Integration of an optical system for 3D reconstruction

Filipe Ferreira Monteiro

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisor: Prof. Jorge Manuel Mateus Martins

Examination Committee

Chairperson: Prof. Carlos Baptista Cardeira

Supervisor: Prof. Jorge Manuel Mateus Martins

Members of the Committee: Prof. João Carlos Prata dos Reis

November 2020

Acknowledgments

I would like to acknowledge my supervisor, Prof. Jorge M. M. Martins, for all he taught me and for guiding me through this thesis.

Sincere thanks to Fravizel and their employees for making to us available their machines and for helping with the assembly of the hardware.

I would also like to thank my family and friends, for their patience, guidance and support, through all the years of my degree and especially this thesis.

Resumo

Considerando a importância da indústria pedreira para qualquer economia, o seu continuo progresso é de interesse significante. Para melhorar a sua indústria através dos seus produtos, Fravizel pretende usar ferramentas de metrologia Tridimensional para obter modelos 3D dos blocos de pedra extraídos e tratados nas pedreiras. Modelos 3D dos blocos não-processados podem ser usados para otimizar os processos de tratamento intermediários, através da identificação de falhas e os cálculos para os cortes respetivamente necessários, enquanto que modelos 3D dos blocos processados podem ser usados para publicitar. Esta tese oferece uma solução para o design dos scanners 3D que tratam da obtenção de dados necessários para a reconstrução 3D por visão computacional. O primeiro design de scanner consiste em integrar um sistema ótico com as máquinas de corte da Fravizel, visa a aliviar o custo de novas infraestruturas. O segundo design de scanner consiste num sistema ótico capaz de adquirir imagens enquanto circula o bloco de pedra, para permitir reconstruir um modelo 3D de alta qualidade, sem buracos ou pontos cegos para publicitar. A informação recolhida pelos scanners consiste no conjunto de imagens, necessárias ao processo de reconstrução, e nas calibrações intrínsecas e extrínsecas com fins de aumentar a eficiência do mesmo processo. Ambos os scanners recolheram data funcional embora nem sempre ideal para o processo de reconstrução. O primeiro scanner obteve imagens com 12 MP e em 461.14 segundos obteve-se uma calibração intrínseca fraca, resultando num erro medio absoluto de 33.99 pixels. O segundo scanner obteve imagens com 8MP e em 124.03 segundos obteve-se uma calibração intrínseca forte, resultando num erro medio absoluto de 0.13 pixels. As estimações de pose em ambos os scanners foram mais consistentes usando algoritmos sem aplicação de RANSAC, obtendo calibrações corretas com erro medio absoluto entre 0.27 até 0.43 pixels no primeiro scanner e entre 0.14 até 8.87 pixels no segundo scanner.

Palavras-chave: Indústria Pedreira, Metrologia 3D, Scanner 3D, Visão Computacional, Estimação de Pose, Calibração de Câmara

Abstract

Given the stone industry vitality to any economy, its continuous progress is of significant interest. To improve their products and industry, the company, Fravizel, desires to use 3D metrology tools in order to obtain 3D models of the stone blocks extracted in quarries. 3D models of the unprocessed stone blocks could be used to optimize intermediary processes, such as identifying flaws and calculating required cuts and 3D models of the processed stone blocks could be used for advertising. This thesis offers a solution for the design of 3D scanners that acquires the information required for 3D reconstruction through computer vision. The first scanner consists of integrating an optical system to Fravizel's machines, alleviating the cost of new infrastructures. The second scanner consists of a camera system capable of acquiring images while circling the stone block, for an advertisable gapless high-quality 3D model. The data, the proposed scanners acquire, consists of a set of images required for reconstruction and the calibrations of the image sensors that could increase the efficiency of the reconstruction process. Both scanner configurations yielded positive yet not ideal data for the reconstruction step. The first scanner obtains 12Mp images and was poorly intrinsically calibrated in 461.14 seconds with a mean-absolute-error of 33.99 pixels. The second scanner obtains 8Mp images and was well intrinsically calibrated in 124.03 seconds with a mean-absolute-error of 0.13 pixels. Pose estimations across both scanners were more reliable using algorithms without RANSAC obtaining correct calibrations with mean-absolute errors of 0.27-0.43 pixels in the first scanner and 0.14-8.87 pixels in the second scanner.

Keywords: Quarry industry, 3D Metrology, 3D Scanner, Computer Vision, Pose Estimation, Camera Calibration

Table of Contents

List of figures

List of tables

Abbreviations

AP3P
  Efficient Perspective-Three-Point ....viii, 18, 20, 81, 82, 83, 84, 86, 87
CAD
  Computer Assisted Design .... viii, ix, 25, 26, 38, 45, 47
CMM
  Coordinate-Measuring Machine... 8, 9, 10, 93
DLT
  Direct Linear Transform 22
EPnP
  Efficient Perspective-n-Point...18, 21, 81, 82, 85, 86, 94
GUI
  Graphical User Interface .... viii, ix, 29, 30, 48, 49
P3P
  Perspective-Three-Point ....viii, 18, 19, 20, 81, 82, 84, 86, 87, 90, 91
PLA
  Polylactic Acid 34
PnP
  Perspective-n-Point...17, 18, 21, 58, 60, 84, 87, 90, 91, 94
PWM
  Pulse-Width Modulated 39, 40
RANSAC
  Random Sample Consensus ....iii, iv, 58, 59, 61, 62, 80, 83, 84, 85, 87, 90, 91
VCC
  Voltage Common Collector 39

Chapter 1

1 Introduction

1 Introduction

This chapter will provide the context regarding the industry we chose to improve, why it was chosen and the way through which we intent to contribute to its development. The first section will supply the context of the activity intended to be improved, clarifying the importance of the industry involved in order to justify the need for its development. The observed problems we seek to minimize are then defined in the second section. After describing the context and problem, the methods chosen to minimize the problems are described in the third section. The final section delineates the structure that the rest of this thesis will follow.

1.1 Context

Mineral resources are highly sought out recourses generated by natural geological processes. These processes are large scaled, slow, outside the control of humanity and not replicable, making these resources, practically speaking, nonrenewable [1].

Humanity has used mineral resources since pre-historic time. In fact, the importance of our use of these recourses has led to a commonly used designation for one of the earliest periods of our history, the Stone Age. Pre-historic humans used stones they simply found and collected, only when we began looking for minerals that required extraction did, we developed mining practices. Oldest quarries found dates back to ancient Egypt. Over 200 quarries haven been studied and dated between the Late Pre-dynastic Period and the Late Roman Period, covering 3500 years [2]. We have continuously evolved our use of more and more mineral resources, leading to subsequent periods to be referred to as the Bronze Age and the Iron Age, as we updated our tools and methods with better minerals. Presently we have found applications for most discovered natural resources. Mineral recourses can be categorized based on their application, fuel minerals, industrial minerals and metallic minerals [3]. Fuel minerals are minerals used as a source of energy, oil, coal and uranium minerals used to produce vehicle fuel, thermal energy and nuclear energy, respectively. Metallic minerals are minerals that contain significant concentrations of metals, such as iron, gold, silver and bronze. Metals are used in nearly all industries, every day we use tools made of metallic materials, from cutlery to heavy machinery, even our building are constructed not just out of industrial minerals but also metallic ones. Industrial minerals are minerals, with physical and chemical properties that make them useful a wide arrange of areas. Industrial processes and production of chemicals and fertilizers make use of chemical properties of industrial minerals. Construction has been the most common application to minerals with good aesthetics and physical properties [3].

1.2 Problem

The limestone is a rock, mostly composed of the mineral calcite, commonly used worldwide in construction, used structurally or decoratively, and in the production of cement, one of the most important building materials. Also the Portuguese pavements, made mostly from limestone, are a significant cultural reference, as can be seen in the University of Coimbra in a mosaic with the image of Saint-Queen Elizabeth of Portugal (Figure 1). Given the economic importance of these resources to worldwide necessities and Portuguese culture, the extraction of limestone is an activity worth improving.

Figure 1: Pavement mosaic of Saint-Queen Elizabeth of Portugal in Coimbra [4]

In a quarry the limestone is removed from its bedrock in large parallelepiped blocks. These blocks must be processed before they can be sold, one of the steps in this process involves cutting the block to remove flaws or reshape it into an acceptable shape. The ideal product is a properly shaped rectangular parallelepiped of limestone. A picture of limestone quarry is presented in Figure 2.

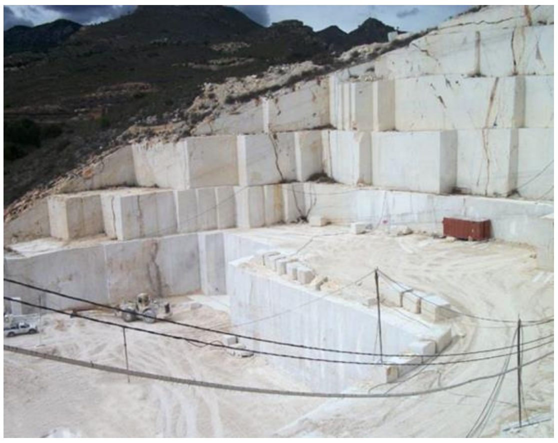


Figure 2: A Limestone Quarry [5]

Due to the size of the limestone blocks, they are difficult to maneuver and analyze. Due to these difficulties, processing the blocks is time consuming. Maneuvering the blocks can only be accomplished using cranes and trucks, while analyzing the blocks for flaws is done through a rough visual inspection, which determines the position of the cuts needed resulting in possible waste or mistakes.

To perform cuts on these large blocks the quarries use heavy machinery equipped with saws, known as Fravizel's machine, (Figure 3). The saw has limitations, it only moves on rails restricting its work area, it is positioned by the workers and it can only perform cuts transversal to its rails. This results in a process where, the stone blocks must be placed in a row between the machine's rails, the saw is positioned visually and if cuts in more than one direction are required the block must be moved and rotated for each direction.

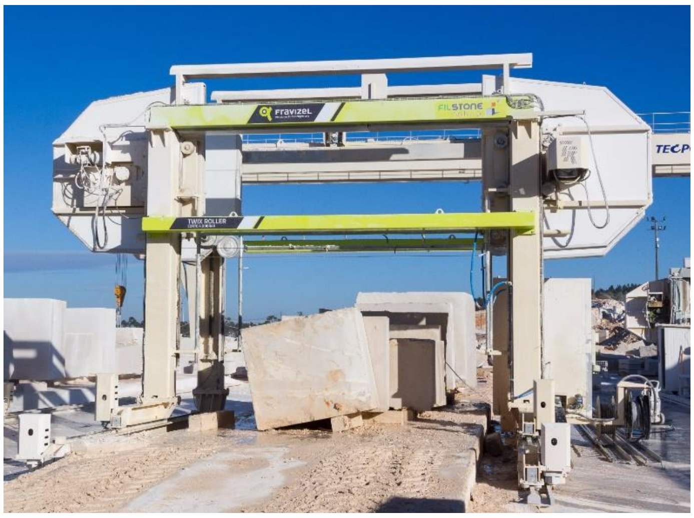


Figure 3: Fravizel's cutting machine

After the blocks has been shaped to an acceptable shape, as exemplified in Figure 4, the quarry wants to advertise the block's qualities as best it can. Advertising their products qualities is also hindered by the sheer size of the blocks. In order to capture the entirety of the block, a photograph must be taken from a distance that doesn't allow the quality of the block's surface to be seen.

Summarizing, there are shortcomings in the methods through which the stone blocks are processed and advertised. Concerning the stones processing, the duration and waste can be reduced by automating both the analysis of the blocks and the operation of the cutting machine. The advertising limitations can be lessened by generating a new medium or platform through which it is possible to portray the shape and surface qualities of the stone block without sacrificing one or the other.

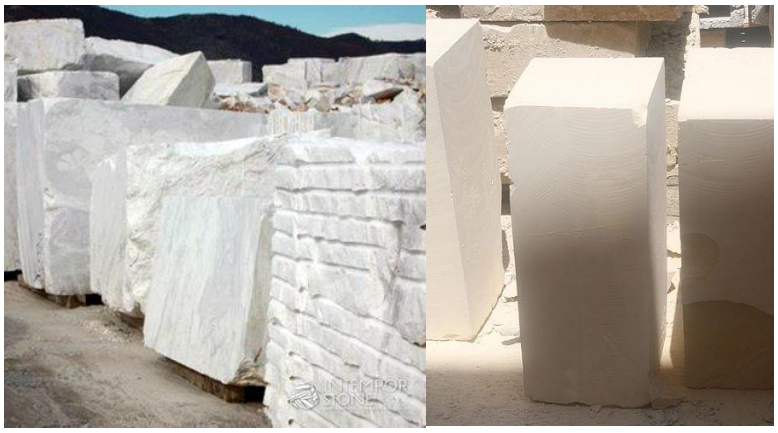


Figure 4: Limestone blocks
(a) Rough [6]; *(b) Clean* [7]

This automation can be accomplished by acquiring a digital three-dimensional model of the blocks allowing the analysis to be computationally accomplished. The scope of this thesis resides within the data acquisition step of a 3D model acquisition process. In other words, designing and physical installing the hardware required as well as implementing software for the strict purpose of acquiring data required for 3D reconstruction, highlighted in Figure 5.

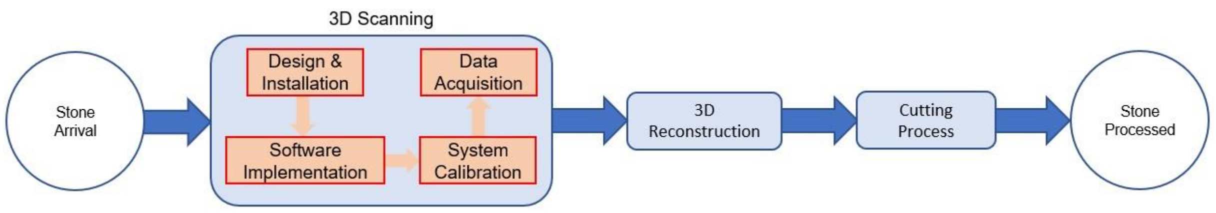


Figure 5: Diagram of proposed solution

1.3 Objective

This thesis intends to be a step towards improving the quarry industry, by increasing the efficiency and safety of one of its processes. Therefore, this thesis statement is:

Design and implement a 3D scanning system that optimizes the following 3D reconstruction, improving its efficiency and viability.

The research question for which this work intends to contribute is the following:

- How effectively does the proposed solution collect the data required for 3D reconstruction?

In order to answer the research question, the following complementary questions should also be answered:

Q.1. What type of scanner is most viable towards replacing the workers visual analysis of the stone blocks?

Q.2. Can the proposed solution be designed around the Fravizel's machine in order to use the existing structure and functionalities?

Q.3. What other mediums can be used to advertise the blocks?

1.4 Structure

This dissertation is organized as follows: Chapter 2 delineates an overview of the various types of 3D scanners developed and employed over the years. This overview shows the evolution of the scanners, displaying the developments made by new scanners, with respect to the limitations of the previous existing scanners. A classification based on their operating principals and technologies is established along with their respective advantages and disadvantages. This chapter goes on to explain the calibration of image sensors which are used in certain types of scanners.

After researching the various types of scanners and choosing the appropriate type for our application, the following chapter 3 describes the configurations used in the process of developing the final scanners. Initial set ups used basic components to merely test the viability of ideas, as previous configuration's limitations are addressed, new scanner configurations are developed and explained. The calibrations are performed using different software and methods, until settling on a final programing language that allows to compare different available calibration pipelines.

The results are presented in chapter 4 to allow an analysis to be conducted regarding the hardware configurations and the calibration methods. The hardware configurations are evaluated by examining the quality of the images acquired with regards to their synchronicity, resolution, scanning reliability. The speed and accuracy of different intrinsic and extrinsic calibration algorithms are displayed, in order to assess the most accurate calibrations with regards to the time it takes to calculate, to ensure an increase in the efficiency of the quarry process. Finally, in chapter 5 the conclusions based on the discussion of the results are presented and possible future steps to improve on the work done here are discussed.

Chapter 2

2 Literature Review

2 Literature Review

Three-dimensional reconstruction is the process of determining the three-dimensional profile and coordinates of the surface points of an object. This application is at the core of a wide variety of fields. The construction of the three-dimensional profile of an object may be done through a variety of data. The process of collecting the required data for 3D reconstruction is referred to as 3D Scanning. Three-dimensional scanning can be divided into 2 categories: Methods that required physical interaction with the object, contact scanning, and methods that do not physically interact with the object, non-contact scanning [8].

2.1 Contact Scanning

Scanning involving physical interaction is commonly categorized as contact scanning. Contact scanning is a straightforward process, the object is fixed to a precision flat surface plate or fixture while a high precision kinematic chain system repeatedly touches the objects surface recording measurements of the location of the kinematic chain's probe to be used as data for the 3D reconstruction. Various types of kinematic systems can be used for this methodology.

The most common configuration is a system of rigid arms, mounted with rails, glide along each other on a constant perpendicular angle. The position of each arm on its respective rail is registered at every point, this data allows the calculation of the 3D coordinates of the tip of the arm that is in contact with the object resulting in a mapping of its surface point by point. An example of this are the coordinate-measuring machines more commonly called CMM scanners [9].

More complex configuration using translational and rotational joints have been designed to achieve higher accuracy and higher maneuverability that would allow scanning of objects with higher order of complexities. With the improved knowledge over kinematic chains and forward kinematics any chain built from a mixture of prismatic and rotational joints can be researched for improvements over the common CMM configuration [10]. Configurations using rotation joint and parallel mechanisms have been tested to achieve better results than more traditional CMMs [11].

2.1.1 Introduction

An example of a contact 3D Scanning device is the coordinate measuring machine, CMM, this system is used for measuring the physical geometric aspect of an object. A CMM operates on a fixed platform where the object is set and immobilized. An articulated mechanical arm is equipped with a probing system, this arm's probe position can be controlled manually by an operator or via computer programming. The rigid sections of the arm are connected orthogonally to each other, forming a traditional three-dimensional coordinate system, where each section corresponds to axis X, Y or Z. To measure the value of the probe's coordinates in the systems Cartesian reference frame, each bone has a scale system for each coordinate. The component of the CMM that physically interacts with the surface being scanned is attached at the end of the last section of the mechanical arm. This component is commonly designated as the probing system. The mechanical arm movement comes from the translations of each of its sections along their respective rails. The rails come equipped with a scale that returns the values of the probe's positional coordinates on its corresponding axis. Reading all three scales gives a complete determination of the probes position in the systems reference frame [12][9].

Coordinate measuring machine are often composed of three main systems. The machine main body which includes the three axes of motion, the probing system which is used to touch the object and the computer system to collect the data, control the movement of the machine body and typically used as an interface between operator and the scanner.

Counter measuring machines bodies have a high variety of configurations and sizes available. In 1950s the Ferranti company in Scotland created the first CMM, however this contact scanner is only had 2 axes of motion. The first three-dimensional version of a contact scanner would premiere in Italy in the 1960s [12][13].

2.1.2 Hardware

CMM systems consist of two system, the kinematic system that composes the main body of the machine and the probing system that interacts with the object.

Kinematic System - Currently the most common configurations of the modern coordinate measuring machine use a bridge type structure. The bridge is composed of a horizontal beam supported by two vertical beams. One of the vertical beams is attached to the systems platform, where the object will lie, with its motion controlled along a rail attached to one side of the table. The other vertical beam rests on the table moving along the edge of the opposite side of the table, simply to assist the support of the horizontal beam. The horizontal beam supports a carriage allowing motion along the beam. The movement of the vertical beams along the table's guide rail and the movement of the carriage along the horizontal beam are perpendicular forming an XY plane. The last axis, for the Z coordinate, is formed by the vertical movement of a last beam along the center of the carriage, supporting a probing system. Finally, the probing system is built on the end of this vertical beam corresponding to the Z-axis, forming the sensing device that will touch the object. This is in essence a kinematic chain composed of four bones connected by three prismatic joints. The combined movement of the vertical beams along the side of the table (one of the XY-axis), the carriage along the horizontal beam (the other XY-axis) and the last beam moving vertically through the carriage (Z-axis) delimit the work area or measuring envelop of the CMM [9].

A variant of the machine body of the CMM is a kinematic chain with rotational joints connecting the bones instead of linear scaled guide rails, providing angular readings from the joints of the arm instead of linear readings of the positions of the arm's bones with respect to one another. While the linear readings directly give the coordinates of the probe's location in the reference frame, the angular reading are used to calculate the coordinates of the probe using basic forward kinematics. These mechanical arms with rotational joints are not constricted by the need for a fixed platform or bed, conferring the scanner a level of portability and flexibility not available in traditional CMMs. The scanners flexibility results in the capacity to scan more complex objects, with steeper curves, concave and convex shapes.

Probing system – The most common type of probe system used nowadays is the electronic touch-trigger probe system. The accuracy and precision of measurements improved drastically with the creation of the touch-trigger probe by David McMurtry [14]. A probe is essential consists of three parts, a system of displacement transducer, a suspension structure and a stylus. The stylus is a relatively long and slender shaft with a sphere at the end that is placed in contact with the surface. The sphere is the ideal shape to allow acceptably accurate measurements over a wide range of surfaces. New probe models can be dragged along a surface and register the position data at specified intervals. This method of CMM inspection is often more accurate and faster than conventional touch-probes [12].

2.1.3 Considerations

Contact systems applicability is limited due to its physical approach. The used methodology requires the mapping of the surface point by point limiting the speed of the process, the dimensions and resistance of the object.

Since CMM scanners operate point by point, scanning each reading is accomplished by moving the probe to a new position and then recording the location. Given the necessity of high precision control of the kinetic chain, the fastest contact systems operates at a few hundred hertz [14][15]. The range of the scanner's work area limits the dimensions of the object. The impracticality of building larger and larger contact scanners makes these methods less suitable for larger objects.

The repeated physical interaction required for accurate mapping also makes this process highly unsuitable to sensitive or fragile objects, such as antiques, and inapplicable to non-rigid objects such as objects made of cloth or textile. All these limitations developed a need for faster and ranged 3D scanning systems.

2.2 Non-Contact Scanning

In order to collect information on an object's shape without physically interaction, non-contact scanners were developed using known properties of several types of emissions such as radiation and sounds by detecting or recording their interaction with the object. Non-contact scanners are divided into two categories, active and passive [16]. Scanning systems are categorized as active if they are responsible for the emissions, they detect in the process of collecting data. Scanning systems are categorized as passive if they rely only on ambient radiation such as visible light spectrum.

2.2.1 Active Scanners

Non-contact active systems collect data regarding object's dimensions by emitting radiation or light/sound at the object and detecting the radiation that passed through the object, such as x-ray techniques or the reflection of the light/sound, such as camera or sonar systems. Time-of-flight, triangulation and phase measurement are examples of methods used to calculate the distance or slope distribution of a surface with the detected information from the emissions sent towards the object [17]. Time-of-flight is an active measuring method that consists in the emission of a laser or sound with known travel speed and timing the return of the reflected emission. Phase measurement works similarly to the time-of-flight method measuring the difference in the phase between the emitted signal and detected signal instead of the time, translating to the slope distribution of a surface instead of direct distance to emitter [18][19]. Triangulation is an active measuring method that consists in the emission of a laser at the object with the simultaneous recording of the incident area. The emitter, the camera and the point where the laser contacts with the object are the three vertices of a triangle. Using the known distance between emitter and camera, the angle of the projected laser and the images captured by the camera we can fully define the triangle's dimensions and therefore the coordinates of the laser incidence with the object [17]. The speed of these scanners will be considerably faster than contact scanners given the lack of need to physically interact with the object, but how much faster depends on the method, hardware and the type data collected for the reconstruction. Active scanners are varied in hardware and operating principles, from a simple set up consisting of a digital camera and an emitter acquiring images for reconstruction through triangulation, to more technically specific system such as 3D terrestrial laser scanners based on the various measuring principles mentions above. Reliable 3D active scanners have been designed using common electronic devices, namely cameras, common video projectors or laser emitters, resulting in very cost-effective reconstructions [19][20]. On the other side of the active scanner spectrum, there are 3D terrestrial laser scanners, taking time-of-flight, phase or triangulation measurements at high speeds from 1000 to 500k Hz [21][22].

2.2.2 Passive Scanners

Non-contact passive systems collect data regarding object's dimensions by detecting naturally available emissions, such as ambient radiation, reflected by the object. The most practical systems in this category involve image sensors detecting the ambient radiations in the visible spectrum, through photogrammetric processes[23]. The most common type of passive scanner are basic cameras that acquire simple images of the object, with no use of projections or laser to enhance the data in the images, Figure 6-b. The reconstruction process is done through triangulation with the camera poses and the corresponding features between the respective acquired images, as shown in Figure 6-a.

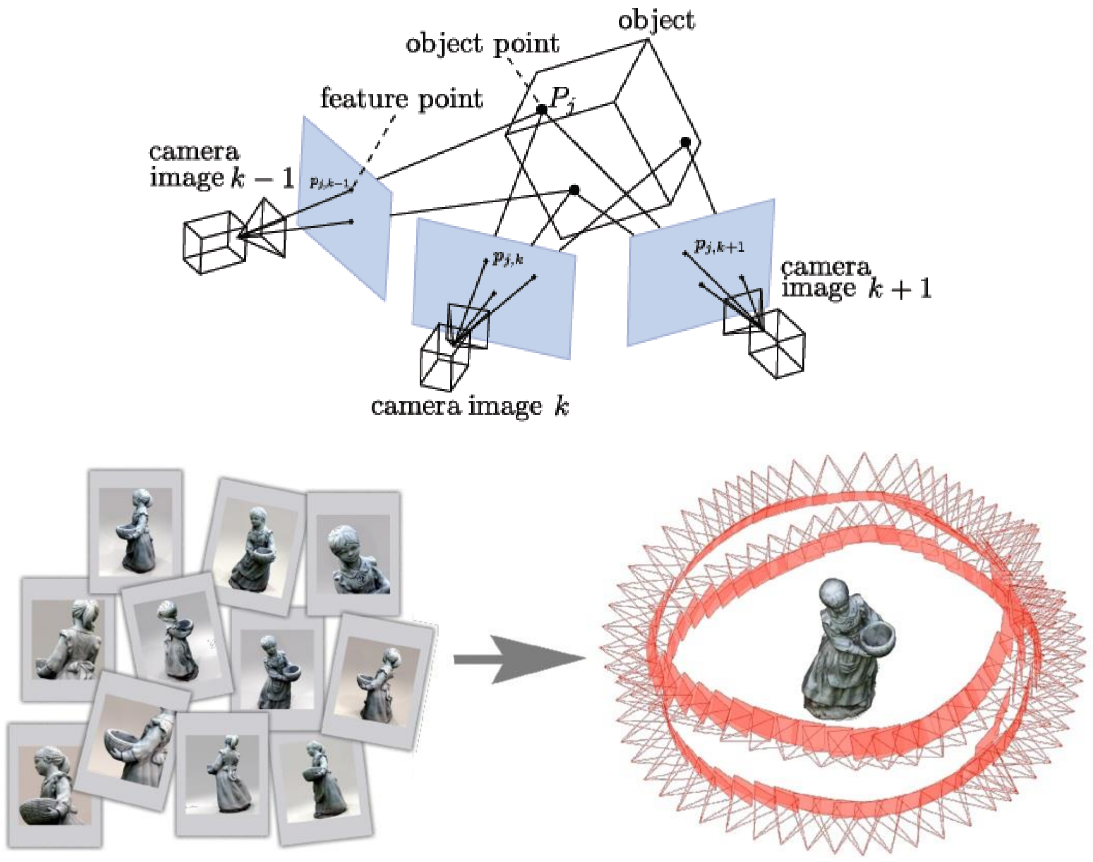


Figure 6: Passive Scanner process
Top-Feature Triangulation [24]*; Bottom-Image 3D reconstruction* [25]

2.2.3 Considerations

Compared to contact scanners, non-contact scanners can present several advantages, such as speed, economic cost, operational difficulty, thoroughness and the lack of physical interaction with the object. This suggests that non-contact scanners are more suitable to our intended applications.

Active scanners function based on the detection of an emitted signal. The emission's detection can be rife with noise in such harsh conditions as that of a quarry. This mean that active scanners are more suitable when there are less noise-inducing conditions. However, passive scanners function based on the detection of ambient radiation, namely natural radiation in the visible spectrum, making them more reliable in more arbitrary work conditions, relative to active scanners.

2.3 Calibrations

Image sensors used in passive scanner require two different types of calibrations. An intrinsic calibration to determine the internal parameters of the image sensor and an extrinsic calibration to determine the pose of the image sensor, the internal parameters remain constant while the pose must be estimated for of each image acquisition.

2.3.1 Intrinsic Calibration

The intrinsic parameters are the internal specifications of the image sensor during the time of the image acquisitions.

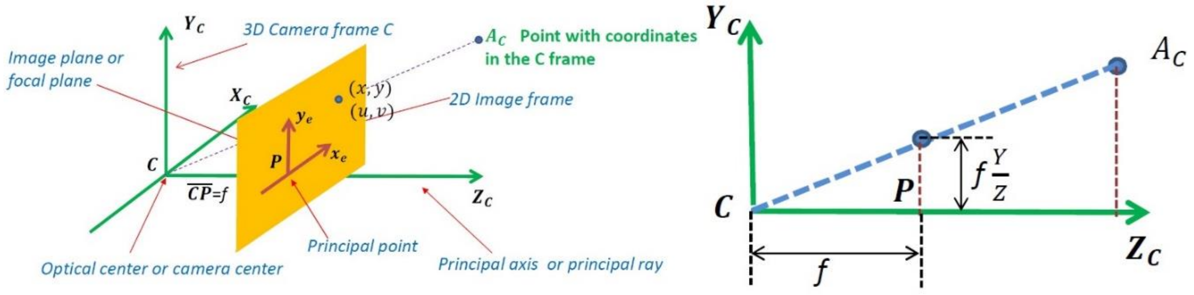


Figure 7: Intrinsic Parameters
f = focal length; P = Principal Point [23]

For the pinhole model of optical imaging, depicted in Figure 7, the camera equation to obtain a three-dimensional point in the image plane is:

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = P \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = K[R\ t] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \qquad 2.1$$

Where λ is a nonzero scale factor; (x, y) are the point's coordinates in the image coordinate system; (x_w, y_w, z_w) are the point's coordinates in the world coordinate system; P is the camera matrix composed of $K[R\ t]$.

K is an upper triangular matrix containing the intrinsic parameters, properly referred to as the matrix of the intrinsic camera parameters. The intrinsic parameters, f_x and f_y are the focal lengths in the x-axis and y-axis directions respectively; (c_x, c_y) is the principle point coordinates; and s is a skew parameter that corrects for tilted pixels, safely assumed to be zero for most cameras [23].

R is the 3x3 rotation matrix and t is the 3x1 translation matrix that encodes the orientation and positions of the camera, in the world coordinate system, these are the extrinsic parameters further explained later.

2.3.1.1 Calibration method:

Determining intrinsic parameters can be accomplished in two main steps:

1. Computing the camera matrix P through known scene points coordinates, known as the resection problem.
2. Factorizing P into K and $[R\ t]$ using RQ-factorization.

First step - The resection problem

Assuming the scene points X_i and their projections x_i are known, the goal is to solve:

$$\lambda_i x_i = PX_i \qquad 2.2$$

Where λ_i and P are unknown. Each point gives three equations and one additional unknown λ_i. Therefore, we need at least six points to be able to define matrix P's 12 elements.

The direct linear transformation approach can be used to solve the equations. Direct linear transformation formulates a homogeneous linear system of equations and solves this system by finding a null space of the systems matrix. Due to noise in the measurements, the system will not wield an exact solution. Therefore, we will search for the solution solving for a homogeneous least squares problem. Singular value decomposition (SVD) computes eigenvectors to solve the homogeneous system [26]. Succinctly, computing P is achieved by firstly establishing a linear homogeneous system, $Cp = 0$, where p is a vector composed from the values of matrix P, that lies in the null space of C, therefore applying a singular value decomposition to C grants a solution to p, and by consequence to the matrix P as well.

Second step – RQ-factorization of P

The now known camera matrix P must be factorized so we can extract the elements of matrix of intrinsic parameters, K. Knowing that the matrix K is an upper right triangular matrix and matrix R is an orthogonal matrix allows the use of the RQ-factorization.

RQ-factorization theorem says that for an n-by-n matrix there is an orthogonal matrix Q and a right triangular matrix R such that $A = RQ$.

Splitting the camera matrix P into two portions such that:

$$P = K[R\ t] \Leftrightarrow \qquad 2.3$$
$$\Leftrightarrow\ [A\ a] = K[R\ t] \Longrightarrow$$
$$\Longrightarrow A = K[R] \qquad 2.4$$

Where matrix A is composed of the first three columns of matrix P, vector $[a]$ is the last column of matrix P, K is the three-by-three upper triangular matrix with intrinsic parameters and matrix R is the three-by-three orthogonal rotation matrix. If the rows of the matrices A and R are represented by A_i and R_i respectively, then:

$$A = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix}, \qquad K = \begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & f \end{bmatrix}, \qquad R = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix},$$

$$A = K[R] \Leftrightarrow$$

$$\Leftrightarrow \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} = \begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & f \end{bmatrix} \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} aR_1 + bR_2 + cR_3 \\ dR_2 + eR_3 \\ fR_3 \end{bmatrix}$$

Solving for $R_i,\ for\ i = 1,2,3$ we get intrinsic matrix K.

$$K = \begin{bmatrix} a/_f & b/_f & c/_f \\ 0 & d/_f & e/_f \\ 0 & 0 & 1 \end{bmatrix}$$

2.3.1.2 Zhang's Calibration method

Commonly used today, camera calibrations are based on a technique developed by Zhengyou Zhang. Zhang proposed a new flexible calibration method by using known constraints and assumptions. [27][28]

Considering still the pinhole model, 2D points are written as $\tilde{m} = [u \quad v \quad 1]^T$ and 3D points are written as $\tilde{M} = [X \quad Y \quad Z \quad 1]^T$, resulting in the model equation $s\tilde{m} = A[R \quad t]\tilde{M}$. In this equation, s is the arbitrary scale factor, $[R \quad t]$ is the extrinsic matrix composed of rotation matrix and translation vector, and A is the intrinsic matrix containing the all the parameters to be determined.

$$A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

In this notation, $(u_0 \quad v_0)$ are the principal point's coordinates, $(\alpha \quad \beta)$ are the scale factors in the u and v axes respectively and the γ parameter is the image's skew.

This method starts by considering the restraints of the intrinsic parameters determined from a single chessboard pattern plane. Assuming the model plane is on $Z = 0$ from the world coordinate system, we can rewrite the model in a simpler form, like such:

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A[r_1 \quad r_2 \quad r_3 \quad t]\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \Rightarrow s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A[r_1 \quad r_2 \quad t]\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \qquad 2.5$$

For each image of the chessboard, with $H = A[r_1 \quad r_2 \quad t]$, the equation $s\tilde{m} = H\tilde{M}$ relates the chessboard point's world coordinate, $\tilde{M}$, to their respective image coordinates, $\tilde{m}$. Using these points detected from the chessboard, the homography matrix, H, is estimated applying singular value decomposition and then the results are optimized through the Levenberg-Marquart method [23].

Knowing the values of the matrix H the intrinsic parameters must be extracted. This extraction is done is the following steps:

- Exploit the constraints regarding, k, r1 and r2;
- Define a matrix $B = A^{-T}A^{-1}$ generating a new homogeneous linear system;
- Solve the new homogeneous linear system for B;
- Decompose matrix B with the Cholesky decomposition.

By expressing $H = [h_1 \quad h_2 \quad h_3]$ we get the equation $[h_1 \quad h_2 \quad h_3] = \lambda A[r_1 \quad r_2 \quad t]$. Using the known constraints of the r_1 and r_2 orthonormality we get two basic constraints:

$$\begin{cases} h_1^T A^{-T} A^{-1} h_2 = 0 \\ h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \end{cases} \quad 2.6$$

Defining $B = A^{-T}A^{-1}$, where B can be defined by vector $b = [B_{11} \quad B_{12} \quad B_{22} \quad B_{13} \quad B_{23} \quad B_{33}]^T$ we can arrive at the following equations,

$$\begin{cases} h_1^T B h_2 = 0 \\ h_1^T B h_1 - h_2^T B h_2 = 0 \end{cases} \Rightarrow \begin{cases} v_{12}^T b = 0 \\ v_{11}^T b - v_{22}^T b = 0 \end{cases} \quad 2.7$$

Where $v_{ij} = [h_{i1}h_{j1} \quad h_{i1}h_{j2} + h_{i2}h_{j1} \quad h_{i2}h_{j2} \quad h_{i3}h_{j1} + h_{i1}h_{j3} \quad h_{i3}h_{j2} + h_{i2}h_{j3} \quad h_{i3}h_{j3}]^T$, establishing a matrix $V = \begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix}$ for each chessboard image. Using at least three chessboard images we can stack the matrix V and solve $Vb = 0$, through singular value decomposition, to obtain b and hence B. Finally, matrix A is calculated by applying the Cholesky decomposition [23] to the matrix B, obtaining the intrinsic parameters.

