to offset the downsides of using budget components and further the aim to achieve the best accuracy possible. A maximum of 50 iterations is set to prevent the algorithm from looping indefinitely if the epsilon value is not reached. After the algorithm has finished refining the corner locations, the camera calibration can be implemented.

### 4.2.3 Monocular Camera Calibration

The next step in the camera calibration process is to determine the intrinsic, distortion, and per-view extrinsic parameters of each camera. This ensures that each camera's internal optical properties are accurately interpreted and accounted for before proceeding to stereo camera calibration. OpenCV provides the necessary functions to complete monocular camera calibration. The primary function for this purpose is called `calibrateCamera()` and requires 12 input parameters. This primary function encapsulates the entire multi-stage calibration process and consists of multiple smaller functions used to complete the process.

The primary function calls the `calibrateCameraInternal()` function, which acts as the core for executing Zhang's calibration method. Zhang's method requires a simple, planar pattern to be observed from multiple different orientations. The architecture of the `calibrateCameraInternal()` function follows a sturdy two-stage process. First, the function conducts an initial parameter estimation. Then, it performs non-linear refinement to simultaneously adjust all parameters by minimising the overall error metric.

The initial parameter estimation has two possible avenues: either using the user-provided guessed intrinsic parameters or a linear solution derived from a homography. This project will not supply an initial guess for the intrinsic parameters and, therefore, will implement the second approach. It is assumed that the camera rig is planar because it is mounted on a levelled aluminium extrusion, which allows the closed-form initialiser used during the linear solution.

There is, however, a significant consideration to keep in mind when working with the Raspberry Pi Camera Module V3. Every time the camera uses autofocus, the intrinsic parameters must be recalculated. This is due to the physical changes in the camera's internal geometry when autofocus is applied. The camera module physically moves the lens elements within the lens barrel to achieve focus, which changes the focal length, shifts the principal point, and alters the distortion coefficients. To combat this, the camera system will follow the steps seen in Figure 13, locking the lens in position before starting the calibration process.

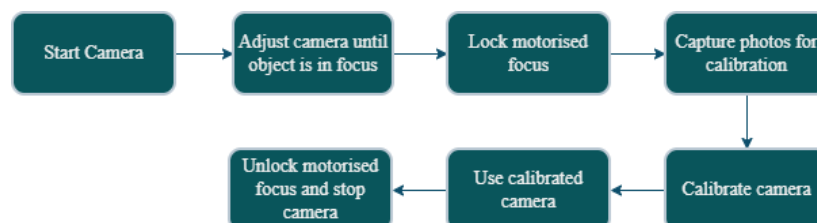


Figure 13: Process of Handling Autofocus during Image Capturing.

After the function has calculated the focal lengths, they are saved in the camera matrix, ready for further refinement. The next step is to initialise the extrinsic parameter estimation using the function `findExtrinsicCameraParams2()`. As with the intrinsic parameters, no initial guess for the extrinsic parameters is provided; instead, they are estimated from a homography matrix.

So far, the intrinsic and extrinsic parameters have been initialised and estimated. The final step in monocular camera calibration is to optimise these estimated parameters using a Levene-Marquardt (LM) optimisation loop to simultaneously refine both the intrinsic and per-view extrinsic parameters. The LM solver computes the per-point reprojection errors (also known as residuals) at each iteration. It aims to minimise the sum of squared residuals across all points and images. The solver compares the total squared reprojection error and the change in parameters to determine the next step. The decision is based on error improvement: if the new error is lower than the previous one, the parameter change is accepted and the damping factor is decreased. Decreasing the damping factor allows the solver to take larger, bolder steps. Conversely, if the new error exceeds the previous error, the parameter change is rejected, and the damping factor is increased to prevent larger steps. The solver terminates when either the parameter change or the error reduction becomes very small, indicating convergence. Alternatively, the process stops when the maximum number of iterations is reached.

#### 4.2.4 Stereo Camera Calibration

The next step in calibrating the stereo camera system is to implement the function `stereoCalibrate()` to obtain the extrinsic camera parameters. The intrinsic parameters will be fixed because they have already been optimised during monocular camera calibration. This will be achieved by passing the flag `CALIB_FIX_INTRINSIC` to the function, indicating that it should only optimise the relative pose between the cameras. The function implements a small Levenberg-Marquardt (LM) solver to find a consistent pose for each image, also accounting for distortion. Immediately after, the code calls the function `findExtrinsicCameraParams2()` to refine the pose. It is implemented in the same way as mentioned in the monocular camera calibration section.

The function provides six components that are the relative stereo pose parameters between the two cameras. The first three components are the rotation from camera 1 to camera 2 in radians. The other three components are the translation from camera 1 to camera 2 in millimetres. These components are the parameters computed per image. To ensure robustness, the median of each component is used as the initial global parameter guess before the stereo LM refinement. The parameter refinement is implemented in the same way as described in the

monocular camera calibration section. The `stereoCalibrate()` function concludes by returning the overall RMS reprojection error across both cameras.

## 4.3 Rectify Stereo Image Pair

Stereo rectification is a geometric transformation applied to a pair of stereo images. This transformation adjusts both images so they appear as if taken by two cameras with a perfectly parallel setup. The primary aim of this process is to ensure that the epipolar lines in both images become horizontal and parallel. Rectifying stereo image pairs makes it easier to find corresponding points between the two images, which is essential for 3D reconstruction. The primary purpose is to convert the process of searching for matching points from a 2D problem into a 1D problem. This is achieved by searching along the epipolar lines to identify the corresponding points.

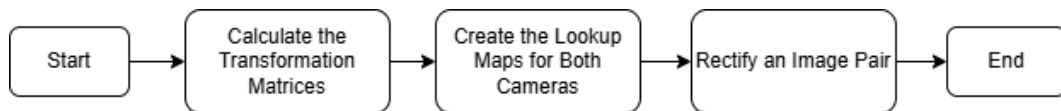
OpenCV provides a function, called `stereoRectify()`, that calculates the transformation matrices needed to align a pair of stereo images. This function takes the calculated intrinsic camera matrices and distortion parameters for both cameras, along with the rotation and translation matrices computed in the previous sections. The outputs generated from this function are two 3x3 rotation matrices for the left and right cameras. These rotation matrices define the exact rotation needed for each camera to make them parallel. The function also outputs two 3x4 matrices that represent the new projection matrices for the rectified cameras. These matrices project the 3D points onto the new, rectified image planes. The last output of the `stereoRectify()` function is a 4x4 matrix, called the disparity-to-depth mapping matrix. This is an essential component used during 3D reconstruction to convert a 2D pixel's disparity value into a 3D coordinate in real-world space.

The five output parameters are passed to the OpenCV function called `initUndistortRectifyMap()` to precompute two efficient lookup maps for each camera. These lookup maps are created for the x- and y-coordinates. This step is performed to ensure that the computationally expensive task of calculating the complex transformations needed to rectify and undistort an image is only executed once. If these maps were not created, complex transformations would need to be done for every image being rectified. To rectify an image, the OpenCV function `remap()` is called by passing the original image and the two lookup maps of the camera on which the image was captured. The function creates a new image by copying pixels from the original, distorted photo, using the corrected values obtained from the lookup maps.

It starts by examining a pixel coordinate that needs to be filled in the new image, for example, (10, 5). To determine the new pixel's x-coordinate, it looks at the lookup map for the x-coordinates at the position (10, 5). This provides a value for



the new x-coordinate of that pixel, for example, 12.7. The same is done to find the y-coordinate, but with the lookup map for the y-coordinate. If the value found is 8.2, the function now knows that to fill the new pixel at coordinate (10, 5), it must find the colour from the original, distorted image at the coordinate (12.7, 8.2). The non-integer coordinate leads to a new problem. There is no pixel at the coordinate (12.7, 8.2) in the original image. To account for this, the flag `INTER_LINEAR` is passed to the `remap()` function. This indicates that bilinear interpolation will be performed to determine the colour of the new pixel. The function repeats the task of looking up and assigning the new colour until all pixels in the new image are filled. The process of rectifying an image pair is illustrated in Figure 14.



**Figure 14: Image Rectification Process.**

## 4.4 Illustration of Disparity Map

The rectified images will be used to compute a disparity map of the scene observed by the cameras. This disparity map provides a visual representation of the disparity in various areas within the image scene. This creates a map that can be used to see the relative distance of objects from the camera system. This is done by implementing a Block Matching (BM) algorithm. This advanced stereo matching algorithm computes the disparity by finding the corresponding points between the left and right rectified images. For each pixel in the left image, the BM algorithm searches for the matching pixel in the right image along the same epipolar line. The displacement between the two matching points is the disparity, and is inversely proportional to the depth. OpenCV provides a total of four BM algorithms. The four methods, from fastest to slowest, are 3WAY, HH4, SGBM and HH. To ensure that the system employs a balance of accuracy and time efficiency, the Semi-Global Block Matching (SGBM) method will be implemented

OpenCV provides a function called `StereoSGBM_create()` to create the SGBM matcher, which can compute the disparity between an image pair. The `StereoSGBM_create()` function accepts the parameters used to develop the BM algorithm and, in turn, creates an object that can be triggered to start the SGBM algorithm. The parameter `minDisparity` has a value of 0, indicating that the algorithm will begin its search by assuming pixels can be at the same horizontal position. The parameter `numDisparity` indicates the total range of disparity values that the algorithm will search for. This parameter must be divisible by 16 to return an integer value in future calculations. The parameter was set to 512, allowing the algorithm to calculate the disparity of objects close to the camera system, albeit at the expense of increased computational time. The `blockSize` parameter is the side length of the pixel window used for matching. The algorithm compares a 5x5 block of pixels from the left image to the right, not just a single

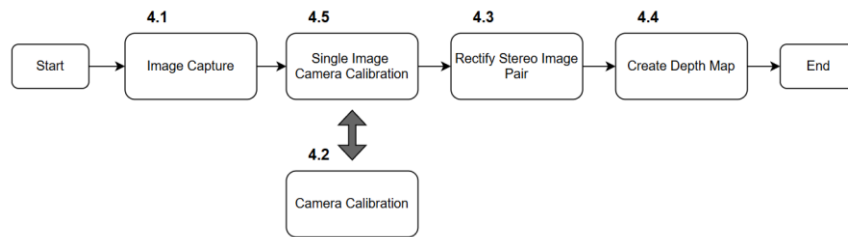
pixel. This ensures the robustness of the algorithm in finding matching areas and requires that the number be odd. The algorithm has two penalty parameters,  $P1$  and  $P2$ . These parameters help control the smoothness of the final disparity, especially when  $P2 > P1$ .  $P1$  is the penalty for small disparity changes between adjacent pixels, and  $P2$  is the penalty for significant, sudden disparity changes. The values for  $P1$  and  $P2$  were found iteratively and are 8 and 24, respectively. The final two parameters are the `uniquenessRatio` and `mode`. The `uniquenessRatio` ensures that the match is only accepted if it is unique, determined by the percentage improvement of the best match over the second-best match. This project will utilise a 5% improvement requirement after iteratively determining the best value. The `mode` parameter will be `STEREO_SGBM_MODE_SGBM`, which indicates that the SGBM algorithm will be implemented.

The disparity will be computed by triggering the SGBM algorithm, passing the grayscale image pair and scaling the output to its true pixel value by dividing it by 16 to reverse the internal multiplication done by the OpenCV function. The final step before the disparity map can be created is to normalise the disparity data into a standard 8-bit grayscale image. This is done using the OpenCV function `normalize()`, which accepts the computed disparity and normalises it between its minimum and maximum values. This is done by passing the parameter `NORM_MINMAX` and indicating the minimum and maximum values of the output image as 0 and 255, respectively.

