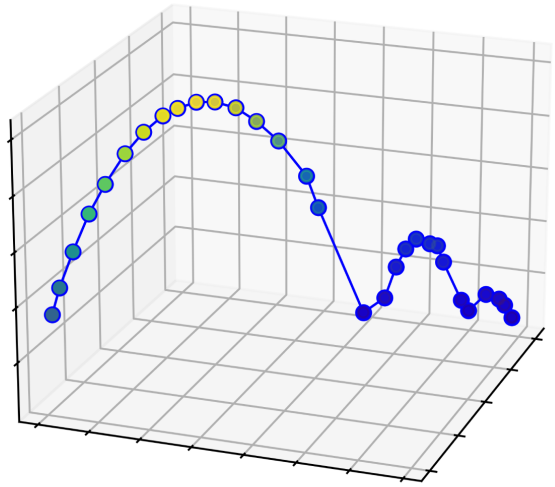
CHALMERS



Design, construction and testing of a portable inexpensive indoor positioning system

A study about designing, constructing and testing an inexpensive portable indoor positioning system using low-cost cameras operating in the infrared spectrum

Bachelor's thesis in Electrical Engineering

Tintin Axelsson, Johan Ewaldsson, Gustaf Ivarsson,
Edvin Nilsson, Christian Ristevski

DEPARTMENT OF Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Bachelor's thesis 2024

Design, construction and testing of a portable inexpensive indoor positioning system

A study about designing, constructing and testing an inexpensive portable indoor positioning system using low-cost cameras operating in the infrared spectrum

Tintin Axelsson, Johan Ewaldsson, Gustaf Ivarsson, Edvin Nilsson, Christian Ristevski

CHALMERS

Department of Electrical Engineering
Division of Systems and Control
EENX16-2024-12
Chalmers University of Technology
Gothenburg, Sweden 2024

A study about designing, constructing and testing an inexpensive portable indoor positioning system using low-cost cameras operating in the infrared spectrum
Axelsson, Ewaldsson, Ivarsson, Nilsson, Ristevski

Supervisor: Nikolce Murgovski, Department of Electrical Engineering
Examiner: Jonas Fredriksson, Department of Electrical Engineering

Bachelor's Thesis 2024
Department of Electrical Engineering
Division of Systems and Control
EENX16-2024-12
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Visualization of tracked trajectory and bounce of a thrown marker in addition to one of the camera modules developed..

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Design, construction and testing of a portable inexpensive indoor positioning system

A study about designing, constructing and testing an inexpensive portable indoor positioning system using low-cost cameras operating in the infrared spectrum

Tintin Axelsson, Johan Ewaldsson, Gustaf Ivarsson, Edvin Nilsson,
Christian Ristevski

Department of Electrical Engineering
Chalmers University of Technology

Abstract

A portable inexpensive optical indoor positioning system using four cameras was developed, constructed, and tested for a total cost of 4421.6 SEK. The system was made portable by the use of self-calibration routines, small lightweight cameras that are easy to set up and do not require any modification to the room in addition to using small and lightweight passive retro-reflective markers. The maximum trackable volume was determined to be 3x3x2 m with a relative RMS error to the origin of 2.51 cm. The system can track multiple objects simultaneously and can handle occlusion as long as the object is within the field of view of at least two cameras. Hardware developed during the project supports an update frequency of 92 Hz which is limited by performance issues of the software, limiting the system update frequency to 20.9 Hz.

Keywords: Indoor, Camera, Positioning system, camera position estimation, camera pose estimation, camera calibration, portable, inexpensive, computer vision, infrared.

Acknowledgements

First and foremost we would like to thank our supervisor Nikolce Murgovski for his guidance and feedback during the project. Special gratitude goes out to Michal Dzialak who helped look over the final design of the circuit board and pointed out potential flaws.

Tintin Axelsson, Johan Ewaldsson, Gustaf Ivarsson, Edvin Nilsson,
Christian Ristevski

Gothenburg, May 2024

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

CAD	Computer Aided Design
DC	Direct Current
GNSS	Global Navigation Satellite Systems
IC	Integrated Circuit
IPS	Indoor Positioning System
IR	Infrared
LED	Light Emitting Diode
MCU	Microcontroller Unit
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
PETG	Polyethylene Terephthalate Glycol
PWM	Pulse Width Modulation
PCB	Printed Circuit Board
RMS	Root Mean Square
SLAM	Simultaneous Localization And Mapping
USB	Universal Serial Bus
UWB	Ultra Widedband
WiFi	Wireless Fidelity

Nomenclature

Below is the nomenclature of parameters and variables that have been used throughout this thesis.

Parameters and variables

$\mathbf{x_c}$	position in the cameras coordinate system
$\tilde{\mathbf{x}_\mathbf{c}}$	position in the cameras homogeneous coordinate system
x_c	X coordinate in the cameras coordinate system
y_c	Y coordinate in the cameras coordinate system
z_c	Z coordinate in the cameras coordinate system
$\mathbf{x_w}$	coordinate in world coordinate system
$\tilde{\mathbf{x}_\mathbf{w}}$	coordinate in world homogeneous coordinate system
x_w	X coordinate in the world coordinate system
y_w	Y coordinate in the world coordinate system
z_w	Z coordinate in the world coordinate system
m_x	Pixel density in x dimension
m_y	Pixel density in y dimension
f	Focal length
o_x	Pixel offset in x direction
o_y	Pixel offset in y direction
u_i	Pixel coordinates
v_i	Pixel coordinates
R	Rotation matrix
$\boldsymbol{t}$	Translation vector
P	Projection matrix
E	Essential matrix

F	Fundamental matrix
M_{int}	Intrinsic calibration matrix
M_{ext}	Extrinsic calibration matrix
M	Transformation matrix between two cameras
$\boldsymbol{c_w}$	Cameras position in world coordinates
$\tilde{\mathbf{x}_\mathbf{w}}$	Estimated coordinates of tracked object in world coordinate system
$\tilde{\mathbf{x}_\mathbf{c}}$	Estimated coordinates of tracked object in a cameras coordinate system

Contents

List of Figures

List of Tables

1
Introduction

The need for cheap and accurate indoor positioning has increased in recent years as the field of robotics and life sciences has grown rapidly. Commercial systems for indoor localization and tracking of objects, robotics, and humans in 3D space are notoriously expensive and often require fixed installations. These systems are therefore often unavailable to low budget and/or portable projects that could have otherwise benefited from them. Cheaper solutions such as global navigation satellite systems, abbreviated GNSS, do not work reliably indoors because of signal attenuation and multipathing issues. In [1], Global Position System was used together with a gyroscope to determine the position of a smartphone, which had an average error of 1.35 m. This raises an interest in developing a cheap and portable positioning system that can be used, for example, in low-volume commercial robotics, or in student and research projects at universities.

An indoor positioning system can be constructed using several different methods. A common method uses preexisting Wi-Fi infrastructure, where the signal strength between the target and Wi-Fi access points can be used to triangulate the position. This method does not require a free line of sight to the tracked target and can often be implemented with existing infrastructure which, may reduce the cost [2]. Compared to the other methods, the precision is low, often no smaller than 1 m [2]. It is therefore not a relevant method for this project which seeks higher precision.

Another method of tracking is to use an optical positioning system using multiple cameras to track markers attached to objects. Systems using this approach can often track multiple objects at once and with great accuracy, with some achieving sub millimeter precision [3]. While the accuracy is high, this type of system is often difficult to reconfigure and expensive. The method is widely used in consumer products.

Another concept considered is to use ultrasonic sound. By having several receiver nodes and a transmitter, the distance to each receiver can be calculated by measuring the time difference, which can then itself be used to calculate the position using trilateration [4]. This system is greatly influenced by the noise and temperature in the environment and could therefore be unreliable in different conditions. The precision achieved with this method can vary from ± ten centimeters [5] to sub centimeter [6].

Ultra Wideband (UWB) is the use of high bandwidth radio waves for a short-range transfer of information. It can be used for indoor positioning by measuring the time of arrival of multiple signal pulses from different transmitters [7]. UWB functions

similarly to Wi-Fi and therefore has the same benefit of not having to rely on free lines of sight [7]. While the accuracy is often greater than that of Wi-Fi, it is less common that sufficient infrastructure already exists making installation costs higher. With the current technology the accuracy is still less than what is achievable with optical [3] and ultrasonic systems [2].

Prior to the study, a comparison between these approaches was performed. The optical system was determined to be the most suitable technology and the focus of this bachelor's thesis because of its ability to maintain high precision while operating at a high refresh rate. Furthermore, it could do so while staying within the project budget if a lower number of inexpensive consumer grade cameras are used.

1.1 Purpose

The purpose of this report is to design, construct and evaluate a portable and inexpensive optical indoor positioning system using cameras. More specifically, the project aims to achieve the following specifications and goals.

1. The positioning system should be capable of measuring the position in 3D space of at least two moving objects simultaneously in $6 \times 6 \times 3$ m room.
2. The system should be accurate and have low relative errors of the tracked points with respect to the true room coordinate and with respect to other tracked points.
   (a) The relative root mean square (RMS) error e_1 between a measured position $\hat{p}$ and it´s true position p can be calculated as

   $$e_1 = \sqrt{\frac{1}{N} \sum \left((\hat{p_x} - p_x)^2 + (\hat{p_y} - p_y)^2 + (\hat{p_x} - p_z)^2 \right)} \tag{1.1}$$

   for N number of measured points were the indices x, y and z denote the different axes in Cartesian coordinates. This error represents the absolute error in the position of the point with respect to the world reference frame. *The RMS error should be lower than 10 cm.*
   (b) The relative RMS error e_2 of the distance between two measured points $\hat{p_i}$ and $\hat{p_j}$, and the true real world distance d can be calculated as

   $$e_2 = \sqrt{\frac{1}{N} \sum \left(\sqrt{(\hat{p_{ix}} - \hat{p_{jx}})^2 + (\hat{p_{iy}} - \hat{p_{jy}})^2 + (\hat{p_{ix}} - \hat{p_{jz}})^2} - d \right)^2} \tag{1.2}$$

   for N numbers of measurement were the indices x, y and z denote the different exes in Cartesian coordinates. This error is between two points, both measured by the system. If the system has an offset to all positions, this will not shown up with this test. *The relative RMS error of any two points should be lower than 5 cm.*
3. In line with the purpose of the project, the cost of the system must lie within the strict budget cap of 5000 SEK. This cost constraint applies to the total system cost including all components.