2.3.2 Extrinsic Calibration

The extrinsic parameters are the external specifications of the image sensor at the time of the acquisitions. These parameters are the position and orientation of the camera's reference frame relative to the world frame. The position is noted in the form of a three-by-one vector containing the cartesian coordinates of the frames origin in relation to the world frame, Figure 9, and the orientation is noted by a three-by-three matrix describing the rotation of the image sensor frame in relation to the world frame, Figure 8.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

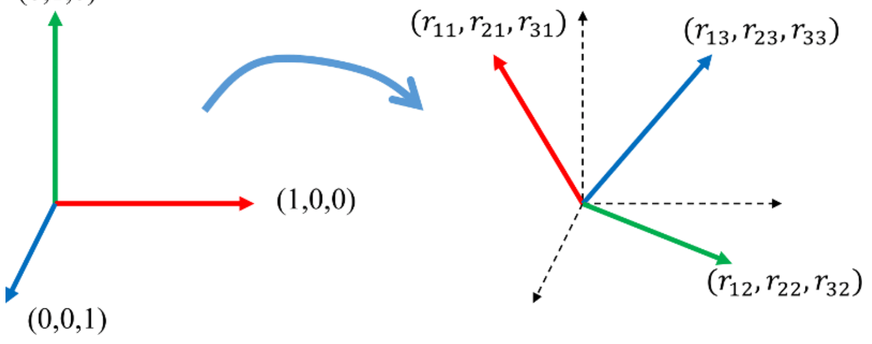

(0,1,0)
(1,0,0)
(0,0,1)
(r_{11}, r_{21}, r_{31})
(r_{13}, r_{23}, r_{33})
(r_{12}, r_{22}, r_{32})

Figure 8: Rotation Parameters [29]

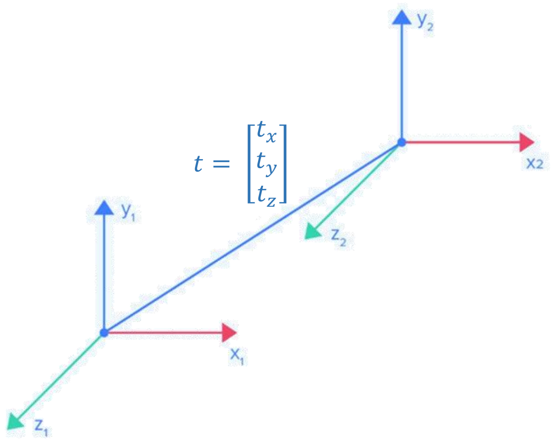


Figure 9:Translation Parameters [30]

Determining these extrinsic parameters is accomplished by solving the Perspective-n-Point problem, commonly referred to as PnP problem. Usually, algorithms used to solve PnP problem require already known intrinsic parameters, therefore a camera calibration is usually executed beforehand. Basically, this problem consists of using a set of known 3D points and their corresponding 2D points in the image, as seen in Figure 10, to estimate the extrinsic parameters of a calibrated camera.

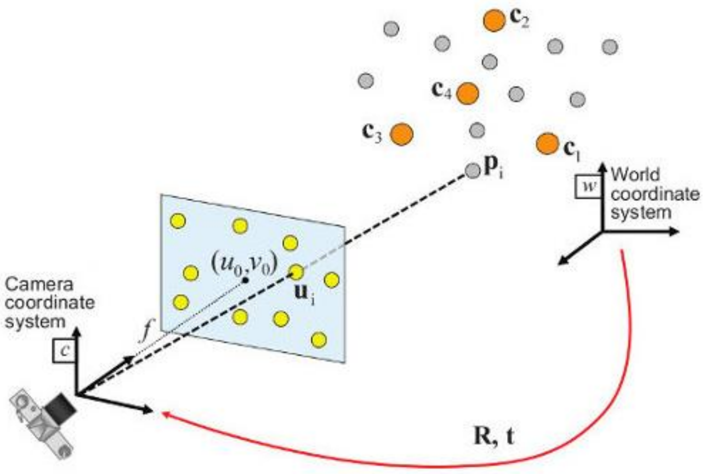


Figure 10: Perspective-n-Points problem [31]

The pose of a camera has overall six degrees-of-freedom. Three degrees-of-freedom come from the camera orientation which consists of roll, pitch and yall rotations and the last three degrees-of-freedom come from the cameras position consisting of translations in each of the 3D world frame's axis. Each point correspondence resolves two degrees-of-freedom, therefore, to reach a solution to the PnP problem, at least three pairs of corresponding points are needed. It should be noted that a PnP problem doesn't return a singular solution, but rather a set of possible solutions, requiring a post-processing step to determine the best solution from the set. Some algorithms are built to estimate a solution using only four pairs, $n = 4$, three pairs to arrive at a set of four possible solutions and the fourth pair to help determine the best solution, this variant is usually noted as P3P algorithms. However, most algorithms are built to support more than three pairs, $n \geq 3$, the extra pairs are used to optimize the solution by reducing the impact of noise in the data, these are referred by the common PnP notation [32].

To keep within the scope of this thesis, we restricted our search to established and open sourced algorithms, instead of programming algorithms from root. The open source library OpenCV has available various implementations of algorithms for the solution of the PnP problem. We will test and compare this established functions to decide the most efficient method to estimate the extrinsic parameters.

The available OpenCV algorithms [33] are based on the following papers:

- P3P algorithm-Complete Solution Classification for the Perspective-Three-Point Problem [34][32].
- AP3P algorithms-An Efficient Algebraic Solution to the Perspective-Three-Point Problem [35].
- EPnP algorithm-An Accurate O(n) Solution to the PnP Problem [36][32].

The default method used by OpenCV is an iterative algorithm. This algorithm uses $n \geq 4$ points, performs an initial estimation for the pose, through a direct linear transform [23], and iterates this estimation using a Levenberg-Marquardt optimization to minimize the reprojection error.

2.3.2.1 P3P algorithm

The P3P algorithm accesses the problem with an algebraic approach, applying Wu-Ritt's zero decomposition algorithm [37][38]. Figure 11 shows the P3P problem, where P is the center of perspective and A, B and C are the control points.

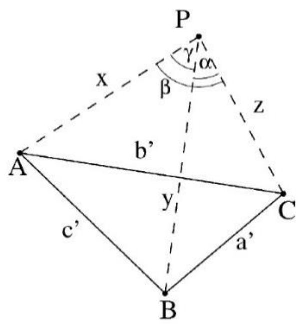


Figure 11: Representation of the P3P problem [34]

With $|PA| = X,\ |PB| = Y,\ |PC| = Z,\ \alpha = \angle BPC,\ \beta = \angle APC,\ \gamma = \angle APB,\ p = 2\cos\alpha\,,\ q = 2\cos\beta\,,$ $r = 2\cos\gamma\,,\ |AB| = c',\ |BC| = a',\ |AC| = b'$ we can write the P3P equation system from the triangles PBC, PAC and PAB:

$$\begin{cases} Y^2 + Z^2 - YZp - a'^2 = 0 \\ Z^2 + X^2 - XZq - b'^2 = 0 \\ X^2 + Y^2 - XYr - c'^2 = 0 \end{cases} \qquad 2.8$$

The following conditions are assumed so that the solutions for the control points are physical:

$$\begin{cases} X > 0,\ \ Y > 0,\ \ Z > 0,\ \ a' > 0,\ \ b' > 0,\ \ c' > 0 \\ a' + b' > c',\ \ a' + c' > b',\ \ b' + c' > a' \\ 0 < \alpha, \beta, \gamma < \pi, 0 < \alpha + \beta + \gamma < 2\pi \\ \alpha + \beta > \gamma, \alpha + \gamma > \beta, \gamma + \beta > \alpha \\ I_0 = p^2 + q^2 + r^2 - pqr - 1 \neq 0 \end{cases}$$

The equation system is simplified by setting $X = xZ, Y = yZ, |AB| = \sqrt{v}Z, |BC| = \sqrt{av}Z, |AC| = \sqrt{bv}Z$. With $Z = |PC| \neq 0$ the system of equations becomes:

$$\begin{cases} y^2 + 1 - yp - av = 0 \\ x^2 + 1 - xq - bv = 0 \\ x^2 + y^2 - xyr - v = 0 \end{cases}$$

The system is reduced to two equations by eliminating v resulting in:

$$\begin{cases} p_1 = (1-a)y^2 - ax^2 - py + arxy + 1 = 0 \\ p_2 = (1-b)x^2 - by^2 - qx + brxy + 1 = 0 \end{cases} \qquad 2.9$$

To solve this P3P system of equations, the positive solutions of the two quadratic equations must be determined. This mean the P3P problem can have infinite solutions or four solutions at most. The Wu-Ritt's zero decomposition method can be used to represent the zero set of a polynomial equation system as the union of zero sets of equations in triangular form, like such:

$$f_1(u.x_1) = 0, f_2(u.x_1, x_2) = 0, \ldots, f_p(u.x_1, \ldots, x_p\ ) = 0,$$

With u being a set of known parameters and x being the unknown variables. Solutions for an equation system in triangular form are well-determined, easily reduceable to the solution of univariate equations. Considering PS as a polynomial set and I as a polynomial, let Zero(PS) be the set of solutions of the equation system $PS = 0$, and $Zero\left(\frac{PS}{I}\right) = Zero(PS) - Zero(I)$. The earlier assumption, $I_0 \neq 0$, can help simplify the computation, considering $Zero(\frac{ES}{I_0})$ we decompose it with Wu-Ritt's method into 10 disjoint components, as such:

$$Zero\left(\frac{ES}{I_0}\right) = \bigcup_{i=1}^{10} C_i. \qquad 2.10$$

Where $C_i = Zero\left(\frac{TS_i}{T_i}\right), i = 1, \ldots 9$ and $C_{10} = Zero\left(\frac{TS_{10}}{T_{10}}\right) \cup Zero\left(\frac{TS_{11}}{T_{11}}\right)$, where T_i are polynomials and TS_i are polynomials equations in triangular form. The decomposition allows us to make the following observation:

- Given that the solutions for each triangular set are well-determined, this decomposition provides the complete set of analytical solutions for the P3P problem.
- Within the assumptions made earlier, there are at most four different solutions, as seen in Table 1.
- This decomposition proves to be a complete and robust method to determine the solutions for the P3P problem.

Table 1: Maximal number of solutions for each component

$\mathbf{C}_i, i =$	1	2	3	4	5	6	7	8	9	10
NO. of solutions	4	3	2	2	1	1	1	2	4	3

2.3.2.2 AP3P algorithm

Using the positions, ${}^{G}p_i$, of three known features, $f_i, for\ i = 1{,}2{,}3$, expressed in frame G, and the corresponding bearing measurements, ${}^{C}b_i$, the AP3P algorithms estimates the position, ${}^{G}p_C$, and orientation, ${}^{G}_{C}C$, of camera, C.

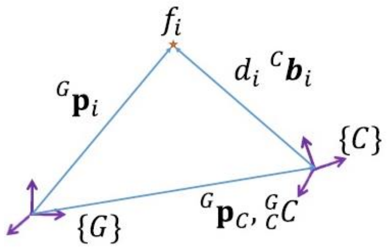


Figure 12: AP3P notation [35]

From Figure 12, we see the geometry of the problem to be solved for each point used in calibration.

$$ {}^{G}p_i = {}^{G}p_C + d_i {}^{G}_{C}C\, {}^{C}b_i, for\ i = 1, 2, 3 \qquad 2.11 $$

Where $d_i = \|{}^{G}p_C - {}^{G}p_i\|$ being the distance from the camera position to the feature, f_i, position. The algorithm first solved the problem for the orientations and then solves for position using the estimated orientation. To solve for orientation the position is eliminated from the previous equation, resulting in a system of three equations:

$$ \begin{cases} ({}^{G}p_1 - {}^{G}p_2)^T {}^{G}_{C}C({}^{C}b_1 \times {}^{C}b_2) = 0 & 2.12 \\ ({}^{G}p_1 - {}^{G}p_3)^T {}^{G}_{C}C({}^{C}b_1 \times {}^{C}b_4) = 0 & 2.13 \\ ({}^{G}p_2 - {}^{G}p_3)^T {}^{G}_{C}C({}^{C}b_2 \times {}^{C}b_3) = 0 & 2.14 \end{cases} $$

Each of the equations in this system is factorized with, ${}^{G}_{C}C = C(k_1, \theta_1)C(k_2, \theta_2)C(k_3, \theta_3)$, where:

$$ k_1 \triangleq \frac{{}^{G}p_1 - {}^{G}p_2}{\|{}^{G}p_1 - {}^{G}p_2\|}, \qquad k_2 \triangleq \frac{{}^{C}b_1 \times {}^{C}b_2}{\|{}^{C}b_1 \times {}^{C}b_2\|}, \qquad k_1 \triangleq \frac{k_1 \times k_3}{\|k_1 \times k_3\|} $$

From the factorization of 2.12 results θ_2, while the factorizations of 2.13 and 2.14 lead to

$$ u_i^T C(k_1, \theta_1)C(k_2, \theta_2)C(k_3, \theta_3)v_i = 0 $$

Where $u_i \triangleq {}^{G}p_1 - {}^{G}p_3,\ v_i \triangleq {}^{C}b_1 \times {}^{C}b_3, for\ i = 1, 2.$ The results of these factorizations are simplified further through several substitutions, with the goal of eliminating θ_3 and arriving at a quadratic polynomial involving a trigonometric function of θ_1. Solving in closed form for the roots of this quadratic polynomial, we get for each root two solutions for the rotation matrix, ${}^{G}_{C}C$.

The rotation matrix is then recovered through the following method. Considering we have

$$ {}^{G}_{C}C = C(k_1, \theta'_1)C(k''_3, \theta_3)C(k_1, \phi)C(k_2, \theta_2) \qquad 2.15 $$

where $k'_3 \triangleq C(k_2, \theta_2)k_3 = k_2 \times k_1$, $\theta'_1 \triangleq \theta_1 - \phi$ and $k''_3 \triangleq C(k_1, \phi)k'_3$. Since k_1 and k_1 are perpendicular to one another a rotation matrix can be constructed such that

$$ \bar{C} = [k_1 \quad k''_3 \quad k_1 \times k''_3] $$

Therefore, $k_1 = \bar{C}e_1$, $k''_3 = \bar{C}e_2$ where $[e_1 \quad e_2 \quad e_3] \triangleq I_3$. Substituting k_1 and k''_3 in 2.15, we get

$$ {}^{G}_{C}C = \bar{C}C(e_1, \theta'_1)C(e_2, \theta_3)\bar{\bar{C}} $$

Where

$$ \bar{\bar{C}} \triangleq C(e_2, \theta_3 - \theta'_3)\bar{C}^T C(k_1, \phi)C(k_2, \theta_2) = $$

$$ = C(e_2, \theta_3 - \theta'_3)[k'_1 \quad k_3 \quad k'_1 \times k_3] = $$

$$ = [{}^{C}b_1 \quad k_3 \quad {}^{C}b_1 \times k_3] $$

With the rotation matrix calculated we proceed to solving for the position. By substituting in 2.11 the expression for d_i yields

$$ {}^{G}p_C = {}^{G}p_C - \frac{\delta \sin \theta'_1}{k_3^T {}^{C}b_3} {}^{G}_{C}C {}^{C}b_3 $$

Where

$$ \delta \triangleq \sqrt{(u_1^T k'_3)^2 + (u_1^T k_2)^2} = \|u_1 \times k_1\| $$

2.3.2.3 EPnP algorithm

As mentions before, the redundancy in having more points than necessary is used to reduce the impacts of noise, however using larger data sets also creates a new problem, processing this data becomes a much more complex procedure requiring heavier computation. For this reason, it becomes necessary to use a more efficient algorithm to solve PnP problems.

The Efficient PnP (EPnP) technique's approach is to express all 3D points as a weighted sum of four virtual control points, reducing the complexity to $O(n)$ while keeping a good accuracy of the estimation. With the provided coordinates for the 3D points and the corresponding 2D image projections, the first step is to retrieve their coordinates in the camera coordinate system. Given that the central idea is to express all the point's coordinates as a weighted sum of four non-coplanar virtual control points, it means that the coordinates of the control points in the camera coordinate system become the unknown in the problem. The efficiency of this technique comes from the notion that for large n's cases, it becomes a much smaller number of unknowns compared to the n depth values that conventional methods work with. Noting the reference points as, $p_i \; for \; i = 1, \dots, n$ and the control points as, $c_j \; for \; j = 1, \dots, 4$, we can express the reference points as a weighted sum of the control points in the world and camera coordinate systems, with the following expressions:

$$p_i^w = \sum_{j=1}^{4} \alpha_{ij} c_j^w \, , with \sum_{j=1}^{4} \alpha_{ij} = 1$$
$$p_i^c = \sum_{j=1}^{4} \alpha_{ij} c_j^c \qquad 2.16$$

Where the α_{ij} are homogeneous barycentric coordinates, uniquely defined and easy to estimate. The superscripts ^{w} and ^{c} indicate the frame in which the points coordinates are expressed, world or camera coordinate system respectively. While the control points can be randomized, taking the centroid of the reference points as one control point, and setting the rest in order to form a basis aligned with the principal directions of the data, results in a more stable method. This choice is similar to conditioning the linear system of equations, presented below, through the normalization of the points coordinates similarly to the one recommended for the classic DLT method.

$$\forall i, w_i \begin{bmatrix} u_i \\ 1 \end{bmatrix} = A p_i^c = A \sum_{j=1}^{4} \alpha_{ij} c_j^c$$

In this expression, A is the camera intrinsic calibration matrix, u_i are the 2D image projections of the reference points, p_i for $i = 1, \dots, n$ and w_i are scalar projective parameters. This equation can be rewritten considering the 3D coordinates for each c_j^c control point, noted as $[X_j^c \quad Y_j^c \quad Z_j^c]^T$, the 2D coordinates of the u_i projections, noted as $[u_i \quad v_i]^T$, the f_u, f_v focal length coefficients and the principal point coordinates, (u_c, u_v), which are present in the matrix A:

$$\forall i, w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & u_c \\ 0 & 0 & 1 \end{bmatrix} \sum_{j=1}^{4} \alpha_{ij} \begin{bmatrix} x_j^c \\ y_j^c \\ z_j^c \end{bmatrix} \qquad 2.17$$

The 12 control points coordinates $\left(x_j^c,\ y_j^c,\ z_j^c\right) for\ j = 1, \dots, 4$ and the n projective parameters $w_i\ for\ i = 1, \dots, n$ are the unknowns of this linear system. It can be deduced from the last row of the above system of equations that $w_i = \sum_{j=1}^{4} \alpha_{ij} z_j^c$, applying this expression to the other two rows results in the following two linear equations for each reference point:

$$\sum_{j=1}^{4} \alpha_{ij} f_u x_j^c + \alpha_{ij}(u_c - u_i) z_j^c = 0$$
$$\sum_{j=1}^{4} \alpha_{ij} f_v y_j^c + \alpha_{ij}(v_c - v_i) z_j^c = 0 \qquad 2.18$$

Since there are no projective parameters w_i in these equations, a linear system based on the concatenation of the two equations can be established for all n reference points:

$$Mx = 0$$

In the equation above, M is a 2n-by-12 matrix consisting of the coefficients of the two linear equations above, for each reference point, while $x = \begin{bmatrix} c_1^{c\,T} & c_2^{c\,T} & c_3^{c\,T} & c_4^{c\,T} \end{bmatrix}^T$ is a vector made of the 12 unknowns. The solution belongs in the null space, or kernel, of M. A final step is a Gauss-Newton optimization that can be performed to increase the accuracy of the solution with minimal computational cost.

Chapter 3

3 Solutions Proposal

3 Solutions Proposal

With the study conducted in the literary review we can assess the advantages and disadvantages of different scanner categories. Given that the stone blocks have dimensions in the range of two to four meters in length, width and height, designing a contact scanner with a work area large enough to fit the stone blocks would not only be economically and spatially costly but the scanning process would also be too time consuming to be applicable. Non-contact active scanners require a measure of control over the working environment. Therefore, a passive scanner capable of acquiring the data for a quality 3D reconstruction is the most viable solution answer to the first complementary question, Q1.

Currently the workers determine the location of the cuts simply by visually inspecting the blocks for flaws. One of our objectives will be to optimize this part of the cutting process. Designing a system capable of scanning the stone blocks for the purpose of reconstructing a three-dimensional model granting the workers the ability to analyze the model using computer software. This allows the quarry to optimize the amount of stone slabbed saved by calculating the positions of the cuts with a higher accuracy. Another objective will be to optimize the advertising quarry methods. Designing another system capable of scanning a processed block for the acquisition a 3D model for advertisement purposes, this allows quarries to show their customers the full volume and surface quality of their stone blocks if a more comprehensive medium.

3.1 Considered scanner configurations

The method used for three-dimensional reconstruction uses as main inputs the images of the object, the intrinsic and extrinsic parameters are optional inputs. The only hardware required to obtain all these inputs are image sensor and a computer. The images for the purpose of the reconstruction must be well-illuminated and sufficiently high resolution for a good quality 3D model and reasonable low resolution to avoid lengthy reconstructions. The intrinsic and extrinsic parameters are computed through distinct calibration processes using images containing easily identifiable points of known coordinates.

3.1.1 Linear scanning system configuration

This configuration is based on the idea to use the known motion of the Fravizel's cutting machine. The cutting machine consists of a portico which supports the cutting instrument, this portico is installed on top of rails allowing it to move over the stone blocks. Figure 13 shows a simple CAD drawing of the portico with the cameras field of views exemplified. The green structure is the portico and the black arrows indicate the motion allowed by the rails.

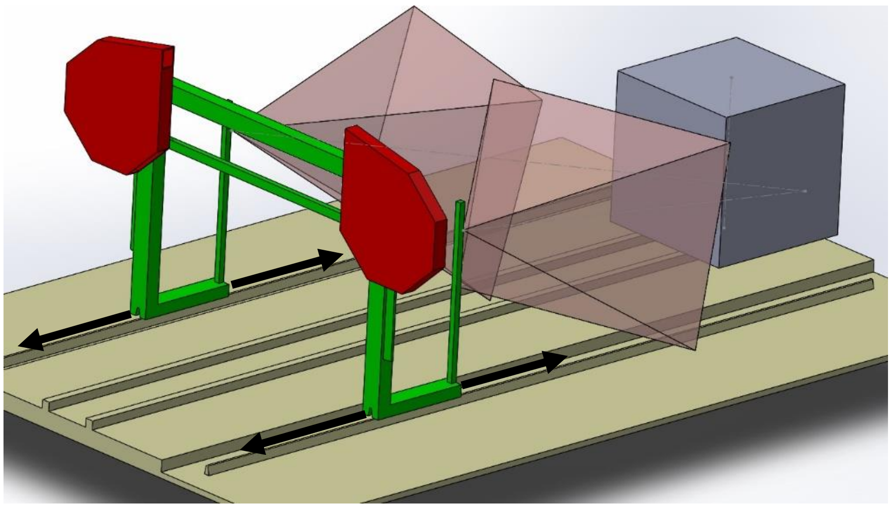


Figure 13: 3D CAD model of the linear scanner configuration

The portico rails are encoded and can be used to accurately perform acquisitions at defined intervals of distance. This way we can define a world frame with one of its axis colinear to the portico's rails, so that the motion provided to the camera would only alter one the extrinsic parameter corresponding to the axis that is colinear with the rails, maintaining all others constant. The full extrinsic parameters of each image can be easily determined with a single pose estimation to determine all the extrinsic parameters of the initial acquisition and the rails motion to calculate the only positional coordinate that changes between acquisitions. Figure 14 shows a 3d graph where the black dot represents the origin of the world frame, the blue markers are the known points used for initial pose estimation and the colored dots and lines represent the cameras position and orientation.

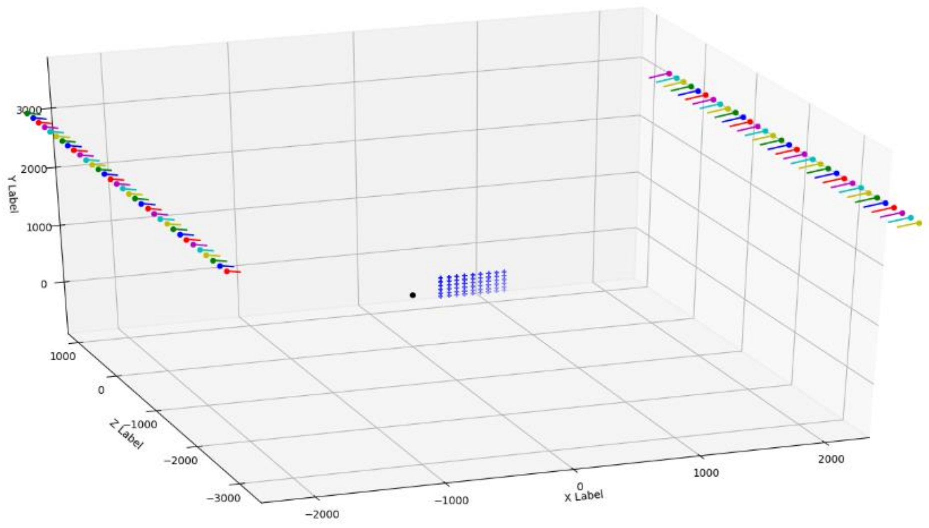


Figure 14: 3D graph plot of the extrinsic calibration

3.1.2 Circular scanning system configuration

Considering the advantages provided by a 3D reconstruction, this seems a medium through which the advertisement limitations could be overcome, answering the question, Q3. A 3D model of the finished limestone block would be possible to display on websites through a 3D displayer. This would allow possible buyers to rotate and zoom in and out of the block to obtain a more comprehensive appreciation of its dimensions and surface quality without having to sacrifice either. This configuration is based on this desire to obtain an accurate full gapless 360-degree view of the stone blocks that quarries intend to sell. The pictures are obtained with cameras positioned in a circle around the object, oriented towards the object, this ensures that all surfaces are captured and that there is significant overlap between consecutive cameras perspectives. The Figure 15 shows the positions and orientations of a set of images taken around an object to exemplify the desired notion of a circular configuration.

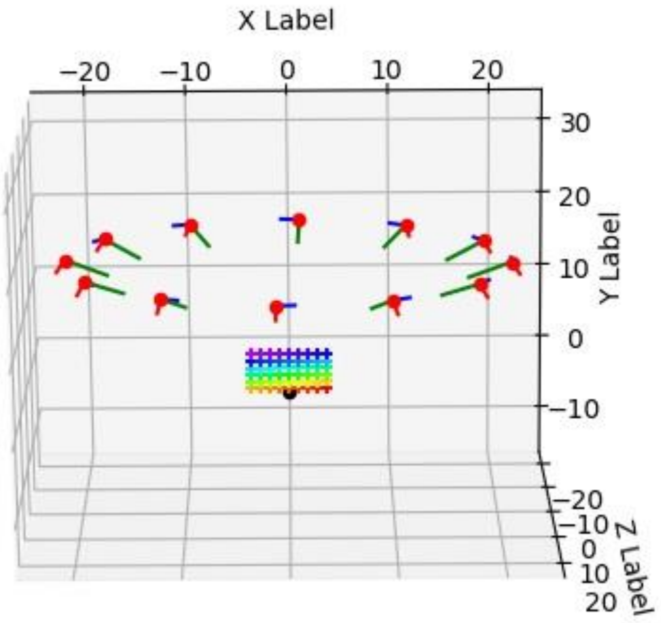


Figure 15: 3D graph plot of an example for the circular scanner configuration

3.2 Hardware configurations

3.2.1 USB webcams and computer

Two Trust Exis webcams were used as image acquisition devices for small scaled simulations of configurations corresponding to desired applications. To simulate a small scale version of the first configuration (linear movement), both cameras were supported by the same homemade tripod, as seen in Figure 16, placed at constant distance from each other and fixed orientations towards the same point between them, where the object is placed. The cameras were connected to one personal computer via USB ports and were accessed via MATLAB software installed with the MATLAB support package for USB webcams. A MATLAB script was created to control the triggering of the cameras. In this configuration, the acquisition of images needs to be synchronized between the cameras to ensure each pair of images corresponds to the same position in the portico’s rail, in the real scale application. The triggering method needs to be highly responsive to ensure the image pairs are captured at consistently regular specified intervals. To test the viability of this configuration the MATLAB scripts accomplished two functions triggering the cameras and timing the triggers. The results of this test allows to conclude that while MATLAB can be used to control and trigger multiple cameras, using one computer to send the triggering signal to multiple cameras results in significant delays making it impossible to synchronize the acquisition of images between multiple cameras, as seen in Table 2. Concluding that synchronicity is not viable, rules out the linear scanning configuration, for this specific hardware.

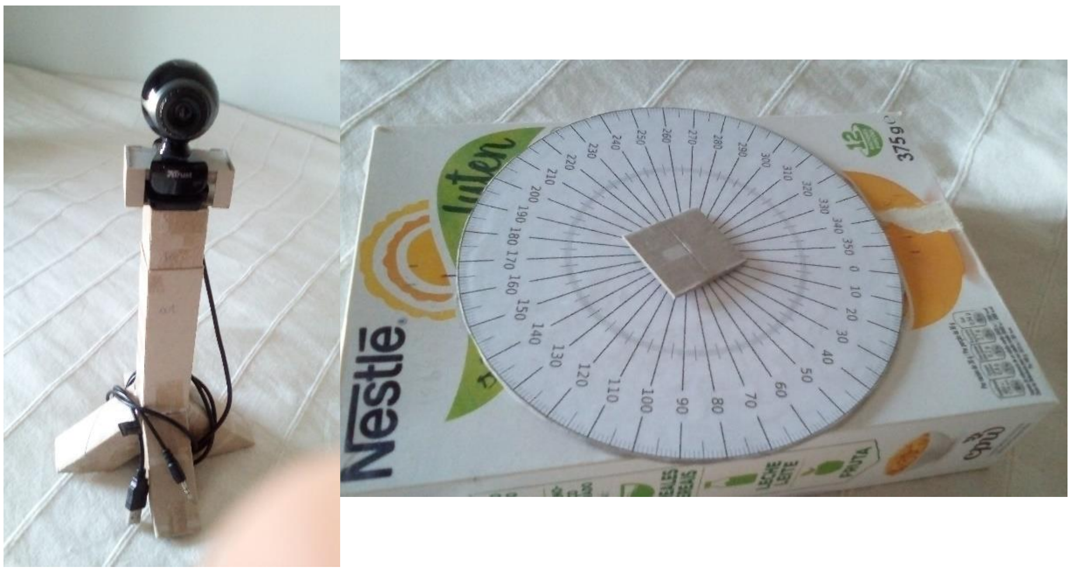


Figure 16: Webcam on makeshift stand and base with rotation palate

Table 2: Timestamps of the camera's triggers

Cam 1 (seconds)	0.2420	1.2022	2.1621	3.2346	4.1943	5.1545	6.2425	7.20235
Cam 2 (seconds)	1.0622	2.0221	3.1104	4.0702	5.0304	6.1025	7.0625	8.0222

To simulate the circular scanning configuration a camera must move in a circle of established radius centered on the object. Images are acquired at a constant specified angular interval for the positions to be known and a full 360º rotation is required to ensure reconstruction of all sides of the object. The maximum allowed angular difference between images supported by the different software used for reconstruction is 30 degrees, this implies a minimum of 12 photos are required to cover the full 360 degrees, this can be seen in Figure 17. To achieve this simulation, the same Trust Exis webcam was used to acquire the images, and the circular motion was achieved by rotating the object itself while maintaining the camera in a known fixed position and unknown fixed orientation. Camera’s position is measured in relation to the objects position and while the camera’s orientation isn’t measured, the cameras center is pointed at the object’s rotation axis. The cameras position can be transformed into the required format later using a viable world coordinate system. The orientation can be calculated using identifiable points in the acquired image with known locations in the same world coordinate system. The object is rotated manually with the help of a platform, as seen in Figure 16, which comes equipped with protractor to be used as a rotating plate where the object would be placed. To ensure this small-scale configuration works, the object was rotated by 30 degrees between each image acquisition. The cameras are triggered manually using a simple windows application and the images are saved as required before using them in the reconstruction software.

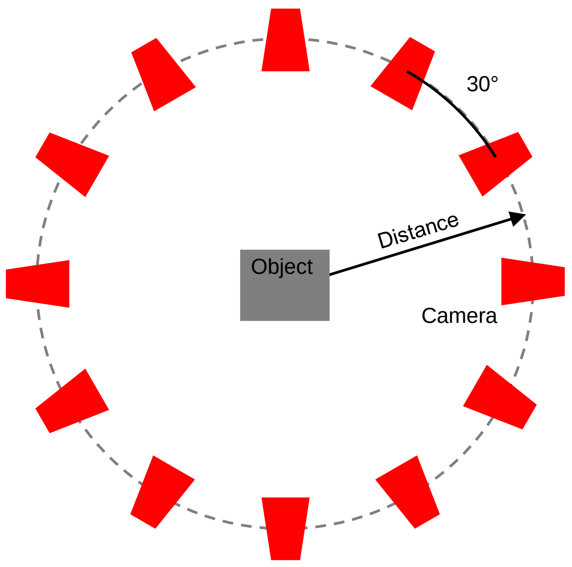
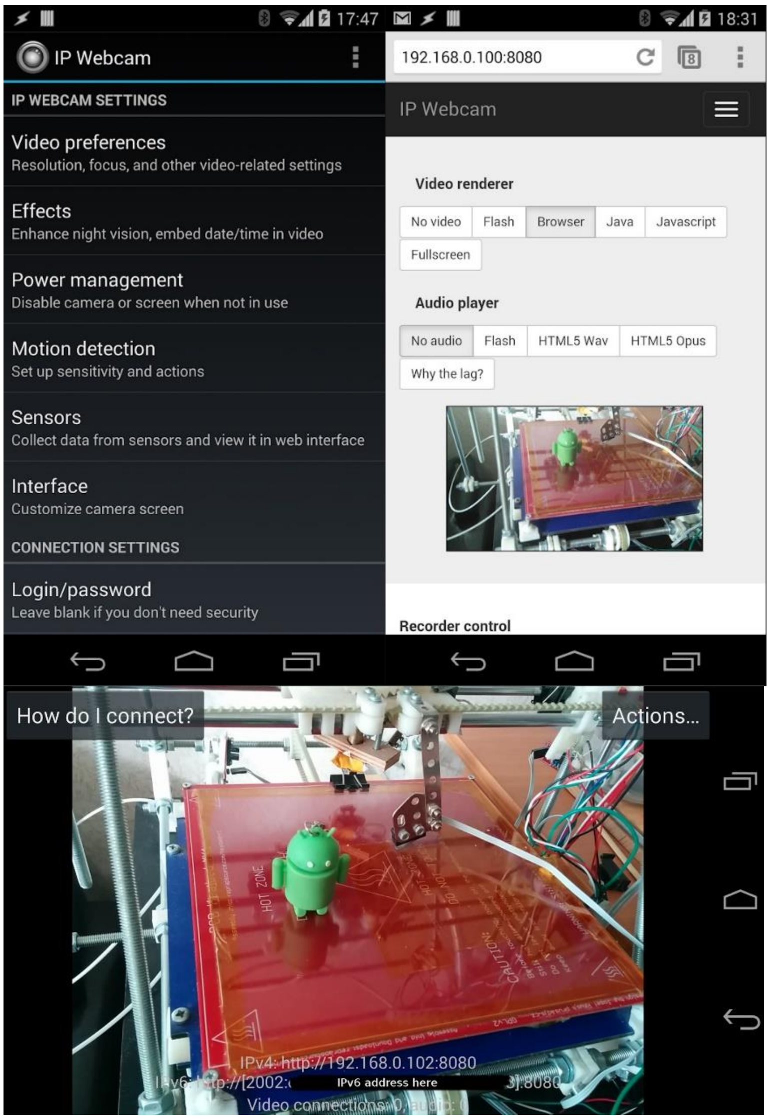


Figure 17: Top view sketch for circular scanner configuration

3.2.2 Computer and remote accessed camera phone

The Trust Exis webcams, while a good option to test the viability of the image acquisition system, produced images with a very low resolution, resulting in a low-quality three-dimensional model. To improve the quality of the reconstructed model, images with higher resolutions are required. While researching for an image sensor of higher resolution, we discovered a simple app, IP Webcam [39], that can be installed in one of our personal mobile telephone allowing us to remotely access its camera for testing purposes, GUI seen in Figure 18. An adjustable phone clamp with suction cup is used to position and support the phone in a desired pose. The suction cup allows the adherence of the phone clamp to a variety of flat surfaces, allowing the positioning of the phone while the clamp's adjustable joint, allows the phone orientation to be set as necessary. The phone's camera was used to capture images and the rotation was simulated, using the same platform to rotate the object while acquiring images at known angular intervals.


17:47

IP Webcam

IP WEBCAM SETTINGS

Video preferences
Resolution, focus, and other video-related settings

Effects
Enhance night vision, embed date/time in video

Power management
Disable camera or screen when not in use

Motion detection
Set up sensitivity and actions

Sensors
Collect data from sensors and view it in web interface

Interface
Customize camera screen

CONNECTION SETTINGS

Login/password
Leave blank if you don't need security

18:31

192.168.0.100:8080

IP Webcam

Video renderer
No video Flash Browser Java Javascript Fullscreen

Audio player
No audio Flash HTML5 Wav HTML5 Opus Why the lag?

Recorder control

How do I connect?

Actions...

IPv4: http://192.168.0.102:8080
IPv6: http://[2002: IPv6 address here]:8080
Video connections: 0. Audio: 0

Figure 18: IPWebcam app GUI [39]

3.2.1 Single-board computer and an 8MP camera

After confirming the necessity and the viability of the images acquired by our phones in the previously described trial configuration, a cost-effective alternative to the use of our personal phones was sought out. This alternative should be able to allow us to create a small-scale version of the scanner we intent to develop, for further testing. The single-board computers developed by the Raspberry Pi foundation [40] have optional accessories and modules, such as an eight-megapixel camera module [41]. A single-board computer is, as it sounds, a fully functional computer built on a single small sized circuit board, without any expansion slots. Since they are computers, however basic, these Raspberry Pis are a very flexible and easy to use tool, they can be used as camera controllers, signal generators and more, allowing a good degree of flexibility for the purpose of testing solutions to our designs. For our purposes, these computers need only enough processing power to manage and trigger one camera, therefore single-board computers are an appropriate cost-effective choice.

Using a Raspberry Pi model 3 B+, shown in Figure 19 (a), as a controller for a V2 eight-megapixel camera module, shown in Figure 19 (b), establishes the optical sensor to be used in the following small-scale versions of the circular scanner configuration designs. In these small-scale circular scanners, the scanned object is a common pavement limestone, shown in Figure 20, as they are a good small-scale approximation of the quarry's limestone block, given that they were a small piece of what was once before a large limestone block. A single camera set up is used to test the cameras optical viability and the Raspberry Pi capabilities. A very simple tripod was used to support the camera module, shown in Figure 21 (a). The assembled camera mounted tripod allows adjusting the position and pose of the camera, Figure 21 (b). Using the OpenCV pose estimation method, the camera pose can be calculated from any position around the object, by providing an image of a chessboard pattern in a known position. To take advantage of this pose estimation method, contrary to previous setups, the object is maintained stationary while the camera is placed around the object. At each position the camera is placed, two images are captured, one with the object and another with the chessboard pattern for pose estimation. The chessboard patterns were attached to a rigid structure of known dimensions to ensure the known coordinates of the pattern's points, shown in Figure 22.