## 4.5 Single Image Camera Calibration

This section will walk you through the implementation of a stereo camera calibration technique using a single 3D calibration plate. Inspiration was taken from the LaVision camera system and its implementation of stereo camera calibration using only a single image pair. The LaVision software DaVis automatically detects the dots on the 3D calibration plate and uses them to calibrate the system. The conventional planar calibration targets, like the chessboard used in the previous section, require taking several views to combat the limitations of implementing Zhang's method (Zhang & Research, 2000). According to Zhang's calibration method, the planar calibration target must be viewed from multiple orientations and distances to calculate the intrinsic parameters of a camera. This means that single-view camera calibration will not be possible when using a planar calibration, due to it not being viewed from multiple angles. Using a 3D calibration target provides more information for calibration. The single image camera calibration process discussed in this section

will replace the camera calibration process that were introduced in Section 4.2 as illustrated in Figure 15.



**Figure 15: Single-Image Calibration Places within the Existing Flow.**

### 4.5.1 Preprocess Images

The captured images require preprocessing to extract all the necessary features while remaining robust in many situations. All preprocessing steps are grouped into a custom function called `preprocess_image()`, which accepts the captured images. The images captured must once again be converted into the grayscale colour space. This is done because thresholding and circle detection operate on grayscale images, instead of colour images. Like in the previous methods, the conversion will be done using the `cvtColor()` function. The converted images are then further processed by applying the function `GaussianBlur()` to smooth the image and prevent random bright pixels from being falsely identified as dots. Multiple other blurring methods are available from the OpenCV library. These methods can be seen in Table 3 with a short explanation on how they work and their downsides.

**Table 3: Blurring Methods Provided by the OpenCV Library.**

Method	How it works	Downside
Averaging Blur	Replaces each pixel with the average value of all pixels in a square area.	Tends to create box-like artefacts and is not good at preserving edges.
Median Blur	Replaces each pixel value with the median value of its neighbouring pixels.	Significantly slower than a Gaussian blur, it creates regions of uniform intensities with block sharp gradient changes, which is not ideal for detecting circles. It can distort the shapes, leading to false classifications.
Bilateral Filter	Replaces each pixel with an average value of nearby pixels that are also similar in brightness.	It is significantly slower than the other methods, requiring increased computational cost. Its edge-preserving feature prevents the smooth gradient needed by the circle detection function.

After testing the different blur methods and comparing their effect on the circle detection function, it was found that the Gaussian blur is the best option for this project. The averaging and median blur methods do not preserve the circle's edges when passed to the circle detection function, as discussed in Section 4.5.2. The bilateral filter method does not work with the circle detection function, as mentioned in Table 3. The Gaussian blur accepts two parameters: the kernel size and a sigma value. The kernel size controls how many pixels are averaged together, controlling the spread of the blur. After testing, the kernel size was determined to be 5x5, meaning 25 pixels will be averaged together. The sigma value was set to 0, allowing the function to automatically calculate the optimal sigma value for the given kernel size.

The next step in preprocessing the image for circle detection is to apply thresholding to the blurred image. This is done to convert the blurred grayscale image into a pure black and white image. It simplifies the image by removing all ambiguous grey values by forcing all pixels to be either black or white. This is accomplished by determining whether a given pixel is brighter or darker than a given threshold. When the pixel is brighter than the threshold, it is set to a value of 255, indicating that it is white. Conversely, if it is darker than the threshold, it is set to a value of 0, indicating that it is black. A clean, high-contrast image is created where the circles are clearly distinguished from the background. To ensure the circle detection is robust against varying lighting conditions, the adaptive thresholding method is implemented. Adaptive thresholding is chosen above other methods, like global thresholding and Otsu's binarisation, due to its ability to apply thresholding to multiple smaller local areas. This is better because it does not rely on one brightness value to work for the entire image. Adaptive thresholding is a more robust solution and can work even if the image has shadows or bright glares on it.

To implement this feature, the function `adaptiveThreshold()` is called and receives the blurred images as input. The function also receives additional parameters needed to optimise the function for this use case. `Max value` indicates the value that will be assigned to pixels greater than the threshold. The local threshold of an area is calculated using a Gaussian-weighted average of the neighbours, indicated by passing the flag `ADAPTIVE_THRESH_GAUSSIAN_C`. The thresholding rule is set to binary using the flag `THRESH_BINARY`, indicating that values less than or equal to the threshold will be set to 0 (black). The neighbourhood size that will be used to calculate the local threshold is set to 5, indicating a 5x5 pixel area. This was determined through testing with values ranging from 3 to 101. A small value, for example, 3, is susceptible to noise and could lead to an unstable threshold. Larger values, like 101, can fail to adapt to lightning changes and act more like a global threshold.

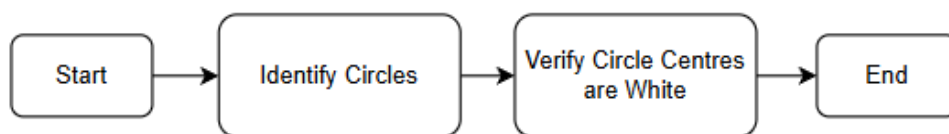
The last part in preprocessing is to clean the images by implementing morphological operations. These image preprocessing techniques modify the shape and structure of objects within an image. This is done by sliding a kernel across the image and deciding whether the pixel at its centre should be changed. The decision is based on a set rule, either erosion or dilation. The rule of erosion is to set a pixel white only if all pixels under the kernel are also white. Thus, the centre pixel is eroded and turned black even if only one pixel in the kernel is black. The rule of dilation is the inverse of erosion, meaning that the centre pixel is set to white if at least one pixel under the kernel is white. The function used to achieve these processes is `morphologyEx()`, provided by the OpenCV library. This function takes the image after adaptive thresholding has been applied and, depending on the flag, applies either erosion or dilation. The two flags that indicate what operation is applied are `MORPH_CLOSE` and `MORPH_OPEN`.

In this project, the code first applies the function `morphologyEx()` with the `MORPH_OPEN` flag. This is done to clean the background by removing small, isolated white specks from the black areas. These white specks are due to any remaining noise and are removed by erosion. After erosion, the function performs dilation to restore the circles to their original size without the presence of white specks. The `MORPH_OPEN` flag indicates erosion followed by dilation.

The next step is to apply the `morphologyEx()` function again, but this time passing the `MORPH_CLOSE` flag to indicate that dilation will be done, followed by erosion. This step is necessary to fill in any black holes that might appear inside the circles.

### 4.5.2 Circle Identifier

Circle identification will be handled in a custom-made function called `detect_dots_hough()`. This function identifies the circles of the calibration plate and verifies that all detected circles have a white centre. Figure 16 indicates the flow of the function `detect_dots_hough()`.



**Figure 16: Flow of Circle Identification Function.**

The `detect_dots_hough()` function implements the OpenCV function called `HoughCircles()` to identify the circles. This function accepts seven parameters to identify the circles, most of which require fine-tuning to detect dots at various distances. To simplify the implementation of the circle finding in this project, the circle identification will be conducted on the same 3D calibration plate at a constant distance of 400 mm. This ensures that the circles of the calibration

plate will remain approximately the same size, regardless of the plate's orientation. Fine-tuning of the parameters was done using a custom-written Python script that can be found at the following [link](#) to the GitHub page. This script took a preprocessed image and provided a live GUI to tune the parameters until all circles were detected. The parameters are provided in Table 4, along with their explanations and fine-tuned values that were passed to the `HoughCircles()` function.

**Table 4: Parameters Passed to HoughCircles Function.**

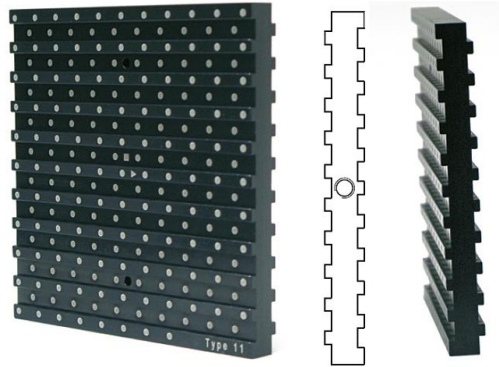
Parameter	Explanation	Value
<code>dp</code>	Inverse ratio of accumulator resolution to image resolution.	1
<code>minDist</code>	Minimum distance between the centres of the detected dots.	25
<code>param1</code>	Upper threshold for the internal Canny edge detector.	50
<code>param2</code>	Accumulator threshold for the centre detection.	15
<code>minRadius</code>	Minimum circle radius to detect.	9
<code>maxRadius</code>	Maximum circle radius to detect.	17
<code>method</code>	The detection method used.	<code>HOUGH_GRADIENT</code>

The second part of the `detect_dots_hough()` function is to verify that the detected circles are from the 3D calibration plate and to filter out false positives. This is done by calling the custom function `is_center_white()`, which verifies that the detected circles have a white centre as expected. It takes a coordinate of the detected circle's centre and verifies whether it is brighter than the threshold. When the circle's centre is greater than the selected threshold of 180, the function will return `True`, indicating that this is a circle. When false is returned, that circle is discarded from the detected circles. The `is_center_white()` function will iterate through all the detected circles in the image until all the false positives are cleared. When this is completed, the function `detect_dots_hough()` will return a variable that contains a list of all the coordinates of the circle centres that have been confirmed.

### 4.5.3 Assigning Dots into Columns

The next step in implementing the custom calibration method is to sort the detected circles into columns based on their coordinates. The calibration plate is

shown in Figure 17, illustrating the multiple layers that provide its 3D characteristics.

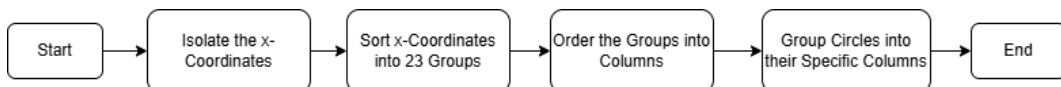


**Figure 17: LaVision's 3D Calibration Plate.**

Source: "c6e347fc-5772-4522-a501-af7b22bc4204", 2025

The circles are arranged into two grids, one being elevated above the other. To ensure that the columns are correctly assigned, the calibration plate will always be captured in one specific orientation. This will ensure that the system is more robust when defining columns and does not rely on detection markers to know its orientation. The calibration plate will always be captured in the same approximate orientation, with the white triangle located North-East of the centre square. In this orientation, the calibration plate will have circles oriented into 23 columns and 23 rows. Since this will remain constant for the orientation in which the plate is captured, the code will assign dots accordingly.

To assign the circles to the correct columns, a custom function called `cluster_into_columns()` was created, seen in Figure 18. This function works by isolating the x-coordinates of the circles' centres and then using K-means clustering to create exactly 23 groups based on these coordinates. The groups created during the K-means clustering are in an arbitrary order and must be ordered into the correct column order before they can be assigned to one of the two grids. The function ends by saving the circle's full coordinates into the correct column list based on the sorted mapping.



**Figure 18: Column Clustering Function Flow.**

To isolate the x-coordinates, the detected circle centres are taken, and only the x-coordinates are saved into a NumPy array. K-means clustering is achieved by first configuring the `KMeans()` function. This tells the algorithm to find exactly 23 clusters by passing the parameter `n_clusters=23`. To ensure that the clustering is more accurate, the entire algorithm will be run 10 separate times with different random starting points. This is done by passing the `n_init=10` parameter to the function, allowing it to pick the best one out of the ten runs automatically. After the function is configured, the `fit_predict` method is

called, and the x-coordinates array is passed into it. This method runs the algorithm in two steps: first, it “fits” the model by determining the 23 best x-positions for the column centres from the data. Second, it “predicts” by going back through every circle and assigning it the ID label of the cluster it is closest to. The final result is stored in an array which contains the cluster ID for every circle.

The K-means algorithm identifies the 23 groups of circles but does not determine which groups correspond to which columns. To address this, the function `np.argsort()` is used. It will sort the list of group IDs based on the x-coordinates, from smallest to largest. An ID-to-column dictionary is created to map the ID of the 23 groups to the corresponding column of the calibration plate. The final step in the `cluster_into_columns()` function is to group the circles into their closest column and refine where necessary. This is achieved by looping through every circle and adding the full circle centre’s coordinates to its column list. The average x-coordinate of the circles in a column is calculated and passed to the custom-created function called `refine_column_assignment()`. After the columns are finalised, the code loops through each column and sorts the dots within it based on their y-coordinates, in a top-down manner.