4. For the system to be portable, multiple sub-requirements must be fulfilled

   (a) It must be easy to move the system into different rooms and locations by not requiring permanent changes to the structure or layout of the room.
   (b) It must not require outside measurements during setup or calibration and be able to function in different configurations, hence being effortless to set up and calibrate.
   (c) The markers should be attachable to multiple kinds of objects and should therefor be smaller then a sphere with a diameter of five centimeters and weigh less then 10 grams.
5. Another crucial aspect is the speed of the system. The update rate must be high enough to track multiple fast-moving objects accurately. Some low-end commercial systems have an update rate of 100 Hz [8], however this project aims to be more affordable and can therefore not be expected to have the same requirements. If the update rate is too low, the system could have trouble keeping track of moving objects and their position will be uncertain for longer intervals. An update rate of 30 Hz is determined to be sufficient for this project.

1.2 Limitations

To retain feasibility of the project and make sure it can be accomplished withing the given time frame, a few limitations are enforced.

- The tracking volume that the system is set up to track is expected to be empty of objects other than those which are meant to be tracked. The system will be designed to handle self-occlusion but other forms of occlusion as imposed by objects left in the tracked volume are outside the scope of this project.
- The ambient light interference that the system experiences is to be reduced for the test scenarios. In the case of the project, reducing the ambient light entails choosing an environment with controllable light levels through, for example, windows blinds, adjustable lights and limited reflective surfaces.
- In addition to the limitations mentioned above, there are general budgetary limitations and time constraints to adhere to. With this in mind, the hardware is intended to cost less than 5000 SEK and the project is to be completed within 16 weeks.

2
Theory

This section contains the basic aspects which lay a foundation for developing the optical based positioning system. Further details and implementations are included throughout section 3.

2.1 Overview of the optical system

The two major topologies commonly used for indoor positioning systems are outside-in and inside-out tracking [9]. For outside-in systems, the tracking of the object is facilitated or assisted by external sensors or beacons. In inside-out systems, the object to be tracked is equipped with all the technology required to know its own position and in some cases the positions of other objects as well [9]. The bulk of commercial indoor tracking systems appear to be optically based outside-in systems, whereby multiple fixed cameras surround the tracking space and track fiducials or markers attached to the tracked object.

For an optical system using cameras, both passive and active markers can be used. Passive markers use a retroreflective material to reflect light from, for instance infrared (IR) light emitted near the camera lens, back into the sensor. Active markers use a light emitting diode (LED), to emit light directly instead. The passive markers that are tracked are generally retroreflective spheres in various sizes but can also be other shapes such as circles as long as they are retroreflective. The property of retroreflectivity implies that an incident ray of light is reflected back to the source of the emitted light [10]. A comparison to other basic types of reflectivity is illustrated in Figure 2.1.

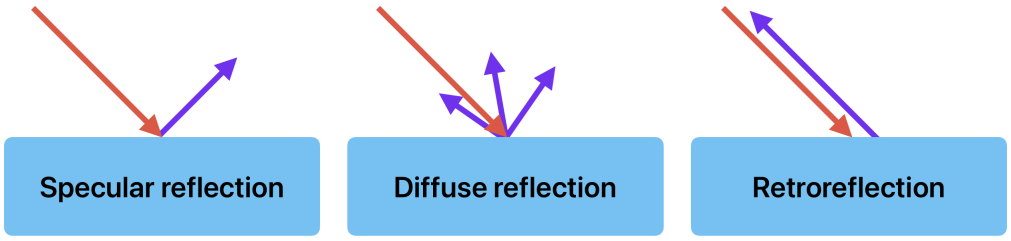


Figure 2.1: Basic types of reflectivity. The red and purple arrows represent incident and reflected light respectively.

These systems take this into advantage by placing strong infrared LEDs in close proximity to the camera lens. Once the light strikes the markers it is reflected back towards the camera appearing as a bright spot. The amount of light returned from the retroreflective markers greatly outweighs the light returned from normal diffuse and secular reflections. An image processing algorithms can find the coordinates of the bright spots that correspond to the markers. The general term for this is blob detection [11]. By then comparing the positions of the markers from the different cameras, a position in 3D space can be found.

These systems allow for tracking multiple objects simultaneously with high accuracy and reliability [12]. Commercial optical positioning systems can have a RMS error of 0.20 mm [3]. Since a large number of cameras in a fixed installation are used, these systems are known to be expensive and hard to reconfigure or move.

To overcome the issue of multiple cameras, inside-out positioning systems can be used in which the camera is placed on the object to be tracked. A simultaneous localization and mapping (SLAM) algorithm, can then be used to compute the position of the object [13]. The benefit of such a system is that it only requires a single camera. The drawbacks are that the SLAM algorithms are often computationally intensive and that the system can compute only the position of itself and possibly positions of objects in the cameras' field of view. Consumer devices such as virtual or augmented reality head mounted displays and controllers, are often tracked optically. Both using cameras and also using multiple photo diode sensors spread out over the tracked object.

When developing optical systems, the inverse square law can be used to determine the intensity of the light in the image plane of a camera. The law states that the intensity is inversely proportional to the square of the distance from its source [14] and can mathematically be described as

$$intensity \propto \frac{1}{distance^2} \tag{2.1}$$

Distortion is a common optical aberration in imaging systems, arising from imperfections in the optical assembly and geometric factors. There are two main types of distortion that can effect the captured image and cause it to not perfectly represent reality.

Radial distortion appears in two opposite variants and is caused by the unequal bending of light that occurs near the edge of the lens which results in an image where straight lines appear curved, either inward or outward [15]. Tangential distortion occurs due to misalignment between the optical center of the lens and image sensor and manifests as stretching of the image [15][16]. Figure 2.2 illustrates the difference before and after distortion correction.

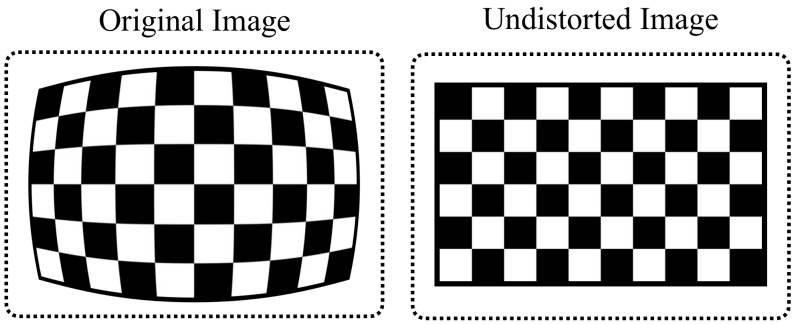


Figure 2.2: Illustration of camera distortion compensation on a checkerboard image.

Undistorting images before using them for geometric calculations or quantitative analysis is essential to ensure accuracy, reliability, and consistency. The process involves applying a transformation to negate the distortion caused by the lens and is implemented as a basic function in many computer vision libraries.

Bundle adjustment is a computer vision and photogrammetry optimization technique aimed at refining the parameters of a camera system and 3D reconstruction of a scene simultaneously. It is an important step to accurately be able to reconstruct a three-dimensional scene from multiple two-dimensional images [17].

The parameters to be optimized include the so called intrinsic camera parameters containing focal length and lens distortion information, along with the extrinsic parameters including both the position and rotation matrix of every camera used. Furthermore, the initially estimated 3D coordinates are refined as part of the same process [18].

To optimize these parameters, a cost function that represents the discrepancy between the observed image measurements and the predictions of the camera and scene model is established and optimized as a non-linear optimization problem [18]. Using bundle adjustment, systems can achieve a sub-pixel level of accuracy [19]. The discrepancy when reprojecting back into the image plane can be a result of multiple

factors.

Some noteworthy sources of error are image noise, an inadequate camera model and inaccuracies in the calibration target [20]. Noise can adversely affect the blob detection algorithm and therefore output a bad position estimate. The intrinsic camera model does not necessarily accurately describe the complex optics of the camera lens and sensor, causing bad trilateration. Lastly, the intrinsic calibration targets are unlikely to be perfect themselves and able to provide perfect calibration for the intrinsic matrix [20].

2.2 Software and libraries

The software of the project is based around Python to enable rapid development and broad library support. A library heavily utilized is the open source library OpenCV which focuses on computer vision and contains functions and algorithms relevant to the domain of image processing. It implements relevant functions for, among other things, optical-based positioning systems, including camera calibration routines, blob detection, and functions for camera pose estimation [21]. For other calculations, the Python library NumPy has extensive support for tasks related to linear algebra such as vector and matrix operations [22].

3

Development of an optical positioning system

In this chapter, a fully functional optical indoor positioning system is designed, constructed, tested and evaluated. The following covers multiple design aspects such as part selection, hardware design, hardware testing, software design and system evaluation.

The system developed during this project is akin to the optical system described in the theory section. The main differences is that the project is developed with low-cost hardware and utilizes a lower number of cameras. Except for this, the underlying principles are the same. The system uses cameras with infrared LED floodlights mounted around the lens to track retroreflective markers within the tracking volume. Regarding the hardware, custom camera modules with infrared LEDs and corresponding driver circuitry were constructed within the budget and time frame of the project. The first sections in this chapter are dedicated to this hardware construction. In total, four camera units were built and a simplified exploded view of the parts that compose a single unit is presented below in Figure 3.1.