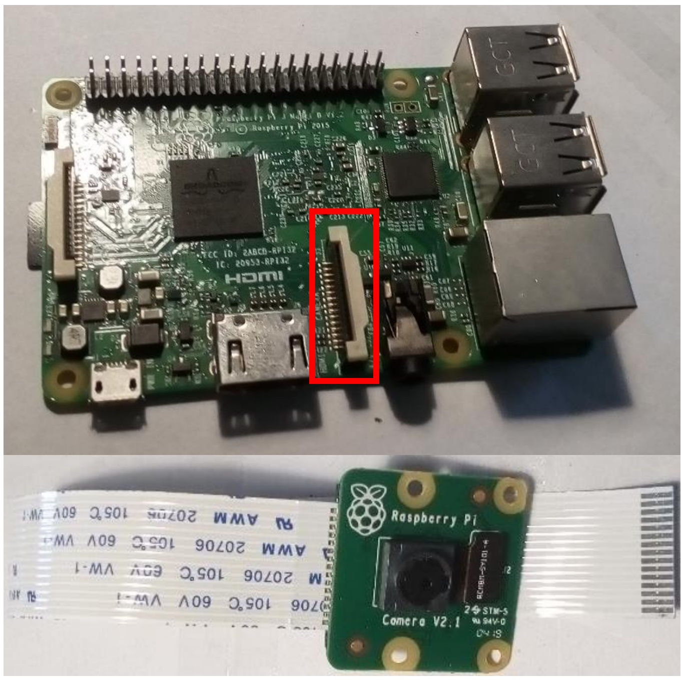


Figure 19: image sensor hardware
(a) Raspberry Pi model 3B+ (Camera port highlighted)
(b) V2 Pi Camera

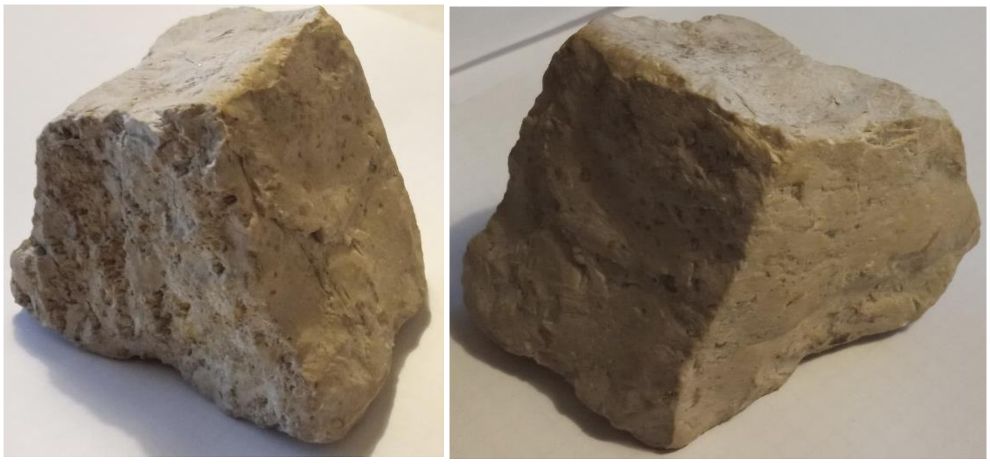


Figure 20: Pavement Stone

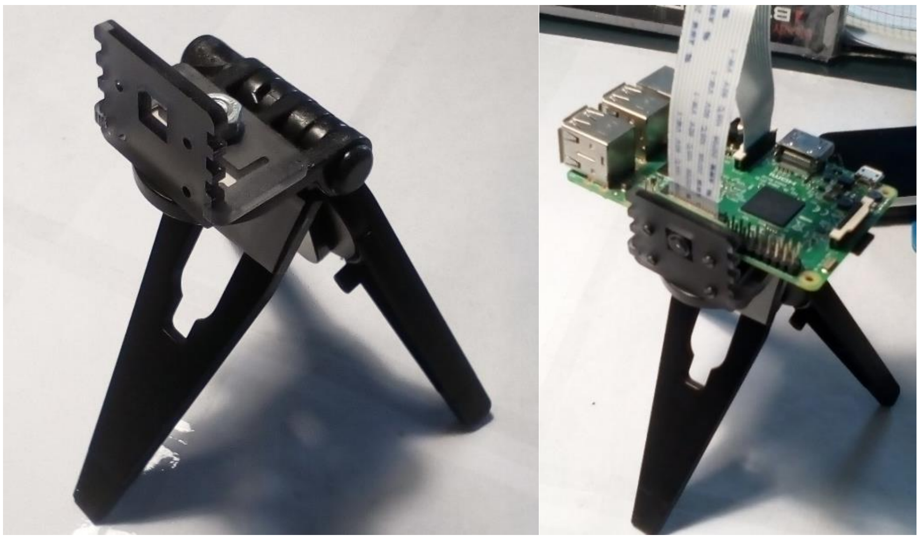


Figure 21: image sensor support
(a) tripod (b) camera mounted tripod

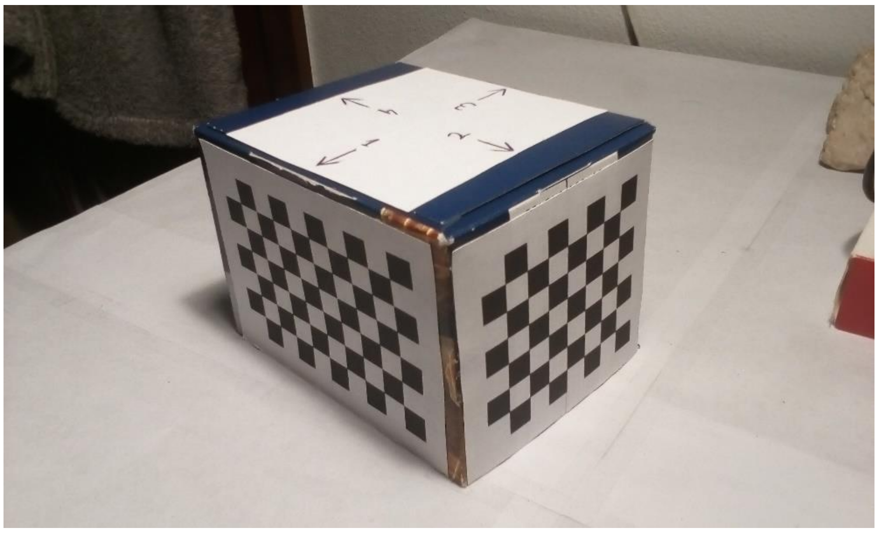


Figure 22: Chessboard Patterns box

3.2.2 Circular scanning configuration with Raspberry Pi

After determining the viability of the single-board computer hardware, the circular scanning configuration is improved. In this configuration, the camera is maintained in a stationary location while the object is rotated between each frame acquisition by a known amount of degrees. For this approach, the extrinsic calibration is accomplished by estimating the pose of the camera with same OpenCV estimation method used before. Performing the extrinsic calibration on the static camera gives us the pose corresponding to the first image, the poses for the subsequent images are acquired by applying basic algebra to rotate the estimated camera pose in accordance with the rotation applied to the object. For instance, if we maintain a stationary camera and we rotate the object by twenty degrees in a clockwise direction between each acquisition, then we can algebraically rotate the initial pose by twenty degrees in a counterclockwise direction, simulating the corresponding inverse scenario. This corresponding scenario would be having the object in a stationary position while the camera moves in a circumference centered on the object, for this example this would mean, the camera is moved, in the circumference, by twenty degrees in a counterclockwise direction between each image acquisition.

This method to obtain images, for the three-dimensional reconstruction, presents a problem; while the object is rotated, the background behind it will remain stationary. The features detection will collect features from the object and the background as well, which will cause conflicts. This problem is solved by applying a background removal process to the images prior to reconstruction, for example, capturing an image of the background and using this image to perform a logical Exclusive OR bitwise operation to all images used for reconstruction. This leaves us with a set of images of the object without background eliminating the detection of the features that would cause problems.

To rotate the object by known intervals a simple device was three-dimensionally drawn and printed. This device is a simple box housing a rotating pad attached to a protractor, as seen in Figure 23.

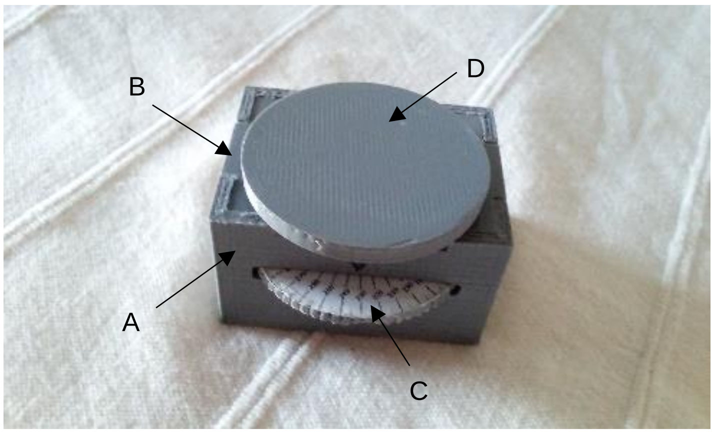


Figure 23: Manually rotating support
A-Base; B-Lid; C-Protractor; D-Platform

This support is comprised of four parts drawn in SolidWorks and printed. The base, the lid, the protractor, and the rotating platform. The object rests on top of the rotating platform, and the protractor is used to rotate and measure the amount of rotation. The base and protractor were printed simultaneously with the lid and platform respectively, Table 3 shows the printing durations.

Table 3: Pieces 3D printing times

Part	Time
Base + Lid	2h 30min
Protractor + Platform	1h 16min
Total	3h 46min

3.2.3 Automated Circular scanning process

The tripod, Figure 21 (b), while viable was not sturdy enough to be a consistently reliable support structure. A better method was required to control the position and orientation of the V2 Pi camera. However, a short flexible flat cable is the only connection between the board and the camera, therefore a case capable of housing both the Raspberry Pi board and the camera module is the best solution. To maintain costs down, a case was designed in SolidWorks, using the boards documents dimensions [42], and 3D printed in PLA instead of purchasing one. Upon some research we discover, common 3D printing errors with large and complex pieces include warping and curling respectively. Considering these common issues, the case was modeled two different ways. The first case was designed in six parts, one for each side of the case, as seen in Figure 24 (a), sacrificing structural integrity for an easier printing process. The case is assembled together with standoffs and nuts keeping the board and camera in place, as shown in Figure 24 (b). This design while also facilitating assembly, by allowing the sides of the case to be attached after the computer and camera board were screwed in place, it introduced issues with the accuracy of the pieces attaching mechanisms. The attaching mechanism between the parts might introduce deviations in the desired positions of the parts and by consequence the access to the board's ports. The method designed to attach each part to one another consisted two mortise and tenon joints between each pair of parts, one part possessed two through mortises, cavities that passes entirely through the part, and the corresponding part possessed two tenon, projections of material on the end of the part, for insertion into the mortise. Usually mortise and tenon joints rely on glue or simple friction between the surfaces to hold the parts in place but in this case, the use of the standoff screws will not only keep the raspberry pi board in place but also close the case and maintain its structure.

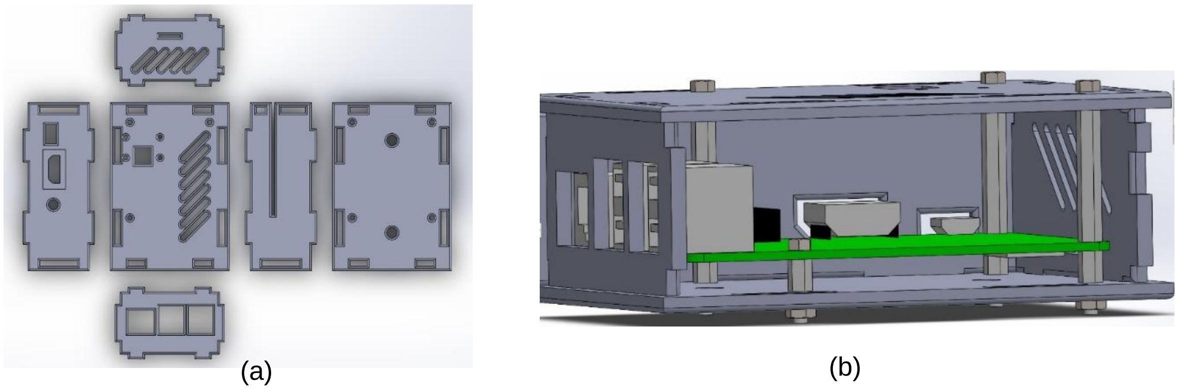


Figure 24: Raspberry Pi case first model
(a) Separate pieces (b) Assembled case

The second designed case consisted of only two parts, the top lid, with the housing for the camera, and the box, with the housing for the computer board, as seen in Figure 25.

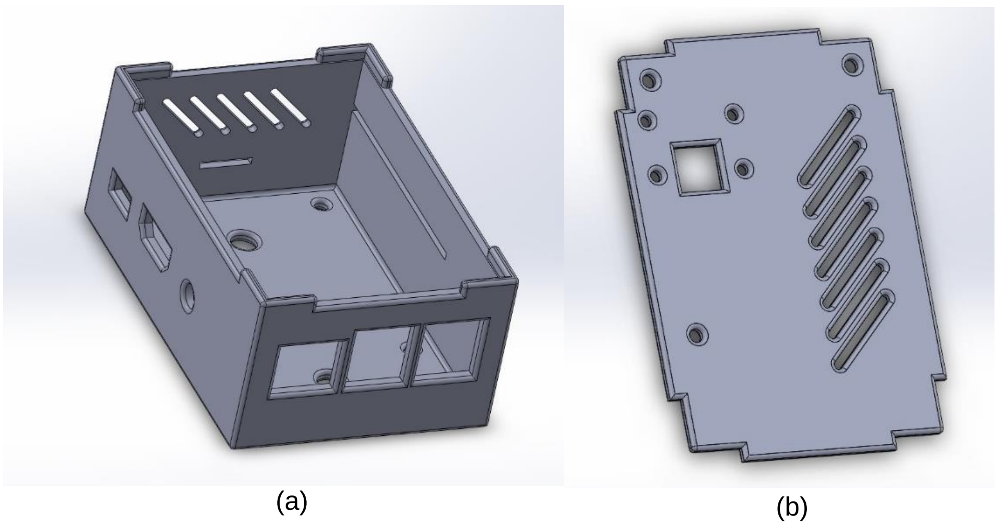


Figure 25: Raspberry Pi case second model
(a) Box (b) Lid

Since the box, that houses the raspberry pi board, was not divided into separate parts the structure of the case was significantly better, and the absence of joints eliminated any errors caused by mismatching swelling in the tenons, as it is unavoidable in 3D printing.

The common issues with warping and curling were occasionally present but minimal and restricted to the corners of the box, which presented no practical problem, and were aesthetically offset by the lack of joints, improving the overall quality of the case, both structurally and aesthetically. Another aspect that was considered when choosing the more appropriate design was the time it took to print. Table 4 shows the time it took for each design to be printed. The second designed proves faster to print by almost an hour, proving to be the preferable design.

Table 4: 3D printing times for each design

1st Design		2nd Design	
Part	Time	Part	Time
Lid	1h40min	Lid	1h20min
Bottom face	1h40min	Box	4h02min
All lateral faces	2h46min		
Total	6h06min	Total	5h22min

Table 5: 3D printing times for pieces common to both designs

Part	Time
Axis Joint 1	1h22min
Axis Joint 2	1h44min
Axis Joint 3	1h10min
3 M2.5x25mm Standoffs	28min
4 M2.5x11mm Standoffs	13min
Slider	5min
Total	5h02min

To position and orient the case in a sturdy way, a kinematic chain, was designed with three degrees of freedom, two rotational joints with axis perpendicular to one another and a prismatic joint, seen in Figure 26. From the base to the tip of this kinematic chain: The first joint is prismatic allowing translations along the vertical axis, Figure 26 (A); the next joint is rotational, Figure 26 (B); the last joint is also rotational, Figure 26 (C). The first rotational joint allows rotation of the case around a vertical axis, turning the camera left and right (yaw rotation), Figure 27 (b), the second rotational joint allows rotation of the case around a horizontal axis, tilting the camera up and down (pitch rotation), Figure 27 (a), and the prismatic joint allows translation of the case along a vertical axis. Three bones of the arm were designed in SolidWorks and 3D printed, the forth bone consists of a aluminum shaft with a squared cross-section, Figure 28.

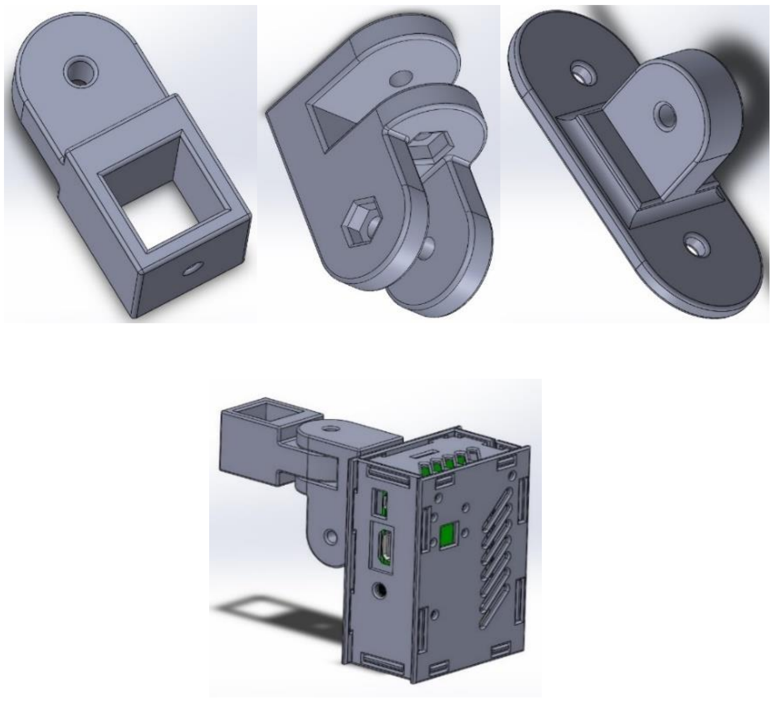


Figure 26: Raspberry Pi support kinematic chain

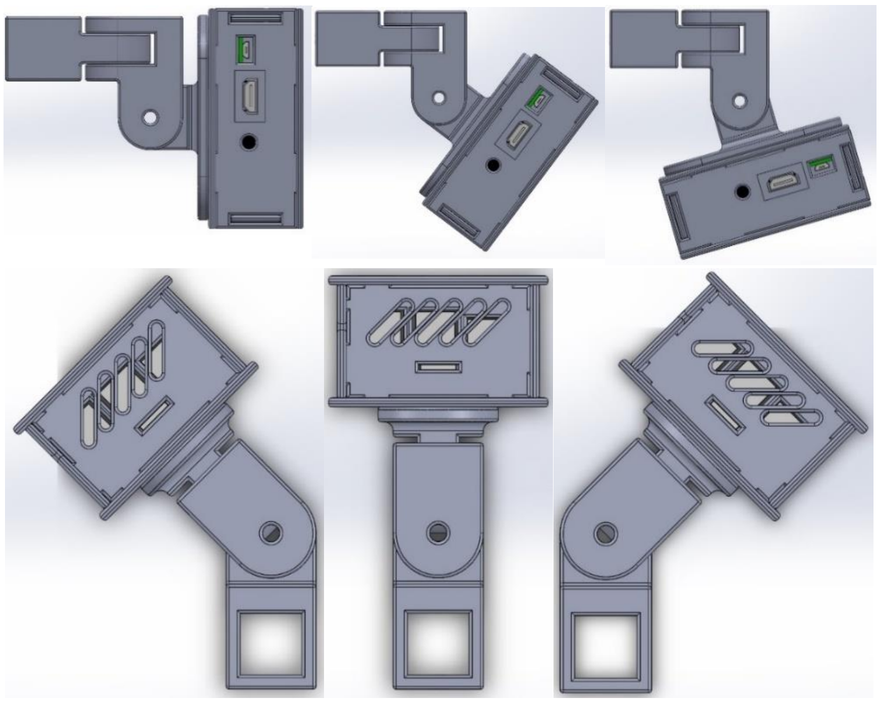


Figure 27: Rotational degrees of freedom motion

The vertical movement, provided by the last axis, is accomplished through the squared shaped slot, which accommodates the aluminum shaft. A T-nut and bolt allows the tightening of the prismatic joint to the aluminum shaft to lock it at any height along its length.

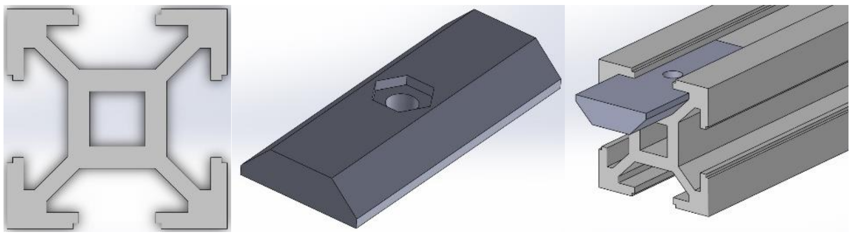


Figure 28: T-slot aluminum beam and T-nut CAD models

While the prismatic joint gives control over the vertical coordinate, the other positional coordinates are control by choosing where to position the aluminum shaft. To maintain the beam in a vertical position a desk clamp with housing for the beam was designed in SolidWorks, Figure 29.

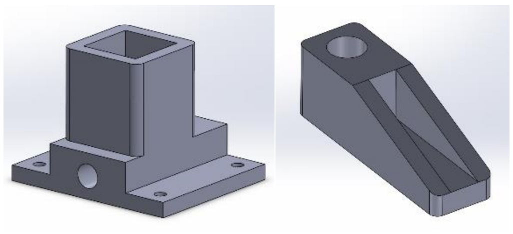


Figure 29: Clamp pieces CAD models

The designed clamp parts were then 3D printed, Figure 30 (a). The clamp parts were connected using an L-shaped screw and bolt, as shown in Figure 30 (b).

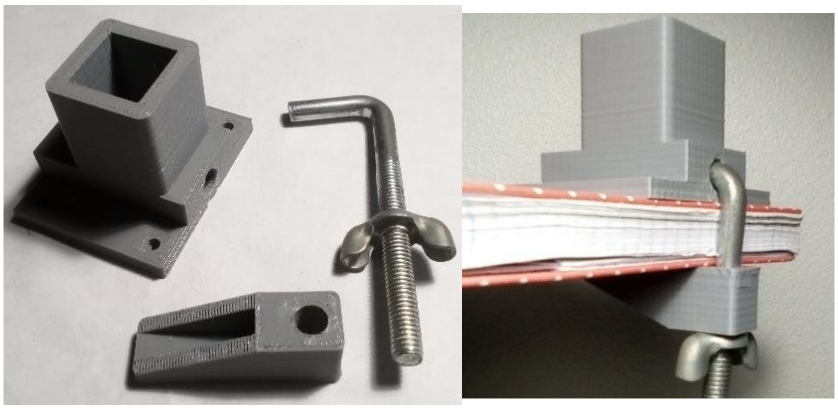


Figure 30: Custom Clamp
(a) 3D printed parts (b) Assembled clamp

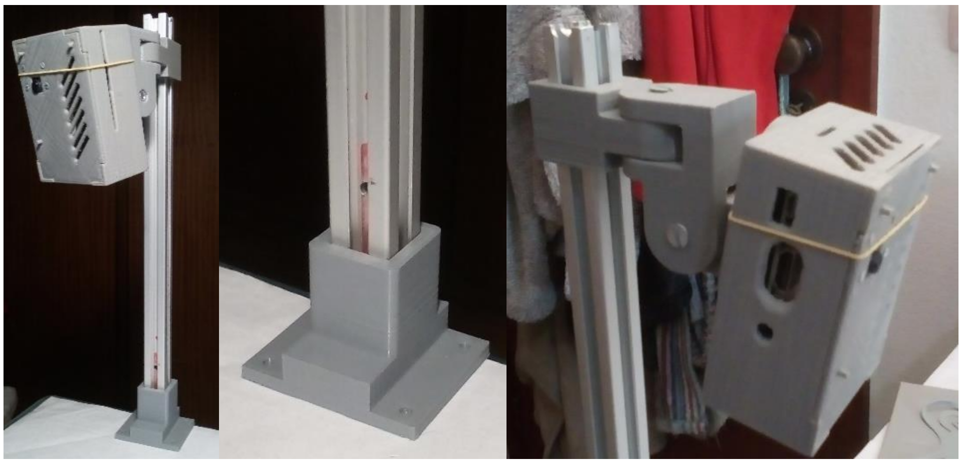


Figure 31: Raspberry Pi case and support structure clamped on a desk

In addition to controlling the image acquisition, the same computer can be programmed to control a servomotor, with the goal of rotating the object. Attaching a servomotor to a support plate creates a simple rotating bed for the object, allowing the automation of the entire circular scanning process.

This hardware set up uses one image sensor, supported by the previously described structure, as shown in Figure 31, maintained in a static position and a servo to rotate the object. To obtain a full reconstruction without gaps, the set of images used must view the object from enough directions as to eliminate blind spots. Since a single camera is used in this set up, a servo capable of 360-degree rotation is required.

There are most commonly two types of servomotors, positional servomotors and continuous servomotors. The positional servomotors receive a PWM control signal, which translates to the angular position of the output shaft within a 180-degree range. The continuous servomotors, the PWM control signal sets the speed and direction of the shaft's rotation instead of its angular position, as it happens in traditional positional servos. In continuous servos the output shaft is capable of rotating the full 360-degrees indefinitely, hence the name [43].

For our set up the Tower Pro MG995 continuous servomotor was used [44]. As most servos, MG995 comes with three wires, orange, red and brown. As shown in Figure 32, the servo has an orange wire for pulse width modulated (PWM) input, a red wire for voltage common collector (VCC) input and a brown wire for grounding. Using the raspberry pi to power the servo, is ill advised so an external power supply was used, a simple four AA battery support with a VCC and Ground wires can be used to power the servo, while the PWM input signal is generated by the Raspberry Pi. Considering that the servo must be grounded with the power supply and the signal generator, the components must be connected according to the electrical diagram shown in Figure 33.

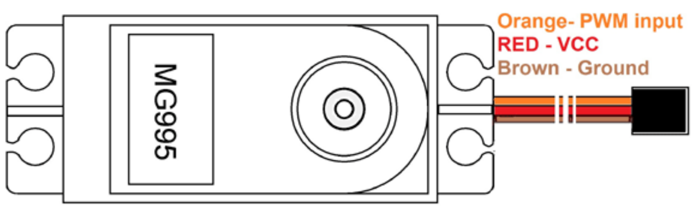


Figure 32: Servo specifications
Left: Tower Pro MG995 Servo; Right: Wire color-coding [44]

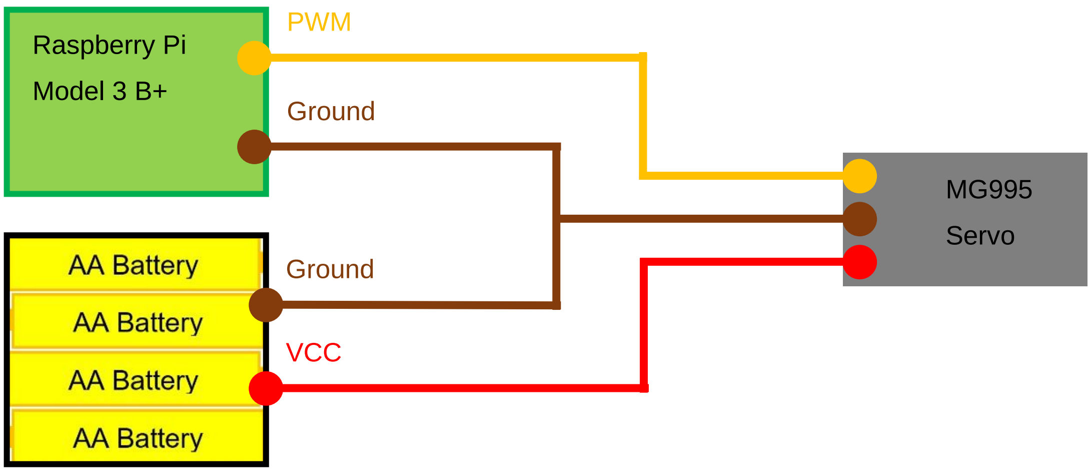


Figure 33: Servo system's electrical diagram

The Figure 34 shows the functions of all the Raspberry Pi pins. The Raspberry Pi GPIO [45] pin six can be used for grounding and pin twelve can output a PWM signal to control the servo.

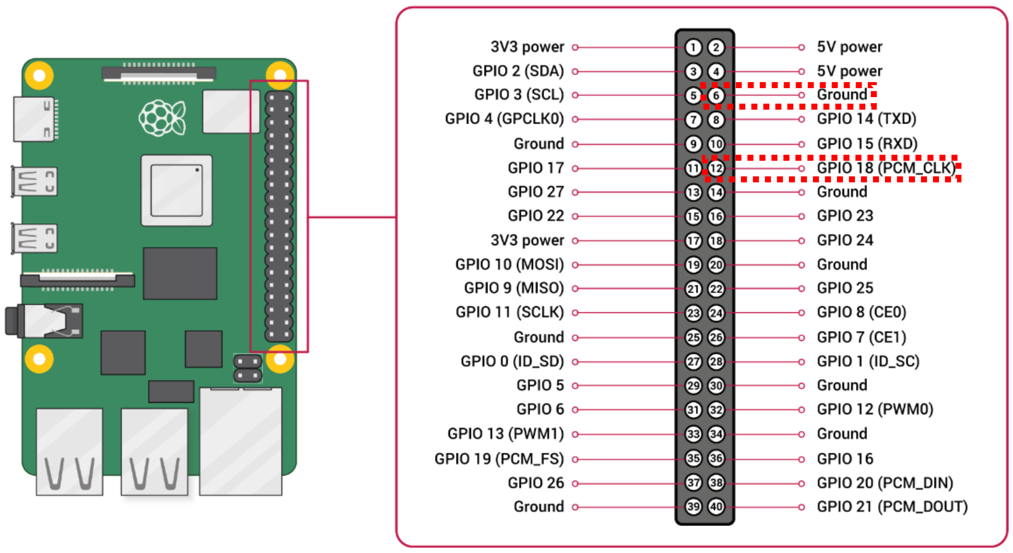


Figure 34: Raspberry Pi 3B+ Pinout specifications [45]

3.2.4 Full-scale linear scanning configuration

The real application of the 3D scanner on the quarry machine requires cameras with a broad field of view and a case to protect it from weather and harsh working conditions. The chosen hardware is a pair of Dalsa Genie Nano C4020 GigE cameras, shown in Figure 35, equipped with 1.1-inch format 12-megapixel optical lenses.

Figure 35: Camera Genie Nano C4020 [46]

The Dalsa cameras use the Sony IMX304 imaging sensor model, which use CMOS technology. This hardware's specifications are detailed in Table 6 and Figure 36.

Table 6: Camera system specifications

Camera type	Color
Dimensions (mm)	38.9 x 29.0 x 44.0
Resolution (pixels)	12 Megapixel ⇒ 4112 horizontally x 3012 vertically
Pixel size (μm)	3.45 x 3.45
Field-of-View (degrees)	83.12° horizontally x 66.00° vertically x 106.14° diagonally

Sensor size (inches)	1.1 in
Video Output	GigE
Trigger type	External or programmable
Lens Mount	C and CS-mount
Mass	~46g
Operating Temperature	-20˚ to +60˚ Celsius
Power Supply	10 to 36V or PoE
Data Connector	Ethernet RJ-45
Power and/or I/O Connector	SAMTEC TFM-105 type
Software Platform	Teledyne DALSA Sapera LT8.0 for windows

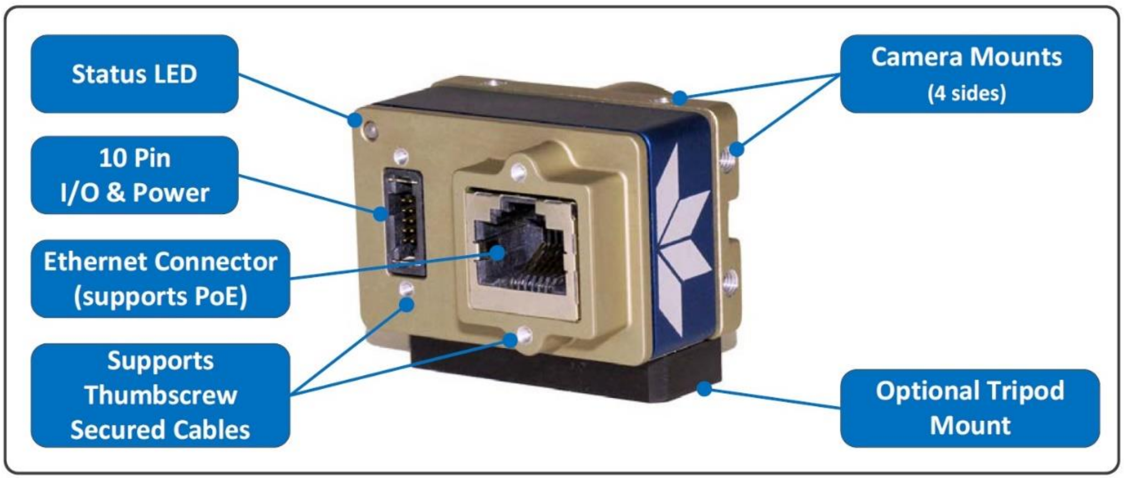
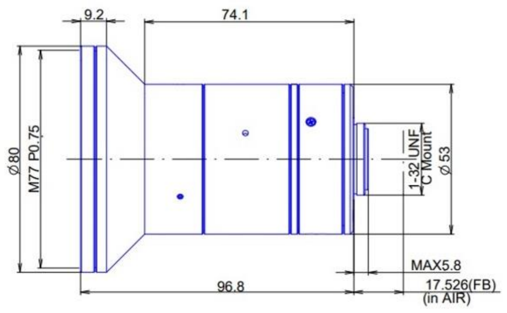


Figure 36: Camera ports

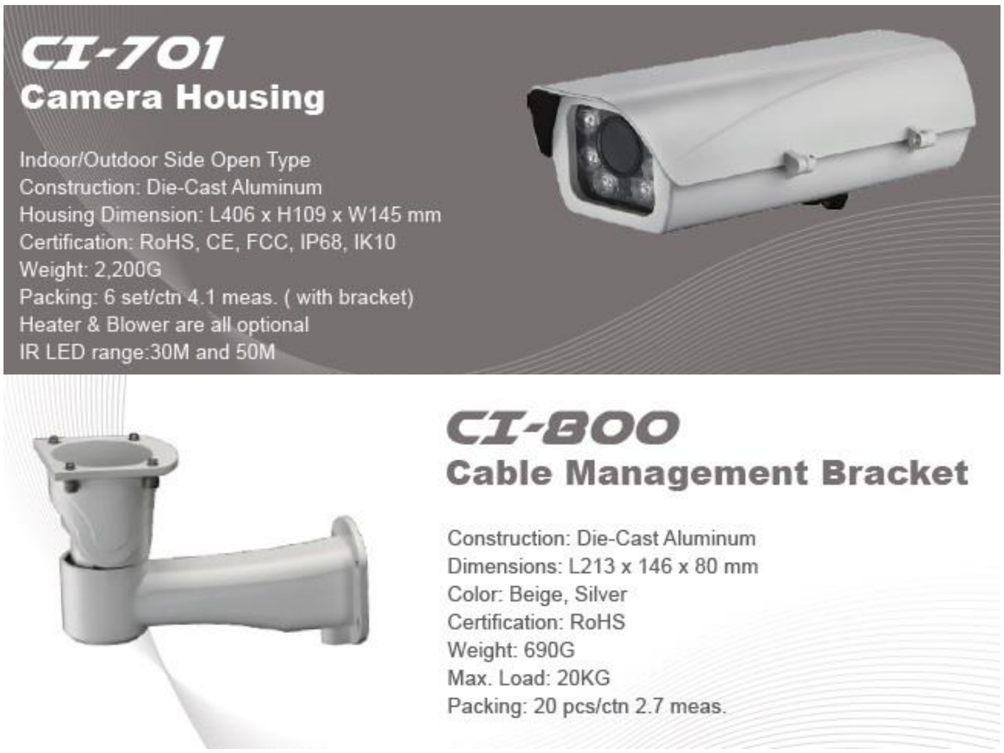
The lens used is the 1.1-inch format high-resolution 12-megapixel lens manufactured by GOYO Optical. This lens general dimensions and technical specifications can be seen in Figure 37.



Focal Length	8 (mm)
Iris Range	F1.8-22
Angle of View	83˚50′×66˚7′
MOD	0.3 (m)
Filter Thread	M=77, P=0.75
Dimension (D×L)	Ø80×96.8 (mm)
Weight	500 (g)

Figure 37: Optical lens specifications [47]

The camera is installed into a housing to protect it against environmental or other external elements. The camera housing used is the model CI-701 supported by the cable management bracket CI-800, Figure 38, and mounted by the pole mount bracket CI-815, Figure 39-(a), supplied by MASSLOAD.