#### **4.5.4 Sort Columns into Two Grids**

The 3D calibration plate can be divided into two grids to utilise its 3D characteristics. These two grids have a depth separation of 1 mm between them, as seen in Figure 17. This depth separation will be used to assist the single-image calibration process. Grid A is identified by the circles on the plane furthest from the camera system and corresponds to all the odd-numbered columns. Grid B contains all the even-numbered columns and is located closest to the camera system, with a 1 mm offset from Grid A.

The sorting of the columns is achieved by utilising the custom-created function `assign_grids_and_sort()`. This function iterates through all 23 columns and checks if the column number is odd or even. If it is odd, all the circles’ coordinates in that column are saved into a list created for Grid A. If the column number is even, the coordinates are saved into the list for Grid B. Once it has iterated through the columns, the lists are converted into arrays and returned as outputs. The columns’ internal order is kept consistent; therefore, all circles are in the correct locations.

#### **4.5.5 Calibrating Cameras**

The Zhang’s calibration method requires that the calibration target be viewed from multiple orientations. Because the suggested single-image calibration method will not capture the target in various orientations, the intrinsic parameters will be guessed. The flag `CALIB_USE_INTRINSIC_GUESS` will be passed to the `calibrateCamera()` function. The camera calibration functions as



discussed in Sections 4.2.3 and 4.2.4, with the only difference being that the two grids will be used as if they were from two separate images. The number of circles in each column will be made equal by trimming the outer layer of circles. The `stereoCalibrate()` function will be implemented in the same way with no changes needed. The function will return the same value types as when running the chessboard calibration method.

## **4.6 Chapter Summary**

This chapter discussed the design for the software created to ensure the operation of the stereo camera system. It explained how the images will be captured, preprocessed and used to calibrate the cameras. The requirements needed to create a disparity map were explained and also how the single image calibration method are created. The next chapter will walk through the processes of implementing the code and the analysis of the results.

## 5 Result and Analysis

This section presents the results and analysis obtained from conducting tests and experiments using the camera system. Attention was given to the safety of all people and equipment during the testing phase. Refer to Appendix C.2 to find more information on the safety aspects considered and the safety report compiled for working in the laboratory. All testing was conducted in a safe and secure environment.

A mouse, keyboard and monitor were connected to the RPi 5 to control and navigate the implemented code. The camera system could have also been controlled from another device using the Raspberry Pi Connect service. This is a secure remote access service provided by Raspberry Pi to control most Raspberry Pi models from a web browser.

### 5.1 Calibration Target Preparation

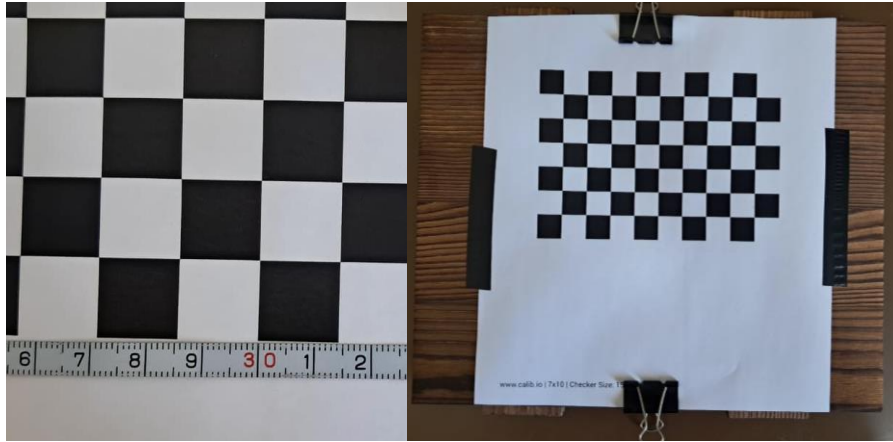
Accurate camera calibration relies heavily on the condition of the calibration target. It is crucial that the chessboard calibration target is accurately printed and that the dimensions of the squares are verified. A second aspect that can influence the calibration accuracy is how flat the calibration target is. If the paper that the chessboard is printed on is not flat, it can lead to inaccurate calibration. It is essential to mount the chessboard pattern on a rigid, flat surface to prevent any distortion of the pattern. The website [www.calib.io](http://www.calib.io) was used to generate the chessboard pattern. It has a pattern generator feature that allows complete control of the specification of your calibration pattern. The specification of the chessboard pattern used during the test in this project can be seen in Table 5.

**Table 5: Chessboard Pattern Specifications.**

Parameter	Target Type	Board Width	Board Height	Pattern Rows	Pattern Columns	Checker Width
Value	Checkerboard	210 mm	297 mm	7	10	15 mm

To verify that the pattern is accurate after printing, multiple measuring tapes were used to measure the dimensions of the checker pattern. It was determined that the pattern dimensions were consistently 14.5 mm, as seen in Figure 19, leading to an adjustment in the square size parameter before calibration. The pattern was kept flat by mounting it to a rigid, flat wooden slab. The paper that the pattern is printed on was mounted tightly to the wooden slab and fastened with tape, glue stick, and metal clips. This ensured that the pattern stays flat during testing and

does not move when manoeuvred around. Figure 19 demonstrates how the chessboard pattern is fastened to the wooden slab.



**Figure 19: Chessboard Square Measured (left), Chessboard Fastened to Wooden Slab (right).**

When analysing the 3D calibration plate, it is crucial to ensure the background is black. This helps the algorithm to determine the correct dots and reduce the number of falsely identified circles. The calibration plate was also wiped with isopropyl alcohol to ensure it is clean and in perfect condition. The 3D calibration plate is of known dimensions as it is machined to extreme accuracy. A measuring tape was, however, used to validate that the dimensions are equal to those provided by the manufacturer. Reusable putty adhesive was used to mount the 3D calibration plate securely to a black background. This protected the plate from permanent damage, kept it fixed in position when capturing calibration photos and allowed it to be moved around into different orientations.

## 5.2 Camera System Setup

The camera system setup included the assembly of the components into the final form, as indicated in Section 3.7. The tripod was expanded and placed on an elevated surface. This made the process of moving the calibration plate around easier. The 3D printed connector was attached to the aluminium extrusion with bolts and T-nuts. The tripod centre mounting bolt was fastened to the threaded insert in the connector piece. The Raspberry Pi 5, in its case, was mounted to the centre of the aluminium extrusion, followed by the two cameras. They were spaced at an equal distance away from the centre. The two LED lights were connected to the bottom of the extrusion and the electronics enclosure, which is mounted to the centre pipe of the tripod. All remaining connections between the RPi 5 and peripherals were made before levelling the camera system using the added bubble level. The final step is to connect all the power cables and log in to the RPi 5.

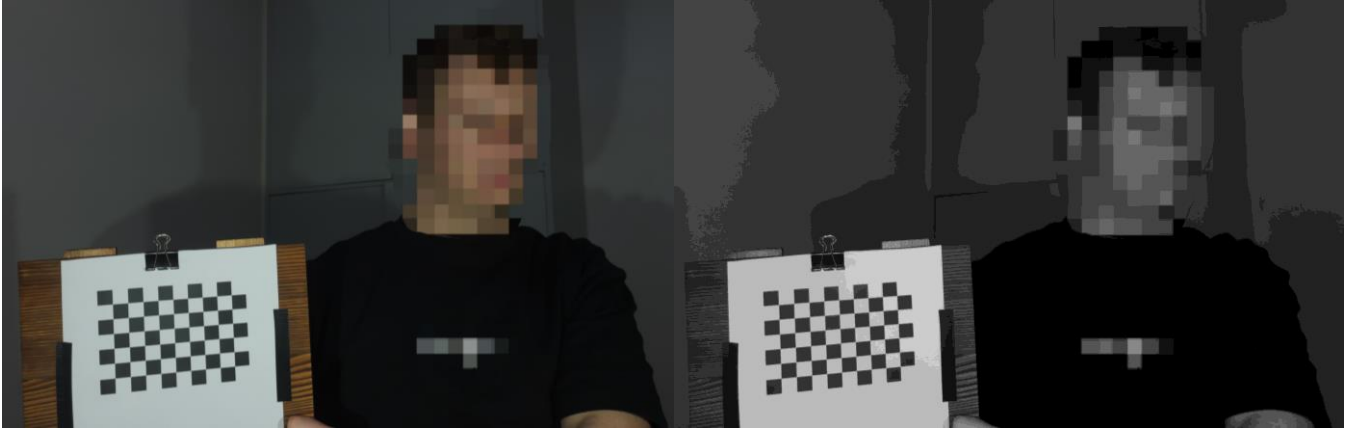
## 5.3 Image Capture

The images are captured by implementing the code designed in Section 4.1. Image capturing begins by prompting the user to specify the number of image pairs to be captured. The number of image pairs that must be captured to ensure a good calibration result depends on the quality and placement of images. After the number of image pairs is selected, both cameras are started, and a GUI window with a live preview is displayed. The preview features a green centre cross on both previews. This ensures that the cameras are pointed to the same point in the image frame. The GUI window displays the number of image pairs captured and the number remaining, along with the options to control camera settings. Autofocus on both cameras can be enabled, and it will focus on the area around the centre cross. The white balance and autofocus can be locked to ensure that there is no change in camera parameters while capturing calibration images. A three-second timer between each capture ensured adequate time to reorient the calibration target.

During testing, 30 image pairs were captured at various orientations and distances. It was ensured that the calibration target was visible in both images in each capture. The orientations were changed to ensure that the entire image frame was covered with the calibration target. Meaning that the target was moved around the image frame during the captures. After the captures have been completed, the image pairs are saved into a new folder, ready to be used for calibrating the camera system. Images are saved as a .png file and named `leftxxx` or `rightxxx`, depending on which camera captured it, with `xxx` representing the image number. The folder also contains a log file of the captures.

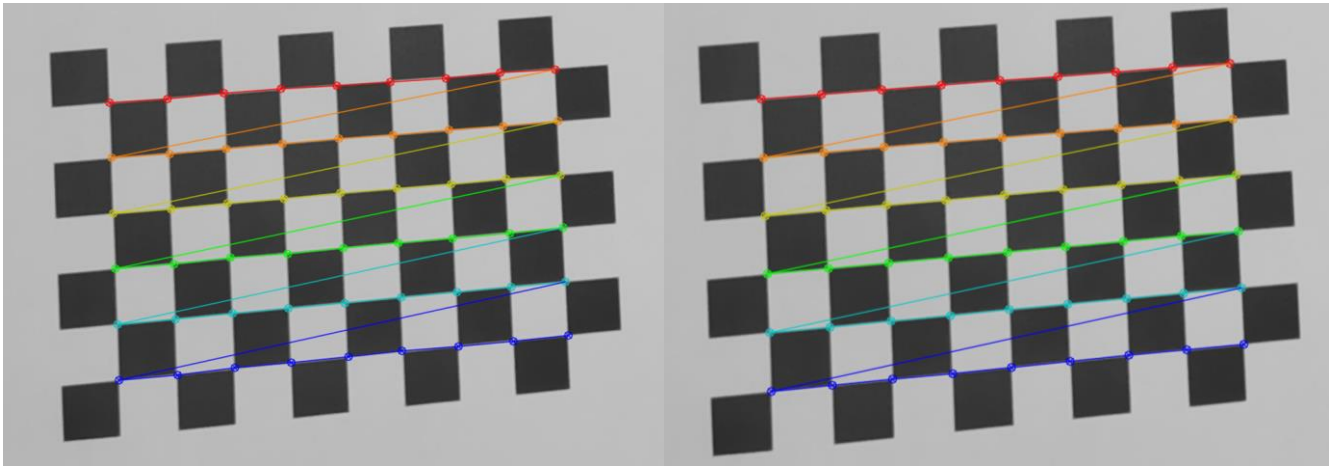
## 5.4 Camera Calibration

The camera calibration process is completed by implementing the code designed in Section 4.2. The code processes the images captured and returns the calibration data in a .npz file format. A human-readable .json file was also created with a calibration report that summarises the information in a .txt file. The first step that the camera calibration code does is to convert the images to grayscale. This conversion can be seen in Figure 20, with the original image shown for comparison.