All software written for the project that is not listed in this report is available at request from the authors.

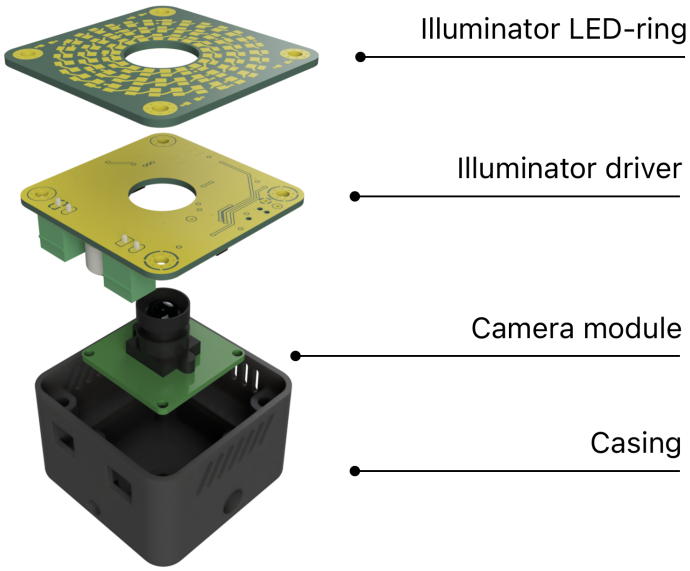


Figure 3.1: Exploded view of the four main components that constitute a camera module.

In addition to the hardware, software is written to find the markers within the images and transform the 2D image points into points within 3D space. This also requires multiple calibration routines to compensate for camera distortion and find the positions of the cameras relative to each other. A block diagram of the proposed system is shown in Figure 3.2

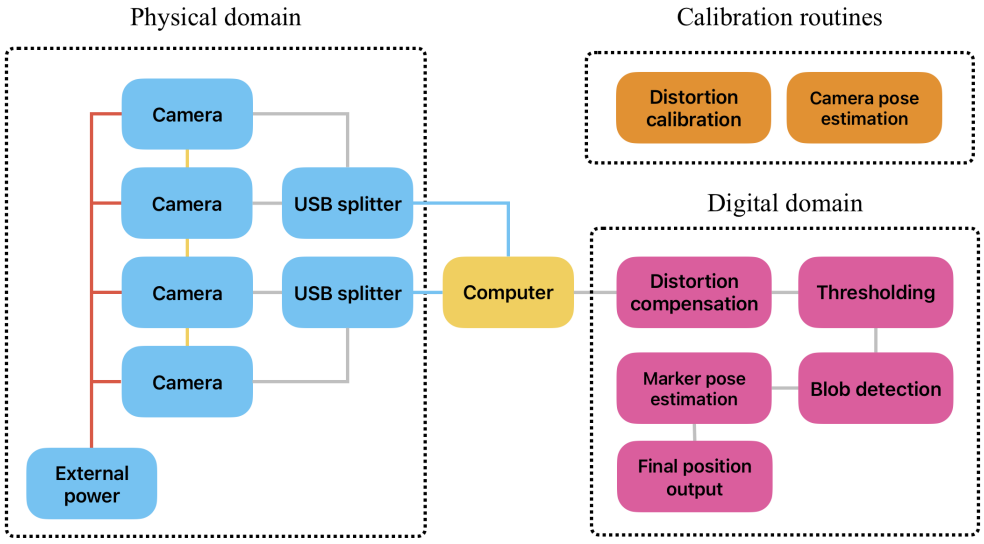


Figure 3.2: A simplified block diagram of the optical system, including physical and digital domain.

3.1 System layout and cameras

To keep the system portable and reduce cost, the system operates with a total of four cameras. The four cameras are placed in a square layout to minimize possible occlusion and to capture varied images of the target with respect to each other. The cameras are connected to the main computer via USB, utilizing splitters to reduce the number of cables as in Figure 3.3. Camera power and synchronization signals are also connected to the cameras in a similar way. The synchronization signal is used to synchronize the cameras and ensure all of them take a picture in the same instance of time. The importance of and the troubles encountered regarding synchronization will be discussed further in the report.

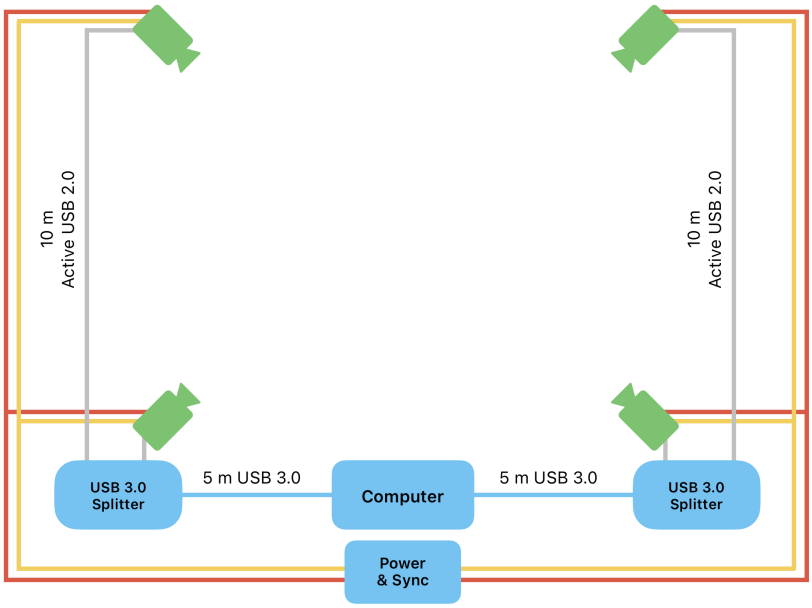


Figure 3.3: Proposed four camera setup and layout with power and synchornization signals connected.

The cameras used require a high refresh-rate to keep up with rapid moving objects and minimize motion blur for accurate tracking. The camera sensor does not need to be chromatic since the image will be converted to gray-scale in the image processing chain. A monochromatic sensor is also advantageous since the data sent over USB only contains luminance and no color data, thus reducing the required bandwidth. This leaves more bandwidth for transmitting images with lower compression which should reduce the workload on the image processing algorithms since the conversion step from color to gray-scale can be omitted.

Choosing a camera module with a global shutter, in which the entire image is captured at the same time, will likely increase tracking performance compared to a rolling shutter camera. The reasoning behind this being that rolling shutter causes different parts of the scene to be captured at different points in time. Moving targets can appear skewed and thus not truly represent the real environment. In addition

to this the cameras also benefit from being synchronized to each other in hardware. This requires cameras with pins broken out for hardware synchronization and cables to be routed between all cameras. Since the tracked markers are illuminated by infrared light, the cameras must be sensitive to this wavelength and not have a infrared blocking filter.

The cameras selected for the project are the Arducam B0332 USB2.0 UVC camera modules with the Ominivison OV9281 sensors. The cameras have a 1 megapixel monochrome sensor running at 120 Hz with a global shutter. They also have hardware input for external synchronization and, according to the manufacturer Arducam, the module also has a synchronization output [23]. Arducam also states that this signal is driven logic level high when the shutter is opened and the camera is capturing an image. During testing it was discovered that only the synchronization input is functional and that the synchronization output is defective, even tough it is advertised as a feature by the manufacturer. This is further elaborated on in the discussion section but is mentioned here since it has a profound impact on the circuit design described in the upcoming section.

3.2 Illuminator light construction

In addition to the cameras, the system also requires a light source to illuminate the markers as previously described. A bright light source will grant a high contrast between marker and background and simplify the initial image processing steps. It will also increase the range of the system as markers can be detected from a greater distance. To prevent the light source interfering with humans and other camera equipment, it is constructed with near-infrared LEDs. The near-infrared light emitted is almost invisible to the human eye [24] and greatly reduced by blocking filters for most regular cameras. Since the camera only captures an image while the shutter is open, the LEDs can be turned off during the majority of the time when the shutter is closed. The timings are illustrated in Figure 3.4, where t_{on} is the duration during which the shutter is open, and t_{delay} is the delay between the positive flank of the synchronization pulse and the opening of the camera shutter. This switching allows for a higher peak current to be driven through the LEDs during the brief exposure, while the average current remains below the manufacturers maximum rating. This in turn results in a higher peak brightness and greater efficiency compared to the case where the LEDs are constantly turned on.

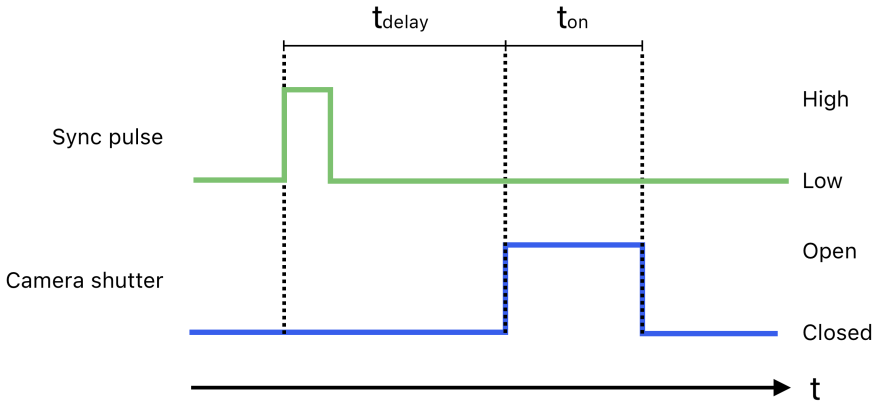


Figure 3.4: Timing diagram of the camera shutter in relation to a synchronization pulse.