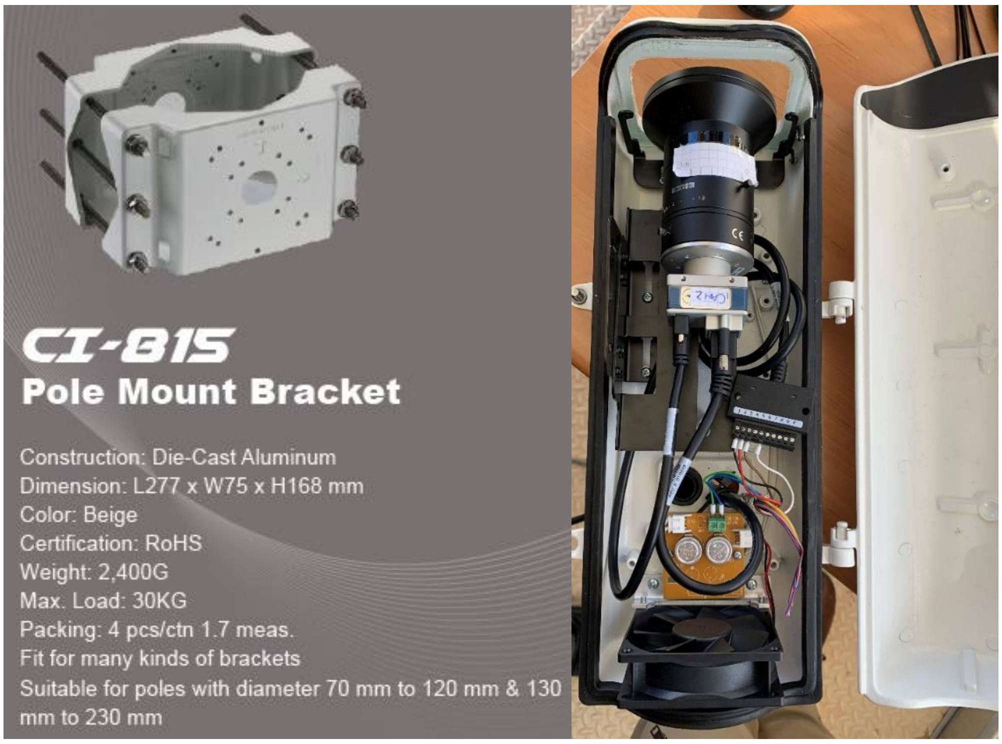
CI-701
Camera Housing

Indoor/Outdoor Side Open Type
Construction: Die-Cast Aluminum
Housing Dimension: L406 x H109 x W145 mm
Certification: RoHS, CE, FCC, IP68, IK10
Weight: 2,200G
Packing: 6 set/ctn 4.1 meas. (with bracket)
Heater & Blower are all optional
IR LED range:30M and 50M

CI-800
Cable Management Bracket

Construction: Die-Cast Aluminum
Dimensions: L213 x 146 x 80 mm
Color: Beige, Silver
Certification: RoHS
Weight: 690G
Max. Load: 20KG
Packing: 20 pcs/ctn 2.7 meas.

Figure 38: Camera housing and support specifications [48]


CI-815
Pole Mount Bracket

Construction: Die-Cast Aluminum
Dimension: L277 x W75 x H168 mm
Color: Beige
Certification: RoHS
Weight: 2,400G
Max. Load: 30KG
Packing: 4 pcs/ctn 1.7 meas.
Fit for many kinds of brackets
Suitable for poles with diameter 70 mm to 120 mm & 130 mm to 230 mm

(a) (b)

Figure 39: Pole mount bracket specifications and camera housing inner installation [48]

The camera system is attached to the Fravizel's machine. As seen in Figure 40, this machine is composed of two main parts, a chainsaw mechanism that rotates a diamond chain, A, and a portico mechanism that supports and moves the chainsaw, B. The portico base attaches to a horizontal rail, 1, to allow it to move into position for the cut while the chainsaw mechanisms are attached to rails built on the vertical beams of the portico to move the chainsaw up and down, 2. The cameras are attached to beams on the portico structure to take advantage of its encoder and respective linear movement, as seen in Figure 41. This allows images of the stone blocks to be acquired with controlled linear motion provided with known position and orientation for each image. The machine possesses two porticos, a main portico that has a secondary shorter rail system rigidly attached to it and moves along the main rail system covering the entire working area, and a secondary portico that moves along the secondary rail system from the main portico. On each portico a pair of cameras can be installed, these camera pairs compose a stereo camera system to simultaneously acquire a pair of stereo calibrated images.

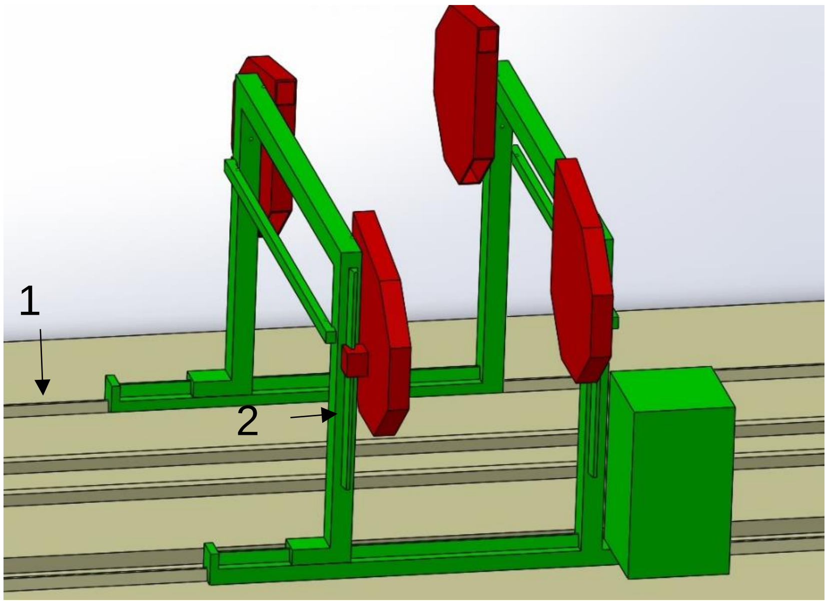


Figure 40: CAD model of Fravizel machine

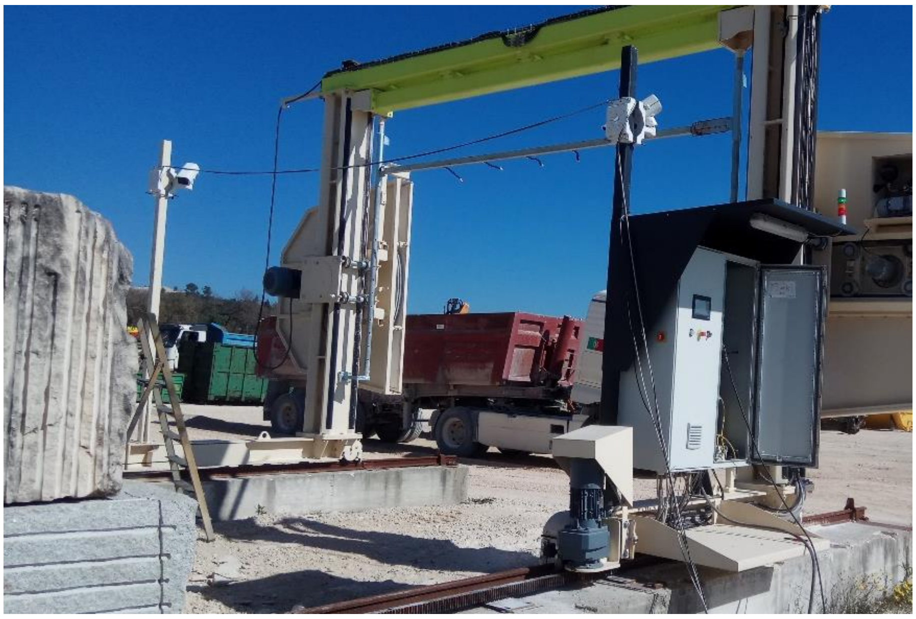
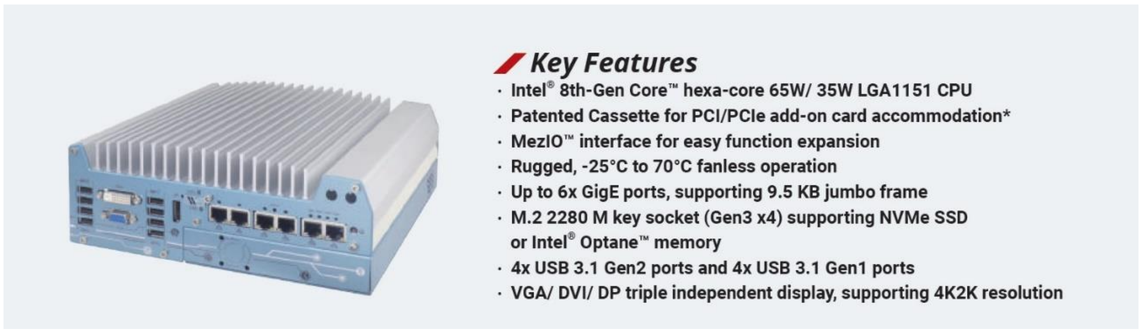


Figure 41: Scanning system installed on Fravizel cutting machine

The data transfer is accomplished by connecting the cameras to a computer through a network using Ethernet cables. The camera is powered and triggered through the I/O input pins. The computer used has several Ethernet ports to allow the installation of more cameras if needed. The full specifications can be seen in Figure 42.

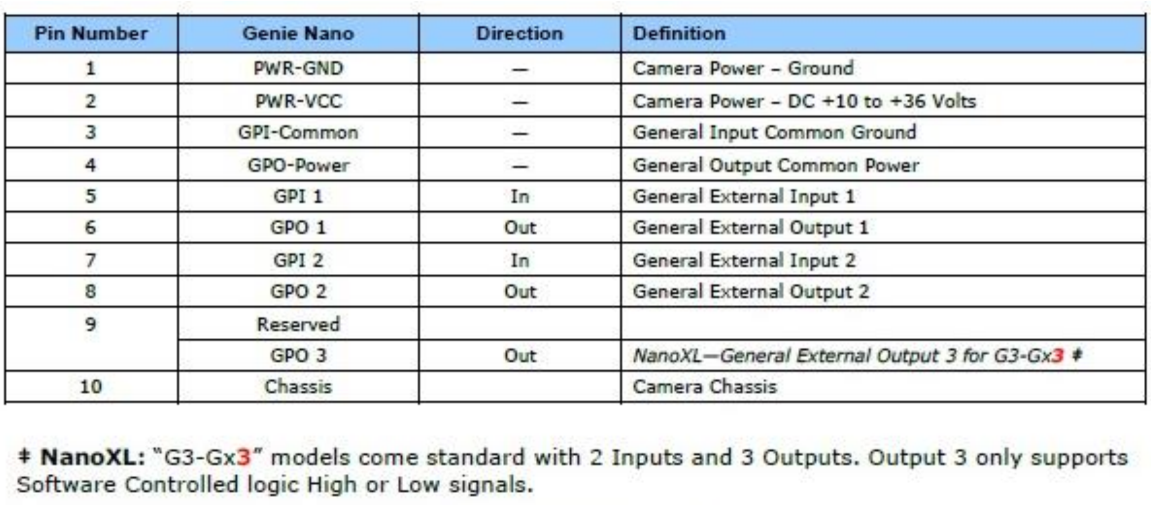


Key Features

- Intel® 8th-Gen Core™ hexa-core 65W/ 35W LGA1151 CPU
- Patented Cassette for PCI/PCIe add-on card accommodation*
- MezIO™ interface for easy function expansion
- Rugged, -25°C to 70°C fanless operation
- Up to 6x GigE ports, supporting 9.5 KB jumbo frame
- M.2 2280 M key socket (Gen3 x4) supporting NVMe SSD or Intel® Optane™ memory
- 4x USB 3.1 Gen2 ports and 4x USB 3.1 Gen1 ports
- VGA/ DVI/ DP triple independent display, supporting 4K2K resolution

Figure 42: Computer Specifications [49]

The cameras have I/O pins that allow power and extra functionalities to be accessed, Figure 43 shows the pins specifications.



Pin Number	Genie Nano	Direction	Definition
1	PWR-GND	—	Camera Power – Ground
2	PWR-VCC	—	Camera Power – DC +10 to +36 Volts
3	GPI-Common	—	General Input Common Ground
4	GPO-Power	—	General Output Common Power
5	GPI 1	In	General External Input 1
6	GPO 1	Out	General External Output 1
7	GPI 2	In	General External Input 2
8	GPO 2	Out	General External Output 2
9	Reserved		
	GPO 3	Out	*NanoXL—General External Output 3 for G3-Gx3 ‡*
10	Chassis		Camera Chassis

‡ **NanoXL:** "G3-Gx3" models come standard with 2 Inputs and 3 Outputs. Output 3 only supports Software Controlled logic High or Low signals.

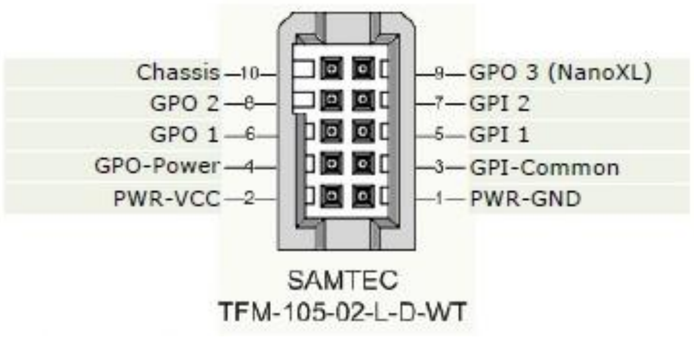


Figure 43: Cameras I/O Connector Pins Details

As proved by the earlier experiment with the Trust Exis webcams, using one computer to send multiple signals to trigger multiple cameras has proved too asynchronous. A simple solution is having one computer generating one signal to be sent, in parallel, to trigger multiple cameras. With the use of a Raspberry Pi, a custom signal can be scripted and transmitted, using one of the GPIO pins, to simulate and test this solution.

This solution was tested in the lab by generating a signal with a Raspberry Pi and transmitting it, in parallel, to the cameras. The cameras external pins 3 and 5, which function as input common ground and external input line 1 as shown in Figure 43, were connected to the RPI GPIO pins for grounding and signal transmission respectively. The results, described in detail in section 4.1.2, show that synchronicity is achievable through external triggering. Therefore, to synchronize the pairs of cameras and match the acquisitions to the porticos position the camera's pins 3 and 5 were connected to the portico rail encoder's ground and signal generator.

To provide power to the cameras, pins 1 and 2 were connected to an independent power supply ground and voltage common collector's pins, respectively, through standard copper wires. Figure 44 represents a simple diagram for the scanner system, the cameras Ethernet connections to the main computer are shown in blue, black and red lines represent the ground and voltage connections to the power supply and lastly the brown and green line represent the ground and signal transmitting wires connected to the trigger.

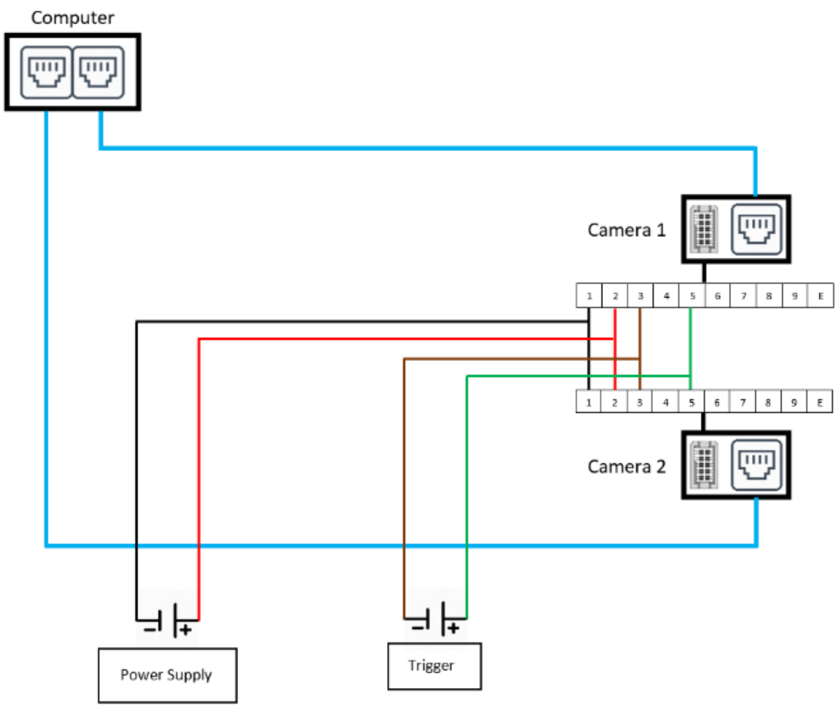


Figure 44: Scanning system electrical diagram

The field of view provided by this system can be viewed more practically in a CAD 3D model drawn in SolidWorks, as shown in Figure 45.

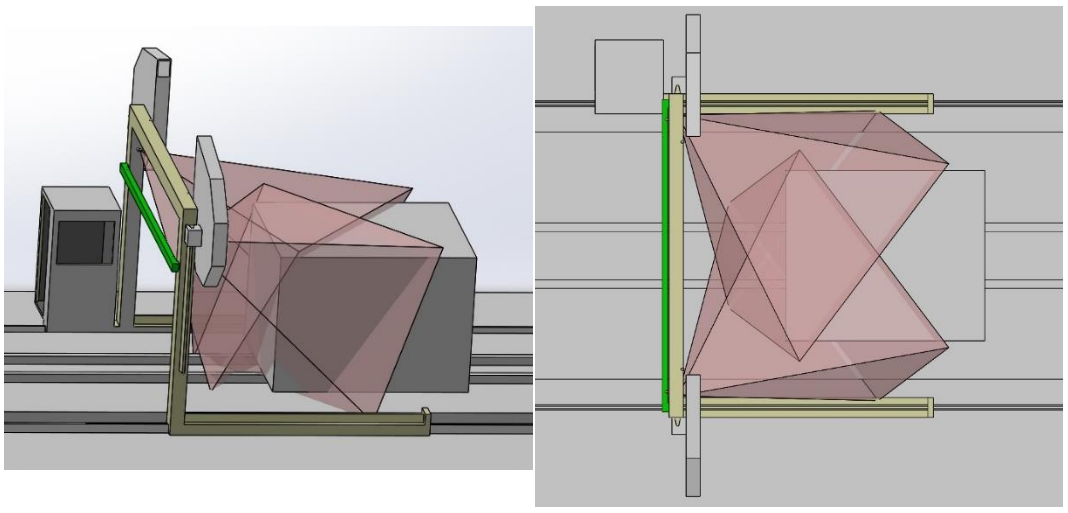


Figure 45: Camera system Field-of-View installed on Fravizel machine

3.3 Intrinsic Calibration methods

The cameras were calibrated using established tools or programmed functions. The MATLAB software comes installed with an app named Camera Calibrator for easy calibrations, later a Python script using pre-built OpenCV functions was used to calibrate the cameras, applying the algorithms explain in section 2.3.1.

3.3.1 MATLAB Scripting

For 3D reconstruction an optional input are the cameras intrinsic parameters, calculated through calibration. MATLAB software comes with the calibrating application, Camera Calibrator, an easy to use application with a simple GUI [50], Figure 47. The Camera Calibrator receives two inputs: a set of images of a chessboard pattern, Figure 46, taken from the camera to be calibrated; and the size of the patterns squares in metric units [51].

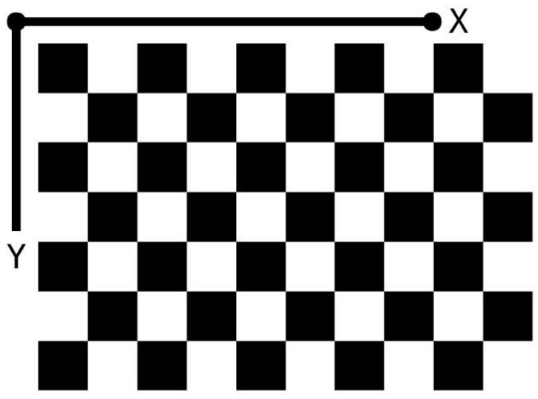


Figure 46: Chessboard pattern for MATLAB calibrations [50]

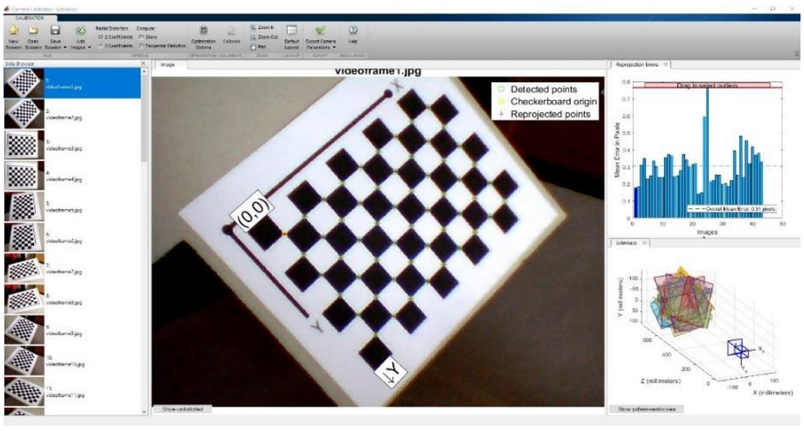


Figure 47: MATLAB Camera Calibration GUI [50]

3.3.2 Python Scripting

Python can be used to calibrate cameras with the use of OpenCV, allowing a larger variety of chessboard patterns to be detected for calibration. A script was written to accomplish calibration through following steps: Input the size pattern squares, generate the array of the points 3D coordinates, analyzing each image to try to detect the pattern, use the points detected across all images to calculate intrinsic parameters.

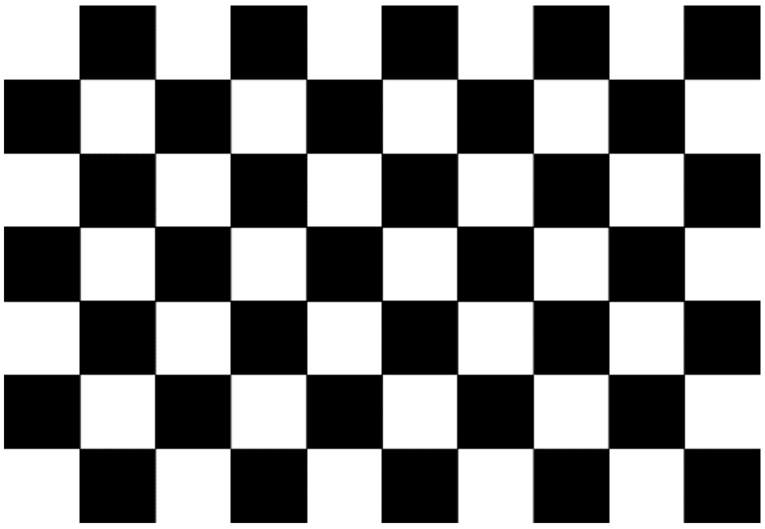


Figure 48: Chessboard pattern for OpenCV

In order to calibrate through images, multiple points must be identified in the image and said points must have, known three-dimensional world coordinates. These points are the inner corners of the chessboard pattern, as shown in Figure 48.

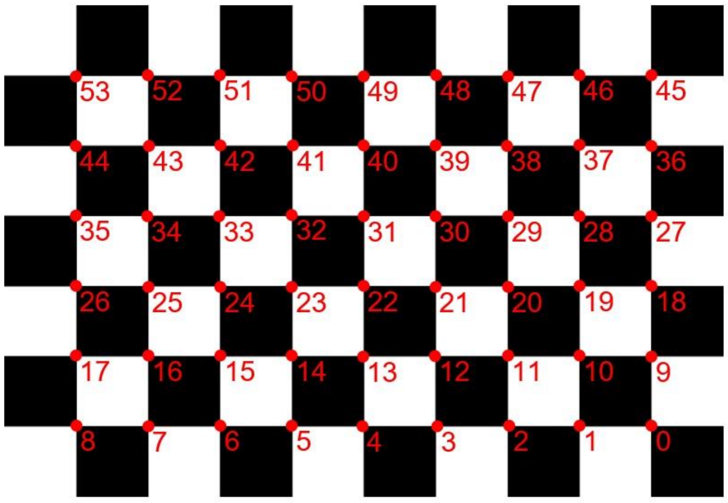


Figure 49: Corners order

The point's three-dimensional world positions must be stored in an array, designated *objectPoints*, in the same order and configuration as the array containing point's two-dimensional image coordinates, the order of the corners is shown in Figure 49. The three-dimensional positions of these points are generated using the measured size of the squares and assuming the board is located along an XY plane. As an example, assuming a pattern with square size of 12 millimeters, the array generated would look like displayed in Table 7.

Table 7: Corners array configuration

Point	X	Y	Z
0	0	0	0
1	12	0	0
2	24	0	0
…	…	…	…
9	0	12	0
10	12	12	0
11	24	12	0
…	…	…	…
18	0	24	0
19	12	24	0
20	24	24	0

The set of images taken of the chessboard pattern are analyzed with the use of the function, *cv2.findChessboardCorners,* this function has two outputs: A Boolean value, indicating whether or not the pattern was detected and an array, designated *Corners*, with the point's image coordinates if the pattern is detected. For preview purposes, the function, *cv2.drawChessboardCorners*, takes as input the image and the array *Corners*, and draws on the image the corners, color coding them in the order in which they were organized, from red to blue as seen in Figure 50.

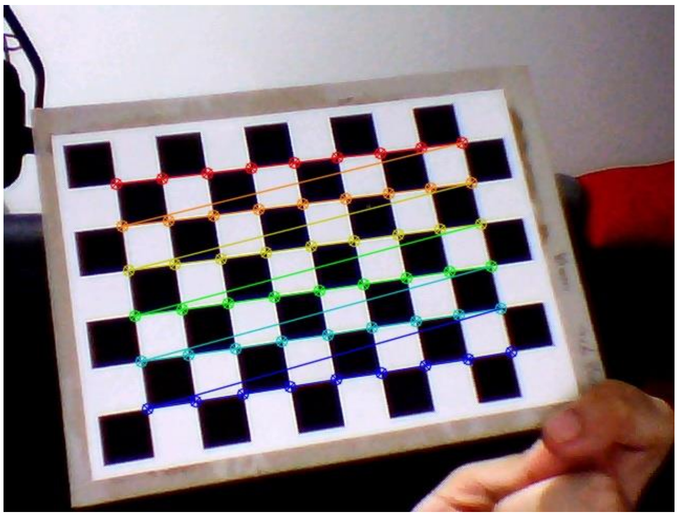


Figure 50: Detected Pattern

After all the images are analyzed and enough patterns are detected, a minimum of 20 patterns are recommended for a good result, all the *Corners* arrays are given to the function *cv2.calibrateCamera.* This function uses the points 2D image coordinates, stored in *Corners* array, and the corresponding 3D world coordinates, stored in *objectPoints* array to determine the intrinsic parameters. The algorithms used by this function follow the technique for camera calibration by Zhengyou Zhang. [27]

This method give us the following intrinsic parameters: a matrix containing, the focal lengths in pixel units, f_x and f_y and the principal point, $(C_x; C_y)$; and a vector containing radial distortion coefficients, K_1 through K_6, and tangential distortion coefficients, P_1 and P_2. Finally, all the intrinsic parameters are stored in an archive with “npz” format, for example *“Calibration_Output.npz”.*

3.4 Extrinsic Calibration Methods

3.4.1 Crude extrinsic calibration

In order to obtain some rough estimates for the orientation of the camera, trigonometric calculations were computed with the pixel positions of three points with known real-world positions. In order to detect points in an image, the Aruco Python module was used. Aruco Python module contains the functions required for accurately detection of a set of custom markers, using these functions different markers can be used to locate specific points in an image containing multiple markers. Figure 52 shows an example of an image of the Aruco markers used to perform the calculations. The camera orientation can be represented using many different sets of angles, in this case the Tait-Bryan angles were calculated for a rough estimate.

The images of the Aruco markers used for this estimation, were acquired under certain constraints. As shown in Figure 51, the markers common plane was placed perpendicular to known distance, L, at a 45-degree angle in the considered world frame. The Aruco marker's dimensions, distance between them and height are known. The camera is placed over the XY plane origin in the world frame at a known height.

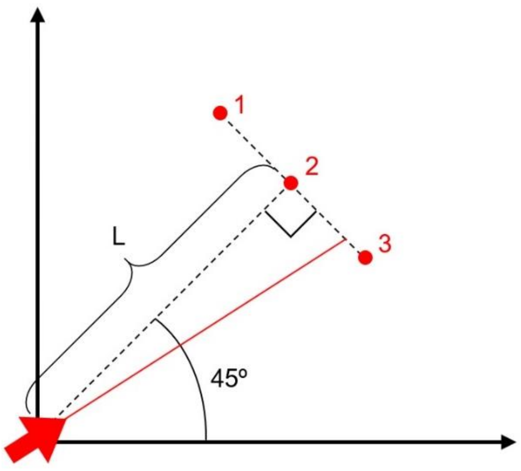


Figure 51: Aruco markers positioning - Top view

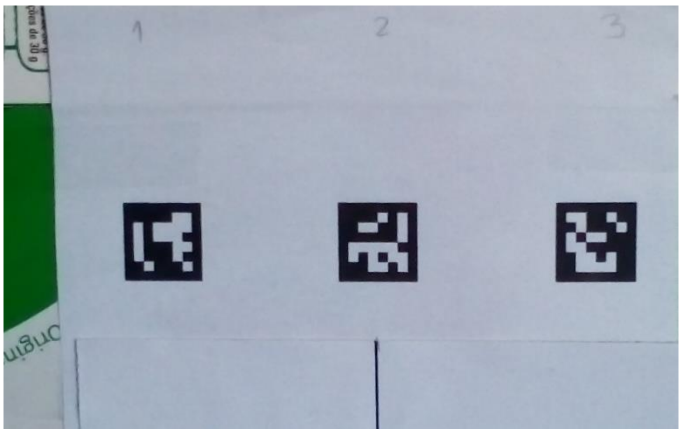


Figure 52: Aruco markers

To calculate the roll of the camera the simplest solution was to compute the angular difference between a horizontal line in the real word and a horizontal line in the respective image, as shown in Figure 53. Using the two most separate markers to describe the world horizontal line the difference between these points coordinates can lead to the equation:

$$Roll = \tan^{-1}\frac{(Marker3_y - Marker1_y)}{(Marker3_x - Marker1_x)} = \tan^{-1}(\frac{\Delta y}{\Delta x}) \qquad 3.1$$

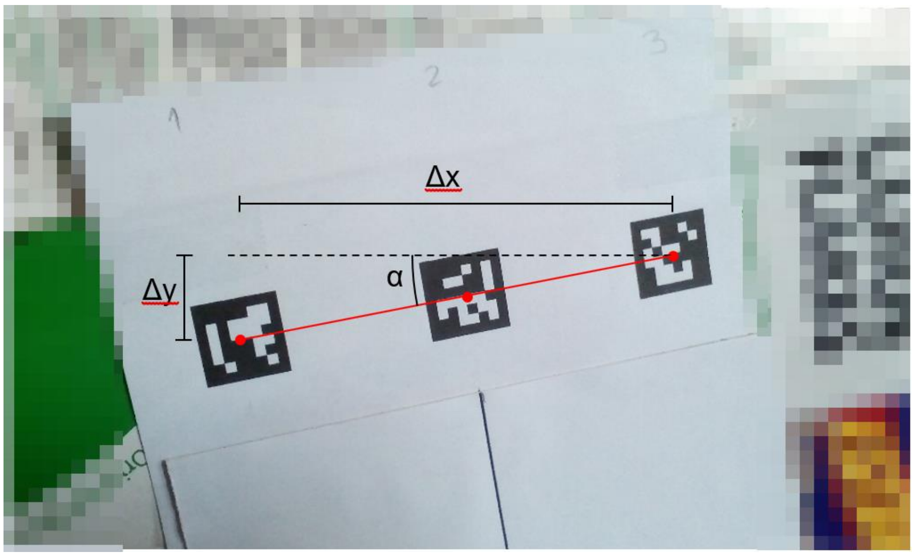


Figure 53: Aruco markers roll

For the purpose of calculating the camera's yaw rotation, the middle marker, 2, is placed at 45º from the X-axis and at a known distance, L, from the camera, while all markers are positioned perpendicularly to the line between the origin and marker 2, as shown in Figure 55. In this arrangement the equation used to estimate an approximate value of the angle yaw is as follows:

$$Yaw = 45 - \tan^{-1}\left(\frac{Image\ Center_x - Marker2_x}{L}\right) = 45 - \tan^{-1}\left(\frac{\Delta x}{L}\right) \qquad 3.2$$

The distance, L, is measured according to Figure 55. While the horizontal coordinates of the image center and the second marker and taken from the image as shown in Figure 54, to determine Δx.

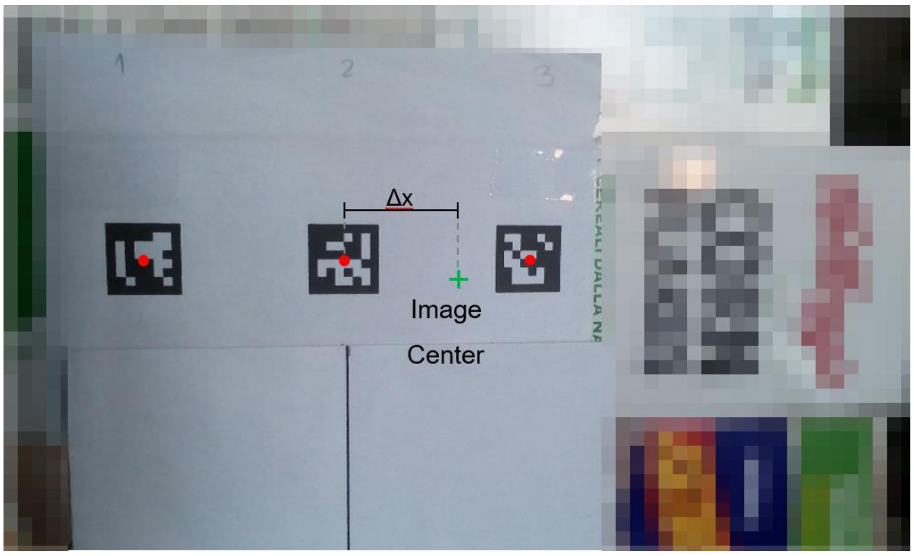


Figure 54: Aruco markers yaw

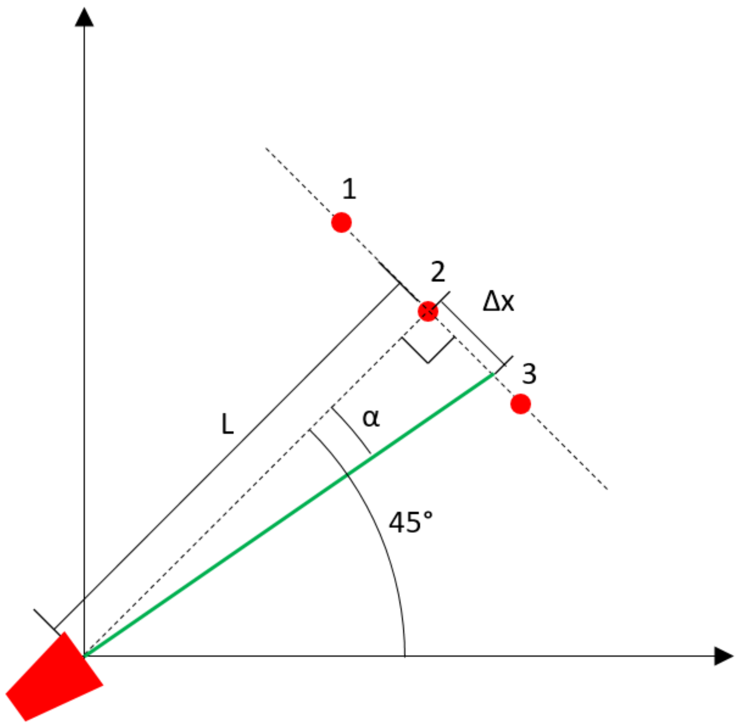


Figure 55: Top view coordinate system for yaw

To calculate the pitch of the camera we need first to calculate the difference, along the vertical axis, between the camera position and the image center in metric units, represented in Figure 57, as Δh. Knowing the camera height, H, and marker height, mH, the height difference Δh is calculated as $\Delta h = H - (mH + \Delta y)$. The value Δy is the difference, along the vertical axis, between the image center and the marker 2 center in metric units, Figure 56. Upon calculating the height difference, the equation to calculate pitch becomes:

$$Pitch = \tan^{-1}(\frac{\Delta h}{L}) \qquad 3.3$$

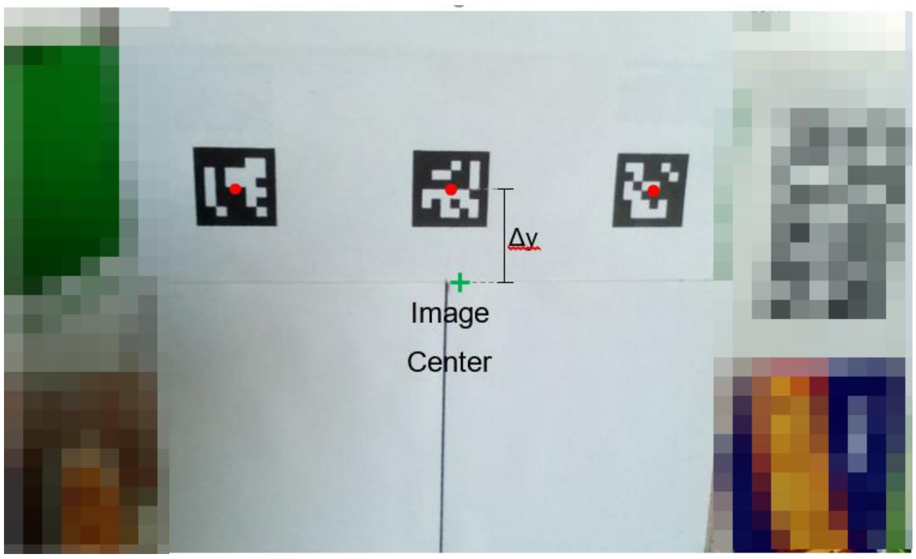


Figure 56: Aruco markers for pitch

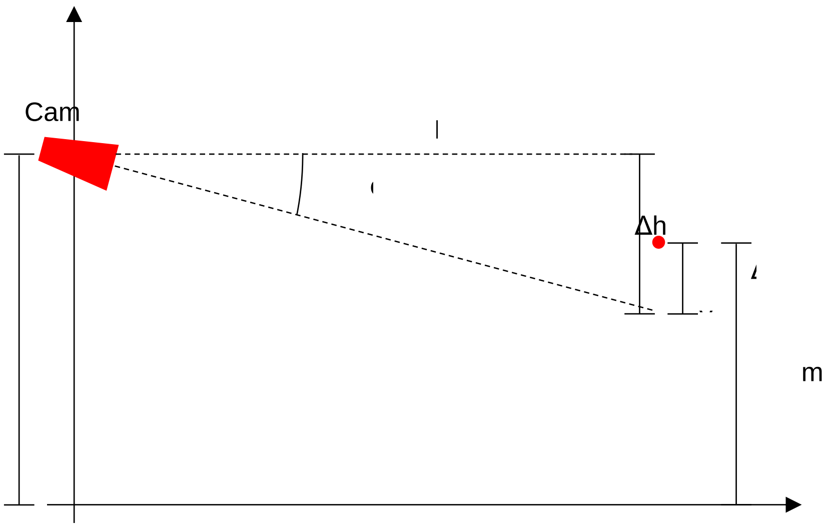


Figure 57: Side view diagram for pitch

The pose for the first image coincides with the cameras real fixed pose, therefore the initial pose is estimated using the Aruco markers and aforementioned equations. To calculate the supposed position and orientation of the camera of each subsequent image, the initial estimated pose is translated and rotated algebraically.