**Figure 20: Original Image (*left*), Grayscale image (*right*).**

Figure 21 illustrates the initial corner detection done by the function `findChessboardCorners()` on this image. It successfully detected all 54 corners of the pattern, which are drawn onto the grayscale image in colour using the OpenCV function `drawChessboardCorners()`.



**Figure 21: Initial Corner Detection (*left*), Refined Corners (*right*).**

The initial corner detection accuracy is enhanced by passing the function `cornerSubPix()`. Noticeable improvements can be observed in the corner precision across the pattern. Figure 21 illustrates the refined corner detection, particularly in the first and last rows of corners. After conducting corner detection and refinement on all 30 image pairs, the corner locations were used to perform both monocular and stereo camera calibration. This calibration yielded the intrinsic and extrinsic parameters of the camera system, which fully define its properties, as seen in Appendix A. The intrinsic camera matrices for cameras 1 and 2 are provided in Equations 5.1 and 5.2, respectively. The focal lengths and principal points of both cameras are almost identical. This is expected due to the use of similar cameras. The horizontal focal length ( $f_x$ ) differs by 0.98%, while the vertical focal length ( $f_y$ ) varies slightly more, at 1.11%. The horizontal and vertical principal points differ by 0.27% and 4.56%, respectively. These

differences are due to the internal properties of each camera and are expected to vary slightly.

$$K_{camera\ 1} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 3433 & 0 & 2406 \\ 0 & 3407 & 1399 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.1)$$

$$K_{camera\ 2} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 3467 & 0 & 2413 \\ 0 & 3445 & 1335 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.2)$$

The lens distortion for each camera is given in the form of a vector containing both radial ( $k_1, k_2, k_3$ ) and tangential ( $p_1, p_2$ ) distortion coefficients. Comparing camera 1's coefficient (Equation 5.3) with camera 2's coefficient (Equation 5.4) shows that the left camera (camera 1) exhibits more distortion, particularly around the edges. This confirms the decision to rectify the images before using them for further analysis.

$$Distortion\ coefficients_{camera\ 1} = [-0.05, 0.68, 0.002, 0.008, -1.52] \quad (5.3)$$

$$Distortion\ coefficients_{camera\ 2} = [0.01, 0.28, -0.001, 0.008, -0.55] \quad (5.4)$$

The extrinsic camera parameters are defined by the rotation ( $R$ ) and translation ( $T$ ) matrices. These matrices describe how the right camera (camera 2) is rotated and translated in relation to the left camera (camera 1). Equations 5.5 and 5.6 present the rotation and translation matrices, respectively. The diagonal entries in the rotation matrix are close to 1, while the off-diagonal entries are near 0. This suggests a near identity matrix, indicating that the cameras are almost aligned. An angle of  $5.38^\circ$  is measured around the camera's y-axis, resulting in a slight inward tilt, which is essential for depth estimation.

$$R = \begin{bmatrix} 0.9956 & 0.0013 & 0.0937 \\ -0.0020 & 0.9990 & -0.0114 \\ -0.0937 & 0.0113 & 0.9955 \end{bmatrix} \quad (5.5)$$

The translation matrix is in vector form, with each value indicating the relative displacement between the cameras in metres. The baseline distance is shown as 215.9 mm and represents the horizontal displacement between the two cameras. To verify this distance, a measuring tape was used to measure the gap between the centres of the cameras. The measured distance was 215 mm, but due to possible human error, this value may have slight deviations from the actual distance. This represents a 0.4% difference and is considered negligible for the reasons previously mentioned. The vertical displacement is 1.7 mm, indicating that the right camera is marginally lower than the left. A 7.2 mm difference is observed in the Z direction, suggesting that the right camera is closer to the observed scene.

$$T = [-0.2159, -0.0017, 0.0072] \quad (5.6)$$

The essential matrix ( $E$ ), which encodes the rotation and translation between calibrated camera coordinates, is also computed and shown in Equation 5.7. The same relationship is provided in the fundamental matrix ( $F$ ), but in pixel coordinates.

$$E = \begin{bmatrix} 0.0001655 & -0.0072552 & -0.0016579 \\ -0.0130304 & 0.0024555 & 0.2156206 \\ 0.0017887 & -0.2158904 & 0.0026254 \end{bmatrix} \quad (5.7)$$

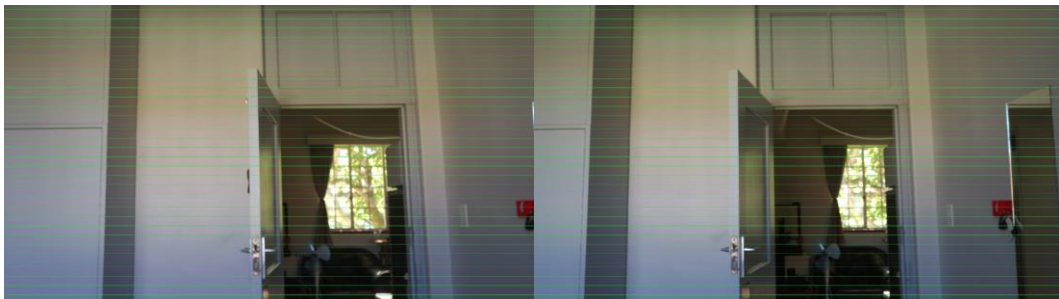
## 5.5 Disparity Map of Scene

To create a disparity map of a scene, another image pair was captured using the same camera settings as those used for calibration. This was done by employing the restore option during image capturing, which applies the camera settings used during calibration. Figure 22 shows the two images captured of a scene for which a disparity map will be created.



**Figure 22: Images Captured to Create a Disparity Map.**

The images must be rectified prior to calculating the disparity to correct for camera distortions. The rectification was performed using the function `stereoRectify()` and can be seen in Figure 23. The epipolar lines, drawn in green, will be used by the SGBM algorithm during the creation of the disparity map.



**Figure 23: Rectified Image Pair with Epipolar Lines.**



The disparity of the scene was calculated, and a disparity map was created using the software designed in Section 4.4. In Figure 25, the disparity of the scene can be viewed in grayscale. Lighter coloured parts represent objects with significant disparity, evident from closure objects. The darker areas represent objects further away from the camera system, which results in less disparity.



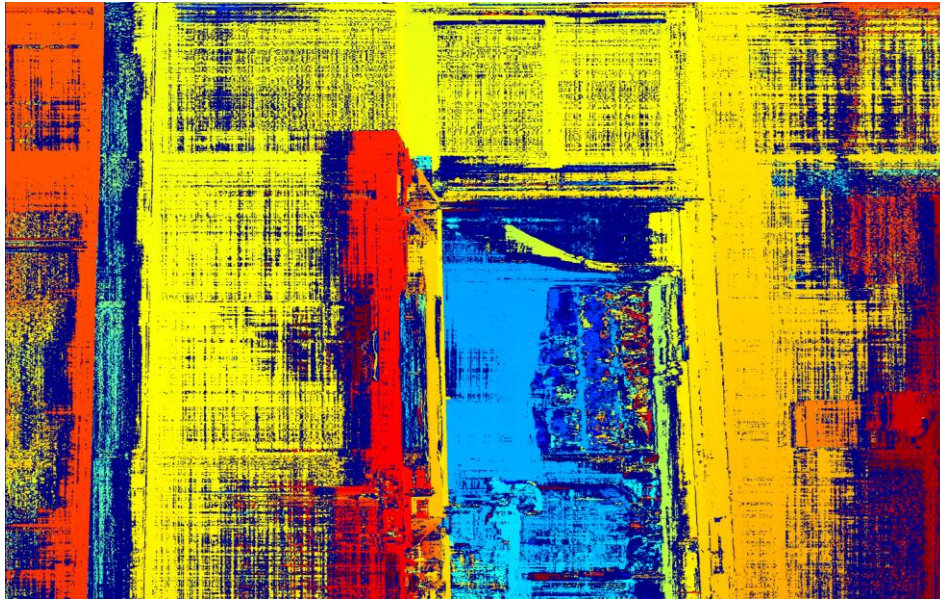
**Figure 25: Grayscale Disparity Map.**

The grayscale format makes it difficult to clearly distinguish between different levels of disparity. To address this, the OpenCV function `applyColorMap()` was used to convert the grayscale image into a colour gradient map of the disparity. The `JET` colour map type was employed, with red indicating high disparity areas and blue representing low disparity areas. Figure 26 illustrates the disparity range in the map. Differences in disparity levels are clearly visible in Figure 27, which shows the colour disparity map of the scene. The objects closest to the camera are shown in bright red, such as the side of the door and the cupboards on the left. On the right, it is clear how the gradient changes from red to orange and then to yellow. This is intuitive, as the wall appears to recede from the camera system. The wall where the door is mounted appears yellow and lies nearly in the middle of the disparity range. Shades of blue depict the back wall and the window within it. The darker blue in the window areas is due to the space beyond the window, while the light blue squares represent the window frame. The standing fan, being closer to the camera, appears in aqua blue, indicating a larger disparity. A cable running behind the door frame can be seen in the upper part of the door opening. The yellow and green shades accurately reflect the disparity of that area indicates.



**Figure 26: Jet Colour Gradient.**



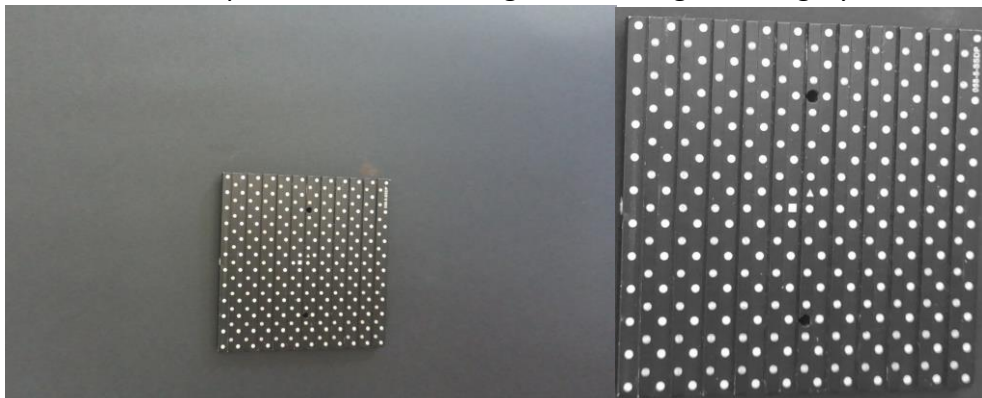


**Figure 27: Jet Gradient Disparity Map.**

The SGBM algorithm used to compute the disparity map struggles with large untextured surfaces, which causes artefacts like those visible on the walls. The disparity maps are not flawless. The large baseline of the camera system results in significant differences between points in the camera views, reducing the precision of the SGBM algorithm. The mirror on the right side of the image also presents a challenge for the SGBM algorithm because of its repetition of an existing part of the scene. Another flaw in the disparity map is the blue vertical strip highlighted on the side of the cupboard. Overall, the disparity map does provide a clear indication of the relative distance of objects from the camera system.