Placing the LEDs as close to the camera lens as possible will allow the largest possible portion of the retroreflected light to fall in to the camera optics. To allow for this it was decided to design a printed circuit board (PCB) to mount the LEDs in a ring around the camera lens. To design the board, the open source electronics design automation software suite KiCad was used. JLCPCB was chosen as the PCB manufacturer because of their short turnaround time and low cost. The dimensions of the PCB were chosen as 50x50 mm since JLCPCB offered a discount for PCBs at that size during the time of designing. The LED selected for the project was the XYC-HIR76C-LX0 surface mounted infrared LED manufactured by Newopto and ordered from LCSC. This particular LED was selected on account of the low cost per unit and high availability at the time of order.

The switching circuitry for the LEDs that is further explained in the upcoming section sets a minimum voltage that can be driven to the LEDs of 10 V. According to the data sheet of the selected LED, the forward voltage is typically 1.55 V at a current of 50 mA and stated as 1.8 V as a general maximum [25]. Since the diodes are operating in a switched environment, the maximum forward voltage of 1.8 V was used as a guideline. To allow for the LED array to handle the minimum voltage of 10 V, the diodes are connected in a series configuration. Six diodes in series raises the maximum forward voltage to $1.8 \cdot 6 = 10.8$ V, which is sufficiently high to avoid damaging the LEDs when operating at minimum voltage. To further increase maximum brightness, the series groups are repeated in parallel to the greatest extent possible until there is no room left on the PCB. This yields a total of twelve parallel groups with six series diodes in each group, resulting in a total of 72 diodes per ring-light. Figure 3.5 shows a computer render of the final LED ring PCB. Full electrical schematics and layout for all PCBs are available in Appendix A.

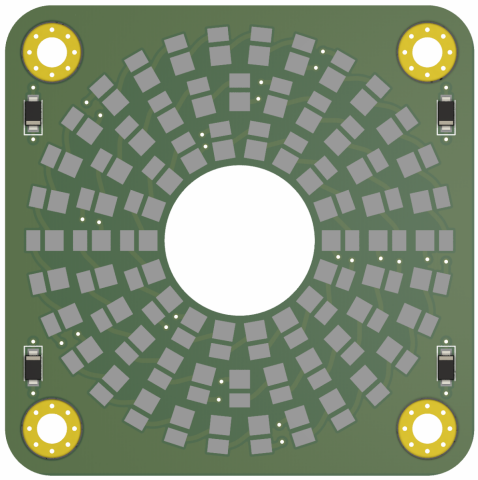


Figure 3.5: Computer render of the designed LED-ring PCB. The center of the PCB contains a hole for the camera lens to see through.

Counter-electromotive force may arise when driving the LEDs as an effect of the high power switching and stray trace inductance. This has the possibility to damage the LEDs if the reverse voltage induced exceeds the reverse breakdown voltage of the LEDs. To reduce this effect and protect the LEDs, there are four conventional diodes placed in reverse, parallel to the LED-array. These diodes clamp any reverse voltage before it becomes high enough to damage the LEDs and also serve as a protection for the LEDs in the event that power is accidentally supplied in reverse.

3.3 Illuminator driver construction

As previously described, the LEDs are switched on and off in synchronization with the opening and closing of the cameras shutter to maximize brightness and overall system efficiency. Initially the synchronization output pin from the camera was expected to be used as a reference signal to trigger a transistor and drive the infrared LEDs. However, since the synchronization output signal was confirmed to be defective, a more advanced driver board capable of flashing the lights at the correct time had to be designed. The general construction of the driver board can be divided into four functional blocks consisting of synchronization input, microcontroller unit (MCU), power switching and input power regulation as illustrated in Figure 3.6. It was decided to make this PCB separate from the illuminator light PCB to avoid the complications of fabrication and general complexity that arises from placing components on two sides of a single PCB.

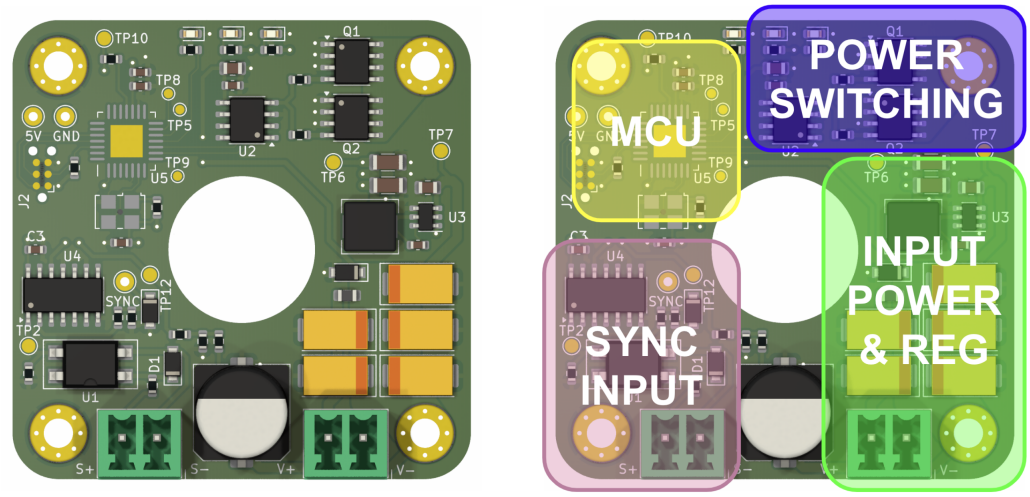


Figure 3.6: Render of the illuminator driver PCB. The left render shows the PCB with components and the right highlights what sections belong to the different functional blocks.

The power input and regulation circuit is responsible for providing a buffer for the high current bursts drawn by the LEDs at switching and to regulate the incoming power to an appropriate level for the rest of the circuitry. External power is necessary to drive the high power LEDs since the USB 2.0 protocol only specifies a maximum current of 500 mA at 5 V [26]. The power is supplied to the PCB through a standard 3.81 mm euro-block connector and any high frequency noise is filtered by a ferrite bead. A large 220 µF aluminum electrolytic capacitor is then connected in parallel with several smaller tantalum capacitors to act as a buffer and smooth out the current spikes caused by switching of the LEDs. The power block also employs a switching DC-DC converter to generate 5 V for the MCU and other logic. The regulator has pads broken out for supplying 5 V to the camera if the power supplied by the long USB cables were to deteriorate out of specification and appear insufficient to power the cameras. A green status LED is connected to the 5 V rail to indicate

the presence of voltage.

The synchronization block accepts a synchronization signal through a standard 3.81 mm euro-block connector. The goal of the circuit shown below in Figure 3.7, is to condition the synchronization signal and send it to the microcontroller as well as the camera at the correct voltage levels. To prevent electromagnetic interference that may be caused from the rapid and high current switching of the LEDs from interfering with the synchronization, the input circuitry is based around an optocoupler. This facilitates a low input impedance at the synchronization input and ensures that small, externally induced currents do not trigger the MCU. The synchronization signal also passes a 7400-series Schmitt trigger inverter that further improves the signal quality by introducing some hysteresis. An output of the Schmitt trigger is routed to the MCU for detection of the sync signal and to the cameras sync input via a voltage divider. A voltage divider is used since the camera sync input only tolerates a maximum voltage of 3.3 V, while the output from the Schmitt trigger is at 5 V. A blue status LED is also connected to the output of the Schmitt trigger to indicate the presence of a synchronization signal. The indication is however only useful at lower frequencies distinguishable by the human eye.

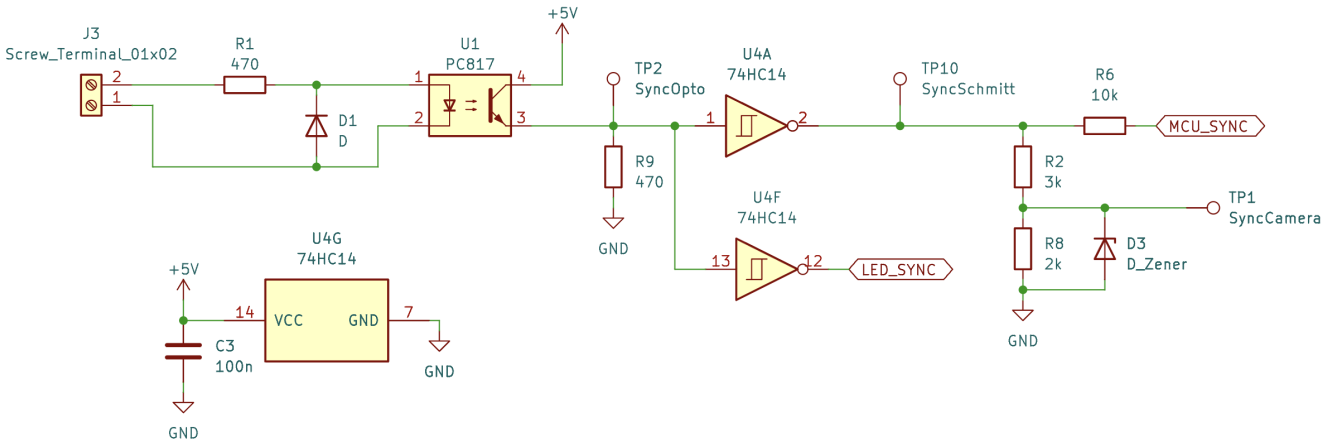


Figure 3.7: Schematic of synchronization input circuitry. The three symbols labeled 74HC14 all correspond to the same IC and are merely split up for clarity.