With all angles estimated and the known position in the established world coordinate system, these extrinsic parameters can be organized in the necessary format required by the software to perform the reconstruction.

3.4.1 Extrinsic Calibration - Perspective-n-Point

The previous extrinsic calibration method was only a simple quick calculation to obtain some rough numbers to attempt reconstructions with the images obtained with the earlier hardware. Estimating the poses of the cameras for each image acquisition can be accomplished more accurately by solving the Perspective-n-Point problem, described in section 2.3.2.

Python Scripting

Similar to the intrinsic calibration method, the image must contain identifiable points with known three-dimensional coordinates, so the chessboard pattern used for the intrinsic calibration is should also be applicable for several of the algorithms used in pose estimation.

The script written to perform pose estimation follows similar steps to the intrinsic calibration, we initialize all known prior information, analyze each image to attempt to detect the pattern's points and use each image's collection of points to estimate the pose of the camera that captured each respective image.

The prior information required is the intrinsic parameters calculated and the three-dimensional coordinates of the point's positions. The intrinsic parameters are simply loaded from the archive created by the calibration script, containing all the parameters.

The array containing the three-dimensional coordinates of the pattern's corners, designated *objectPoints*, is generated the same way as before but in this case the pattern is not assumed to be a in a random XY plane independent from the patterns from other images. The function that calculates the camera poses will return a pose relative to the positions of the points used, in other words the camera pose will be defined in the same three-dimensional coordinate system used to define the point's coordinates. For this reason, all the point's positions from all patterns, must be described using the same coordinate system.

For the case where the images are acquired through a circular motion of the camera, a box was used with a chessboard patterns glued to each lateral surface, as shown in Figure 58. In some poses the image sensor's field-of-views encases two surfaces/patterns, this would cause confusion to the point detection function, which would randomly detect one of the two visible patterns.

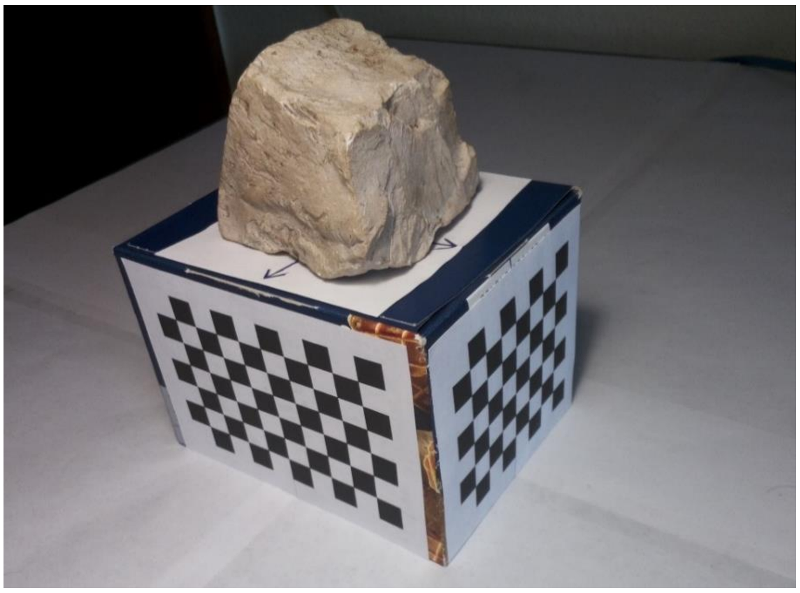


Figure 58: Box with chessboard patterns

To avoid this multiple pattern detection problem, different size patterns were used between consecutive surfaces; the larger faces had nine-by-six chessboard patterns, while the smaller faces had seven-by-six chessboard patterns. This difference in size allows us to specify the pattern we prefer to be detected on each case. The box dimensions and patterns square size are manually measured to generate each of the 3D-coordinates arrays, *objectPoints*, using the same coordinate system. Figure 59 shows a graph with the points of the four patterns, using the three-dimensional coordinates from the generated *objectPoints* arrays.

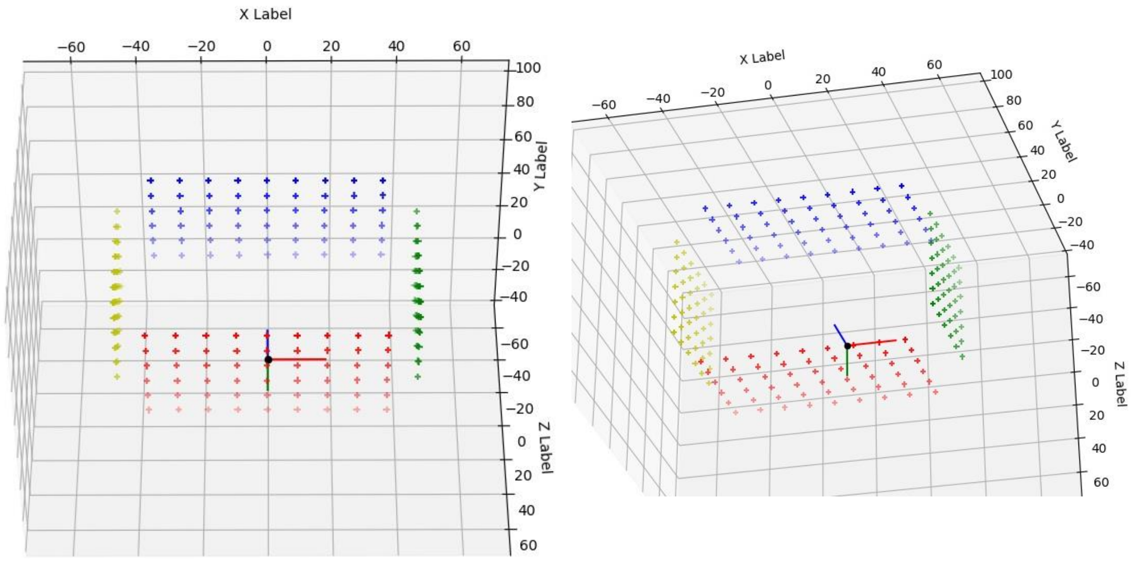


Figure 59: 3D-graph for chessboard corners

For the case where the images are acquired with a static camera and the circular motion obtained through the rotation of the object, the pose of the camera was calculated differently. A single image of the chessboard pattern is required for the set of images acquired through this method. The first pose is estimated through the same extrinsic calibration method and the following poses are calculated algebraically based on the angular motion applied to the object between image acquisitions.

Similarly, to the intrinsic calibration script, each image is analyzed with the function, *cv2.findChessboardCorners,* if a pattern is detected an array, *Corners*, is outputted. The *Corners* array contains the 2D-image coordinates of the points detected from the chessboard pattern. An optional function, *cv2.cornerSubPix*, can be used to refine the corner's locations in the Corners array. Finally, the function, cv2.*solvePnP* or cv2.*solvePnPRansac*, uses the 3D-coordinates array, *objectPoints,* the 2D-coordinates array, *Corners*, and the intrinsic parameters to estimate the pose for the camera that captured that respective image.

RANSAC, which stands for Random Sample Consensus, is a method that can be used to determine from a sample of data possible outliers that might induce wrong results. Using this method in our calibrations means detecting outliers from the set of points used for 3D to 2D correspondence and removing them from the set of points provided to the calibration algorithms.

Figure 60 shows an image, taken with the Pi Camera, shown in Figure 31, and a graph of the pose returned by the default iterative algorithm employed, by OpenCV, for PnP solution without the optimization of RANSAC, function *cv2.solvePnP*.

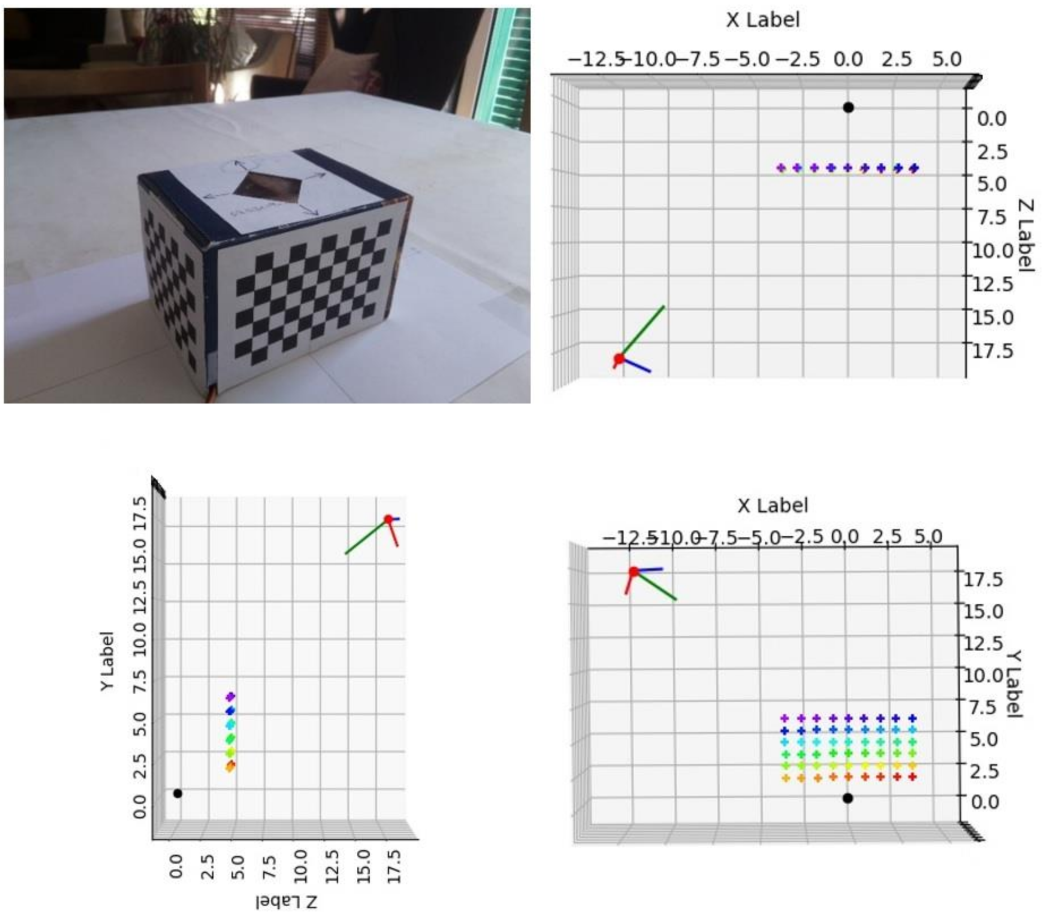


Figure 60: 3D-graph for chessboard and proper camera pose

The black dot is the world frame's origin, the red point is the camera position and the blue, red and green lines represent the camera orientation's X, Y and Z-axis, respectively. In this graph the chessboard corners were color-coded from red to purple to easily display the order in which the corners are stored. As mentioned before, the points coordinates must be ordered the same way to ensure proper correspondence between image coordinates and world coordinates. As it can be seen from the graph, the pose is placed in an expected position and orientation.

However, several extrinsic estimations have gone awry with the reason behind it difficult to determine. To provide an example of these occurrences consider the following use of the same data and algorithm but accompanied with the RANSAC optimization, results shown in the Figure 61.

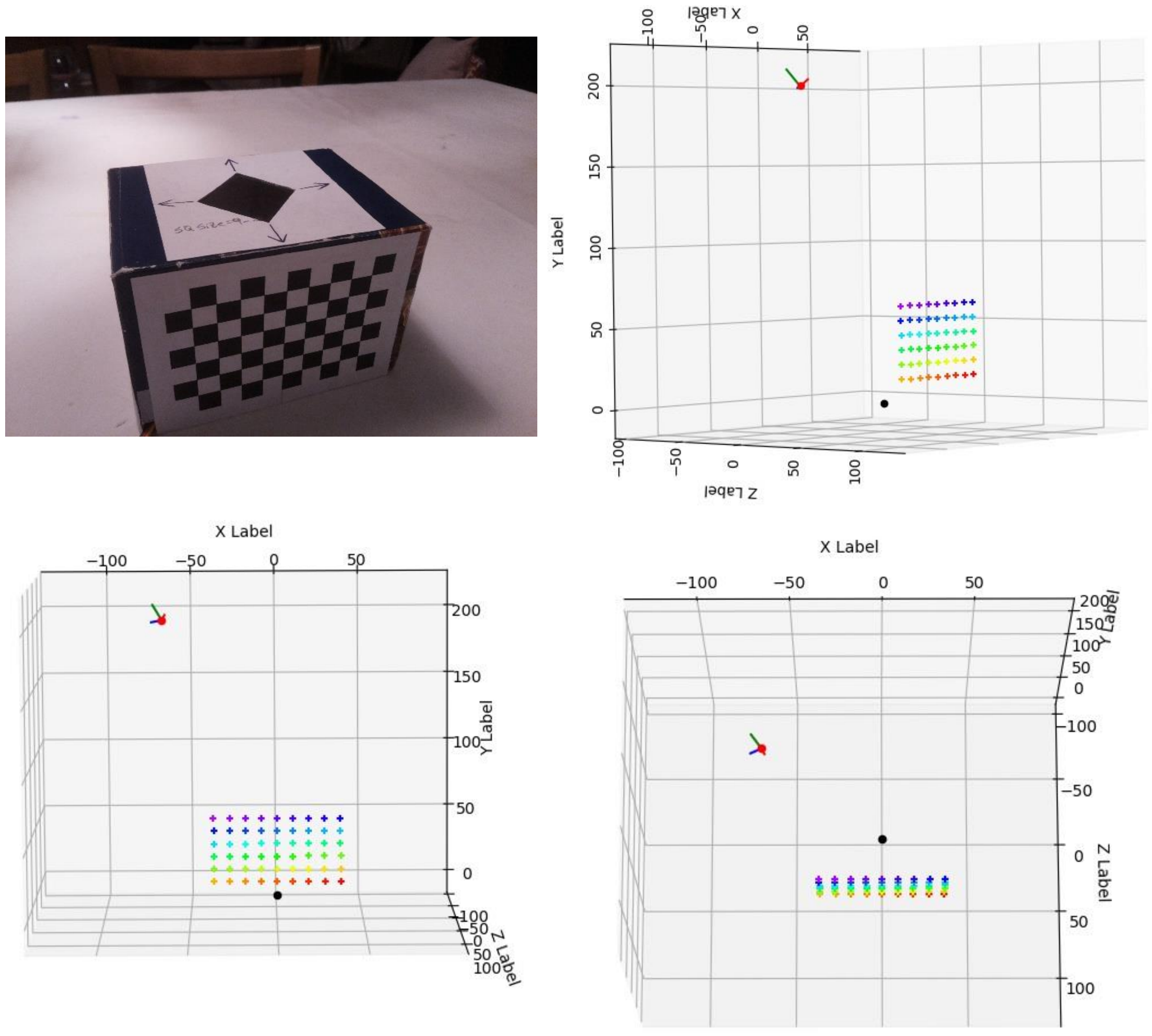


Figure 61: 3D graph for chessboard and wrong camera pose

Both the position and orientation estimated are not what would be expected. Different data and different methods were used to try and determine the cause of this error. The poses are improperly estimated using different methods on the same data, and on different data used by the same methods, making it difficult to determine where the error could be coming from. However, a couple of possible causes were theorized, one of human error and another pertaining to the analytical nature of the problem. The possible human error consisted of the positioning of the chessboard in parallel with the XY-plane, resulting in providing a data set of 3D coordinates, where the Z-coordinate was the same value for all points. This lack of variation in the Z-axis, possibly didn't allow the PnP algorithms to correctly determine the configuration of the world frame. The other theorized error is related to the analytical ambiguity of using coplanar points to determine the pose, inducing the algorithm to converge to one of the possible solutions randomly and erroneously. To rule out both these theorized causes the chessboard was replaced by four Aruco markers, each providing their four corners as points for the pose estimation, shown in Figure 62. The markers we place parallel to the XY-plane, for easy calculation and graphical verification, but at different Z-coordinates, this ensures that the data set of 3D coordinates has different values in the Z-axis and that the points are not all in the same plane, ruling out both theorized causes.

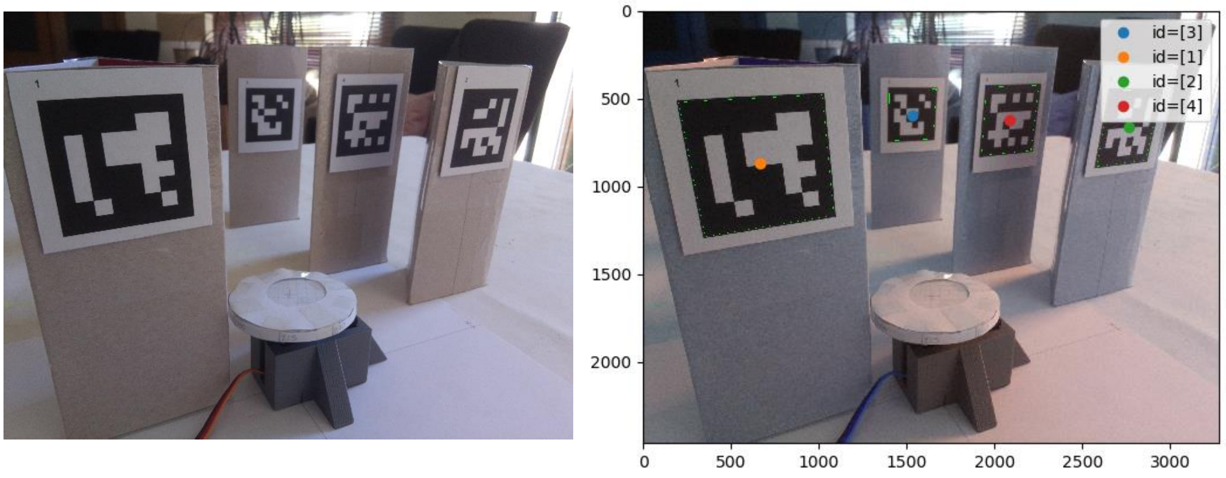


Figure 62: Aruco Markers and Identification

Using this configuration to estimate pose still provided wrong results. The Figure 63 shows the cv2.solvePnPRansac function estimating pose badly yet again. Taking into consideration that RANSAC excludes points from the calculation if it determines those points to be outliers and that this new method is using only 16 points as opposed to the 54 points detected from the chessboard, using RANSAC might be excluding too many points inducing a wrong solution. When using Aruco markers with RANSAC the number of points will be increased by using not only the corners but also the marker's centers and edge's midpoints, resulting in 9 points per marker for a total of 36 points.

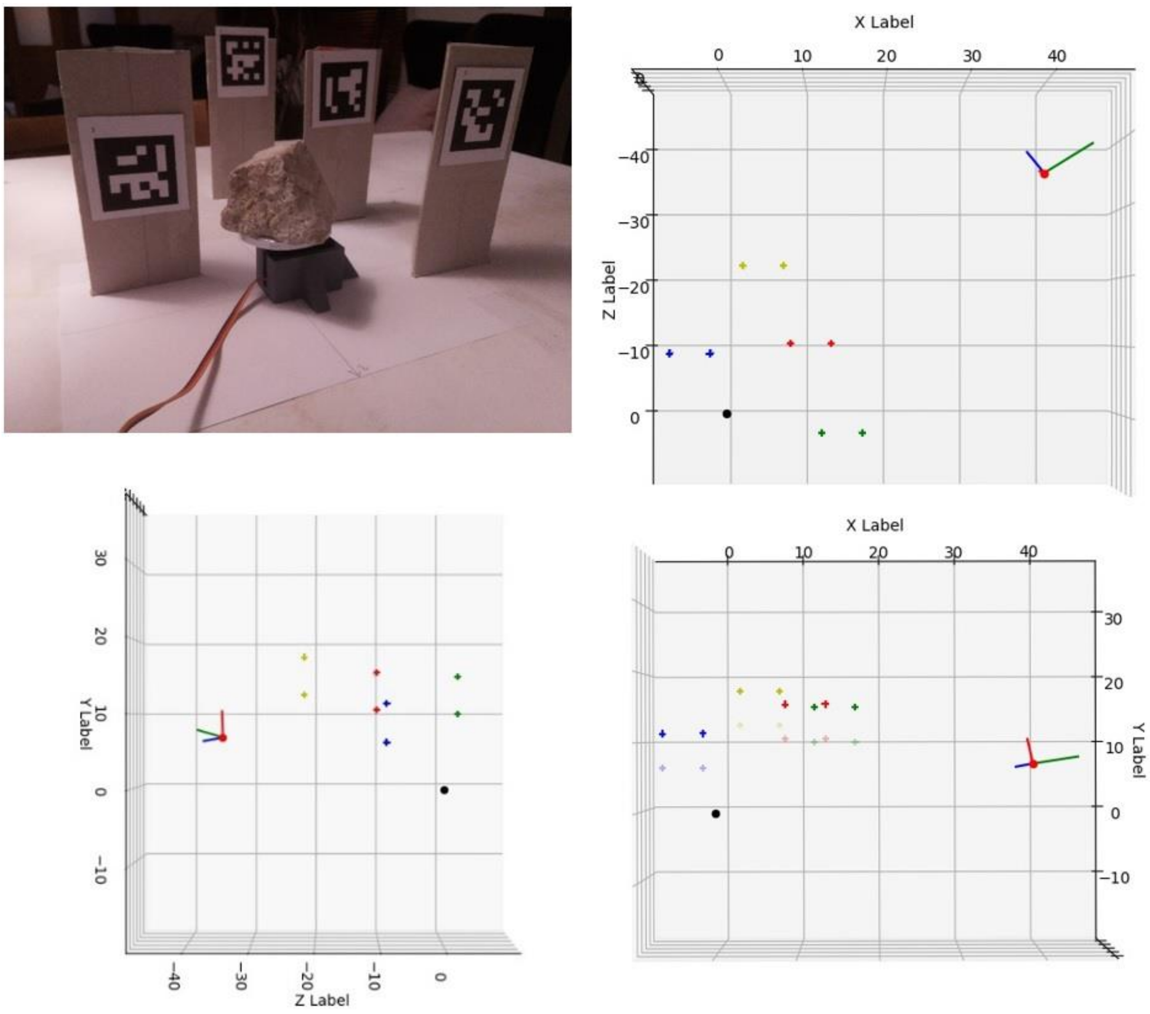


Figure 63: 3D graph of markers and wrong camera pose
***Red**: Marker 1; **Blue**: Marker 2; **Green**: Marker 3; **Yellow**: Marker 4*

Considering for the use of a lower amount of points, RANSAC would be more of a liability then an advantage, the same data was used for pose estimation without RANSAC, Figure 64 shows the results of this estimation. Even though the markers didn't fully resolve the problem they still rule out two possible problems and will be compared with the chessboard pattern performance regarding extrinsic calibrations.

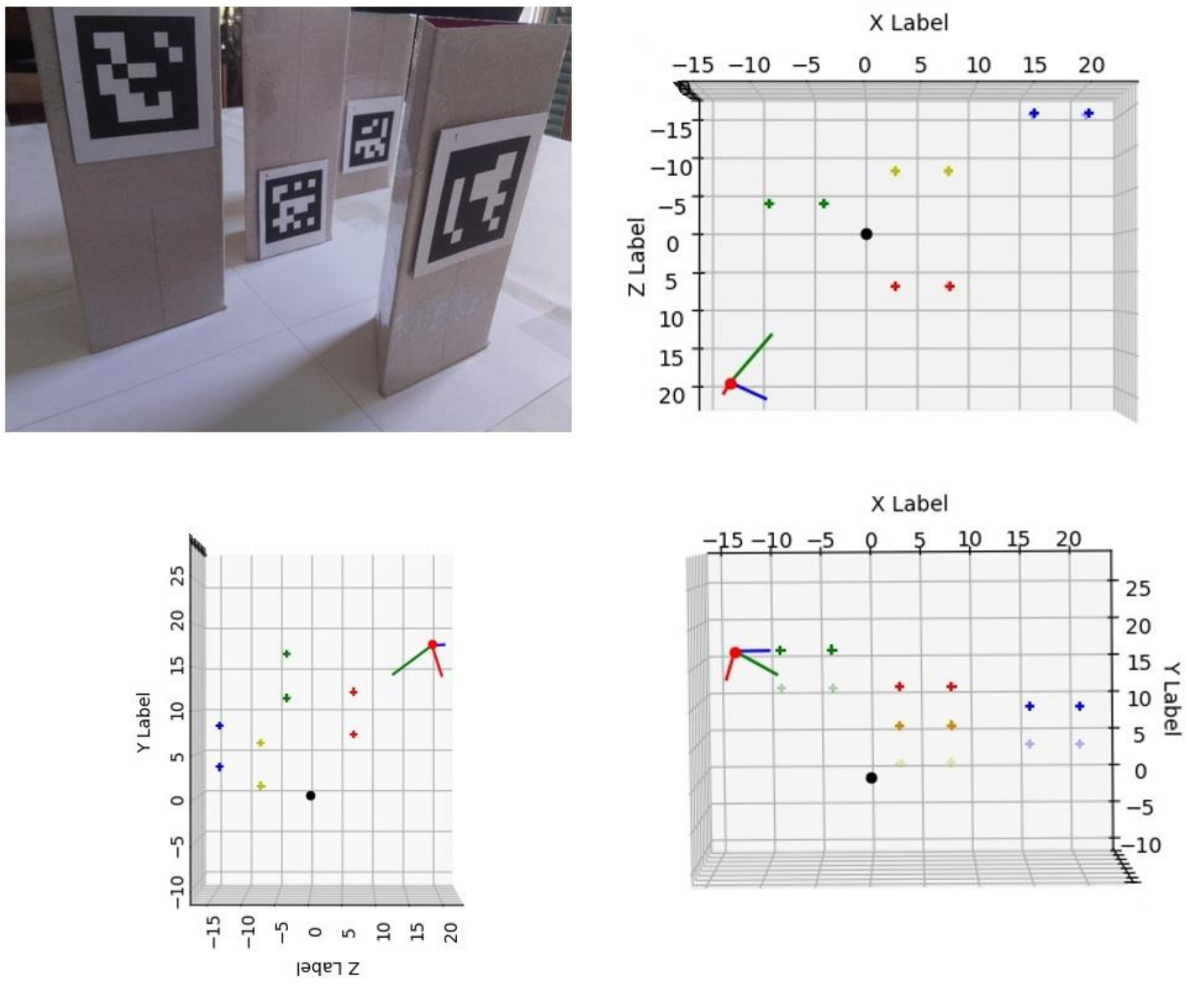


Figure 64: 3D graph for markers and proper pose
***Red**: Marker 1; **Blue**: Marker 2; **Green**: Marker 3; **Yellow**: Marker 4*

The function cv2.solvePnP returns the pose of the camera in the format of a rotation vector and a translation vector. The estimated rotation and translation matrices are given in reference to a camera coordinate system. Therefore, they need to be inverted to reflect the proper information in the world coordinate system.

The 3-by-3 rotation matrix, R_{3x3}, and translation vector, t_{3x1}, compose the transformation matrix, H, as follows:

$$H = \begin{bmatrix} & R_{3x3} & & t_{3x1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} R_{11} & R_{12} & R_{13} & tx \\ R_{21} & R_{22} & R_{23} & ty \\ R_{31} & R_{32} & R_{33} & tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow \qquad 3.4$$

This matrix can be decomposed into two matrixes, one containing the rotation's information, R, and another containing the translation's information, T.

$$\Rightarrow T * R \Rightarrow \begin{bmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & 0 \\ R_{21} & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 3.5$$

Inverting the product of two matrices is equal to the product of the inverse matrices in reverse order, as such:

$$H^{-1} = (T * R)^{-1} = R^{-1} * T^{-1}$$

R is a rotation matrix therefore is an orthogonal matrix. The inverse of the rotation matrix is its transpose.

$$R^{-1} = R^{T} = \begin{bmatrix} R_{11} & R_{21} & R_{31} & 0 \\ R_{12} & R_{22} & R_{32} & 0 \\ R_{13} & R_{23} & R_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The inverse of the translation matrix is the translation matrix with opposite signs on each of the translation components.

$$T^{-1} = -T = \begin{bmatrix} 1 & 0 & 0 & -tx \\ 0 & 1 & 0 & -ty \\ 0 & 0 & 1 & -tz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Multiplying the now inverted matrixes together results in,

$$H^{-1} = \begin{bmatrix} R_{11} & R_{21} & R_{31} & 0 \\ R_{12} & R_{22} & R_{32} & 0 \\ R_{13} & R_{23} & R_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -tx \\ 0 & 1 & 0 & -ty \\ 0 & 0 & 1 & -tz \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{21} & R_{31} & -(R_{11} * tx + R_{21} * ty + R_{31} * tz) \\ R_{12} & R_{22} & R_{32} & -(R_{12} * tx + R_{22} * ty + R_{32} * tz) \\ R_{13} & R_{23} & R_{33} & -(R_{13} * tx + R_{23} * ty + R_{33} * tz) \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} {R_{3x3}}^{T} & -{R_{3x3}}^{T} * t_{3x1} \\ 0\;0\;0 & 1 \end{bmatrix} \qquad 3.6$$

Therefore, a camera's orientation, in the established world frame, is given by ${R_{3x3}}^{T}$ and the position is given by $-{R_{3x3}}^{T} * t_{3x1}$. With everything calculated and expressed properly, last step is to export it in the formats compatible with the software we intend to use for the 3D-reconstruction.

Chapter 4

4 Results

4.1 Hardware

4.2 Intrinsic Calibration

4.3 Extrinsic Calibration

4 Results

4.1 Hardware

4.1.1 Circular Configuration Scanner with Raspberry Pi Camera V2

As described before, in section 3.2.3, this system uses the Raspberry Pi single board computer to control a camera and servo to perform a 360-degree scan of the stone. The entire hardware used for this proposed scanner can be seen in Figure 65.

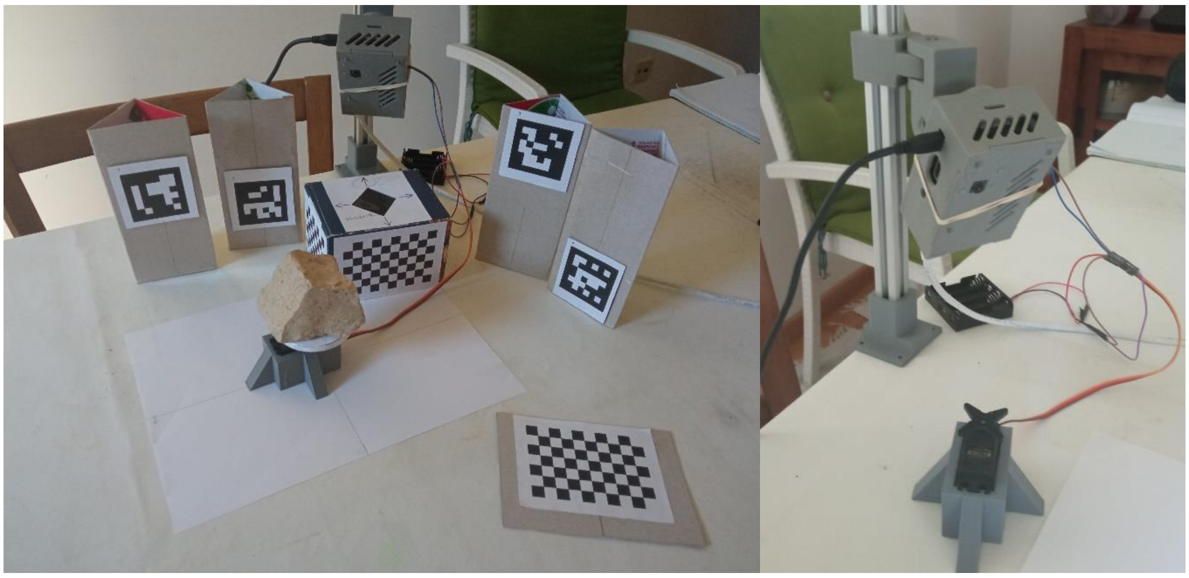


Figure 65: Circular Scanner Hardware

The first image in shows all the devices required for camera calibration and image acquisition, the Aruco markers and chessboard patterns, the servo motor, the Raspberry Pi and its camera supported by a 3D printed kinematic chain and a common pavement limestone rock used as a cheap small-scale alternative for the quarry limestone blocks. The second image provides a closer look at the servo motor, used to rotate the stone, and the kinematic chain, that supports the Raspberry Pi with its mounted camera.

This Scanning system acquired the images, in Figure 66, with a resolution of (3280; 2464) pixels and a 30-degree angle between them.

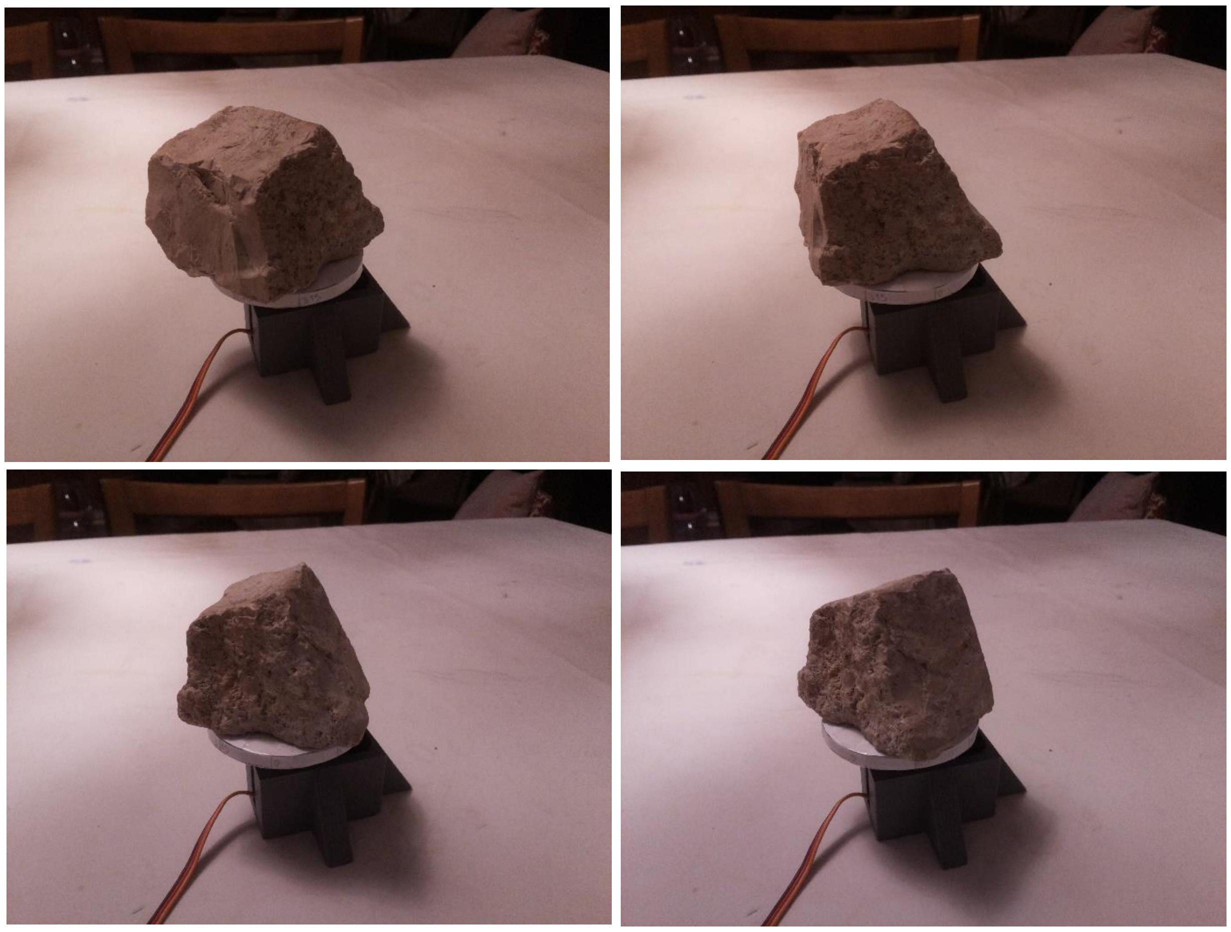


Figure 66: Raspberry Pi Scanner Images

4.1.2 Linear Configuration Scanner with Genie Nano camera

This configuration is the proposed solution to improve Fravizel's cutting machines. The cameras were incorporated into the machine to use its motion to both move and trigger the cameras, as seen in Figure 67. The image on the left shows where the camera's housings were mounted and the image on the right shows the inside of the housing where the camera is attached.