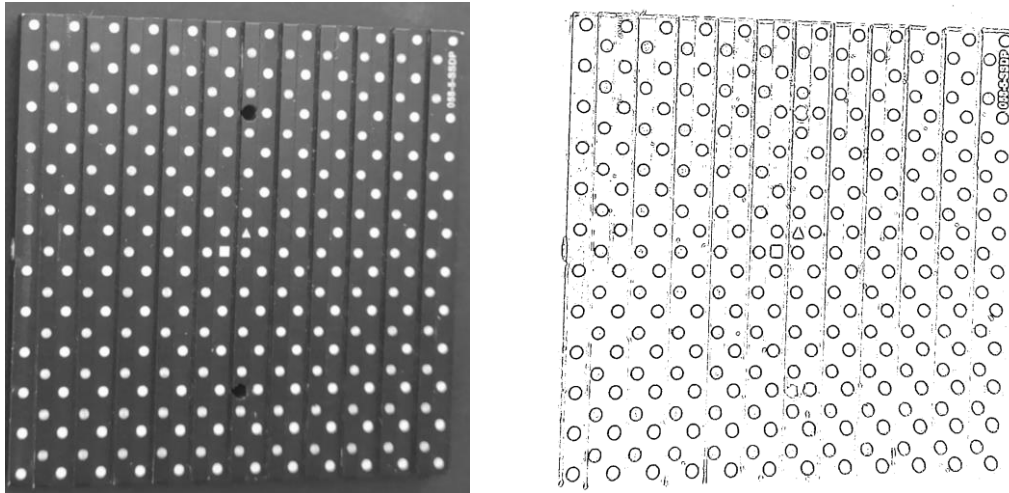
## 5.6 Single Image Calibration

The initial step in implementing single image stereo calibration is to capture an image pair and carry out the preprocessing steps described in Section 4.5. The process of capturing the image and converting it to grayscale follows the same method as in Section 5.2. One of the images demonstrating the single image calibration in this report can be seen in Figure 28, along with its grayscale version.



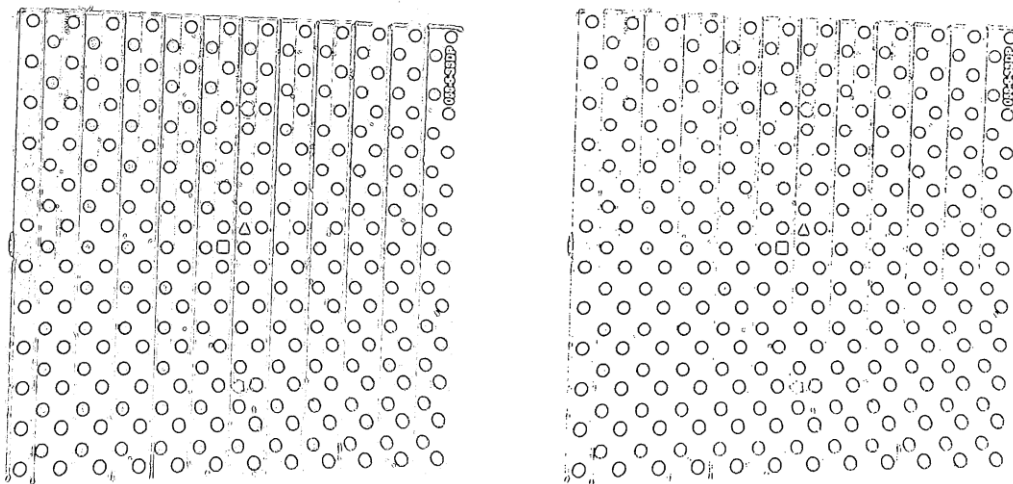
**Figure 28: Single Image of 3D Calibration Plate (*left*), Grayscale Conversion of Image (*right*).**

The images of the preprocessing steps are shown, cropped for easier visualisation of the effects. The next step in the process is to apply the Gaussian blur to the image, as shown in Figure 29. Adaptive thresholding produced an image with circles visible, displayed in Figure 29.



**Figure 29: Gaussian Blur Applied (*left*), Adaptive Thresholding Applied (*right*).**

The morphological opening and closing can be seen in Figure 30. A clear improvement in the image quality can be seen after each step of the morphological operations, and ensures reliable circle detection.



**Figure 30: Morphological Opening (*left*) and Closing (*right*) Applied.**

The circles were detected and assigned to Grid A or Grid B as shown in Figure 31. Grid A is displayed if the centre dots are red, while Grid B is shown if the centre dots are blue. The `is_center_white()` function worked effectively and prevented two holes from being misinterpreted as calibration circles.



**Figure 31: Detected Circles Organised into Two Grids.**

The calibration process was started by passing the locations of the circle centres to the monocular and stereo calibration functions, which were sorted into their correct columns. The results returned after the calibration were provided in a calibration report, as seen in Appendix F. However, the calibration did not yield accurate results. The focal lengths in both cameras remained the same, as the `calibrateCamera()` function did not change their values. The problems started at the principal point calculations, which were off by a factor of ten. The exact reason why they are incorrect is unknown and would require extensive research and testing, which is beyond the scope and timeframe of this report. The camera matrices can be seen in Equations 5.8 and 5.9, illustrating the significant differences seen in the principal points.

$$K_{camera\ 1} = \begin{bmatrix} 3433 & 0 & 206.6 \\ 0 & 3407 & 326.3 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.8)$$

$$K_{camera\ 2} = \begin{bmatrix} 3467 & 0 & 254.5 \\ 0 & 3445 & 409.3 \\ 0 & 0 & 1 \end{bmatrix} \quad (5.9)$$

The distortion coefficients of both cameras were also incorrect, especially considering the radial distortions, which can be seen in Equations 5.10 and 5.11.

$$Distortion\ coefficients_{camera\ 1} = [0.31, -3.53, -0.001, -0.014, 6.34] \quad (5.10)$$

$$Distortion\ coefficients_{camera\ 1} = [0.36, -4.39, 0.003, -0.007, -5.99] \quad (5.11)$$

The errors propagated throughout the calculations and became significantly large due to the inherent nature of matrix calculations. The translation vector ( $T$ ) indicated that the relative distances between the left and right cameras are in the magnitude of meters and hundreds of meters, as seen in Equation 5.12. Checks were conducted to ensure that the units used in calculations remained consistent

and did not lead to misinterpretation of the positions within the functions. The baseline calculated by the provided functions was 333 403.2 mm.

$$T = [-1.429467, 200.408503, 266.443451] \quad (5.12)$$

The significant differences between the values obtained from single-image calibration and those previously obtained using the chessboard calibration method cannot be precisely accounted for. The possible reasons for the significant differences are due to the calibration functions not being able to fully constrain the image space. The main reason why multiple images of a calibration target are captured is to allow the algorithm to compare what is observed in the image to the known truth. The lack of images prevents the algorithm from understanding how the cameras interpret the calibration target and cannot account for any distortions or imperfections. Although this is theoretically possible, as seen by its implementation in the LaVision DaVis software, it could not be accurately replicated during this project.

## 5.7 Chapter Summary

This chapter discussed the implementation of the software using the camera system. The entire process was documented, and images were used to see the results of each part of the process. The images were successfully captured and used to calibrate the camera system. The chessboard calibration yielded the expected results and where correctly implemented to generate intuitive disparity maps.

# Conclusion

The project set out to address the lack of an accessible, open-source stereoscopic camera system that enables accurate, low-cost 3D DIC analysis. This project achieved its aim by completing the three main objectives set out. The first objective that is completed is to design and construct a stereoscopic camera system that fits within an area of 1 m<sup>3</sup>. The aluminium extrusion is 1 meter in length, and the total height of the tripod is 0.7 meters. The width of the entire camera system is less than 0.4 meters, ensuring it fits within the 1 m<sup>3</sup> area specification. The second main objective has two parts that must be completed for the camera system to be considered a success. The first is to develop and implement a stereo camera calibration procedure that allows for accurate depth representation of an image scene. The second is to create a calibration procedure to be completed within 120 seconds. This was all successfully completed during the chessboard calibration procedure shown in Chapter 5, with each image being captured every 3 seconds. The calibration process ran for less than 15 seconds and yielded a stereo RMS error of 1.14 pixels. This indicates that objective 2 is fully met. The last objective is to create intuitive, user-friendly interfaces for performing tasks with the camera system, such as creating disparity maps. This was successfully achieved and demonstrated in Chapter 5.

The project will contribute to the body of scientific knowledge by providing an open-source stereoscopic camera system that can be used to research 3D DIC. It is, however, not limited to only 3D DIC. It can also be used for research in fields such as 3D reconstruction, gesture tracking, and environmental monitoring. The open-source nature of the project allows anyone to use the project files and contribute in meaningful ways. The budget-friendly design focus makes the project more accessible to more people across the world.

No dangerous situations were encountered during the project's lifecycle that posed an imminent risk to individuals' health and safety. This is due to closely following the safety plan provided in Appendix C.

The project was ultimately completed below budget and within the given timeframe, as discussed in Appendix B. The project, however, did not accomplish the additional aim of implementing a single-image camera calibration method. This aim is a perfect starting point for individuals to build on this project. The project is still regarded as a success due to its successful completion of all its main objectives.

# References

12.3MP Motorized Focus Camera Module for Raspberry South Africa. n.d. Available: <https://www.ubuy.co.za/product/539N3V4L0-arducam-raspberry-pi-hq-camera-with-motorized-focus-lens-12-3mp-imx477-camera-module-with-m12-lens-for-raspberry-pi-4-model-b-pi-3-b-and-pi-zero> [2025, October 17].

3nYBxKgncpGoxWARhUYIDSCyy. [n.d.]. [Online]. Available: [https://custom-images.strikinglycdn.com/res/hrscywv4p/image/upload/c\\_limit,fl\\_lossy,h\\_9000,w\\_1200,f\\_auto,q\\_auto/12625443/3nYBxKgncpGoxWARhUYIDSCyy.png](https://custom-images.strikinglycdn.com/res/hrscywv4p/image/upload/c_limit,fl_lossy,h_9000,w_1200,f_auto,q_auto/12625443/3nYBxKgncpGoxWARhUYIDSCyy.png) [2025, October 4].

Abouelnour, Y., Rakauskas, N., Naquila, G. & Gupta, N. 2024. Tensile testing data of additive manufactured ASTM D638 standard specimens with embedded internal geometrical features. *Scientific Data*. 11(1). DOI: 10.1038/S41597-024-03369-Y.

Anwar, A. 2022. *What are Intrinsic and Extrinsic Camera Parameters in Computer Vision? | Towards Data Science*. Available: <https://towardsdatascience.com/what-are-intrinsic-and-extrinsic-camera-parameters-in-computer-vision-7071b72fb8ec/> [2025, August 27].

Anzai, A. & DeAngelis, G.C. 2010. Neural computations underlying depth perception. *Current Opinion in Neurobiology*. 20(3):367–375. DOI: 10.1016/J.CONB.2010.04.006.

Arun KL. 2023. *Ultimate Guide to Single Board Computers (SBCs)*. Available: <https://thesecmaster.com/blog/what-are-single-board-computers-sbcs-and-why-you-should-buy-single-board-computers> [2025, August 03].

Butler, S. 2022. *What Is GPIO, and What Can You Use It For?* Available: <https://www.howtogeek.com/787928/what-is-gpio/> [2025, October 04].

Camera - Raspberry Pi Documentation. n.d. Available: <https://www.raspberrypi.com/documentation/accessories/camera.html> [2025, October 04].

cornersubpix. [n.d.]. [Online]. Available: <https://docs.opencv.org/4.12.0/cornersubpix.png> [2025, September 24].

Crushing Photography. n.d. Available: <https://crushingphotography.com/> [2025, October 24].

Depth Chapter 11. n.d. Available: <https://studylib.net/doc/14486472/depth-chapter-11> [2025, August 26].

DIC and test set up. 2018. [Online]. Available: [https://www.semanticscholar.org/paper/Using-3D-Digital-Image-Correlation-\(3D-DIC\)-to-CTOD-Samadian-Hertel%2C%2F7b83c5e970f1cde79cb85445b5342e518811551c/figure/1](https://www.semanticscholar.org/paper/Using-3D-Digital-Image-Correlation-(3D-DIC)-to-CTOD-Samadian-Hertel%2C%2F7b83c5e970f1cde79cb85445b5342e518811551c/figure/1) [2025, April 12].

Digital Image Correlation (DIC). n.d. Available: <https://www.lavision.de/en/techniques/dic-dvc/> [2025, August 08].