The power switching section consists of two low-side n-channel power metal-oxide-semiconductor field effect transistors (MOSFETs) and a gate driver connected to the MCU. Using low-side n-channel MOSFETs allows the gate voltage to remain below the drain voltage, in essence allowing for the MCU to directly switch the gates. However to improve on efficiency and reliability, a suitable gate driver is used. This allows the MOSFETs to operate at a higher gate-source voltage and therefore improved switching speeds and efficiency. A diagram of the proposed power switching circuitry is shown in Figure 3.8. The gate driver used is the IRS2181 from Infineon, and according to the datasheet it has a minimum supply voltage of 10 V and

a maximum of 20 V [27]. This is what constrains the previously explained minimum input voltage to 10 V. The drains of the MOSFETs are electrically connected to the cathodes of the LED-ring via the PCB mounting hole next to the MOSFETS. The anodes of the LEDs are in a similar fashion electrically connected to filtered input voltage through the mounting hole next to the power input.

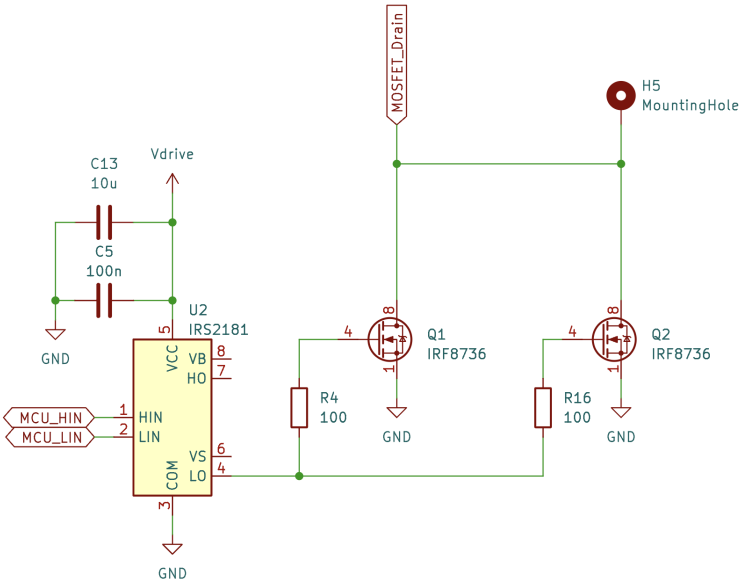


Figure 3.8: Schematic of the power switching circuit depicting the two power MOSFETS and gate driver IC. The power net Vdrive corresponds to the filtered power from the power input section.

The MCU is responsible for timing the switching of the LEDs in accordance with the opening and closing of the camera shutter, recall the timing diagram from Figure 3.4. The MCU used is the PIC16F76 from Microchip and was selected on the basis that it was available for free and that members of the group had prior experience with Microchip MCUs. The MCU accepts power from the 5 V regulator previously detailed and also accepts a clock signal at 16 MHz generated by an external oscillator. The sync signal is connected to an interrupt capable pin and the gate driver is connected to a general purpose pin. Further, a programming interface is broken out to a tag-connect pad for convenient flashing and debugging of the device. In operation the microcontroller uses interrupt routines to achieve the best performance. Once the correct transition of the camera sync signal is detected, an internal timer module is started. This timer is set to the precise time it takes the camera to detect the sync signal and start taking an image. When the timer overflows, it activates a new interrupt to turn on the gate driver and thus the MOSFETS which power the LED ring. At this time a second timer with a timeout corresponding to the time the camera shutter is open, is started. Once this second timer expires, the signal to the gate driver is set to low which turns of MOSFETS and LED-ring subsequently.

In addition to orchestrating the synchronized switching of the LEDs, the MCU also monitors the supplied voltage via another voltage divider. If the voltage is deemed

to be too high, as to possibly damage the LEDs, the MCU does not allow switching of the LEDs. This is indicated by illuminating a red status LED connected to one of the MCUs general purpose pins. All of the functionality outlined was programmed in C using Microchip's MPLAB X integrated development environment and the source code is available in Appendix B. Simplified pseudocode accomplishing the same functionality is outlined below.

```
1: while True do
2:     if Input voltage < Maximum input voltage then
3:         Enable external interrupts.
4:     else
5:         Disable external interrupts.
6:     end if
7: end while
8:
9: if External interrupt then
10:    if Sync interrupt then
11:        Start timer0 to wait for camera shutter opening.
12:
13:    else if timer0 interrupt then
14:        Turn on signal to gate driver.
15:        Start timer1 to wait for camera shutter closing.
16:
17:    else if timer1 interrupt then
18:        Turn off signal to gate driver.
19:    end if
20: end if
```

3.4 Camera and illuminator case

To contain the camera, illuminator LED board and illuminator driver board, a case was designed and built. It was designed to be as compact as possible and account for multiple different mounting configurations. The case was designed using the computer aided design (CAD) software Fusion 360 and 3D printed in polyethylene terephthalate glycol (PETG) plastic.

To account for multiple different mounting configurations, the case can be mounted to external holders with a metric M6 and imperial 1/4 inch thread. The 1/4 inch thread was selected in addition to M6 since it is the most common thread available on camera tripods and allows the system to be mounted to them. To avoid 3D printing the threads and thereby avoiding problems with wear after prolonged use, brass heat set threaded inserts are used. The camera, illuminator LED board and illuminator driver board are also secured to the case via smaller heat set inserts. Since the camera has a pigtail USB cable attached to it, a strain relief was also designed into the case. This strain relief prevents accidental tugging of the cable damaging the camera. The order of the component placements from back to front are USB cable strain relief, camera, illuminator driver board, illuminator LED board.

The case was designed such that the connectors for the illuminator driver board are easily accessible from the outside. A fiber laser was used to mark the plastic above the connector ports for the sync signal and power input to indicate their correspondence. The same fiber laser was also used to mark the type of thread used for the external M6 and 1/4 inch heat set inserts. A computer render of the designed case is shown in Figure 3.9. Slotted cooling louvers were also added to the case as a precaution if the driver board or LEDs would generate excessive heat when in operation.

Figure 3.9: Computer render of the case 3D model.

3.5 Measurements and implementations of shutter and flash timings

To program the MCU into activating the illuminator LEDs at the same time that the shutter is open, it is required to know the precise timings of the camera. Recalling timings in Figure 3.4, the camera receives a sync pulse and then takes some time to open the shutter. The shutter is open for some different period of time until the capture is complete and the camera closes the shutter.

First of all a test was setup to find the delay between the camera receiving a synchronization pulse and the camera opening the shutter. This was done by using two function generators and a regular LED. One of the function generators were used to generate a pulse with a certain pulse length. This generator was connected to the camera modules synchronization input as well as to the second function generators external trigger. The second function generator was set to trigger externally on the falling edge of signal generated by the first function generator. It was then set to generate a short pulse of 1 µs at a voltage large enough to drive the LED. A diagram of the signals generated by the two function generators are shown in Figure 3.10.

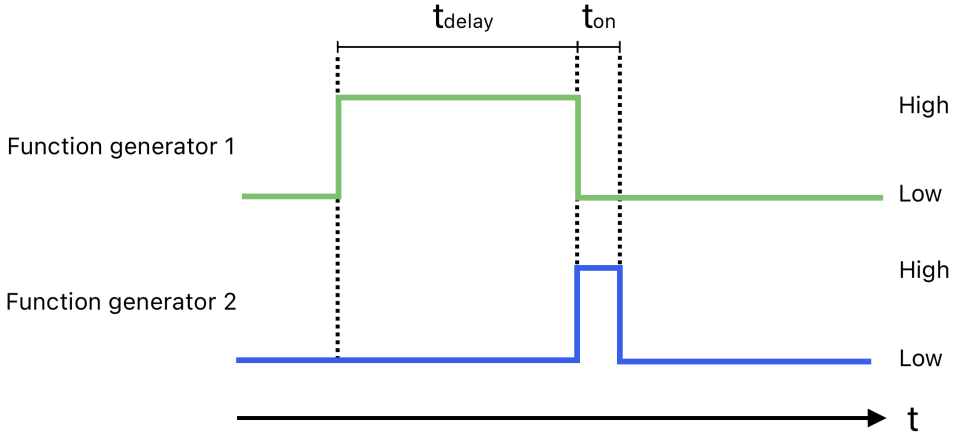


Figure 3.10: Diagram depicting the pulses generated by the two function generators were Function generator 2 is externally triggered by Function generator 1.

Since the camera synchronization input triggers on the rising edge of the signal generated by the first function generator. The pulse generated by the second function generator can be moved in time by varying the length of the pulse generated by the first function generator. By pointing the camera towards the LED connected to the second function generator and varying the pulse length of the first function generator, the LED will appear illuminated for a certain range of pulse lengths. The shortest possible pulse length where the LED is illuminated corresponds to the time the camera requires to interpret the sync pulse and open the shutter. In a similar fashion the longest pulse length when observing a lit LED subtracted with the time

required to open the shutter, corresponds to the time the shutter is open.

The test was done at two different exposure levels which were manually controlled within the software AMCap from Arducam. The two lowest exposure levels -13 & -12 were tested because higher values resulted in the cameras loosing synchronization lock at 90 Hz. Two Agilent 33220A function generators and an infrared LED were used for the test.

Once the timings for the camera shutter were determined, the proper timer values needed by the MCU to flash the LED-ring at the correct time had to be ascertained. This was accomplished experimentally by connecting an illuminator LED-ring to the driver board and varying the timer constants while taking measurements on the output driven to the LED-ring. The measurements were taken on a LeCroy wavepro 954 oscilloscope with channel one connected to the incoming synchronization pulse generated by an Agilent 33220A function generator and channel two connected to the illuminator driver synchronization output. Since the illuminator driver uses a low-side switching regime, the voltage across the LEDs can not be measured directly without damaging the oscilloscope. Therefore a LeCroy AP030 differential probe was used to measure the voltage over the LEDs. The timers within the MCU were then varied until the pulse of the LED-ring measured by the oscilloscope overlapped the previously determined time constants for the shutter opening and closing. Using the camera output synchronization from the illuminator driver board as a reference for time zero.