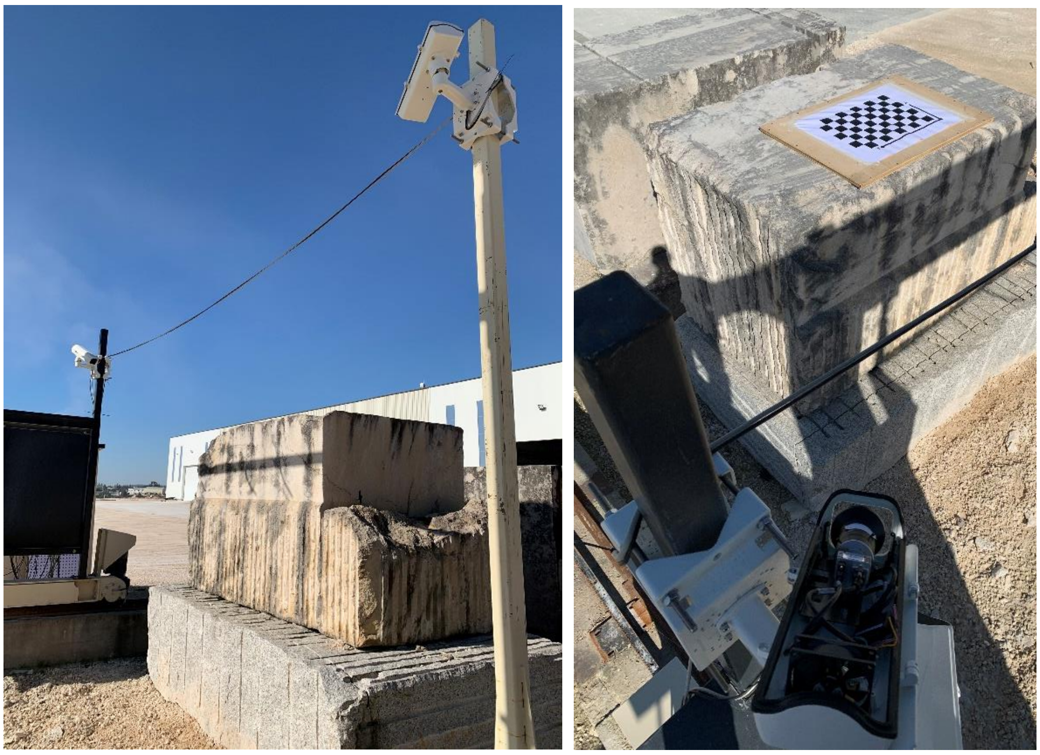


Figure 67: Linear Scanner Hardware

Genie Nano Cameras acquired the following images, in Figure 68 and Figure 69 with a resolution of (4112; 3008) pixels. These images are part of a set of images taken in 15-centimeter intervals in the machine’s rails, however for the sake of visualization, the pictures have a 240-centimeter interval between them.

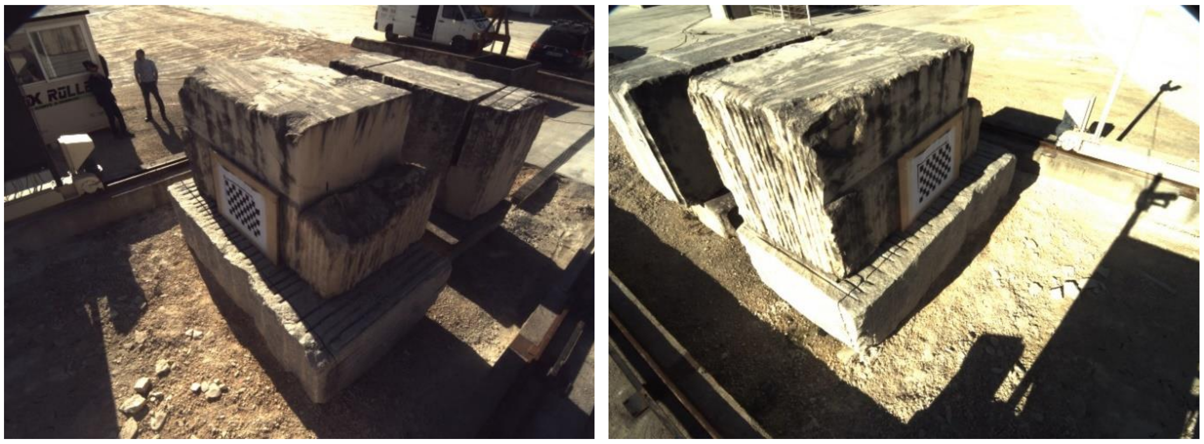


Figure 68: Genie Nano Scanner Images-Part 1

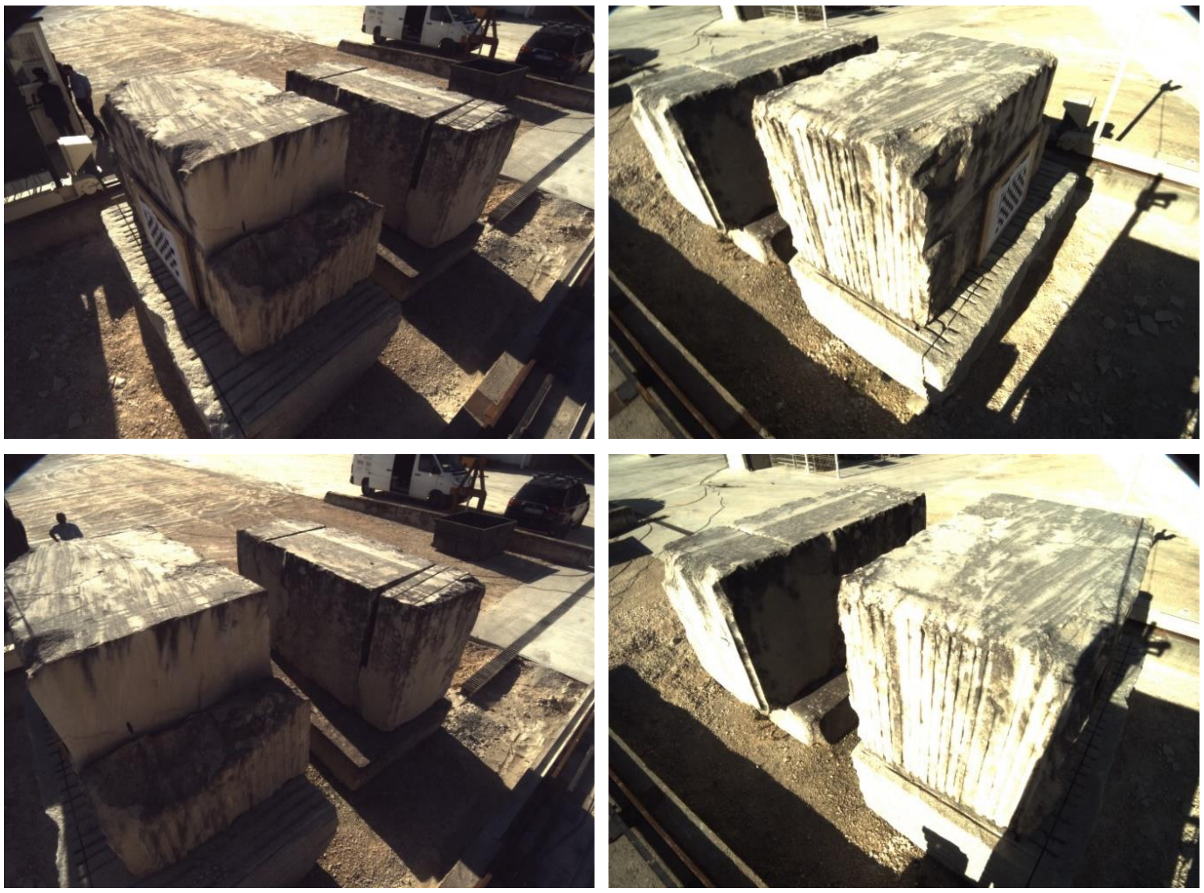


Figure 69: Genie Nano Scanner Images-Part 2

Genie Nano Camera System’s Synchronicity Test:

To test the Genie Nano Cameras synchronicity, both cameras were set up to be triggered externally by the same signal generated by a Raspberry Pi. The generated signal was set to trigger acquisition every 1 second. Both cameras were aimed to a browser stopwatch to obtain a direct timestamp for the acquisitions.

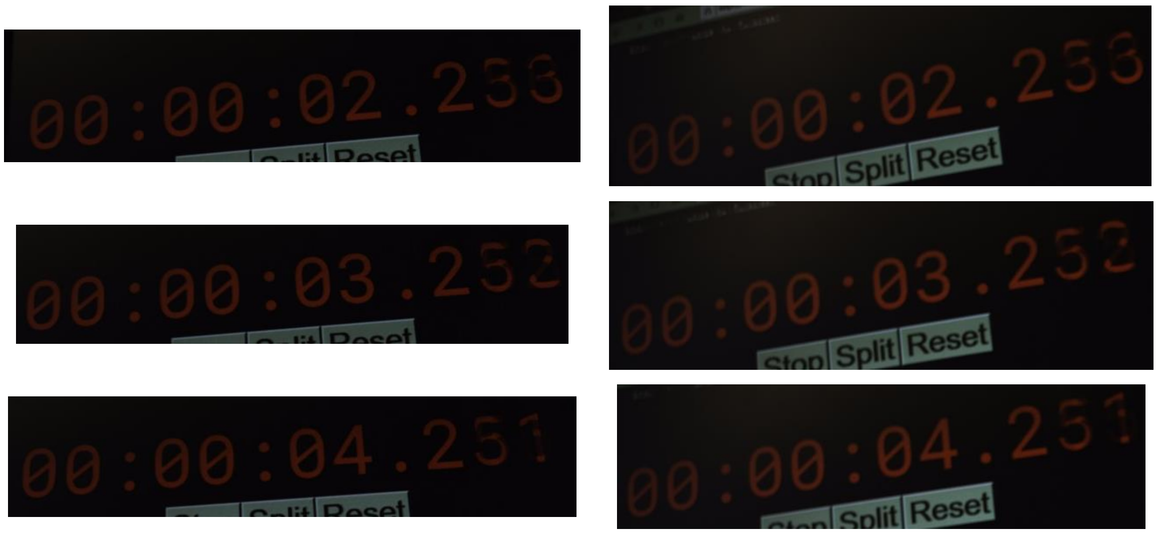


Figure 70: Camera acquisitions through external trigger - Part 1

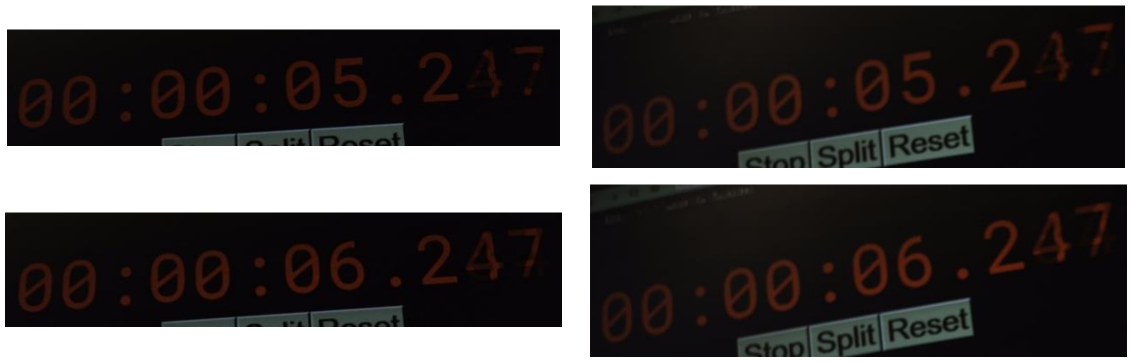


Figure 71: Camera acquisitions through external trigger-Part 2

Figure 70 and Figure 71 shows the images acquired by this test, the results were organized in the following table. As we can see from Table 8, using an external signal to trigger the cameras results in a very satisfying synchronicity. This allows us to provide a positive answer to question, Q2, the scanning system was successfully incorporated into the existing Fravizel machine's structure.

Table 8: Camera trigger's timestamps

Camera 1 (seconds)	2.253	3.252	4.251	5.247	6.247
Camera 2 (seconds)	2.253	3.252	4.251	5.247	6.247

4.2 Intrinsic Calibration

The Intrinsic matrix and distortion coefficients vector are organized and shown in the following format:

$$Intrinsic\ Matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = \begin{bmatrix} K_1 & K_2 & P_1 & P_2 & K_3 \end{bmatrix}$$

Where (f_x, f_y), are the focal lengths and (c_x, c_y), are the principal point's coordinates in $X - axis$ and $Y - axis$, respectively. $K_i, i = 1, 2, 3$ and $P_j, j = 1, 2$, are the radial and tangential distortion coefficients, respectively.

4.2.1 Trust Webcam

Python Calibration:

$$Intrinsic\ Matrix = \begin{bmatrix} 1085.33 & 0 & 231.09 \\ 0 & 1082.48 & 247.71 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 1075.68 & 0 & 227.97 \\ 0 & 1068.68 & 246.02 \\ 0 & 0 & 1 \end{bmatrix}$$

Where α is a scaling parameter by which we refine the intrinsic matrix with OpenCV function, *cv2.getOptimalNewCameraMatrix()*.

$$Distortion\ Coefficients = \begin{bmatrix} -0.1297 & 0.1486 & -0.0073 & -0.0175 & 6.2048 \end{bmatrix}$$

This camera was calibrated in 68.13 seconds using 80 images. These intrinsic parameters result in a mean reprojection error of 0.05 pixels and a root mean square reprojection error of 0.41 pixels.

MATLAB Calibration:

$$Intrinsic\ Matrix = \begin{bmatrix} 1073.9 & 0 & 243.7 \\ 0 & 1072.3 & 239.8 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 1067.62 & 0 & 240.96 \\ 0 & 1062.05 & 238.47 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = \begin{bmatrix} -0.1701 & 1.5050 & -0.0067 & -0.0160 & -0.0951 \end{bmatrix}$$

These intrinsic parameters result in a mean reprojection error of 0.3918 pixels, with the mean reprojection error for each image provided shown in Figure 72. Using these intrinsic parameters to distort the images captured by the respective camera results in the images seen in Figure 73.

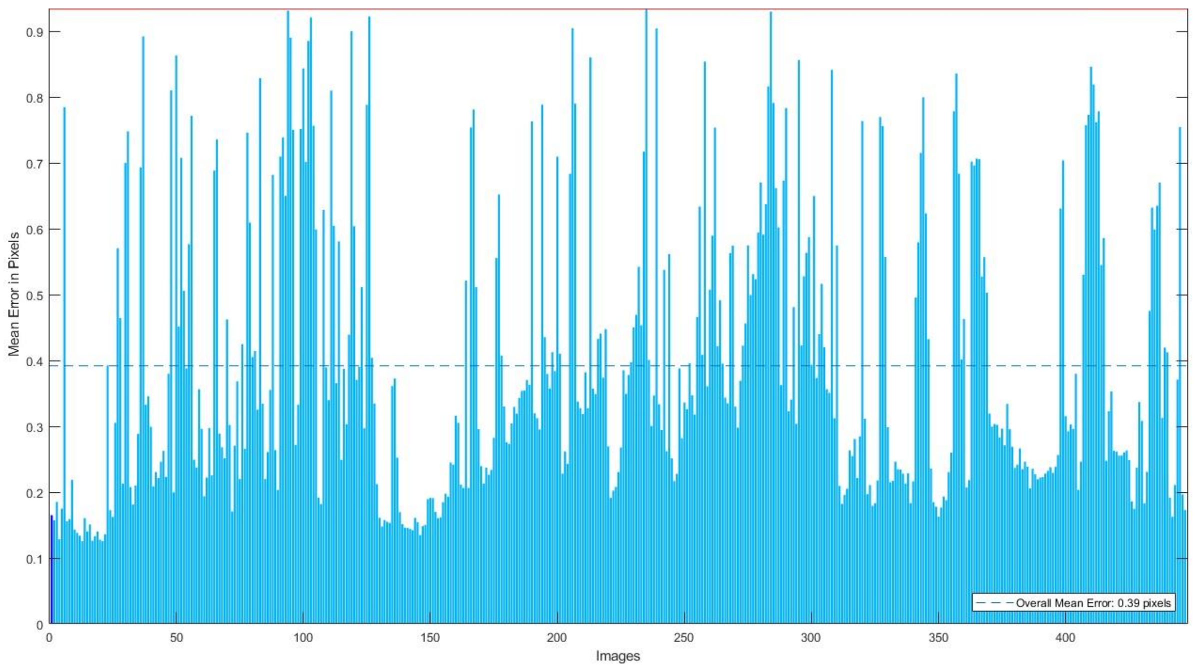


Figure 72: Images Mean Reprojection Errors

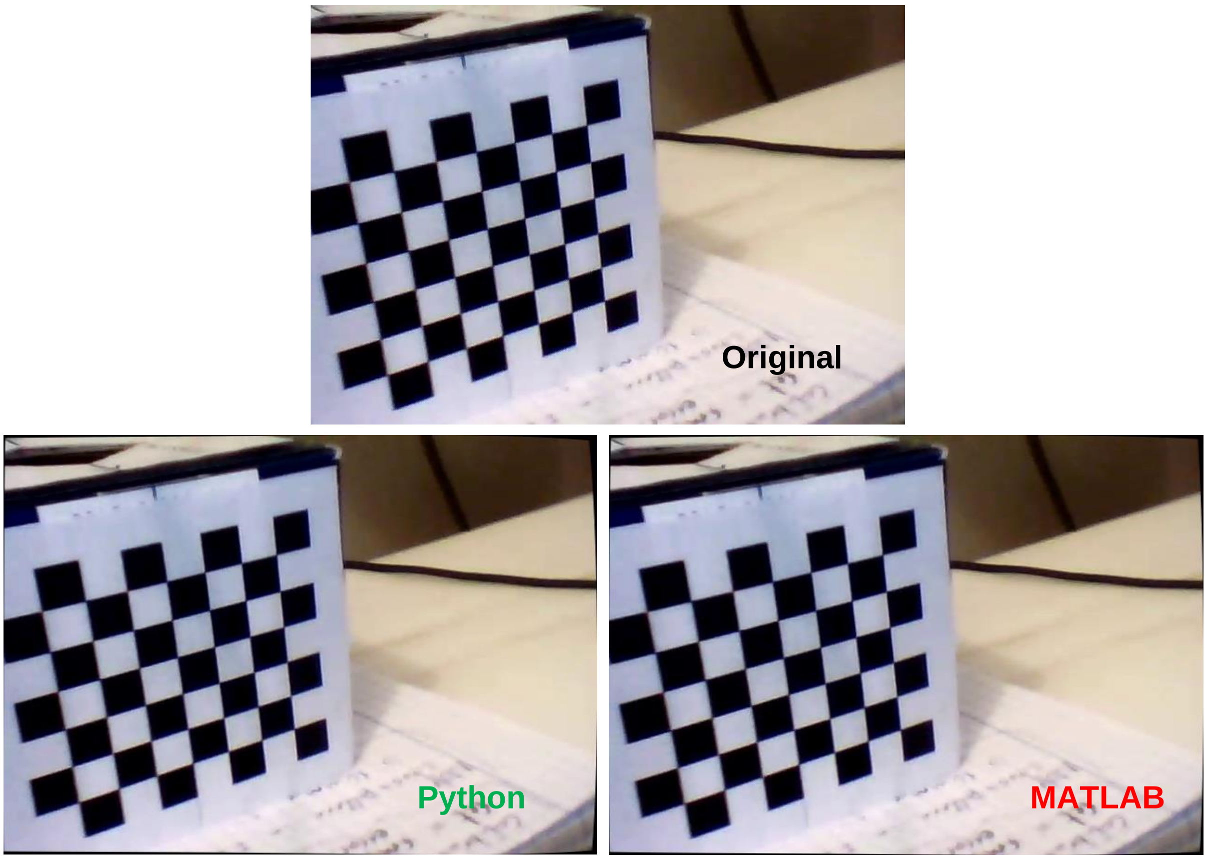


Figure 73: Webcam Image undistortion
***Top**: Original Image; **Bottom Left**: Python Undistortion; **Bottom Right**: MATLAB Undistortion*

4.2.2 Phone Camera (IPWebcam App)

Python Calibration:

$$Intrinsic\ Matrix = \begin{bmatrix} 1506.68 & 0 & 979.05 \\ 0 & 1507.14 & 545.00 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 1534.47 & 0 & 983.72 \\ 0 & 1527.97 & 546.07 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = [0.1653 \quad -0.4934 \quad 0.0021 \quad 0.0027 \quad 0.4999]$$

This camera was calibrated in 10.87 seconds using 60 images. These intrinsic parameters result in a mean reprojection error of 0.11 pixels and a root mean square reprojection error of 0.96 pixels.

MATLAB Calibration:

$$Intrinsic\ Matrix = \begin{bmatrix} 1502.6 & 0 & 976.5 \\ 0 & 1502.1 & 537.7 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 1539.5 & 0 & 978.7 \\ 0 & 1523.8 & 537.4 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = [0.1854 \quad -0.6729 \quad -0.0006 \quad 0.0019 \quad 0.9224]$$

These intrinsic parameters result in a mean reprojection error of 0.5264 pixels, with the mean reprojection error for each image provided, shown in Figure 75. Using these intrinsic parameters to distort the images captured by the respective camera results in the images seen in Figure 74.

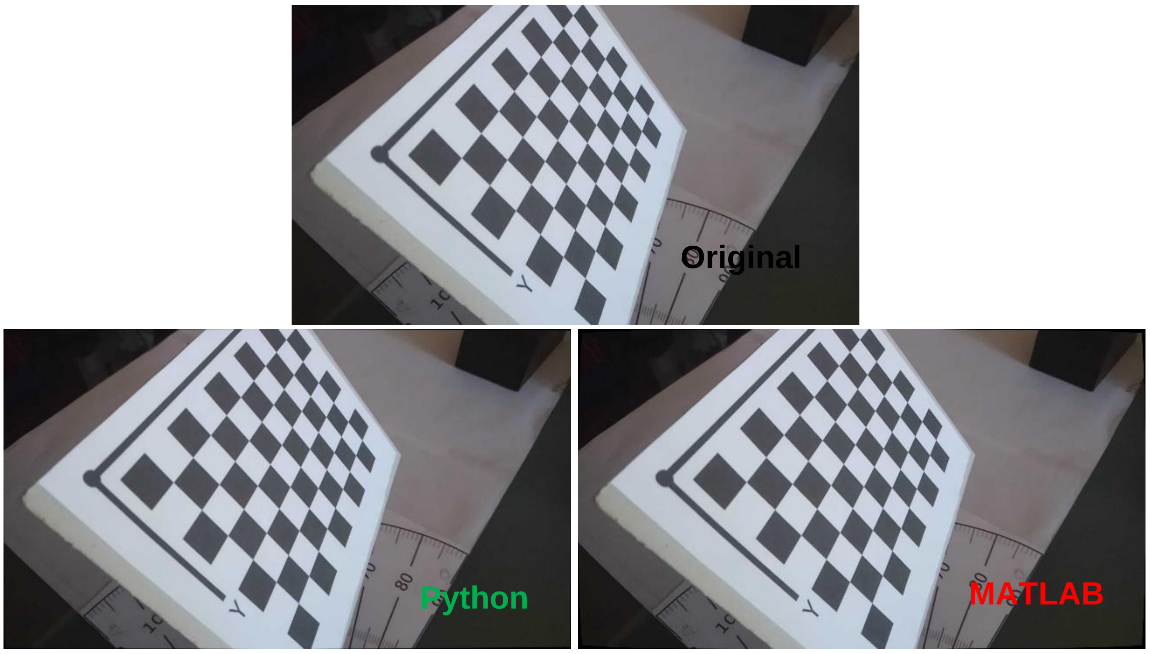


Figure 74: Phone Image Undistortion
***Top**: Original Image; **Bottom Left**: Python Undistortion; **Bottom Right**: MATLAB Undistortion*

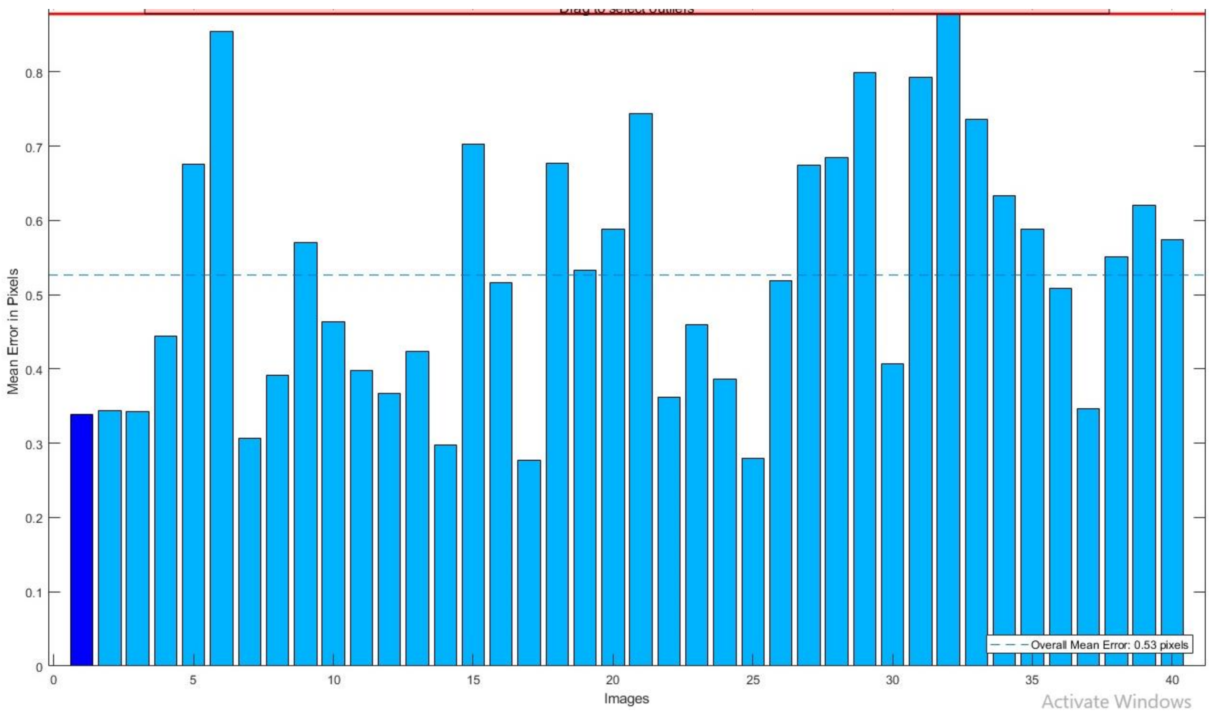


Figure 75: Images Mean Reprojection Errors

4.2.3 Raspberry Pi Camera V2

Python Calibration:

$$Intrinsic\ Matrix = \begin{bmatrix} 2647.41 & 0 & 1625.08 \\ 0 & 2649.74 & 1242.52 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 2600.09 & 0 & 1607.71 \\ 0 & 2594.37 & 1231.05 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = [0.1902 \quad -0.4105 \quad -0.0036 \quad -0.0030 \quad 0.1085]$$

This camera was calibrated in 124.03 seconds using 78 images. These intrinsic parameters result in a mean reprojection error of 0.1336 pixels and a root mean square reprojection error of 1.1248 pixels.

MATLAB Calibration:

$$Intrinsic\ Matrix = \begin{bmatrix} 2645.9 & 0 & 1634.1 \\ 0 & 2648.1 & 1246.4 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 2612.5 & 0 & 1626.2 \\ 0 & 2612.7 & 1238.9 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = [0.1963 \quad -0.4424 \quad -0.0027 \quad -0.0015 \quad 0.1554]$$

These intrinsic parameters result in a mean reprojection error of 0.7676 pixels, with the mean reprojection error for each image provided, shown in Figure 76. Using these intrinsic parameters to distort the images captured by the respective camera results in the images seen in Figure 77.

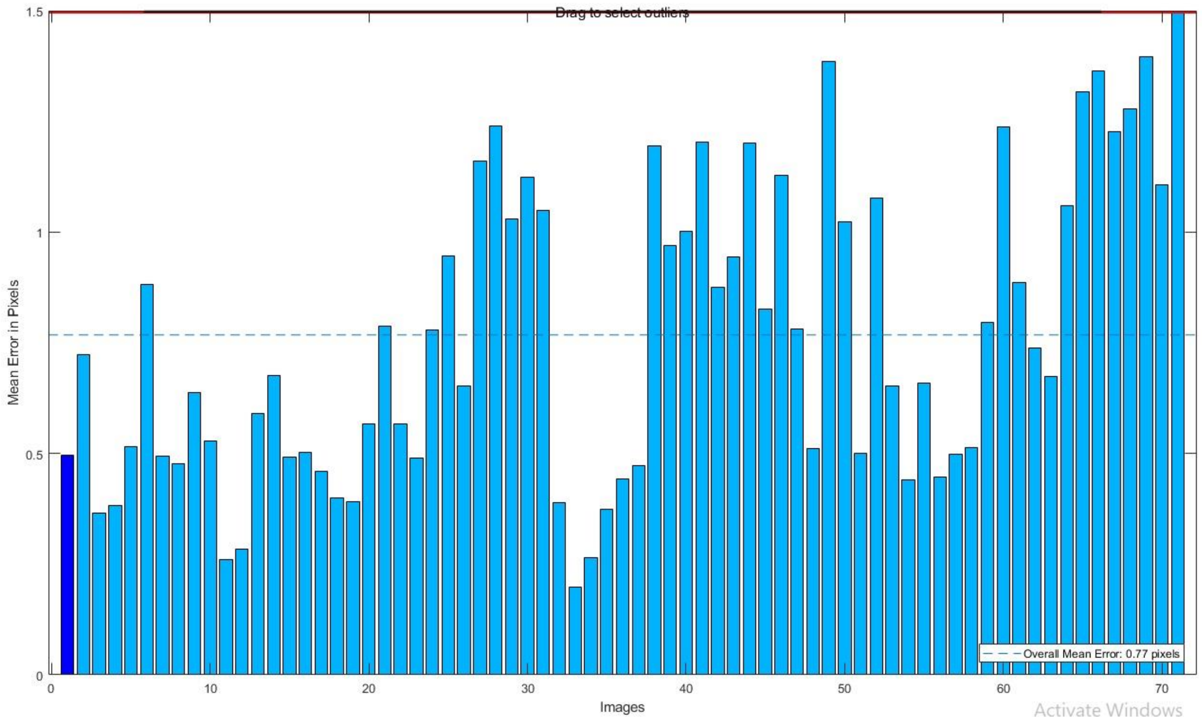


Figure 76: Images Mean Reprojection Errors

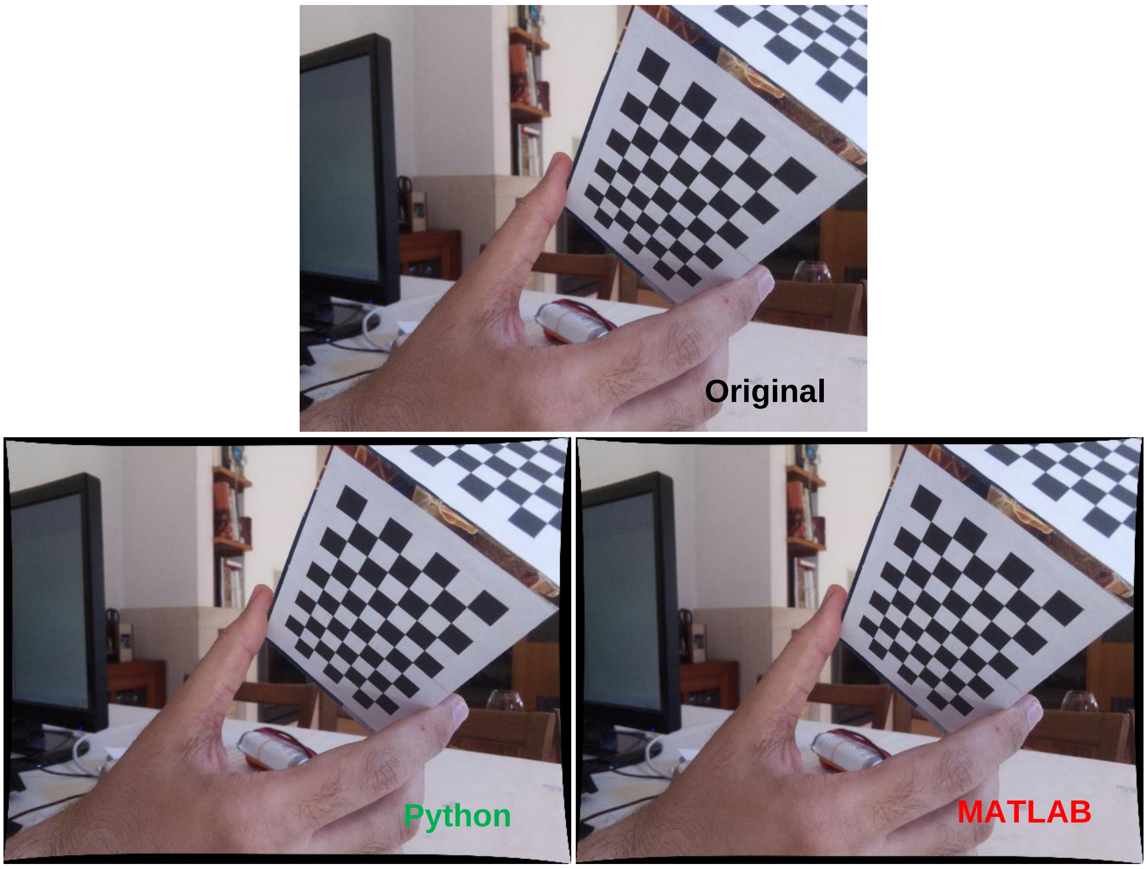


Figure 77: Raspberry Pi Image Undistortion
***Top**: Original Image; **Bottom Left**: Python Undistortion; **Bottom Right**: MATLAB Undistortion*

4.2.4 Genie Nano Camera System

Python Stereo Calibration:

Camera 1:

$$\mathit{Intrinsic\ Matrix}\ 1 = \begin{bmatrix} 2337.89 & 0 & 1954.44 \\ 0 & 2302.29 & 1262.92 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 5076.85 & 0 & 2023.74 \\ 0 & 3990.98 & 1433.45 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathit{Distortion\ Coefficients}\ 1 = \begin{bmatrix} 0.8528 & -35.2602 & -0.0369 & -0.0468 & 443.56 \end{bmatrix}$$

These intrinsic parameters result in a mean reprojection error of 38.55 pixels and a root mean square reprojection error of 1.09 pixels.

Camera 2:

$$\mathit{Intrinsic\ Matrix}\ 2 = \begin{bmatrix} 2369.29 & 0 & 1887.51 \\ 0 & 2301.63 & 1312.52 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 4350.99 & 0 & 2020.21 \\ 0 & 3458.68 & 1433.69 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathit{Distortion\ Coefficients}\ 2 = \begin{bmatrix} 0.3426 & -8.8095 & -0.0208 & -0.0176 & 105.0084 \end{bmatrix}$$

These intrinsic parameters result in a mean reprojection error of 29.43 pixels and a root mean square reprojection error of 0.84 pixels.

Python stereo calibration determines the rotation and translation matrix between the camera pair, such that:

$$\begin{cases} R_2 = RR_1 \\ T_2 = RT_1 + T \end{cases}$$

Where (R_i, T_i), is the pose of the object relative to the i^{th} camera.

$$Rotation\ between\ cameras = \begin{bmatrix} -0.2776 & -0.5233 & 0.8057 \\ 0.5625 & 0.5913 & 0.5778 \\ -0.7788 & 0.6136 & 0.1302 \end{bmatrix}$$

$$Translation\ vector\ between\ cameras = \begin{bmatrix} -3252.09 \\ -1977.43 \\ 3687.30 \end{bmatrix}$$

Lastly Python determines the essential, E, and fundamental, F, matrices through the following equations:

$$E = \begin{bmatrix} 0 & -T_2 & T_1 \\ T_2 & 0 & -T_0 \\ -T_1 & T_0 & 0 \end{bmatrix} R$$

$$F = Intrinsic\ Matrix\ 2^{-T} E Intrinsic\ Matrix\ 1^{-1}$$

Where T_i, are the elements of the translation vector $T = [T_0 \quad T_1 \quad T_2]^T$

$$Essential\ matrix = \begin{bmatrix} -534.12 & -3393.83 & -2388.10 \\ -3556.28 & 66.15 & 3394.21 \\ -2378.24 & -2957.78 & -285.98 \end{bmatrix}$$

$$Fundamental\ Matrix = \begin{bmatrix} 1.25 \times 10^{-6} & 8.17 \times 10^{-6} & 1.51 \times 10^{-4} \\ 8.45 \times 10^{-6} & -1.62 \times 10^{-7} & -3.48 \times 10^{-2} \\ -1.04 \times 10^{-4} & 8.83 \times 10^{-7} & 1 \end{bmatrix}$$

This stereo camera pair was calibrated in 461.14 seconds using 100 images per camera. These parameters, determined through stereo calibration, result in a root mean square reprojection error of 1.42 pixels.

MATLAB Stereo Calibration:

Camera 1:

$$Intrinsic\ Matrix = \begin{bmatrix} 2554.7 & 0 & 1506.8 \\ 0 & 2491.9 & 1296.1 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 2337.9 & 0 & 2915.9 \\ 0 & 2061.4 & 1545.5 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = [-0.0496 \quad 3.5578 \quad -0.0282 \quad -0.0945 \quad -20.3199]$$

These intrinsic parameters result in a mean reprojection error of 0.9142 pixels.

Camera 2:

$$Intrinsic\ Matrix = \begin{bmatrix} 2617.4 & 0 & 2458.4 \\ 0 & 2776.1 & 340.3 \\ 0 & 0 & 1 \end{bmatrix} \overset{\alpha=1}{\Longrightarrow} \begin{bmatrix} 1576.3 & 0 & 3064.5 \\ 0 & 1845.1 & 1682.1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ Coefficients = [-1.4228 \quad 3.3854 \quad 0.0540 \quad -0.0522 \quad -4.3887]$$

These intrinsic parameters result in a mean reprojection error of 0.8634 pixels.