Dong, Y.L. & Pan, B. 2017. A Review of Speckle Pattern Fabrication and Assessment for Digital Image Correlation. *Experimental Mechanics* 2017 57:8. 57(8):1161–1181. DOI: 10.1007/S11340-017-0283-1.

Elmardi, O.M. & Khayal, S. 2024. Experimental Methods of Materials Testing in Tension and Compression. (July). DOI: 10.13140/RG.2.2.22657.88161.



- Fundamentals of Speckling for DIC*. 2023. Available: <https://www.youtube.com/watch?v=z9QHc5K4X4A&t=1s> [2025, August 07].
- Generic representation of the stereo geometry. 2017. [Online]. Available: [https://www.researchgate.net/profile/Tiago-Dias-7/publication/319928547/figure/fig3/AS:540556553973760@1505890133852/Generic-representation-of-the-stereo-geometry\\_W640.jpg](https://www.researchgate.net/profile/Tiago-Dias-7/publication/319928547/figure/fig3/AS:540556553973760@1505890133852/Generic-representation-of-the-stereo-geometry_W640.jpg) [2025, June 18].
- Hartley, Richard. & Zisserman, Andrew. 2004. *Multiple view geometry in computer vision*. 2nd ed. Cambridge University Press.
- Holmes, J., Sommacal, S., Das, R., Stachurski, Z. & Compston, P. 2023. Digital image and volume correlation for deformation and damage characterisation of fibre-reinforced composites: A review. *Composite Structures*. 315. DOI: 10.1016/j.compstruct.2023.116994.
- Junction Box Square 80x80x50 IP54. 2025. [Online]. Available: <https://myzappliances.com/Junction-Box-Square-80x80x50> [2025, June 6].
- “LaVision DaVis 11”. n.d.
- libcamera*. n.d. Available: <https://libcamera.org/faq.html> [2025, October 09].
- Luke. 2013. *IP Ratings Explained - What Are IP Ratings? | NEMA Enclosures*. Available: <https://www.nemaenclosures.com/blog/ingress-protection-ratings/> [2025, October 06].
- Mouser Electronics. 2025. *102991834 BeagleBoard by Seeed Studio | Mouser South Africa* [Online]. Available: <https://www.mouser.com/ProductDetail/BeagleBoard-by-Seeed-Studio/102991834> [2025, September 24].
- OpenCV*. n.d. Available: [https://docs.opencv.org/4.12.0/d1/d9f/classcv\\_1\\_1Stereo\\_1\\_1StereoBinarySGBM.html](https://docs.opencv.org/4.12.0/d1/d9f/classcv_1_1Stereo_1_1StereoBinarySGBM.html) [2025, October 24].
- OpenCV: Feature Detection*. n.d. Available: [https://docs.opencv.org/4.12.0/dd/d1a/group\\_\\_imgproc\\_\\_feature.html#ga354e0d7c86d0d9da75de9b9701a9a87e](https://docs.opencv.org/4.12.0/dd/d1a/group__imgproc__feature.html#ga354e0d7c86d0d9da75de9b9701a9a87e) [2025, October 24].
- Perumal, P.S., Sujasree, M., Chavhan, S., Gupta, D., Mukthineni, V., Shingekar, S.R., Khanna, A. & Fortino, G. 2021. An insight into crash avoidance and overtaking advice systems for Autonomous Vehicles: A review, challenges and solutions. *Engineering Applications of Artificial Intelligence*. 104:104406. DOI: 10.1016/J.ENGAPPAI.2021.104406.
- Pinhole Camera Model | HediVision*. n.d. Available: <https://hedivision.github.io/Pinhole.html> [2025, August 08].
- c6e347fc-5772-4522-a501-af7b22bc4204. 2019. [Online]. Available: <https://img1.17img.cn/17img/images/201908/pic/c6e347fc-5772-4522-a501-af7b22bc4204.jpg> [2025, September 11].
- Carlota, V. 2023. *PLA for 3D Printing*. Available: <https://www.3dnatives.com/en/pla-3d-printing-guide-190820194/> [2025, October 24].

- Raspberry Pi. 2025. *Buy a Raspberry Pi 4B* [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/> [2025, September 24].
- Raspberry Pi. 2025. *Buy a Raspberry Pi 5* [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-5/> [2025, September 24].
- “Raspberry Pi Case for Raspberry Pi 5”. 2024.
- RobotShop. n.d. Available: <https://ca.robotshop.com/> [2025, October 17].
- Saba, N., Jawaaid, M. & Sultan, M.T.H. 2018. An overview of mechanical and physical testing of composite materials. *Mechanical and Physical Testing of Biocomposites, Fibre-Reinforced Composites and Hybrid Composites*. (January, 1):1–12. DOI: 10.1016/B978-0-08-102292-4.00001-1.
- Shanmukha. 2017. *STRAIN MEASUREMENT INSTRUMENTS | CIVIL ENGINEERING*. Available: [https://knowledge4civil.wordpress.com/2017/02/05/strain-measurement-instruments/?utm\\_source=chatgpt.com](https://knowledge4civil.wordpress.com/2017/02/05/strain-measurement-instruments/?utm_source=chatgpt.com) [2025, August 26].
- Standard & specimens for metals tensile test*. n.d. Available: <https://www.zwickroell.com/industries/metals/metals-standards/metals-tensile-test-astm-e8/> [2025, October 24].
- Strain gauges*. n.d. Available: [https://www.learningelectronics.net/vol\\_1/chpt\\_9/7.html?utm\\_source=chatgpt.com](https://www.learningelectronics.net/vol_1/chpt_9/7.html?utm_source=chatgpt.com) [2025, August 26].
- StrainMaster-portable. [n.d.]. [Online]. Available: [https://shop.lavision.de/media/image/07/24/54/StrainMaster-portable\\_600x600@2x.png](https://shop.lavision.de/media/image/07/24/54/StrainMaster-portable_600x600@2x.png) [2025, August 13].
- Sutton, M.A., Yan, J.H., Tiwari, V., Schreier, H.W. & Orteu, J.J. 2008. The effect of out-of-plane motion on 2D and 3D digital image correlation measurements. *Optics and Lasers in Engineering*. 46(10):746–757. DOI: 10.1016/J.OPTLASENG.2008.05.005.
- Tensile test brittle deformation fracture. 2021. [Online]. Available: <https://www.tec-science.com/wp-content/uploads/2021/03/en-tensile-test-brittle-deformation-fracture-768x432.jpg> [2025, September 16].
- The Comparison between Microcontrollers and Single board Computer*. 2024. Available: <https://www.vemeko.com/blog/the-comparison-between-microcontrollers-and-single-board-computer.html> [2025, October 17].
- Tiwari, V., Sutton, M.A. & McNeill, S.R. 2007. Assessment of high speed imaging systems for 2D and 3D deformation measurements: Methodology development and validation. *Experimental Mechanics*. 47(4):561–579. DOI: 10.1007/S11340-006-9011-Y/TABLES/3.
- Young, S.S. 2020. *3D reconstruction from 2D images using binocular disparity*. Available: <https://github.com/youngseok-seo/stereo-vision> [2025, August 26].
- Zaremba, A. & Nitkiewicz, S. 2024. Distance Estimation with a Stereo Camera and Accuracy Determination. *Applied Sciences (Switzerland)*. 14(23). DOI: 10.3390/APP142311444.



Zhang, Z. & Research, M. 2000. A Flexible New Technique for Camera Calibration; a typo in Section 2.4) A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 22(11):1330–1334. Available: <http://research.microsoft.com/~zhang><http://research.microsoft.com/~zhang> [2025, October 12].

## **Appendix A    Resource Use and End-of-Life Strategy**

The project will prioritise the use of recyclable materials wherever feasible within its respective fields. All design and construction decisions will emphasise responsible resource management. The frame construction will be designed for easy deconstruction and potential reuse in future projects after decommissioning. Electronic components will be preserved for future project applications. Notably, no batteries, chemicals, or biological materials will be incorporated into the construction of the 3D DIC system, ensuring simplified end-of-life handling and minimising environmental impact.

## Appendix B    Techno-Economical Analysis

The section will focus on the technological feasibility of the project and its relationship to economic viability. The section will include a detailed budget and the final cost breakdown for the entire project.

### B.1 Budget

The budget will exclude VAT on all expenses listed in the table below. A nominal rate of R500 per hour will be assigned to all tasks conducted by a junior engineer. The budget table will be divided into main sections that group similar items. The budget table will include a column grouping tasks under specific labels: **Engineering Time**, **Running Cost**, **Facility Use**, **Capital Cost**, **MMW Labour**, and **MMW Material**.

The cost of the time spent by the junior engineer will be listed under **Engineering Time** and will adhere to the rate of R500 per hour. The **Running Cost** column will account for all services and consumables used during the project. The **Facility Use** column will include depreciation and maintenance costs incurred for the use of existing equipment and facilities provided by the Engineering Department. The **Capital Cost** column will indicate expenses related to any new equipment purchased for the project. Equipment will be categorised under **Capital Cost** if its purchase price exceeds R2000.

The last two columns will account for costs associated with the **Mechanical & Mechatronic Workshop (MMW)**. The **MMW Labour** column will reflect technician costs based on the time spent manufacturing parts. The **MMW Material** column will include the cost of all materials used by the MMW to manufacture parts.

Table 6 and Table 7 indicate the initial budget and the final cost breakdown of this project, respectively.

The actual cost of this project was R 55 960 less than the initial budget. This reduction is due to various savings throughout the entire project lifecycle. Significant savings resulted from the system not being tested against the commercial system as initially expected at the start of the project. Another major saving was achieved by obtaining an outstanding price for the final project presentation. If this uses the entire budget allocation of R 16 880, the project will still be R 39 080 under budget. This project is relatively low-cost, with most expenses going towards engineering time.

**Table 6: Initial Project Budget**

	Engineering Time		Running Cost	Facility Use	Capital Cost	MMW Labour		MMW Material	Total Cost
Description	hr	R	R	R	R	hr	R	R	R
Concept Designs	27.5	13 750	50	275	0	0	0	0	14 075
Concept Selection	11	5 500	0	110	0	0	0	0	5 610
Components	11	5 500	300	110	3500	0	0	0	9 410
Manufacturing Prototype	45	22 500	200	450	0	0.5	175	180	23 505
Test Prototype	20	10 000	100	200	0	0	0	50	10 350
Develop Control Software	74	37 000	100	740	0	0	0	0	37 840
Manufacturing Final Product	30	15 000	250	300	0	0.5	175	100	15 825
Test Final Product	22	11 000	100	220	0	0	0	100	11 420
Compare Final Product to Commercial System	22	11 000	50	11 220	0	0	0	100	22 370
Literature Study	24.5	12 250	50	245	0	0	0	0	12 545
Design Reports	32	16 000	0	320	0	0	0	0	16 320
Project Progress Report & Presentation	44	22 000	50	440	0	0	0	0	22 490
Final Report Draft	47	23 500	50	470	0	0	0	0	23 970
Final Report	17	8 500	50	170	0	0	0	0	8 720
Final Project Presentation	33	16 500	50	330	0	0	0	0	16 880
Total [R]:	457	228 500	1 400	15 600	3500	1	350	530	249 880

**Table 7: Final Cost Breakdown**

	Engineering Time		Running Cost	Facility Use	Capital Cost	MMW Labour		MMW Material	Total Cost
Description	hr	R	R	R	R	hr	R	R	R
Concept Designs	8	4 000	0	275	0	0	0	0	4 275
Concept Selection	4	2 000	0	110	0	0	0	0	2 110
Components	10	5 000	75	110	3835	0	0	0	9 020
Manufacturing Prototype	25	12 500	200	450	0	0	0	0	13 150
Test Prototype	12	6 000	100	200	100	0	0	0	6 400
Develop Control Software	62	31 000	0	740	0	0	0	0	31 740
Manufacturing Final Product	26	13 000	250	300	0	0	0	0	13 550
Test Final Product	54	27 000	200	220	100	0	0	0	27 720
Literature Study	45	22 500	0	245	0	0	0	0	22 745
Project Progress Report & Presentation	34	17 000	0	440	0	0	0	0	17 440
Final Report Draft	40	20 000	0	320	0	0	0	0	20 320
Final Report	50	25 000	0	320	0	0	0	0	25 320
Final Project Presentation	-	-	-	-	-	-	-	-	-
Total [R]:	370	185 000	825	4 060	4035	0	0	0	193 920

## B.2 Planning

This section will provide the proposed project plan in the form of a Gantt chart. The Gantt chart will be used to outline specific activities required to complete the project. Where possible, activities will be run in parallel to optimise the available time while allowing for potential delays. Overlapping tasks on the Gantt chart represent these parallel activities. Key milestones will also be highlighted on the Gantt chart.