Finally, the acceptable maximum shutoff voltage had to be determine. To do this the current passing though the LEDs had to be measured and compared with datasheet maximum ratings. According to the datasheet the maximum pulse current is less than 0.5 A for a duty cycle of less than 1%. Since there are twelve groups of series diodes connected in parallel, the maximum pulse current is $0.5 \cdot 12 = 6$ A. However, since the pulse time and duty cycle later were found to be slightly larger than the ones stated in the datasheet, a conservative target pulse current of 5 A was selected. To measure this pulse current, a Kyoritsu 8112 clamp adapter was connected to channel four of the oscilloscope. Voltage was then applied to the illuminator driver by an Agilent E3633A DC variable power supply. The supplied voltage was increased until the current approached the set target value and the MCU was subsequently programmed to inhibit switching at a higher supply voltage.

3.6 Image processing pipeline

The image processing aims to determine the pixel coordinates of the objects to track from the camera image. This section explains all the steps required to obtain accurate pixel coordinates, along with reducing the risk of false detections.

All image-processing software was written in Python, using the open-source library OpenCV as the primary computer vision framework [21]. Since the OpenCV Python library uses C++ bindings, lackluster performance caused by using Python should not be an issue.

Initially, unintended bright spots may appear in the image, such as reflection from sunlight, or other light sources. These spots could potentially be detected as a point, which leads to false object detection. To address the spots, a mask is captured of an image of the scene without any objects present. The mask is then subtracted from all subsequent images, effectively filtering out bright spots that were present when the mask was generated. This process results in compensated images where only newly introduced bright sports are passed through.

The next step in the image processing pipeline is to compensate for the inherent distortion caused by the cameras lenses. The undistortion is carried out using the OpenCV function *cv2.undistort* which uses the cameras image and the pre-calculated calibration parameters to generate an undistorted image. The points after this step may be small and hard to detect and in order to make them easier to detect, as well as increase the accuracy of the blob detection, Gaussian blur is added to the image. This is done with the OpenCV function *cv2.GaussianBlur.*

Next, the image is thresholded to set the pixels to either white or black. The pixel brightness is a unit-less value between 0 and 255 where 0 is completely black and 255 is completely white. If the brightness for a pixel is above the threshold, it is set to white and otherwise set to black. This process makes it possible to later detect the white spots and calculate its coordinates. If the threshold is set too low, noise present in the image can be detected as points, and if the threshold is set too high, real points could get missed. However, in most cases the points are large enough to be almost perfectly bright on the initial image, making it hard to set the threshold too high. Through testing, the optimal value has been set to 200.

The blob detection is done with the OpenCV function *cv2.findContours*, which gives a list of contours of the white spots in the image. The contour object given by the function contains the moments of the contour. By dividing the two moments, m_{01} and m_{10} with the third moment m_{00}, all given by the OpenCV function, the x and y coordinates of the centerpoint are obtained [28]. In this way, the blob detection can determine the center coordinates of a point on the image with sub-pixel accuracy. An overview of the complete image processing method is shown in Figure 3.11.

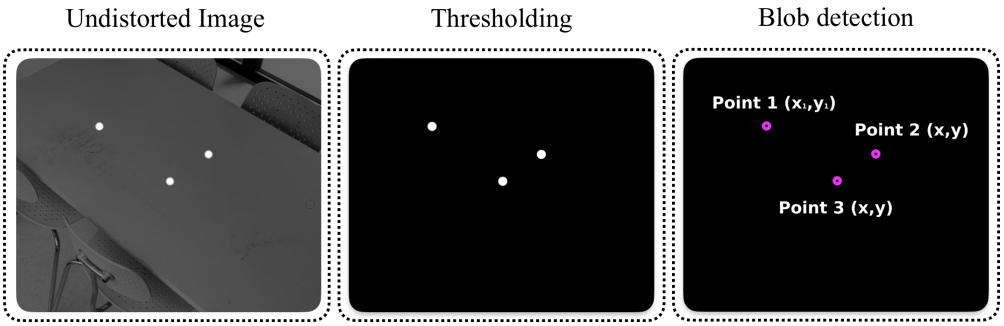


Figure 3.11: Illustration of thresholding and blob detection for a constellation of three markers.

The image points are now identified. The coordinates are returned as a list to the main routine, which uses the information to calculate the positions of the camera as well as the objects to track.

3.7 Camera calibration

This section aims to explain the intrinsic camera calibration routine, that is, the estimation of the internal parameters of the cameras. These parameters are used to understand the correspondence between the image pixel coordinates and the positions of the object in 3D space. To express these parameters in a precise and concise way, a camera model is needed. One such model is presented in the lecture series "Camera Calibration" , presented by Shree Nayar, T. C. [29] [30] [31]. The derivation from Nayar used a stereo setup, that is two cameras. However, the equations are later generalized for this specific project.

In [29] Nayar assumes a forward imaging model, see Figure 3.12, in which the world coordinate system **w** exists, with axes $\hat{x_w}, \hat{y_w}$ and $\hat{z_w}$. The cameras have internal coordinate systems C_l and C_r with axes $\hat{x_{cl}}, \hat{y_{cl}}$, $\hat{z_{cl}}$ and $\hat{x_{cr}}, \hat{y_{cr}}$, $\hat{z_{cr}}$ respectively, where the indices l and r represent the two cameras, l for left and r for right. The 2D image planes, parallel to the camera's coordinate system, have axes u_l, v_l and u_r, v_r, and have units of pixels instead of space coordinates. Assuming a point p is placed in the room with an unknown position that needs to be determined by estimation. The general visualization of the model in shown in Figure 3.12 below.

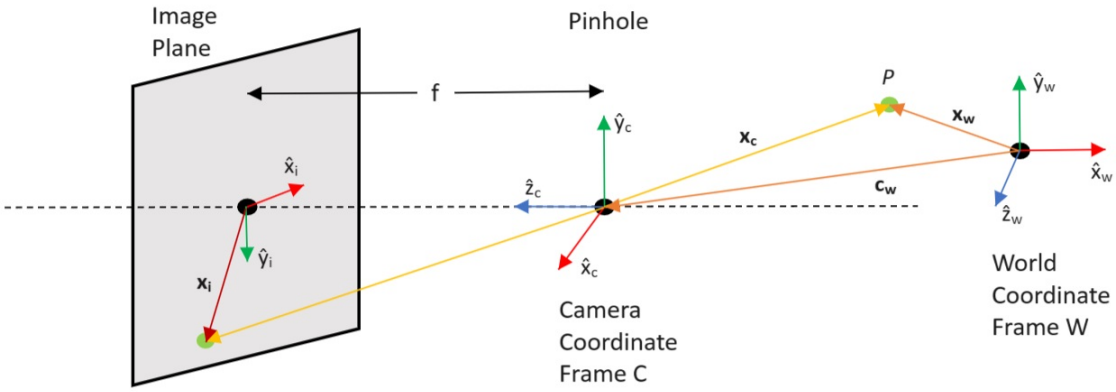


Figure 3.12: A general visualization of the forward imaging model for one camera, transforming world coordinates in 3D to pixel coordinates in 2D via the camera coordinate frame C.

Here, f is the focal length, the distance between the image plane and the point called the pinhole. The image plane is the 2D plane of the camera, defined in special coordinates. When the image is taken by the image sensor, the unit of coordinates is transformed from meters to pixels. The transformation from the camera's coordinate system consists of perspective projection from the camera to the image plane, and the scaling by a factor m_x and m_y for the axes $\hat{x}$ and $\hat{y}$, which is the pixel density in each direction. The translation is given by the constant translation terms o_x and o_y. The difference between the image plane and the image sensor is visualized in Figure 3.13.

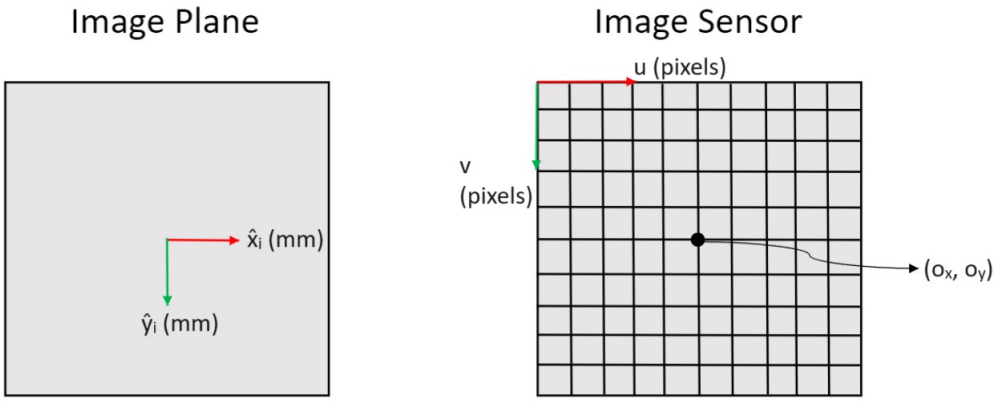


Figure 3.13: Visualization of the mapping from the image frame to the the image sensor.