Stereo calibration outputs the pose of the second camera relative of the first camera:

$$Rotation\ of\ camera\ 2\ relative\ to\ camera\ 1 = \begin{bmatrix} -0.5234 & -0.1411 & 0.8403 \\ 0.4943 & 0.7530 & 0.4343 \\ -0.6940 & 0.6427 & -0.3243 \end{bmatrix}$$

$$Translation\ of\ camera\ 2\ relative\ to\ camera\ 1 = \begin{bmatrix} 2384.4 \\ -1378.2 \\ 4671.0 \end{bmatrix}$$

Lastly the essential, E, and fundamental, F, matrices are determined such that:

$$\begin{cases} [P_2 \quad 1]E\begin{bmatrix} P_1 \\ 1 \end{bmatrix} = 0 \\ [P_2 \quad 1]F\begin{bmatrix} P_1 \\ 1 \end{bmatrix} = 0 \end{cases}$$

Where P_1, are the coordinates of a point in image 1 and P_2 are the coordinates of the corresponding point on image 2.

$$Essential\ matrix = \begin{bmatrix} -499.1 & -4115.8 & -2555.2 \\ -4448.3 & 1273.5 & -2468.6 \\ -1057.8 & 2476.7 & 575.9 \end{bmatrix}$$

$$Fundamental\ Matrix = \begin{bmatrix} -1 \times 10^{-4} & -6 \times 10^{-4} & -4.59 \times 10^{-2} \\ -6 \times 10^{-4} & 2 \times 10^{-4} & -1.827 \times 10^{-2} \\ -1.71 \times 10^{-2} & 2.4826 & 86.5887 \end{bmatrix}$$

These parameters determined through stereo calibration result in a mean reprojection error of 0.8888 pixels, with the mean reprojection error for each image pair provided, shown in Figure 78. Using these intrinsic parameters to distort the images captured by the respective camera results in the images seen in Figure 79.

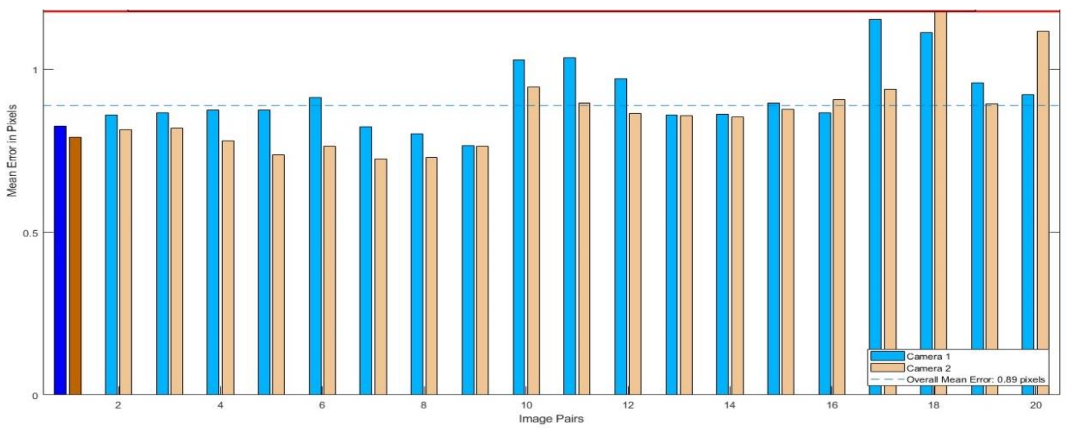


Figure 78: Image Pairs Mean Reprojection Errors

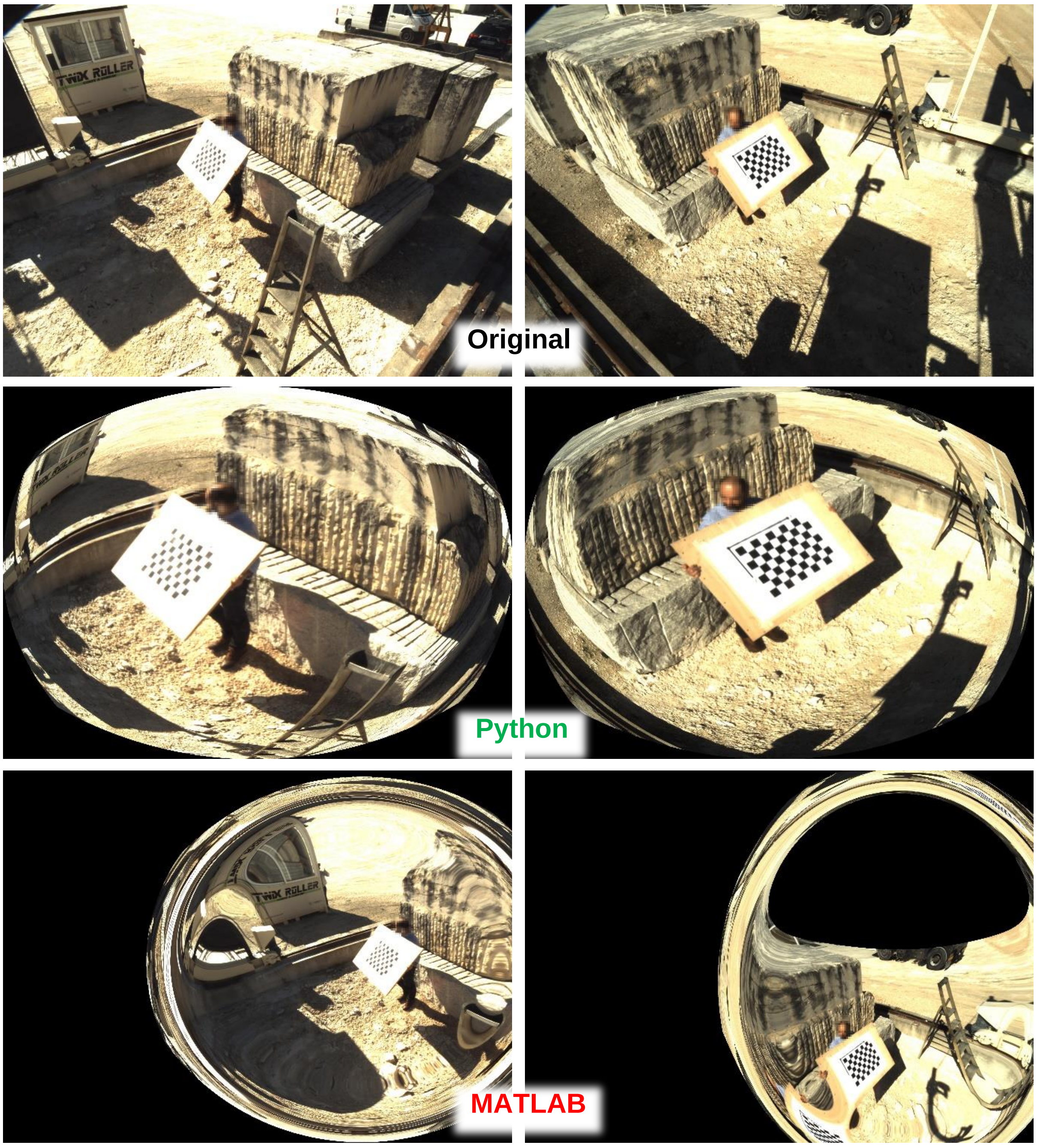


Figure 79: Genie Nano Camera Image Undistortion
***Top**: Original Images; **Bottom**: Undistorted Images*

4.3 Extrinsic Calibration

4.3.1 Raspberry Pi Automated Scan Circular Configurations

4.3.1.1 Pose Estimation with Aruco Marker

Figure 80 shows the Aruco markers image used for pose estimation and 3D plots of the estimated poses of the camera in the world coordinate system. In the 3D graphs, the black dot is the origin of our world frame, the Aruco corners are displayed by color-coded crosses and the cameras pose are displayed by the red dots with the blue, red and green lines indicating the camera's position and X, Y, Z-axis respectively.

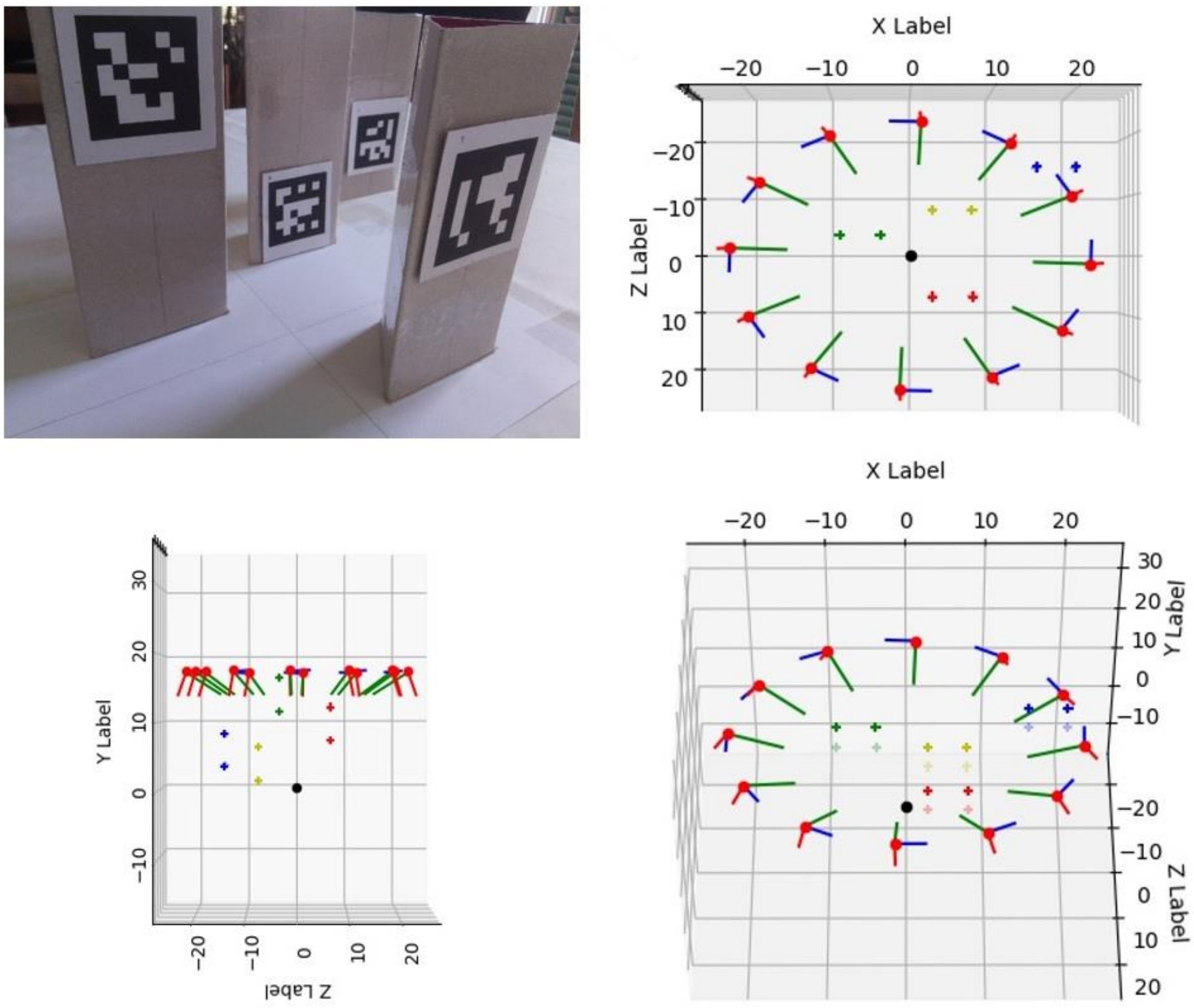


Figure 80: Raspberry Pi extrinsic calibration results with Aruco and iterative algorithm

4.3.1.2 Pose Estimation with Chessboard

Figure 81 shows the image used for pose estimation and 3D plots of the estimated poses of the camera in the world coordinate system. In the 3D graphs, the black dot is the origin of our world frame, the chessboard are displayed by color-coded crosses and the cameras pose are displayed by the red dots with the blue, red and green lines indicating the camera's position and X, Y, Z-axis respectively.

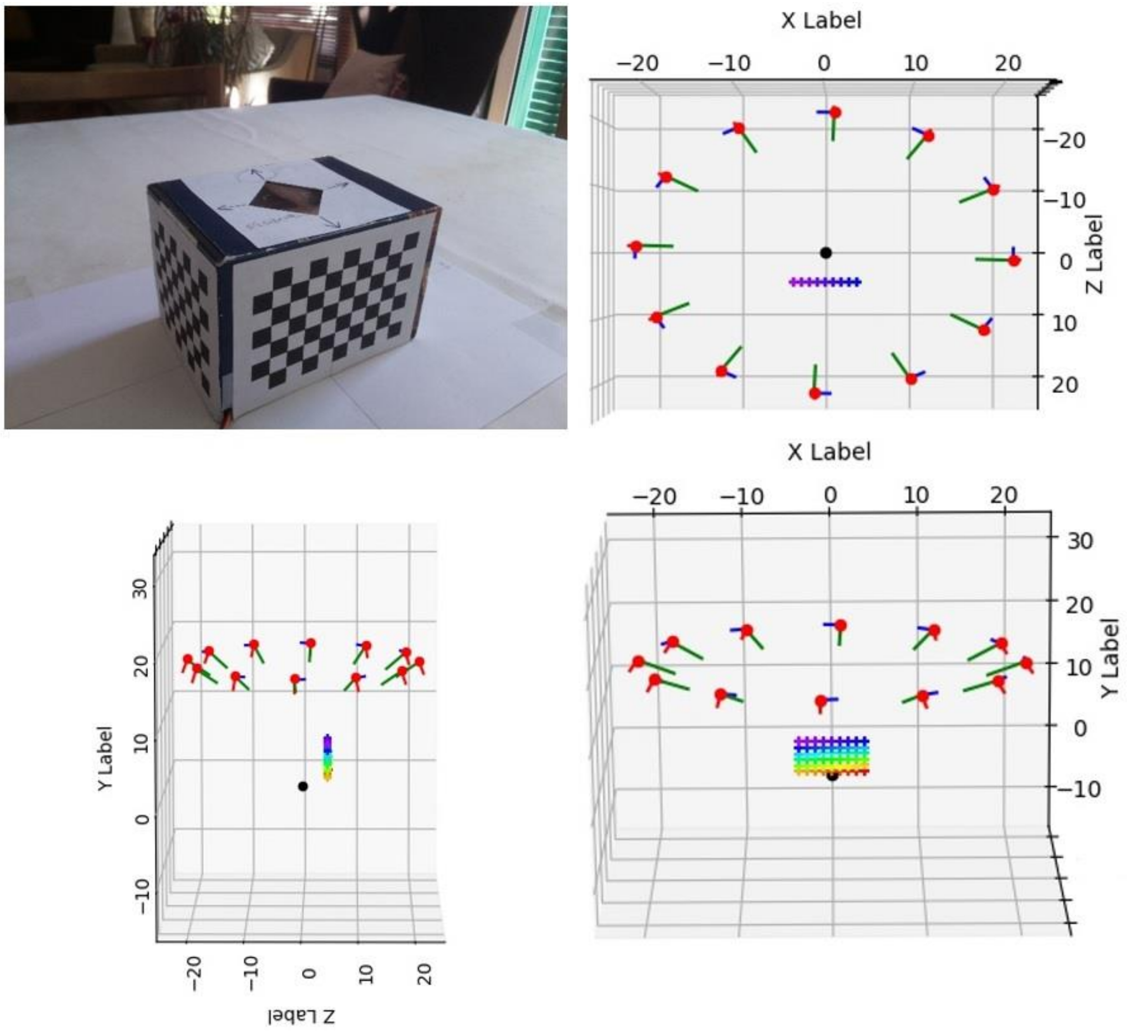


Figure 81:Raspberry Pi extrinsic calibration results with chessboard and iterative algorithm

Table 9 organizes the times, mean absolute and root mean square reprojection errors by algorithm and image used for comparison. The following chart displays the same information for easier visualization and comparison.

Table 9: Results for available algorithms

RANSAC	Pattern	Solver	Points (Inliers/Outliers)	Time (Seconds)	M.A.E. (Pixels)	R.M.S.E. (Pixels)
No	Aruco	Iterative	36/0	0.23	5.31	21.24
		EPnP		0.24	8.87	35.47
		P3P	4/0	0.24	4.89	19.57
		AP3P		0.24	4.89	19.57
Yes		Iterative	15/25	0.24	3.86	15.45
		EPnP		0.25	3.87	15.47
		P3P	14/26	0.24	4.02	16.09
		AP3P		0.25	4.02	16.09
No	Chess	Iterative	54/0	23.5	0.14	1.05
		EPnP		23.2	8.42	61.86
		P3P	4/0	22.3	0.29	2.11
		AP3P		23.0	0.29	2.11
Yes		Iterative	6/48	21.79	45.20	337.14
		EPnP		22.79	18.07	132.82
		P3P	52/2	22.26	11.85	87.11
		AP3P		23.70	11.85	87.11

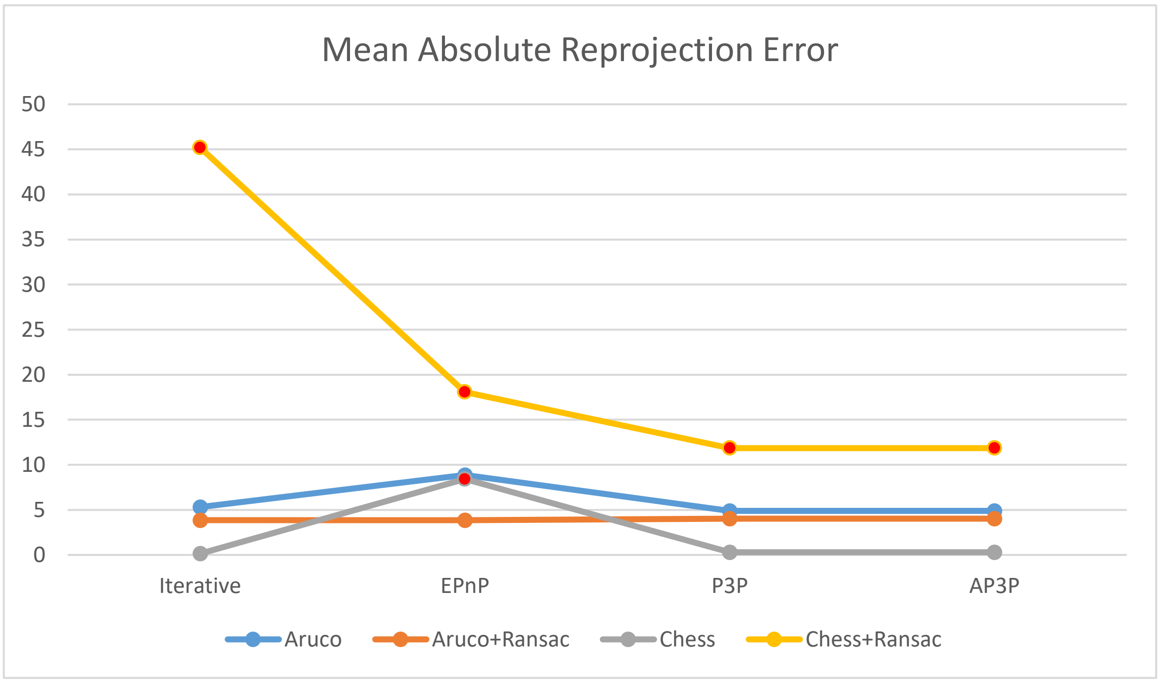


Figure 82: Circular scanner Mean Absolute Error by algorithm
Wrong pose estimations highlighted in red

As we can see from Table 9 and Figure 82, Aruco markers are faster to identify and result in an overall faster calibration than the chessboard pattern. However, the slowest calibration was still under half a minute, making any time concern negligible, this allows us to prioritize the accuracy of the different algorithms without much concern towards delays that might negatively impact the overall 3D reconstruction process efficiency.

Considering the calibrations performed without RANSAC, while the points from the Aruco markers were non-coplanar and more varied coordinates-wise, the lesser amount of points seems to have resulted in higher mean absolute and root mean square reprojection errors compared to the calibration using the chessboard pattern. Using RANSAC resulted in entirely wrong poses across all algorithms applied on the chessboard pattern, using only 11% of points with Iterative or Efficient-PnP algorithms, and using 96% of points with P3P and AP3P, on the other hand the Aruco markers seem to fair better managing to estimate a decent approximations while only using 42% of the points with Iterative or Efficient-PnP algorithms and 39% of the points with P3P and AP3P algorithms.

Overall all algorithms with or without RANSAC when applied on the Aruco markers, have resulted in the correct pose albeit with lower accuracy indicated by the reprojection errors, while the chessboard pattern has with some algorithms resulted in entirely wrong poses, showing however great accuracy when it does work properly.

Given that a small number of poses and patterns were used to test the algorithms, these results might not reflect the cause of their limitations or general behavior.

4.3.2 Genie Nano Camera System Linear Configuration

Figure 83 and Figure 84 shows the images used for pose estimation and 3D plots of the estimated poses of the camera in the world coordinate system, respectively. In the 3D graphs, the black dot is the origin of our world frame, the chessboard are displayed by color-coded crosses and the cameras pose are displayed by the colored dots with the blue, red and green lines indicating the camera's position and X, Y, Z-axis respectively.

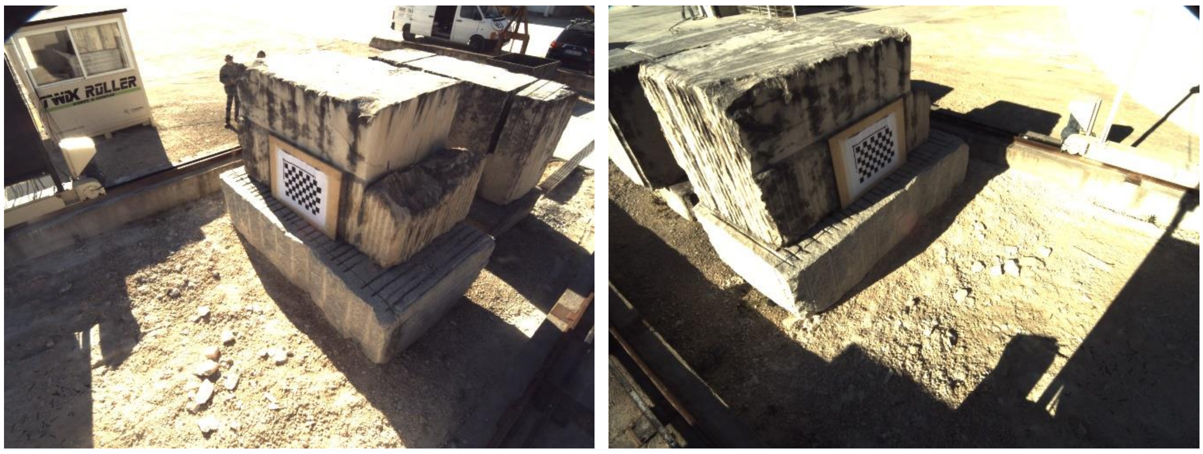


Figure 83: Genie Nano System Images for pose estimation

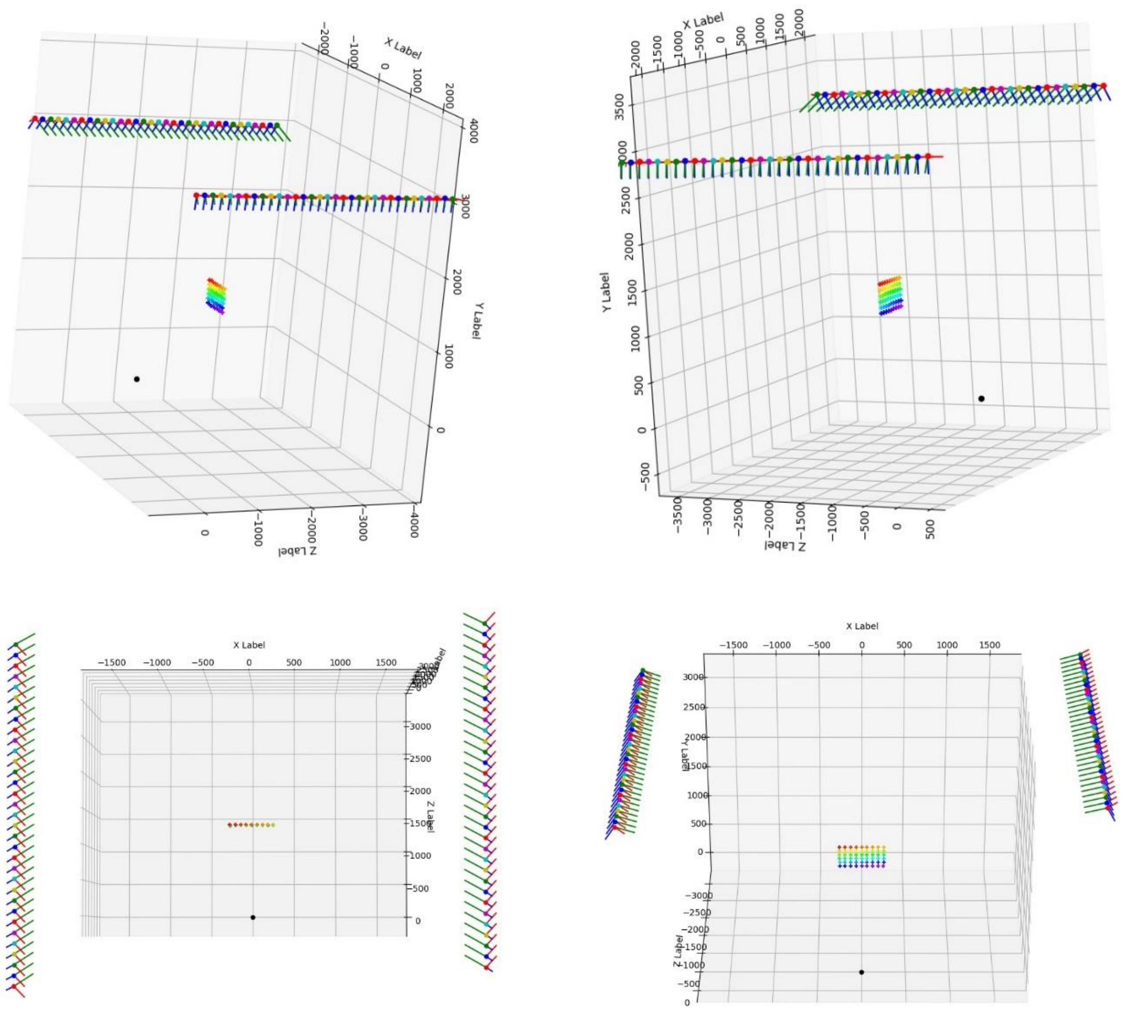


Figure 84: Genie Nano System extrinsic calibrations

Table 10: Results for available algorithms

RANSAC	Solver	Points (Inlier/Outliers)		Time (Seconds)	Mean Absolute Reprojection Error (Pixels)		Root Mean Square Reprojection Error (Pixels)	
		Cam 1	Cam 2		Cam 1	Cam 2	Cam 1	Cam 2
No	Iterative	54/0	54/0	5.77	0.27	2.01	0.26	1.88
	EPnP			5.82	1.07	7.90	2.41	17.68
	P3P	4/0	4/0	5.85	0.43	3.17	0.42	3.11
	AP3P			5.81	0.43	3.17	0.42	3.11
Yes	Iterative	50/4	26/28	5.95	0.29	2.12	0.45	3.30
	EPnP			5.97	1.36	9.97	3.24	23.83
	P3P	48/6	46/8	5.84	0.89	6.50	2.51	18.42
	AP3P		49/5	5.78	0.89	6.50	2.33	17.12

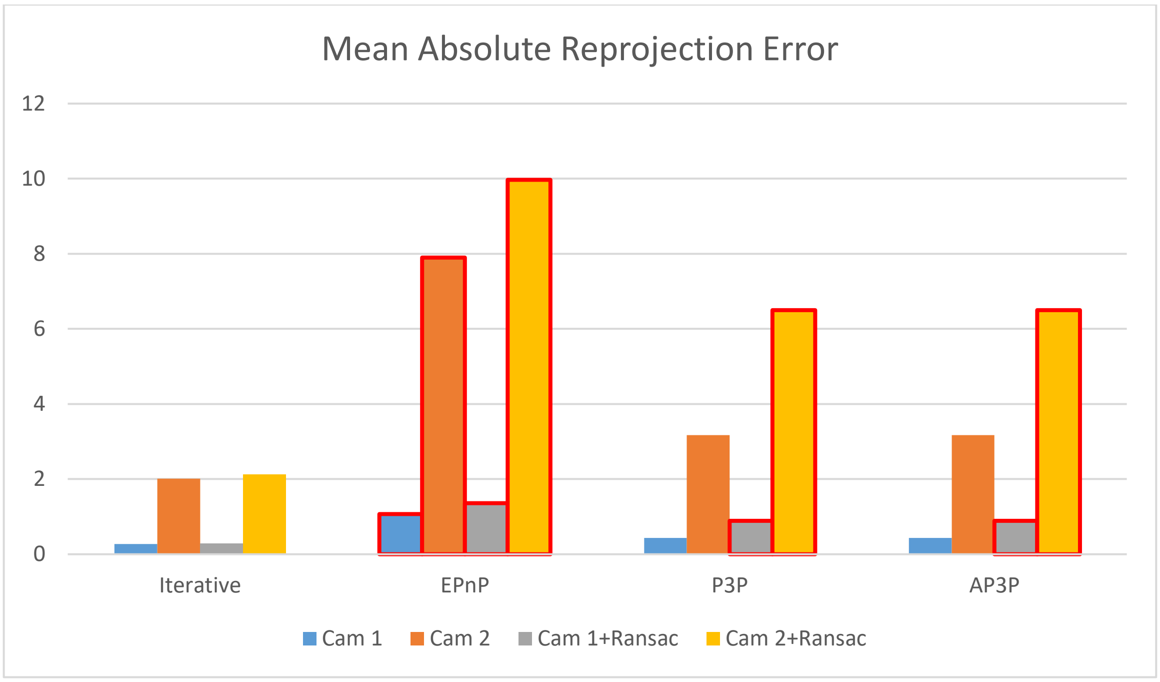


Figure 85: Linear scanner Mean Absolute Error by algorithm
Wrong pose estimations highlighted in red

As we can see from Table 10 that all the calibrations take similarly little less than six seconds making time comparisons irrelevant, this allows us to prioritize comparing the accuracy of the different algorithms. From Figure 85 we can see that RANSAC has induced wrong estimation for the P3P and AP3P algorithms while using 87% and 90% of points respectively. The iterative algorithms, both with and without RANSAC, obtained a proper estimation with significant lower error than the estimation performed by the P3P and AP3P algorithms without RANSAC. The Efficient-PnP algorithms performed poorly, resulting in wrong estimations with and without RANSAC.

It should be noted that while some pose estimations resulted in the wrong pose, its reprojection errors are not overtly higher than other correctly estimated poses. This led us to use the 3D plots, translation vector and rotations matrix as an added support towards evaluating the results. These evaluated wrong pose estimations are highlighted in the table and graphs in red.

Chapter 5

5 Conclusions

5.1 Future Work

5 Conclusions

The categories by which 3D scanners are classified are Contact Scanners and Non-Contact scanner, with the latter being divided into active and passive categories. Given the dimensions of the objects in need of scanning, using contact scanning would require a coordinate-measuring machine with a work area large enough to cover the large objects. Additionally, these contact scanners are slow and only provide volumetric data, no allowing for surface reconstruction. These limitations led us to focus on non-contact scanners which do not require contact, are faster and possibly more cost-effective compared to contact scanners, depending on the type of non-contact scanner considered. Within this category, non-contact scanners are considered active if they generate some form of emission which reflection is captured as data to be used for reconstruction, and passive if the data acquired originates solely from naturally available emissions. Given that the desired applications are set in a quarry, the detection of the non-natural emissions would be hindered by the nature of the working conditions. The level of sun-provided brightness varies from day to day or even hour to hour, the dust present from routine quarry activities and many other aspects would interfere with the recording device's ability to accurately capture the scanners emissions. For these reasons, we determined the most apt category of scanner for the application and working conditions is the non-contact passive scanner category. One of the proposed scanners was designed around the Fravizel's cutting machine to reduce costs and improve data acquisition.

The scanning system proposed to improve upon a quarry's cutting process was design to be incorporated into the existing Fravizel's cutting machine used by quarries. Succinctly, a non-contact passive scanner consists of a system of cameras aimed at the target object to acquired images of the object from different perspectives. For this system, a pair of stereo calibrated cameras was installed onto the cutting machine, using its infrastructure and mechanisms to support, move and trigger the cameras as synchronously as possible. The resulting images from a test scan were very clear and synchronized to the centisecond between the camera pair.

For the purpose of advertising the stone blocks pictures have some limitations, pictures taken close enough to allow the surface to be evaluated wouldn't capture the entire block given its dimensions, while a picture taken far enough away to capture the entire block might not allow the quality of the block's surface to be appreciated. A compromise would be to take a picture of the entire block with high enough resolution to allow surface detail upon zooming, however a better alternative is using a 3D displayers that allows a 3D model to be rotated, for a full appreciation of the volumetric dimensions and allows the model to be zoomed in and out, to inspect the surface's quality. A scanning system with a circular movement was proposed to acquire a set of pictures from perspectives in a 360-degree motion around the object. A small-scale version of this system was constructed, where a single camera remains stationary while the object is rotated, through the use of a servo motor, at consistent intervals between image acquisitions, simulating the motion of the camera itself around a known circle centered on the object.

The real-scale scanner consisted of two Dalsa's Genie Nano C4020 cameras and accessories, such as protective housing, costing a total of 13 575,39€. This linear scanner produced scans of twelve-megapixels synchronized images, and while the intrinsic calibration was sub-optimal the extrinsic calibration produced acceptable results with some algorithms, notably the iterative algorithm. The intrinsic calibration took 461,13 seconds to estimate the intrinsic values, resulting in high mean absolute reprojection errors, 38,55 and 29,43 pixels for camera 1 and 2 respectively, and badly estimated distortion coefficients can be seen by the undistorted images in Figure 79. This can be improved by using a set of images with the detectable chessboard pattern more evenly distributed around the field-of-view, especially the edges of the field-of-view for better distortion coefficients estimation. Considering the use of a sub-optimal intrinsic calibration, the iterative algorithms resulted in correct extrinsic calibrations, with and without RANSAC, showing low mean absolute reprojection errors of 0.27-0.29 and 2.01-2.12 pixels for camera 1 and camera 2 respectively, while P3P-based algorithms only estimated extrinsic parameters correctly without RANSAC, showing mean absolute reprojection errors of 0.43 and 3.17 pixels for camera 1 and camera 2 respectively. The incorrectly estimated extrinsic parameters, using P3P-based algorithms, show mean absolute reprojection errors of 0.89 and 6.50 pixels for camera 1 and camera 2 respectively. The efficient-PnP algorithms failed to properly estimate extrinsic parameters with and without RANSAC, showing for camera 1 and camera 2, mean absolute reprojection errors of 1.07 and 7.9 pixels without RANSAC and 1.36 and 9.97 pixels with RANSAC respectively

However, with proper intrinsic values the algorithms that failed might function properly, conveying reliability to the extrinsic calibration process across all algorithms and circumstances. Considering that intrinsic calibration is only required if the cameras internal properties are altered between scans and that extrinsic calibration are required for each scan, the time consumption of a scan boils down to the time it takes to intrinsically calibrate the cameras divided by the number of scans performed, plus the time to acquire the images and perform the extrinsic calibration for each scan. Considering that the intrinsic calibration took no longer than 8 minutes and the extrinsic calibration slightly less than 6 seconds, the scanning process is accomplished nearly in the same time it takes to move the Fravizel's cutting machine over the stone blocks.

The small-scale circular scanner consisted of a Raspberry Pi 3B+, a Pi Camera v2, a continuous servo motor and a 3D printed kinematic chain, producing scans of eight-megapixel images. The camera was properly intrinsically calibrated, in 124,03 seconds, showing mean absolute reprojection error of 0,13 pixel. Several methods were used to achieve accurate extrinsic calibrations, the chessboard pattern or four Aruco markers were used to generate world to image coordinate correspondences while different algorithms were used for comparison. The Aruco markers resulted in proper extrinsic calibrations across all algorithms used with and without RANSAC, albeit with higher mean absolute reprojection errors, of 3.86 and 4.02 pixels respectively, when compared to extrinsic calibrations accomplished without RANSAC using the chessboard pattern, which resulted in mean absolute reprojection errors within 0.14 to 0.29 pixels, with the exception of the efficient-PnP algorithm which incorrectly estimated extrinsic parameter, showing a mean absolute reprojection error of 8.42 pixels. Using RANSAC on the chessboard pattern improperly estimated extrinsic parameters with all algorithms, resulting in mean absolute reprojection errors of 45.2, 18.07 and 11.85 pixels for iterative, efficient-PnP, and P3P-based algorithms respectively.

5.1 Future Work

While the proposed scanner's data acquisition viability yielded positive results, it also showed room for improvement in the form of possible next steps and optimizations. Considering these scanners are a first step in a contribution to the quarry industry, some of the possible improvements are described subsequently.

5.1.1 Second stereo camera pair for linear scanner

Since the viability of the proposed solution must be tested for the desired application, the linear scanning system incorporated into Fravizel's machine consisted of only one pair of stereo calibrated cameras, attached to parallel poles. Even if successful, this physical configuration leaves a portion of the stone blocks unreconstructed, corresponding to blind spots between the camera's fields-of-view. This hole in the reconstruction can be easily minimized by adding another stereo pair of cameras, onto the cutting machine's secondary portico, aimed at the blind spot of the first stereo camera pair. For the purpose of the cutting process, there is a higher emphasis on the volumetric aspect of the blocks over surface. However, there may be aesthetically unpleasant surfaces that quarries might prefer to remove at an acceptable loss of material, therefore, investment into surface reconstruction might be a future development.