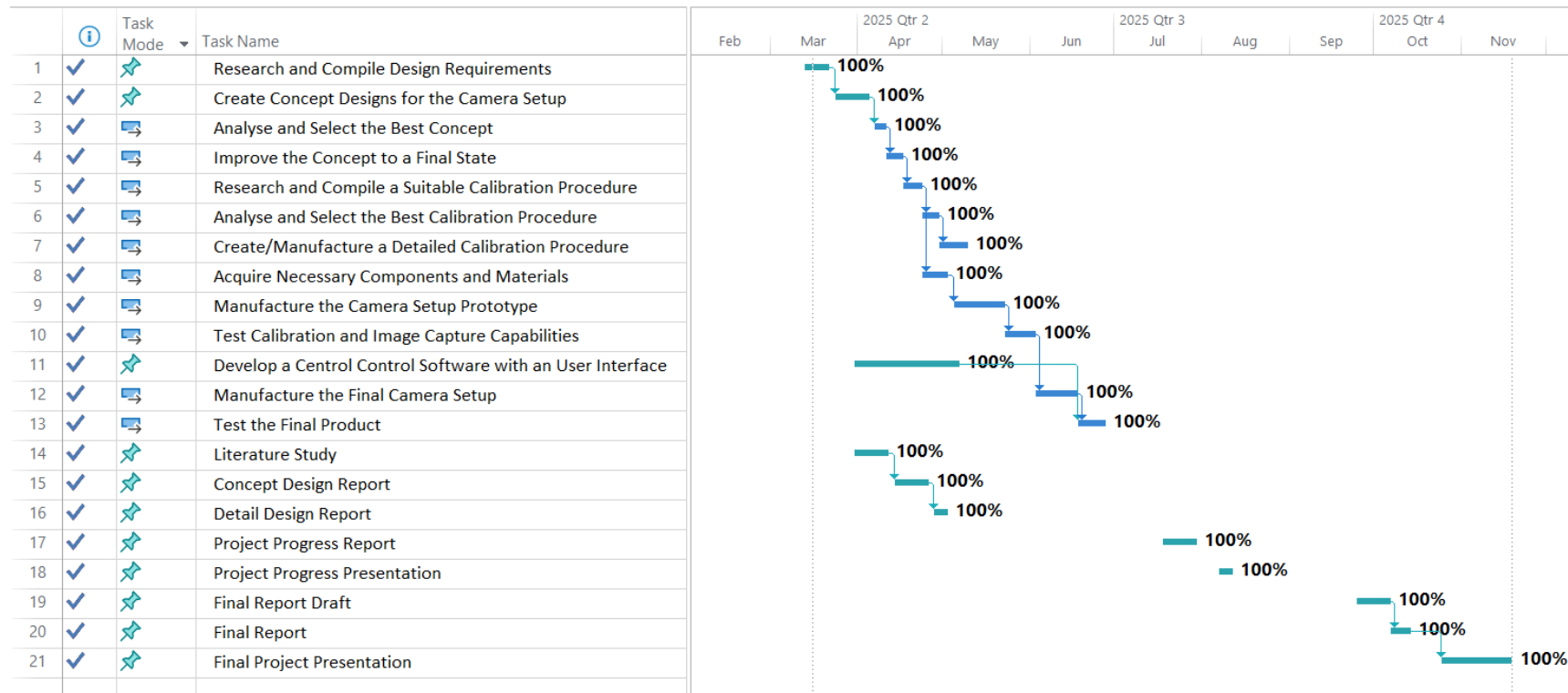


Figure 32: Gantt Chart of Completed Project.

### **B.3            Technical Impact**

Stereoscopic camera systems can be used in many applications, like 3D DIC. The created camera system features a cost-efficient, open-source design capable of stereo image capturing. The calibration process enables anyone to calibrate the system using the intuitive user interface easily. The system enabled the creation of accurate disparity maps, which can be used to visualise the relative distances of objects in the image scene.

The system provides an accurate camera system for the development and testing of 3D DIC. It laid the foundation for implementing a single-image calibration algorithm. Single-image stereo calibration will significantly benefit the industry by reducing the time spent to calibrate a camera system, while remaining robust.

### **B.4            Return on Investment**

The project provides a cost-efficient stereo camera system with an intuitive calibration process, which can be used in future research on an in-house-developed 3D DIC algorithm at Stellenbosch University. The project also started with the implementation of a single-image stereo calibration algorithm. This is an ideal project for a future Mechatronic Project 478 student to further build upon.

### **B.5            Potential for Commercialisation**

The open-source nature of this project enables anyone to access the files and build the system. It also does not prevent them from selling or distributing the system as they wish. This allows the system to be easily commercialised and used in anyway desired. Possible areas where the camera system can be used is in schools and learning workshops. This can allow kids and young adults to be introduced to the concept of computer vision and stereoscopic camera systems.

## **Appendix C     Risk Analysis and Safety Procedures**

The project delineates various risks that require consideration and mitigation to guarantee its successful completion. These risks are primarily categorised into two groups: risks related to project completion and safety risks. Ethical risks are deemed irrelevant, as the project does not entail testing on humans or animals.

### **C.1             Safety Risks**

Working in a workshop environment presents several safety risks. To mitigate these risks, a comprehensive safety report was compiled prior to the commencement of all laboratory setups or manufacturing processes. This will allow experienced individuals to review planned procedures and identify potential hazards. Adhering to established safety protocols and using appropriate personal protective equipment (PPE) will further minimise risks during fabrication and testing.

### **C.2             Safety Report**

The safety report was compiled before conducting testing using the camera system and the LaVision 3D calibration plate. The report is detailed in describing the experimental procedure followed to ensure the safety of all people in the laboratory, as well as the equipment used. It lists all the general laboratory safety instructions followed and how interactions with other laboratory users will be handled. An in-depth activity-based risk assessment was conducted, and mitigating steps to reduce risks were identified and provided. The report concludes with an evacuation plan in the event of an emergency. The safety report is provided in the following pages.



# Ampere Lab Safety Report

## Camera Calibration Testing using LaVision Calibration Plate


<b>Date:</b>	17/06/2025
<b>Student &amp; SU nr:</b>	Franco du Plessis - 25866095
<b>Student contact nr:</b>	084 582 1148
<b>Supervisor:</b>	Dr R.P. Theart
<b>Lab engineer:</b>	Mr K. Zapke
<b>Head of safety:</b>	Mr Cobus Zietsman

### Emergency Contacts:


<b>Contact:</b>	<b>Room nr.</b>	<b>Work nr.</b>	<b>Cell nr.</b>
Mr K. Zapke	M6029	082 714 9512	-
Mr C. Zietsman	M212	021 808 4275	-
Campus Security	-	021 808 2333	WhatsApp: 082 808 233
Fire Brigade	-	021 808 8888	-
Ambulance	-	021 883 3444	-

### Signatures:


Student:  
(Franco du Plessis)

 17/06/2025

Supervisor:  
(Dr R.P. Theart)

 18/06/2025

Lab Engineer:  
(Mr K. Zapke)

 17/06/2025

Head of Safety:  
(C Zietsman)



### **Pressure vessels or pipes (check relevant box):**

- ☒ No pressure vessels or pipes with pressure in excess of 50kPa are involved in this project.  
☐ Pressure vessels or pipes in excess of 50kPa are involved – additional signature and report required (refer to Safety Report Guidelines on SUNLearn).

### **Hot work / working at heights / confined entry / excavation (check relevant box):**

- ☒ No hot work / working heights / confined entry / excavation work involved in this project.  
☐ Hot work / working heights / confined entry / excavation work (underline relevant work type(s)) involved in this project – additional signature and report required (refer to Safety Report Guidelines on SUNLearn).

## Overview of Testing

Type of test and standard (if applicable):

<b>Test type</b>	Calibration data collection and testing.
<b>Standard(s)</b>	iDIC good practices guide

Equipment to be used:

<b>Equipment type</b>	<b>Make &amp; model</b>	<b>Measurement range (if applicable)</b>	<b>Resolution (if applicable)</b>
Calibration plate	LaVision Type 058-5-1	500 mm	-
Stereoscopic camera system	Student developed – Raspberry Pi5 (8 GB) and two Camera Module 3	Maximum frame rate of 60 Hz	3840 x 2160 pixels
Computer	Lenovo IdeaPad Gaming 3	-	-
Measurement	Midas Measuring Tape - 1 m	1000 mm	1 mm

## Detailed Experimental Procedure

The purpose of this experiment is to validate the camera calibration procedure implemented by a student developed stereoscopic camera system. The system consists of a Raspberry Pi 5, two Camera Module 3 units, and a frame connected a tripod. Images of a LaVision Type 058-5-1 calibration plate will be captured and processed using a calibration algorithm. The algorithm will correct for lens distortion, misalignment and intrinsic and extrinsic parameters. This section outlines the setup, execution and shutdown procedures for the experiment.

Room E3020 will be used as a safe storage location for the calibration plate, ensuring the equipment is locked with restricted access. The venue identified for conducting the experiment is room E2029 (Ampere Lab).

### Setup procedure for the camera calibration:

- Step 1: The operator must verify that the venue identified to conduct the experiment is open for use during the planned time and date.
- Step 2: The operator must verify that the planned time and date of the experiment is outside of the load shedding schedule.
- Step 3: The operator must be well informed on the operation of the system and the planned experiment procedure.
- Step 4: The operator must transport the calibration plate in a locked container, containing padding on the inside. Transportation must be completed in a single action with no other stops in-between the testing venue and the safe storage location. The calibration plate must under no circumstances be left unsupervised by the operator.
- Step 5: The area in which the experiment will be conducted must be cleared of any liquid or loose materials which can damaged the equipment.
- Step 6: The camera system must be set up and all connections verified.

Step 7: The operator must create a secure storage location to store all information being collected during the experiment.

### Testing procedure for the camera calibration:

- Step 1: Attach the calibration plate to a tripod 500 mm across from the camera system in the same vertical and horizontal plane.
- Step 2: Using the student developed calibration software's GUI, ensure that each camera has an unrestricted view of the calibration plate. Adjust each camera individually as needed.
- Step 3: Adjust the focus on each camera to ensure that both have the clearest image of the calibration plate.
- Step 4: Take multiple images of the calibration plate at different angles to be used in further development of the calibration software.
- Step 5: Run the calibration software.
- Step 6: Capture multiple images of ruler to verify the calibration algorithm. The calibration plate is replaced with the ruler to ensure the distant between the cameras and capture point is the same.

### Shut-down procedure for the camera calibration:

- Step 1: After all photos have been captured, turn off and unplug the camera system.
- Step 2: Ensure that the calibration plate is in the locked container.
- Step 3: Upload the data captured to a secure cloud storage location.
- Step 4: The operator must disassemble the camera system and verify that it is ready for transportation.
- Step 5: After use, the calibration plate will be put back in its container and returned to the storage venue.
- Step 6: Pack away all remaining equipment and verify that nothing is left behind.
- Step 7: Follow the general housekeeping steps as discussed in this report.

## Warning Symbols

All applicable warning and hazard symbols.