The points in the camera coordinate system can be transformed into an image plane in pixels according to the equations below.

$$u_i = m_x f \frac{x_c}{z_c} + o_x = f_x \frac{x_c}{z_c} + o_x \tag{3.1}$$

$$v_i = m_y f \frac{y_c}{z_c} + o_y = f_y \frac{y_c}{z_c} + o_y \tag{3.2}$$

u_i and v_i are the pixel coordinates in the image plane, f is the focal length of the camera, m_x and m_y are the pixel density in pixels per mm and o_x and o_y are the pixel offset where the optical axis pierces the image sensor. Meanwhile, f_x and f_y are the product of focal length and pixel densities and can be interpreted as focal length in pixels. However, these equations are nonlinear which makes them unfeasible to use. By using homogeneous coordinates, the parameters can be transformed by adding a third coordinate into linear equations, such that a vector $\mathbf{u}\,(u, v)$ is represented as $\tilde{\mathbf{u}}\,(u, v, 1)$, as visualized in Figure 3.14

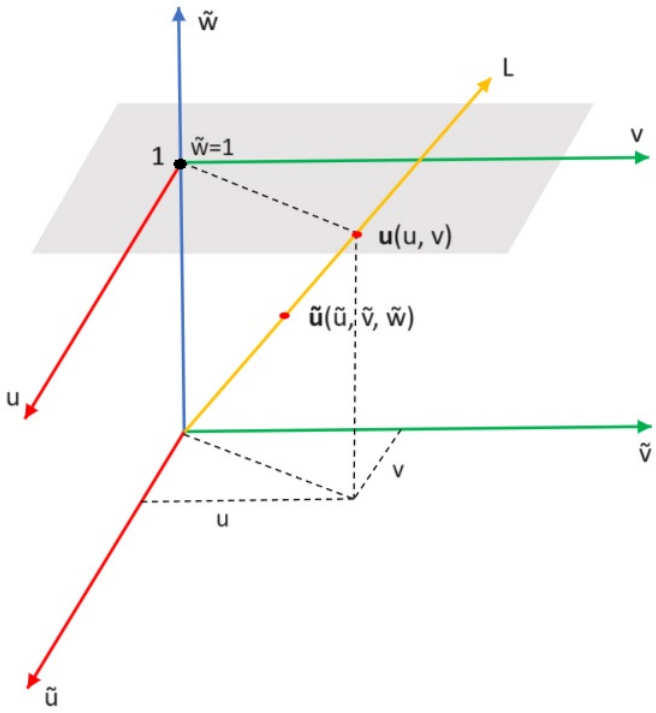


Figure 3.14: Visualization of homogeneous coordinates.

All points on the line L given by scaling the homogeneous vector are considered equivalent, given that the third coordinate $\tilde{w} \neq 0$ is fictitious such that

$$u = \frac{\tilde{u}}{\tilde{w}}$$

and

$$v = \frac{\tilde{v}}{\tilde{w}}$$

With the homogeneous vector representation, the equation can then be expressed as the linear expressions

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} \equiv \begin{bmatrix} z_c u \\ z_c v \\ z_c \end{bmatrix} = \begin{bmatrix} f_x x_c + z_c o_x \\ f_y y_c + z_c o_y \\ z_c \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{3.3}$$

where equations (3.1) and (3.2) have been inserted with the constant z_c acting as the fictitious coordinate. The left 3x4 matrix is the intrinsic calibration matrix, such that

$$\tilde{\mathbf{u}} = M_{int}\tilde{\mathbf{x}_\mathbf{c}} \tag{3.4}$$

The intrinsic parameters describe the cameras distortion properties. To calibrate these parameters, OpenCV is used. By taking multiple pictures of a ChArUco calibration board from multiple angles, the intrinsic matrix as well as distortion parameters can be solved for using the OpenCV function cv2.aruco.calibrateCameraCharuco. A ChArUco board is a calibration board consisting of ChArUco markers which can be detected using OpenCV. The size of the squares are known, which is used to calibrate the model of the lens distortion. A ChAruCo board is shown in Figure 3.15 below.

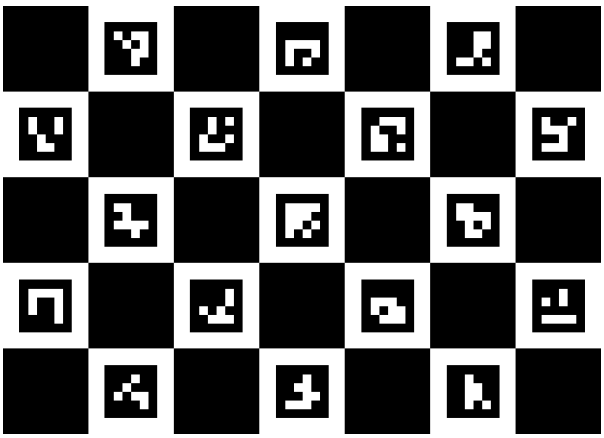


Figure 3.15: A ChArUco board used for calibrating the intrinsic parameters of the camera.

The calibration is done once, and all values are stored in a pickle file which is a python object in a serialized file format [32]. The parameters are then loaded when needed.

3.8 Camera pose estimation

The information given by the cameras is only useful if the position and rotation of the cameras themselves are known. This section aims to express a method for estimating the position of all cameras, without the need to measure the parameters manually in the physical world. Initially, the camera model defined in the above section is developed for expressing the rotation and transformation of the cameras, and then a method for estimating these parameters are given.

The position of the camera, $\boldsymbol{c_w}$, and their orientation are described by a rotation matrix R as defined as in Equation (3.5) and a translation vector $\boldsymbol{t}$ as defined as in Equation (3.6)[29]. These parameters together are referred to as the extrinsic camera parameters.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.5}$$

The rotation matrix describes the camera's rotation around three axes

$$\boldsymbol{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.6}$$

where t_x, t_y, t_z represent the translation from the system origin in the x, y, and z direction respectively. The origin is the zero-point of the coordinate system and are defined by the user during the calibration. The observed point can now be described using the camera coordinate frame C according to Equation (3.7) below.

$$\boldsymbol{x_c} = R\left(\boldsymbol{x_w} - \boldsymbol{c_w}\right) = R\boldsymbol{x_w} - R\boldsymbol{c_w} = R\boldsymbol{x_w} + \boldsymbol{t}, \tag{3.7}$$

The vector $\boldsymbol{x_c}$ is the point referenced in the camera coordinate system, $\boldsymbol{c_w}$ is the position of the camera in the world coordinate frame and $\boldsymbol{x_w}$ is the position of the point in the world coordinate frame. The vector $\boldsymbol{t}$ is the translation vector such that $\boldsymbol{t} = -R\boldsymbol{c_w}$. Rewriting the relation using homogeneous coordinates gives the following system of equations.

$$\tilde{\boldsymbol{x_c}} = \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{3.8}$$

This can be written as

$$\tilde{\boldsymbol{x_c}} = M_{ext}\tilde{\boldsymbol{x_w}} \tag{3.9}$$

Where M_{ext} is the extrinsic matrix that needs to be estimated. This method can be used to transform positions between any two reference systems.
Nayar [29] introduces the projection matrix P through Equation (3.4) and (3.9) put together below.

$$\tilde{\boldsymbol{u}} = M_{int} M_{ext} \tilde{\boldsymbol{x_w}} = P \tilde{\boldsymbol{x_w}} \tag{3.10}$$

P is the projection matrix and $\tilde{\boldsymbol{u}}$ is the pixel location in the image, such that

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{3.11}$$

To calculate the extrinsic parameters the OpenCV function *solvePnP* is used [21]. The function computes the translation and rotation vector between a camera and world coordinates based on the the world coordinates of at least three targets, the pixel coordinates of those targets and the intrinsic matrix of the camera. The rotation vector can then be transformed into the rotation matrix seen in Equation (3.5) using Rodrigues' formula [33]. The estimation is furthermore done with all cameras individually.

This calibration also establishes the coordinate system for the room as one point of the calibration target is used to define the origin. It is therefore important that the target remains in the same place when calibrating the cameras so that all cameras use the same coordinate system. An illustration of the calibration target used to estimate the extrinsic matrices for the cameras can be seen in Figure 3.16.

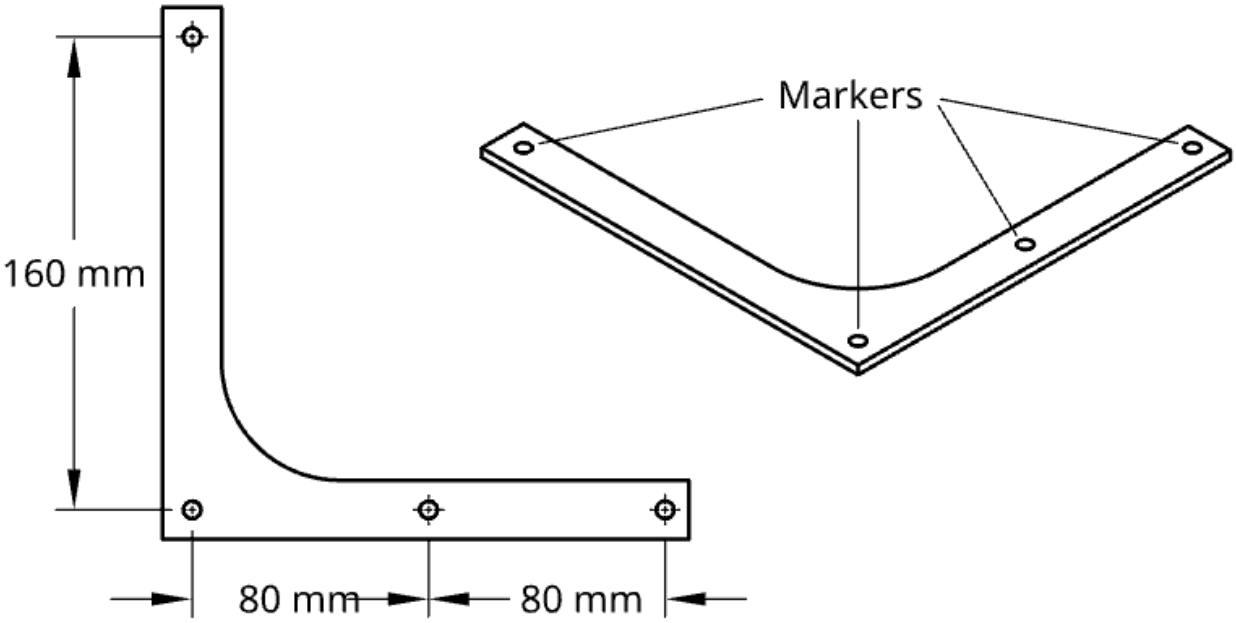


Figure 3.16: Calibration target used to calibrate the cameras extrinsic matrices.