5.1.2 Establish Real scale Circular Configuration

Given that the proposed small-scale circular scanner showed viability in obtaining the data for the reconstruction of good quality 3d models absent of blind spots, the next step would be to design a full-scale version of this scanner and implement it on the processed stone blocks. The small-scale solution took the route of rotating the object itself instead of the camera out of practicality and efficiency, however the real application will be dealing with large, heavy, hard to rotate stone blocks instead of small pavement stones. For this reason, the full-scale application might not employ the proposed circular scanner configuration directly and instead could take one of two routes. A circular rail for the camera to move around the stationary stone blocks can be designed, or applying the proposed solution exactly, the stone blocks could be rotated while maintaining the camera in a known stationary position. The option that would require the rotation of the stone would be more appealing and cost-effective, if the quarries already have such a platform that rotates the stone blocks, requiring the simple installation of the camera and set it to be triggered externally by the encoder of the motor used to rotate the platform, similarly to the system implemented on the cutting machines. Once the 3D models are reconstructed, the use of a website with an 3D model player, would then allow quarries to showcase their products in a far more comprehensive, attractive and accurate way.

5.1.3 Surface Enrichment

Through our research, active scanners were dismissed on account of the working conditions interfering with the capture of the scanner generated emissions, such as lasers or projections. However, a simple and economical addition can be made into the scanner to assist the reconstruction process when the working conditions allow it or in an application with a more controlled working area. 3D reconstruction through images is based on detectable features on the surface of the object across the set of images, the ability to reconstruct and the quality of the model is related with the amount and quality of these features. Considering the uniformity of limestone blocks texture and color-wise, a projector can be used to project noise function-based patterns onto the stone blocks to enrich possibly featureless surfaces, as shown in [52].

References

[1] S. A. Nelson, “Mineral Resources.” https://bit.ly/2H8ntPc (accessed Sep. 25, 2020).

[2] J. A. Harrell and P. Storemyr, “Ancient Egyptian quarries – an illustrated overview,” *Geolocical Suevey Norw. Spec. Publ.*, vol. 12, no. 2009, pp. 7–50, 2009.

[3] L. Johansson Westholm and D. Alderton, *Mineral Resources*. Elsevier Inc., 2015.

[4] “Coimbra_University_seal.jpg (669×500).” https://upload.wikimedia.org/wikipedia/commons/4/4a/Coimbra_University_seal.jpg (accessed Sep. 25, 2020).

[5] “cantera-caliza-capri-845x664.jpg (845×664).” https://grupomos.com/wp-content/uploads/2019/05/cantera-caliza-capri-845x664.jpg (accessed Sep. 26, 2020).

[6] “HTB1CMsWLpXXXXbuXVXXq6xXFXXXD.jpg_350x350.jpg (350×350).” https://sc02.alicdn.com/kf/HTB1CMsWLpXXXXbuXVXXq6xXFXXXD.jpg_350x350.jpg (accessed Sep. 27, 2020).

[7] “400px-Kalkstein_(nahe).JPG (400×600).” https://upload.wikimedia.org/wikipedia/commons/thumb/d/d4/Kalkstein_%28nahe%29.JPG/400px-Kalkstein_%28nahe%29.JPG (accessed Sep. 27, 2020).

[8] F. Franceschini, M. Galetto, D. Maisano, and L. Mastrogiacomo, “Large-scale dimensional metrology (LSDM): From tapes and theodolites to multi-sensor systems,” *Int. J. Precis. Eng. Manuf.*, vol. 15, no. 8, pp. 1739–1758, 2014, doi: 10.1007/s12541-014-0527-2.

[9] E. Savio, “Coordinate Measuring Machine,” in *CIRP Encyclopedia of Production Engineering*, 2019.

[10] X. H. Li, B. Chen, and Z. R. Qiu, “The calibration and error compensation techniques for an Articulated Arm CMM with two parallel rotational axes,” *Meas. J. Int. Meas. Confed.*, vol. 46, no. 1, pp. 603–609, 2013, doi: 10.1016/j.measurement.2012.08.020.

[11] T. Oiwa, “Coordinate Measuring Machine using Parallel Mechanism,” *Proc. 16th IMEKO World Congr.*, 2000.

[12] R. J. Hocken and P. H. Pereira, *Coordinate measuring machines and systems: Second edition*. 2016.

[13] “The History of the Coordinate Measuring Machine.” https://www.cmm-solutions.co.uk/contact-us/cmm-history/ (accessed Sep. 22, 2020).

[14] M. A.-B. Ebrahim, “3D Laser Scanners’ Techniques Overview,” *Int. J. Sci. Res.*, vol. 4, no. 10, pp. 323–331, 2013.

[15] M. M. P. A. Vermeulen, P. C. J. N. Rosielle, and P. H. J. Schellekens, “Design of a high-precision 3D-coordinate measuring machine,” *CIRP Ann. - Manuf. Technol.*, vol. 47, no. 1, pp. 447–450, 1998, doi: 10.1016/s0007-8506(07)62871-6.

[16] B. Curless, “From Range Scans to 3D Models,” *Comput. Graph.*, vol. 33, no. 4, pp. 38–41, 1999, doi: 10.1145/345370.345399.

[17] J. Butime, I. Gutierrez, L. G. Corzo, and C. F. Espronceda, “3D reconstruction methods, a survey,” *VISAPP 2006 - Proc. 1st Int. Conf. Comput. Vis. Theory Appl.*, vol. 2, pp. 457–463, 2006, doi: 10.5220/0001369704570463.

[18] Z. Zhang *et al.*, “Three-dimensional shape measurements of specular objects using phase-measuring deflectometry,” *Sensors (Switzerland)*, vol. 17, no. 12, 2017, doi: 10.3390/s17122835.

[19] L. Huang, M. Idir, C. Zuo, and A. Asundi, “Review of phase measuring deflectometry,” *Opt. Lasers Eng.*, vol. 107, no. March, pp. 247–257, 2018, doi: 10.1016/j.optlaseng.2018.03.026.

[20] C. Rocchini, P. Cignoni, C. Montani, P. Pingi, and R. Scopigno, “A low cost 3D scanner based on structured light,” *Comput. Graph. Forum*, vol. 20, no. 3, pp. 299–308, 2001, doi: 10.1111/1467-8659.00522.

[21] C. Frohlich and M. Mettenleiter, “Terrestrial laser scanning - new perspectives in 3D surveying,” *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 36, no. Part 8, p. W2, 2004.

[22] L. Bornaz and F. Rinaudo, “Terrestrial laser scanner data processing,” *XXth ISPRS Congr. Istanbul*, 2004.

[23] R. Hartley and A. Zisserman, *Multiple View Geometry in computer vision*, Second. 2004.

[24] “sfm — openMVG library.” https://openmvg.readthedocs.io/en/latest/openMVG/sfm/sfm/ (accessed Oct. 17, 2020).

[25] S. Bianco, G. Ciocca, and D. Marelli, “Evaluating the performance of structure from motion pipelines,” *J. Imaging*, vol. 4, no. 8, pp. 1–18, 2018, doi: 10.3390/jimaging4080098.

[26] S. Galliani, “Multi-View 3 D Reconstruction With Geometry and Shading,” no. 24863, 2018.

[27] Z. Zhang, “A flexible new technique for camera calibration,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000, doi: 10.1109/34.888718.

[28] W. Burger, “Zhang’s Camera Calibration Algorithm: In-Depth Tutorial and Implementation,” *Tech. Rep.*, p. 55, 2016, doi: 10.13140/RG.2.1.1166.1688.

[29] “coordinate_axes_3d.png (800×398).” https://motion.cs.illinois.edu/RoboticSystems/figures/modeling/coordinate_axes_3d.png (accessed Sep. 27, 2020).

[30] “image-1582219553536-7e66425f6594fca53e6b3bf340ae116f.png (1720×1122).” https://bs-uploads.toptal.io/blackfish-uploads/uploaded_file/file/190900/image-1582219553536-7e66425f6594fca53e6b3bf340ae116f.png (accessed Oct. 30, 2020).

[31] “pnp.jpg (600×420).” https://docs.opencv.org/3.4.0/pnp.jpg (accessed Oct. 03, 2020).

[32] X. X. Lu, “A Review of Solutions for Perspective-n-Point Problem in Camera Pose Estimation,” 2018.

[33] “OpenCV: Camera Calibration and 3D Reconstruction.” https://bit.ly/358IgtR (accessed Oct. 03, 2020).

[34] X. Gao, X. Hou, J. Tang, and H. Cheng, “Complete Solution Classification for the Perspective-Three-Point Problem,” vol. 25, no. 8, pp. 930–943, 2003.

[35] T. Ke and S. I. Roumeliotis, “An efficient algebraic solution to the perspective-three-point problem,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 4618–4626, 2017, doi: 10.1109/CVPR.2017.491.

[36] V. Lepetit, F. Moreno-Noguer, and P. Fua, “EPnP: An accurate O(n) solution to the PnP problem,” *Int. J. Comput. Vis.*, vol. 81, no. 2, pp. 155–166, 2009, doi: 10.1007/s11263-008-0152-6.

[37] S. Chou and X. Gao, “Ritt-Wu ’ s Decomposition Algorithm ∗,” *Sci. York*, vol. 2, pp. 1–15.

[38] W. Wen-Tsun, “Basic principles of mechanical theorem proving in elementary geometries,” *J. Autom. Reason.*, vol. 2, no. 3, pp. 221–252, 1986, doi: 10.1007/BF02328447.

[39] P. Khlebovich, “IP Webcam - Apps on Google Play.” https://bit.ly/3kd8yBy (accessed Sep. 30, 2020).

[40] “Raspberry Pi hardware - Raspberry Pi Documentation.” https://bit.ly/3dC5TyM (accessed Sep. 30, 2020).

[41] “Camera Module - Raspberry Pi Documentation.” https://bit.ly/3o3Ri3V (accessed Sep. 30, 2020).

[42] “Mechanical Drawings - Raspberry Pi Documentation.” https://bit.ly/2T3vQxL (accessed Sep. 30,

2020).

[43] D. Sawicz, “Hobby Servo Fundamentals,” pp. 1–11, 2002.

[44] “MG995 Servo Motor Pinout.” https://components101.com/motors/mg995-servo-motor (accessed Sep. 30, 2020).

[45] “GPIO - Raspberry Pi Documentation.” https://bit.ly/3452iGs (accessed Sep. 30, 2020).

[46] C. A. Scan, “Genie Nano Series ™ Camera User’s Manual 1 Gb GigE Vision-Monochrome & Color Area Scan,” 2019.

[47] GOYO OPTICAL INC., “Industrial Lens, Item No . GM12HR58018MCN,” vol. 526, no. mm, p. 1.

[48] “Massload.” http://www.massload.com.tw/ (accessed Oct. 30, 2020).

[49] I. Core, G. Ports, P. Cassette, and M. Interface, “Nuvo-7000E / 7000DE / 7000P Series,” pp. 1–3.

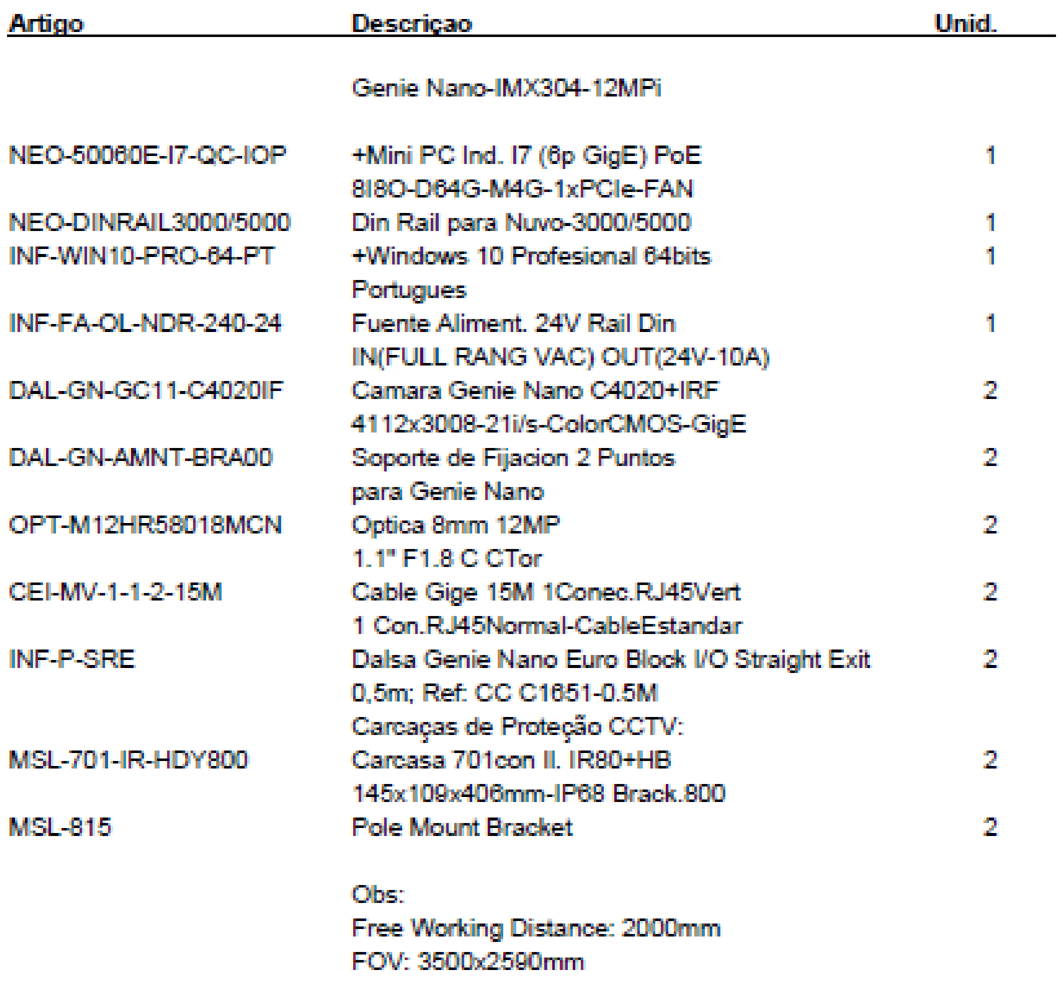
[50] “Single Camera Calibrator App - MATLAB & Simulink.” https://bit.ly/3lTICva (accessed Sep. 30, 2020).

[51] “Camera Calibration and 3-D Vision - MATLAB & Simulink.” https://bit.ly/347w1OT (accessed Sep. 30, 2020).

[52] A. Koutsoudis, G. Ioannakis, B. Vidmar, F. Arnaoutoglou, and C. Chamzas, “Using noise function-based patterns to enhance photogrammetric 3D reconstruction performance of featureless surfaces,” *J. Cult. Herit.*, vol. 16, no. 5, pp. 664–670, 2015, doi: 10.1016/j.culher.2015.01.008.

Appendix

A – Linear Scanner hardware budget



Artigo	Descriçao	Unid.
	Genie Nano-IMX304-12MPi	
NEO-50060E-I7-QC-IOP	+Mini PC Ind. I7 (6p GigE) PoE 8I8O-D64G-M4G-1xPCIe-FAN	1
NEO-DINRAIL3000/5000	Din Rail para Nuvo-3000/5000	1
INF-WIN10-PRO-64-PT	+Windows 10 Profesional 64bits Portugues	1
INF-FA-OL-NDR-240-24	Fuente Aliment. 24V Rail Din IN(FULL RANG VAC) OUT(24V-10A)	1
DAL-GN-GC11-C4020IF	Camara Genie Nano C4020+IRF 4112x3008-21i/s-ColorCMOS-GigE	2
DAL-GN-AMNT-BRA00	Soporte de Fijacion 2 Puntos para Genie Nano	2
OPT-M12HR58018MCN	Optica 8mm 12MP 1.1" F1.8 C CTor	2
CEI-MV-1-1-2-15M	Cable Gige 15M 1Conec.RJ45Vert 1 Con.RJ45Normal-CableEstandar	2
INF-P-SRE	Dalsa Genie Nano Euro Block I/O Straight Exit 0,5m; Ref: CC C1651-0.5M	2
	Carcaças de Proteção CCTV:	
MSL-701-IR-HDY800	Carcasa 701con Il. IR80+HB 145x109x406mm-IP68 Brack.800	2
MSL-815	Pole Mount Bracket	2

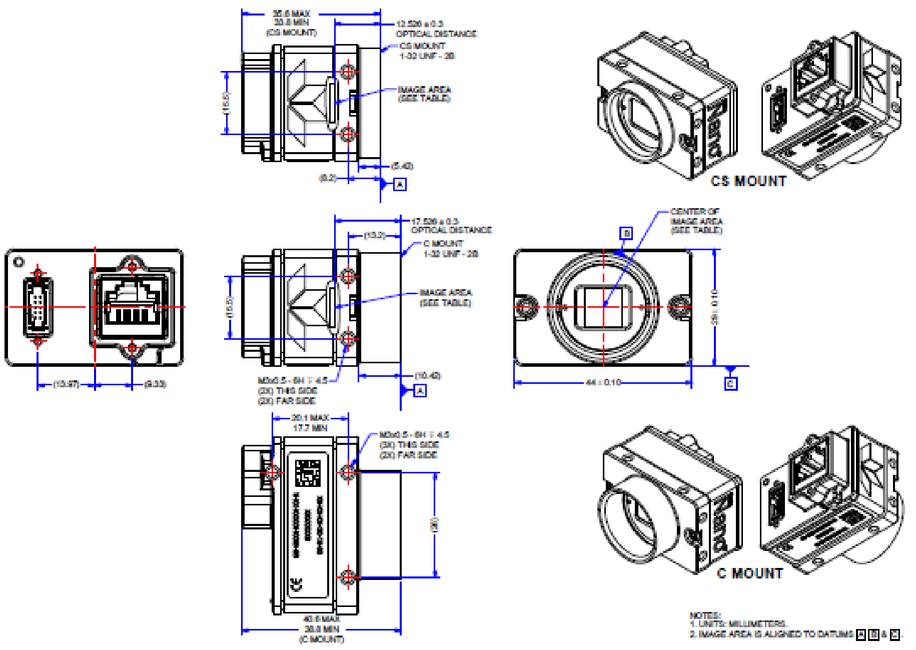
Obs:
Free Working Distance: 2000mm
FOV: 3500x2590mm

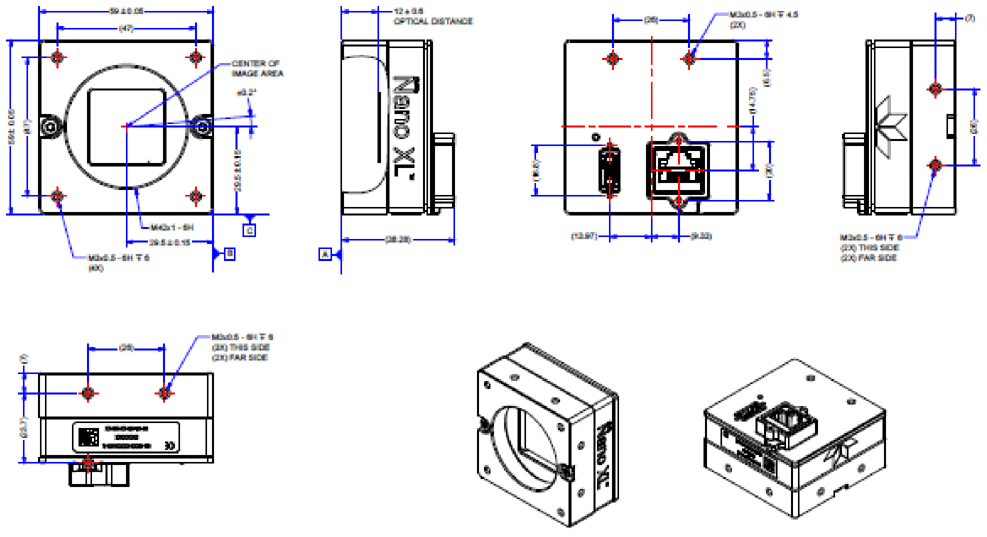
B – Image sensor optical and mechanical specifications

Genie Nano C4020 Color
Part number: G3-GC11-C4020

Genie Nano C4020 Color Part number: G3-GC11-C4020	4112 x 3008 *Sony IMX304*	Color	GigE Vision	3.45 µm	**Standard:** 9.7 fps **TurboDrive:** 20 fps

Total Pixels	Interface Speed	Bit Depth	Operating Temperature
12.4 MP	1 Gbps	8 bit, 12 bit	-20—65 °C
Dynamic Range	**Supported Lens Options**	**Shutter Type**	**Power Requirement**
76 dB	C-Mount, CS-Mount	Global Shutter	PoE or 10-36 VDC
Synchronisation	**Exposure Control**	**General Purpose I/O**	**Dimensions (W x H x L)**
software trigger, free-run, hardware trigger, PTP	hardware trigger, programmable via camera API, automatic	2 digital input, 2 digital output	21 x 29 x 44 mm
Mass			
46 g			





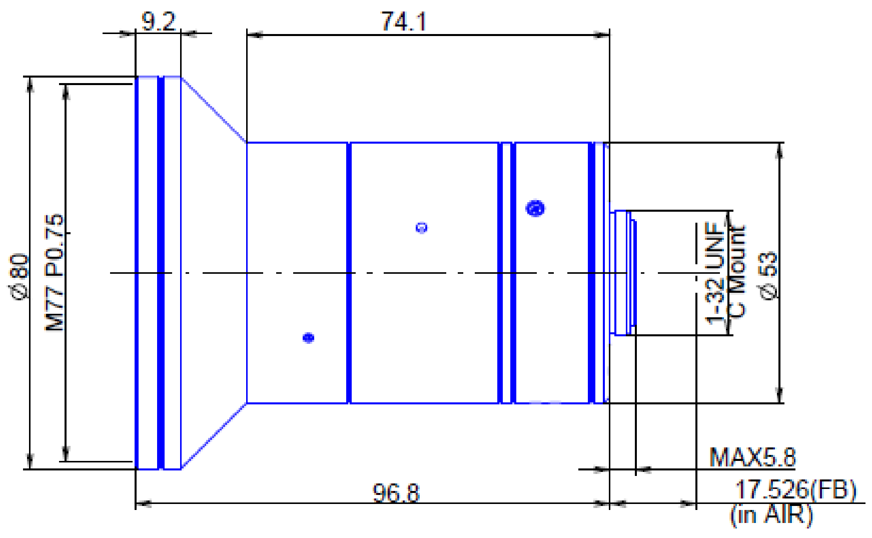
1703

GOYO OPTICAL INC.

Industrial Lens

1.1inch Format
High resolution 12 Megapixel Lenses

Item No.GM12HR58018MCN



ITEM NO.		GM12HR58018MCN	
Focal Length		8	(mm)
Iris Range		F1.8 - 22	
Angle of View	1.1"	83°5' x 66°7'	
(Hx V x D)			
MOD		0.3	(m)
Filter Thread		M=77 , P=0.75	
Dimention (D x L)		Ø80 x 96.8	(mm)
Weight		500	(g)

GOYO OPTICAL Inc.,

3-8-31 HAMASAKI, ASAKA, SAITAMA 351-0033, JAPAN,
TEL:+81-48-474-2235 FAX:+81-48-474-7373
http://www.goyooptical.com E-MAIL:info@goyooptical.com

GOYOOPTICAL Inc.,

D – Camera Housing specifications

INFAIMON

CI-701 Camera Housing

Specifications	Item NO.	CI-701 Camera Housing
	Viewing Window	Tempered Glass Insert Patented
	Construction	Die Casting Al-alloy
	Latch	Duel side locking by screw
	Camera Mounting	Removable Camera sled
	Heater (Optional)	AC24V,110V,230V,12V
	Heater On/Off	15°C On, 25°C Off
	Fan (Optional)	DC 12V
	Fan On/Off	35°C On, 25°C Off
	Housing Dimensions	L 406 mm x H 109 mm x W 145 mm
	Coating	Beige and white epoxy powder coating
	Kit	Pre assembled
	Housing Style	Clam Shell
Environmental	Operating Temperature	-50° to 50° C
	Application Use	Outdoor / Indoor
Certification	IP68	
	RoHS Compliant	
	CE	
IR LED Range	30M & 50M	

Accessories

CI-800

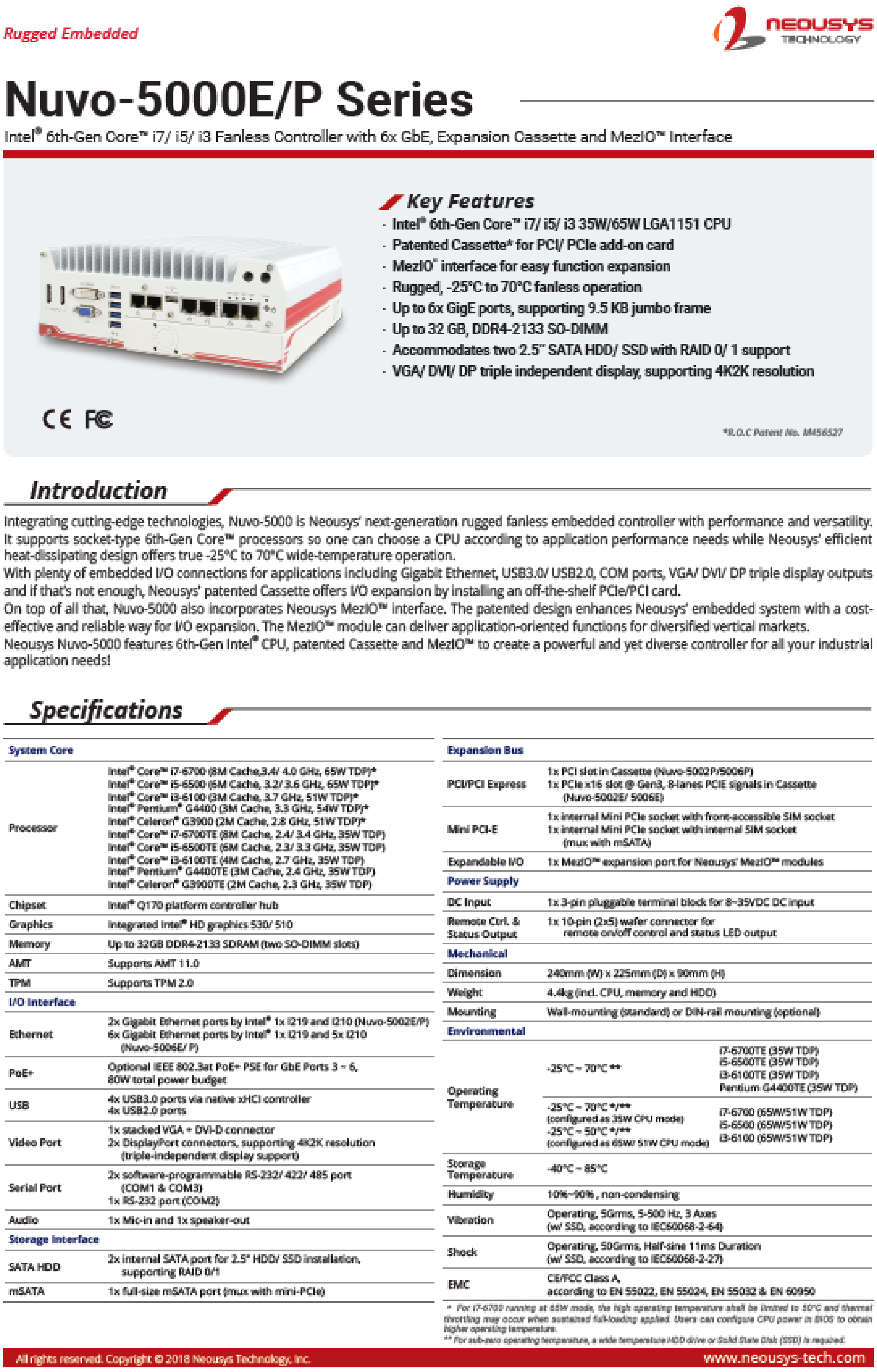
CI-813

CI-701A

INFAIMON

ESPAÑA		MEXICO		BRASIL	PORTUGAL
· Barcelona	· Valencia	· Aguascalientes	· Puebla	· Campinas	· Aveiro
· Bilbao	· Valladolid	· Ciudad de México	· Querétaro	· Curitiba	
· Madrid	· Vigo	· Guadalajara		· São Bernardo do campo	
· Sevilla	· Zaragoza	· Monterrey			

www.infaimon.com



Rugged Embedded

Nuvo-5000E/P Series

Intel® 6th-Gen Core™ i7/ i5/ i3 Fanless Controller with 6x GbE, Expansion Cassette and MezIO™ Interface

Key Features

- Intel® 6th-Gen Core™ i7/ i5/ i3 35W/65W LGA1151 CPU
- Patented Cassette* for PCI/ PCIe add-on card
- MezIO™ interface for easy function expansion
- Rugged, -25°C to 70°C fanless operation
- Up to 6x GigE ports, supporting 9.5 KB jumbo frame
- Up to 32 GB, DDR4-2133 SO-DIMM
- Accommodates two 2.5" SATA HDD/ SSD with RAID 0/ 1 support
- VGA/ DVI/ DP triple independent display, supporting 4K2K resolution

*R.O.C Patent No. M456527

Introduction

Integrating cutting-edge technologies, Nuvo-5000 is Neousys' next-generation rugged fanless embedded controller with performance and versatility. It supports socket-type 6th-Gen Core™ processors so one can choose a CPU according to application performance needs while Neousys' efficient heat-dissipating design offers true -25°C to 70°C wide-temperature operation.
With plenty of embedded I/O connections for applications including Gigabit Ethernet, USB3.0/ USB2.0, COM ports, VGA/ DVI/ DP triple display outputs and if that's not enough, Neousys' patented Cassette offers I/O expansion by installing an off-the-shelf PCIe/PCI card.
On top of all that, Nuvo-5000 also incorporates Neousys MezIO™ interface. The patented design enhances Neousys' embedded system with a cost-effective and reliable way for I/O expansion. The MezIO™ module can deliver application-oriented functions for diversified vertical markets.
Neousys Nuvo-5000 features 6th-Gen Intel® CPU, patented Cassette and MezIO™ to create a powerful and yet diverse controller for all your industrial application needs!

Specifications

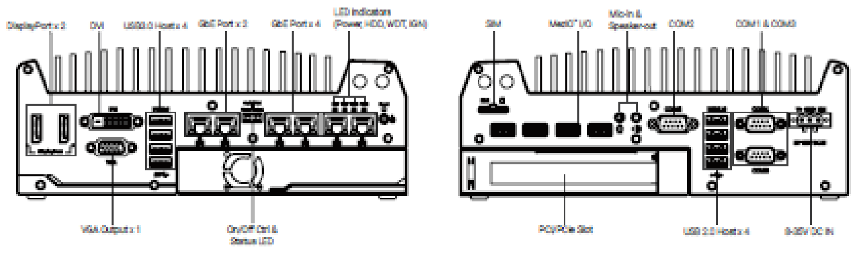
System Core	
Processor	Intel® Core™ i7-6700 (8M Cache,3.4/ 4.0 GHz, 65W TDP)* Intel® Core™ i5-6500 (6M Cache, 3.2/ 3.6 GHz, 65W TDP)* Intel® Core™ i3-6100 (3M Cache, 3.7 GHz, 51W TDP)* Intel® Pentium® G4400 (3M Cache, 3.3 GHz, 54W TDP)* Intel® Celeron® G3900 (2M Cache, 2.8 GHz, 51W TDP)* Intel® Core™ i7-6700TE (8M Cache, 2.4/ 3.4 GHz, 35W TDP) Intel® Core™ i5-6500TE (6M Cache, 2.3/ 3.3 GHz, 35W TDP) Intel® Core™ i3-6100TE (4M Cache, 2.7 GHz, 35W TDP) Intel® Pentium® G4400TE (3M Cache, 2.4 GHz, 35W TDP) Intel® Celeron® G3900TE (2M Cache, 2.3 GHz, 35W TDP)
Chipset	Intel® Q170 platform controller hub
Graphics	Integrated Intel® HD graphics 530/ 510
Memory	Up to 32GB DDR4-2133 SDRAM (two SO-DIMM slots)
AMT	Supports AMT 11.0
TPM	Supports TPM 2.0
I/O Interface	
Ethernet	2x Gigabit Ethernet ports by Intel® 1x I219 and I210 (Nuvo-5002E/P) 6x Gigabit Ethernet ports by Intel® 1x I219 and 5x I210 (Nuvo-5006E/ P)
PoE+	Optional IEEE 802.3at PoE+ PSE for GbE Ports 3 ~ 6, 80W total power budget
USB	4x USB3.0 ports via native xHCI controller 4x USB2.0 ports
Video Port	1x stacked VGA + DVI-D connector 2x DisplayPort connectors, supporting 4K2K resolution (triple-independent display support)
Serial Port	2x software-programmable RS-232/ 422/ 485 port (COM1 & COM3) 1x RS-232 port (COM2)
Audio	1x Mic-in and 1x speaker-out
Storage Interface	
SATA HDD	2x internal SATA port for 2.5" HDD/ SSD installation, supporting RAID 0/1
mSATA	1x full-size mSATA port (mux with mini-PCIe)

Expansion Bus		
PCI/PCI Express	1x PCI slot in Cassette (Nuvo-5002P/5006P) 1x PCIe x16 slot @ Gen3, 8-lanes PCIE signals in Cassette (Nuvo-5002E/ 5006E)	
Mini PCI-E	1x internal Mini PCIe socket with front-accessible SIM socket 1x internal Mini PCIe socket with internal SIM socket (mux with mSATA)	
Expandable I/O	1x MezIO™ expansion port for Neousys' MezIO™ modules	
Power Supply		
DC Input	1x 3-pin pluggable terminal block for 8~35VDC DC input	
Remote Ctrl. & Status Output	1x 10-pin (2x5) wafer connector for remote on/off control and status LED output	
Mechanical		
Dimension	240mm (W) x 225mm (D) x 90mm (H)	
Weight	4.4kg (incl. CPU, memory and HDD)	
Mounting	Wall-mounting (standard) or DIN-rail mounting (optional)	
Environmental		
Operating Temperature	-25°C ~ 70°C **	i7-6700TE (35W TDP) i5-6500TE (35W TDP) i3-6100TE (35W TDP) Pentium G4400TE (35W TDP)
	-25°C ~ 70°C */** (configured as 35W CPU mode) -25°C ~ 50°C */** (configured as 65W/ 51W CPU mode)	i7-6700 (65W/51W TDP) i5-6500 (65W/51W TDP) i3-6100 (65W/51W TDP)
Storage Temperature	-40°C ~ 85°C	
Humidity	10%~90% , non-condensing	
Vibration	Operating, 5Grms, 5-500 Hz, 3 Axes (w/ SSD, according to IEC60068-2-64)	
Shock	Operating, 50Grms, Half-sine 11ms Duration (w/ SSD, according to IEC60068-2-27)	
EMC	CE/FCC Class A, according to EN 55022, EN 55024, EN 55032 & EN 60950	

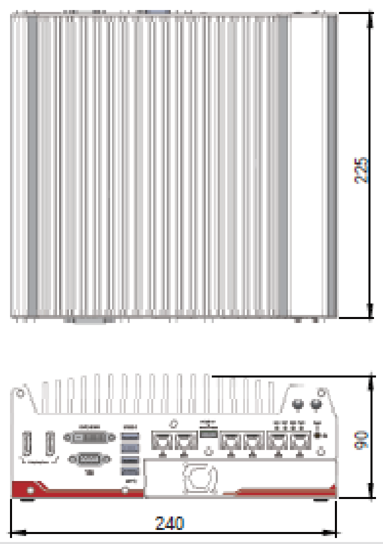
* *For i7-6700 running at 65W mode, the high operating temperature shall be limited to 50°C and thermal throttling may occur when sustained full-loading applied. Users can configure CPU power in BIOS to obtain higher operating temperature.*
** *For sub-zero operating temperature, a wide temperature HDD drive or Solid State Disk (SSD) is required.*

www.neousys-tech.com

Appearance



Dimensions



Ordering Information

Model No.	Product Description
Nuvo-5002E	Intel® 6th-Gen Core™ i fanless controller with 2x GbE, PCI Express Cassette and MezIO™
Nuvo-5002P	Intel® 6th-Gen Core™ i fanless controller with 2x GbE, PCI Cassette and MezIO™
Nuvo-5006E	Intel® 6th-Gen Core™ i fanless controller with 6x GbE, PCI Express Cassette and MezIO™
Nuvo-5006P	Intel® 6th-Gen Core™ i fanless controller with 6x GbE, PCI Cassette and MezIO™
Optional upgrade for GbE ports 3~6 to IEEE802.3at PoE+ ports	

Optional Accessories

DINRAIL-O	DIN-rail mounting assembly for Nuvo-5000 series
Fan-25	Fan assembly for 1-slot Cassette, 25x25x10 mm
PA-120W-OW	120W AC/DC power adapter 20V/6A; 18AWG/120cm; cord end terminals for terminal block, operating temperature : -30 to 70 °C

Cassette Modules	
CSM-PoE354	Cassette module with PCIe-PoE354 and pre-installed passive heat-spreader
CSM-USB380	Cassette module with PCIe-USB380 and pre-installed passive heat-spreader
CSM-NV750	Cassette module with NVIDIA® GTX 750 Ti graphics card, pre-installed heat-spreader and fan
CSM-R800	Cassette module accommodating four 2.5" HDD/ SSD (support RAID 0/ 1/ 10)

MezIO™ Modules	
MezIO™-C180	MezIO™ module with 4x RS-232/ 422/ 485 ports and 4x RS-232 ports
MezIO™-C181	MezIO™ module with 4x RS-232/ 422/ 485 ports and 4x RS-422/ 485 ports
MezIO™-D220	MezIO™ module with 8-CH isolated digital input and 8-CH isolated digital output
MezIO™-D230	MezIO™ module with 16-CH isolated digital input and 16-CH isolated digital output
MezIO™-V20-EP	MezIO™ module with ignition power control function for in-vehicle application
MezIO™-U4	MezIO™ module with 4x USB3.0 ports
MezIO™-G4	MezIO™ module with 4x GigE ports
MezIO™-G4P	MezIO™ module with 4x IEEE 802.3at PoE ports

F – Fravizel machines installed on quarry