1) Fragile equipment



2) Electrical hazard



3) General warning

## General Laboratory Safety

The following general laboratory safety instructions are applicable:

- No afterhours testing may be performed without the necessary permissions<sup>1</sup>.
- Full supervised training is required before testing may be undertaken. Permission to proceed with unsupervised testing should be signed off by the lab engineer.
- Closed shoes must be worn at all times.
- Emergency equipment must be located and easily accessible.
- Students may not work alone in the laboratory.
- Emergency exits must be known. The nearest exits applicable to the setup are provided in Appendix A of this document.
- Loose clothing may not be worn. Loose hair must be tied up.
- Good housekeeping practices should be maintained during testing. The lab should be completely clean, including all equipment stored away, after testing. Refer to the General Housekeeping section for particulars regarding practices to be followed for this specific setup.
- No food or drink is permitted in the laboratory.
- Safety report must be visible and accessible during testing.
- No equipment or test may be left unattended.

## Anticipated Interactions with other Laboratory Users

The venue is not in use during the planned dates of the experiment. Any other users present in the venue will be informed of the experiment being conducted and all possible hazards. A 2 m area around the experiment area will be kept clear to prevent unnecessary interactions with other users while conducting the experiment. A partner will always be present and will be well informed with regards to the experiment.

## General Housekeeping

The following housekeeping steps must be followed at all times:

- Lock the calibration plate in the designated storage location.
- Report any damaged or broken equipment.
- Ensure that the workspace is clean and neat.
- Ensure that no personal belongings are left behind.

## Fire Safety

The experiment does not include any direct fire risk. In the event of a fire, the evacuation plan seen in Appendix A should be followed. The operator is prohibited from wearing earphones while in the venue to remain alert to fire alarms.

## Activity Based Risk Assessment

The following risks and mitigation measures are associated with the experiment setup and procedure:

Activity	Risk	Risk Type* (P/E)	Mitigating Steps	Classification of Risk Severity
<b>General</b>				
Moving around in the lab	Tripping or knocking equipment over	P	Acceptable risk	Never use a cell phone when moving around and be aware of surroundings
Moving any equipment out of the way	Damaging equipment	E	Possible risk	Clear the area around the intended experiment setup before starting to move equipment
Camera system may not be left unattended	Malfunction or damage to equipment	E	Substantial risk	Always stay at workstation while tests are ongoing
Turning the camera system on and off	Electrical shock	P	Possible risk	Inspect cable insulation before switching plugs on
Power outages	Damage to equipment or loss of data	E	Possible risk	Be aware of load shedding schedules and be sure to switch all equipment
Personal valuables in the laboratory	Theft of valuables from the laboratory	P	Acceptable risk	Do not bring unnecessary valuable items to laboratory sessions. Valuables that are brought to the laboratory should be placed in a safe and visual location
Tidying lab	Tripping	P/E	Possible risk	Do not trip over cables
Dropping the calibration plate during transportation	Damage to equipment	E	Possible risk	Transport the equipment in a container filled with padding.
<b>Camera calibration</b>				
Backing up data	Data loss	P	Possible risk	Store data as per guidelines and ensure sufficient independent backups are kept

\*P – personal, E - equipment

## Disciplinary Actions

Failure to comply with any of the aforementioned safety regulations or procedures will result in disciplinary action. Students will be issued an initial warning: after three warnings, the lab access is revoked for a month.

## Appendix A: Emergency Evacuation Plans for Ampere Laboratory:

### EMERGENCY EVACUATION PLAN

Stellenbosch  
UNIVERSITY  
TYUNIVERSITH  
UNIVERSITEIT

### ESCAPE PLAN

#### EVACUATION INSTRUCTIONS

- The automated alarm system or staff will announce the evacuation.
- Follow the instructions and evacuate immediately to safe assembly points.
- When a venue is completely evacuated, close all doors and place markers on the outside door handles to indicate the evacuation is complete.
- Assist disabled individuals as well as visitors to safe assembly points.
- Any missing individuals must be reported immediately to the Evacuation Marshal on duty.

#### ONTUIMING INSTRUKSIES

- Die geoutomatiseerde alarm stelsel of personeel sal die ontruiming aankondig.
- Volg die instruksies en ontruim dadelik na die veilige versamelpunte.
- Wanneer 'n lokaal ontruim is, maak alle deure toe en plaas merkers op buite deurhandvatsels om aan te dui die ontruiming is afgehandel.
- Verleen hulp aan gestremde individue asook besoekers na die veilige versamelpunte.
- Enige vermisde individue moet dadelik aan die Ontruimingsbeampte op diens gerapporteer word.

#### MEDICAL EMERGENCIES

Campus Health Services (CHS): 076 431 0305 (all hours) for CHS ambulance services during office hours and stand-by doctor after hours.  
If the person involved in the medical emergency has medical aid, also contact ER24 ambulance: 084 124

#### MEDIËSE NOODGEVALLE

Kampusgesondheidsdiens (KGD): 076 431 0305 (alle ure) vir KGD se ambulansdiens gedurende kantoorure en na-ure 'n bystandsdokter.  
Indien die persoon betrokke mediese fonds dekking het, kontak ER24 ambulans: 084 124

#### EMERGENCY NUMBERS

CAMPUS SECURITY (USBD)	021 808 2333
CAMPUS HEALTH SERVICES (CHS)	021 808 3496
Police Flying Squad	10111
Ambulance	999/10177
Stellenbosch Medi-Clinic	021 861 2095/021 861 2094
Stellenbosch Hospital	021 808 6100/021 887 0310
Stellenbosch Fire and Rescue	021 808 8888
24-Hour Rape Crisis Stellenbosch	082 977 8581
24-Hour Psychology Crisis Service	082 557 0880

Electrical and Electronic Engineering  
Level 2

YOU ARE HERE

Emergency Assembly Point in Parking Lot

#### LEGEND

IN CASE OF FIRE DO NOT USE THE ELEVATOR	FIRE ALARM	FIRE HYDRANT	FIRE HOSE	EMERGENCY KEY BOX	EVACUATION STAIR CHAIR	SAFETY SHOWER	EYEWASH
FIRE TELEPHONE	FIRE EXTINGUISHER	FIRE BLANKET	FIRST-AID EQUIPMENT	GENERAL DIRECTION	DISTRIBUTION BOARD		

Figure 1: Emergency Evacuation plan

## Appendix D Wiring Diagram of Relay Control

This Appendix includes all the wiring diagram used to control the lights of the system.

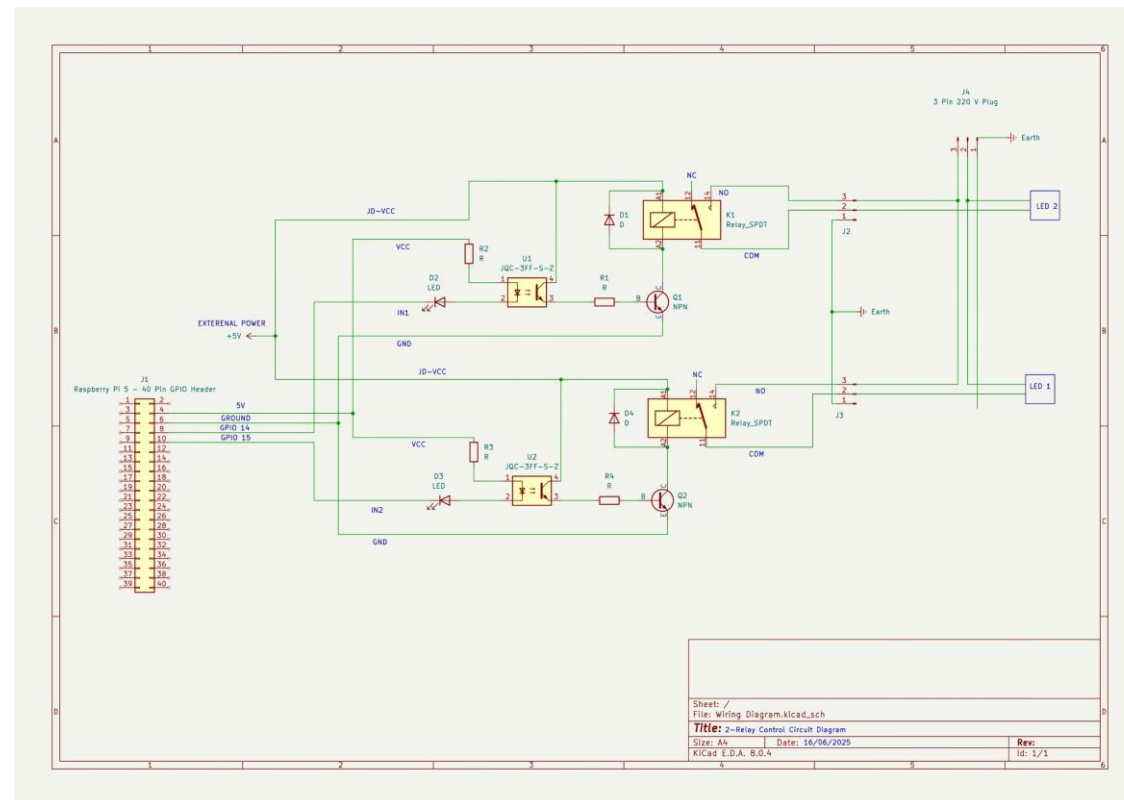


Figure 33: Relay Control Circuit.

## Appendix E Chessboard Pattern Calibration Report

=== STEREO CALIBRATION REPORT ===

Configuration:

- Checkerboard size: 9x6 inner corners
- Square size: 14.5mm
- Image pairs used: 30

Calibration Results:

- Left camera RMS error: 0.597 pixels
- Right camera RMS error: 0.553 pixels
- Stereo calibration RMS error: 1.140 pixels
- Baseline distance: 215.9mm

Left Camera Parameters:

- Focal lengths (fx, fy): 3433.0, 3407.3
- Principal point (cx, cy): 2406.1, 1398.9
- Distortion coefficients: [-0.0480, 0.6814, 0.0017, 0.0081, -1.5204]

Right Camera Parameters:

- Focal lengths (fx, fy): 3466.8, 3445.2
- Principal point (cx, cy): 2412.6, 1335.1
- Distortion coefficients: [0.0068, 0.2815, -0.0005, 0.0076, -0.5480]

Stereo Geometry:

- Translation: [-0.2159, -0.002, 0.007] (meters)
- Rotation angles: 5.42 degrees

Files Generated:

- stereo\_calib.npz: NumPy format
- stereo\_calib.json: JSON format (human-readable)
- calibration\_report.txt: This Report

**Figure 34: Chessboard Stereo Calibration Report.**



## Appendix F    Single-Image Calibration Report

=== 3D PLATE STEREO CALIBRATION RESULTS ===

### CALIBRATION DATA:

- Total correspondences: 264 points
- Grid A (Z=-1mm): 143 points
- Grid B (Z=0mm): 121 points
- Dot spacing: 5.0mm

### CALIBRATION QUALITY:

- RMS reprojection error: 298.9611 pixels
- Final baseline: 333403.2mm
- Baseline calibration RMS: 0.4729 pixels
- Improvement: -298.4882 pixels

### VALIDATION ERRORS:

- Left camera mean error: 68242920049737728.000px
- Right camera mean error: 26723.031px
- Overall mean error: 34121460024868864.000px

### FINAL CAMERA PARAMETERS:

#### Left Camera:

- Focal lengths: fx=3433.0, fy=3407.3
- Principal point: cx=206.6, cy=326.3
- Distortion: k1=0.3102, k2=-3.5303, p1=-0.0010, p2=-0.0144, k3=6.3423

#### Right Camera:

- Focal lengths: fx=3466.8, fy=3445.2
- Principal point: cx=254.5, cy=409.3
- Distortion: k1=0.3587, k2=-4.3928, p1=0.0032, p2=-0.0070, k3=-5.9868

### Stereo Parameters:

- Translation: [-1.429467, 200.408503, 266.443451] meters
- Baseline: 333403.2mm

### FILES GENERATED:

- 3d\_plate\_stereo\_calib.npz: NumPy format
- 3d\_plate\_stereo\_calib.json: JSON format (human-readable)
- 3d\_plate\_calibration\_report.txt: This report

**Figure 35: Single-Image Calibration Report.**