3.9 Position estimation

This section aims to explain the principles of estimating the unknown positions of an object given image points on at least two cameras. One camera can determine the position in two coordinates, but to be able to determine the remaining coordinate of the object, a second camera is needed. As shown at the end of this section, two cameras provide four equations for the three unknown coordinated in 3D space, leading to an over-determined system of equations. By calculating the least squares solution to the system, the position of the object can be estimated. By using a larger amount of cameras, more equations are given, resulting in a better estimation. Nayar estimates the position using the intrinsic and extrinsic matrices from two cameras [30]. The intrinsic matrices from the two cameras give the following expressions.

$$\begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} = \begin{bmatrix} f_{x,l} & 0 & o_{x,l} & 0 \\ 0 & f_{y,l} & o_{y,l} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_l \\ y_l \\ z_l \\ 1 \end{bmatrix} \tag{3.12}$$

$$\begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = \begin{bmatrix} f_{x,r} & 0 & o_{x,r} & 0 \\ 0 & f_{y,r} & o_{y,r} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \\ 1 \end{bmatrix} \tag{3.13}$$

Both of which can be written as Equation (3.14) where Equation (3.15) shows $M_{int,i}$.

$$\tilde{\mathbf{u}_\mathbf{i}} = M_{int,i}\tilde{\mathbf{x}_\mathbf{i}} \tag{3.14}$$

$$M_{int,i} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \tag{3.15}$$

For Equation (3.15) above, i is the index, either l or r for the left and right camera respectively.

The stereo cameras are assumed to be calibrated such that a projection matrix from the right camera frame to the left can be calculated and used to transform positions from the right camera to the left camera. The process is described by the following equation.

$$\begin{bmatrix} x_l \\ y_l \\ z_l \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{43} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \\ 1 \end{bmatrix} \tag{3.16}$$

Which in short can be written as

$$\tilde{\boldsymbol{x_l}} = M\tilde{\boldsymbol{x_r}} \tag{3.17}$$

Note that M in Equation (3.17) is defined between two cameras, while the extrinsic matrix defined previously is the matrix from the world coordinate frame to the camera frame. The new matrix can be calculated from the extrinsic matrices and is derived in the following manner.

$$\tilde{\boldsymbol{x_l}} = M_{ext,l}\tilde{\boldsymbol{x_w}} \tag{3.18}$$

$$\tilde{\boldsymbol{x_r}} = M_{ext,r}\tilde{\boldsymbol{x_w}} \tag{3.19}$$

where $\tilde{\boldsymbol{x_l}}$ and $\tilde{\boldsymbol{x_r}}$ are the positions of the object in the left and right cameras coordinate system respectively, $M_{ext,l}$ and $M_{ext,r}$ is the extrinsic matrix of the left and right camera, and $\tilde{\boldsymbol{x_w}}$ is the position of the object to track in the world coordinate frame. By multiplying the inverse of the extrinsic matrix on the left side of the expressions, the following expressions is given.

$$M_{ext,l}^{-1}\tilde{\boldsymbol{x_l}} = \tilde{\boldsymbol{x_w}} \tag{3.20}$$

$$M_{ext,r}^{-1}\tilde{\boldsymbol{x_r}} = \tilde{\boldsymbol{x_w}} \tag{3.21}$$

as both equations are equal to $\tilde{\boldsymbol{x_w}}$, the left-hand sides of the equations can be set to equal each other as in Equation (3.22).

$$M_{ext,l}^{-1}\tilde{\boldsymbol{x_l}} = M_{ext,r}^{-1}\tilde{\boldsymbol{x_r}} \tag{3.22}$$

By then multiplying both sides with the extrinsic matrix of the left camera, the final expression is given as follows.

$$\tilde{\boldsymbol{x}}_{\boldsymbol{l}} = M_{ext,l} M_{ext,r}^{-1} \tilde{\boldsymbol{x}}_{\boldsymbol{r}} \tag{3.23}$$

$$M = M_{ext,l} M_{ext,r}^{-1} \tag{3.24}$$

Where M is the sought after camera to camera transformation matrix.
Now, substituting equation (3.16) into Equation (3.12) gives the left camera imaging equation.

$$\begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} = \begin{bmatrix} f_{x,l} & 0 & 0_{x,l} & 0 \\ 0 & f_{y,l} & o_{y,l}, 0 & \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{43} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \\ 1 \end{bmatrix} \tag{3.25}$$

Which in short can be written as

$$\tilde{\mathbf{u}}_{\mathbf{l}} = P_l \tilde{\mathbf{x}}_{\mathbf{r}} \tag{3.26}$$

where

$$P_l = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \tag{3.27}$$

By rewriting the terms in Equation (3.13) and Equation (3.25) the following system of equations are given

$$\begin{bmatrix} u_r m_{31} - m_{11} & u_r m_{32} - m_{12} & u_r m_{33} - m_{13} \\ v_r m_{31} - m_{21} & v_r m_{32} - m_{22} & v_r m_{33} - m_{23} \\ u_l p_{31} - p_{11} & u_l p_{32} - p_{12} & u_l m_{33} - p_{13} \\ v_l p_{31} - p_{21} & v_l p_{32} - p_{22} & v_l m_{33} - p_{23} \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = \begin{bmatrix} m_{14} - m_{34} \\ m_{24} - m_{34} \\ p_{14} - p_{34} u_l \\ p_{24} - p_{34} v_l \end{bmatrix} \tag{3.28}$$

Which in short can be written as

$$\mathrm{A}\tilde{\mathbf{x}}_{\mathbf{r}} = \mathbf{b} \tag{3.29}$$

The matrix A and vector $\mathbf{b}$ are known, implying that the position can be estimated by solving the system of equations. Since the system is overdetermined, and by assuming that the cameras are not perfectly calibrated, the position can not be calculated exactly but can be estimated by calculating the least square solution using pseudo-inverse as

$$\mathrm{A}\tilde{\mathbf{x}}_{\mathbf{r}} = \mathbf{b} \tag{3.30}$$

$$\mathrm{A}^T \mathrm{A}\tilde{\mathbf{x}}_{\mathbf{r}} = \mathrm{A}^T \mathbf{b} \tag{3.31}$$

$$\tilde{\mathbf{x}}_{\mathbf{r}} = (\mathrm{A}^T \mathrm{A})^{-1} \mathrm{A}^T \mathbf{b} \tag{3.32}$$

In contrast to the model presented by Nayar, our system used multiple cameras to estimate the position of the object. One camera, by the equations above called the right, hereafter called the first, is used as the reference. By using the intrinsic matrix M_1, the following expression is known

$$\begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_1 \\ 1 \end{bmatrix} \tag{3.33}$$

which can be rewritten as

$$\tilde{\mathbf{u_1}} = M_{int,1}\tilde{\mathbf{x_1}} \tag{3.34}$$

By rewriting the equations, the following underdetermined system of equations is given

$$\begin{bmatrix} u_r m_{31} - m_{11} & u_r m_{32} - m_{12} & u_r m_{33} - m_{13} \\ v_r m_{31} - m_{21} & v_r m_{32} - m_{22} & v_r m_{33} - m_{23} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} m_{14} - m_{34} \\ m_{24} - m_{34} \end{bmatrix} \tag{3.35}$$

Which in short can be written as

$$\mathrm{A}\tilde{\mathbf{x}_\mathbf{r}} = \mathbf{b} \tag{3.36}$$

All other cameras that have detected a point corresponding to the object are used by adding their equation to the system of equations. First, the projection matrix between the first and new camera is calculated as below. Here, i correspond to the i:th camera.

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_{x,i} & 0 & 0_{x,i} & 0 \\ 0 & f_{y,i} & o_{y,i} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{43} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} \tag{3.37}$$

Which in short can be written as

$$\tilde{\mathbf{u}_\mathbf{i}} = P_i\tilde{\mathbf{x_1}} \tag{3.38}$$

where

$$P_i = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \tag{3.39}$$

By rewriting the equations, the following rows are added to the system of equations

$$\begin{bmatrix} u_l p_{31} - p_{11} & u_l p_{32} - p_{12} & u_l m_{33} - p_{13} \\ v_l p_{31} - p_{21} & v_l p_{32} - p_{22} & v_l m_{33} - p_{23} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} p_{14} - p_{34} u_i \\ p_{24} - p_{34} v_i \end{bmatrix} \tag{3.40}$$

These equations are added to the A and b matrices, which increases the number of equations of the system. This equation are appended for all cameras, resulting in a system of equations with the size of two times the number of cameras. The least square method can once again be used, which gives an estimate of the position from an arbitrary number of cameras larger or equal to two. The estimated position is given in the reference camera's coordinate frame. To transform the position into world coordinated, homogeneous coordinates can be used with the extrinsic parameters as

$$\tilde{\boldsymbol{x_w}} = M_{ext}^{-1} \tilde{\boldsymbol{x_1}} \tag{3.41}$$

where $\tilde{\boldsymbol{x_w}}$ is the positions in the world coordinate frame, and $\tilde{\boldsymbol{x_1}}$ is the position in the reference cameras coordinate frame. $\tilde{\boldsymbol{x_w}}$ is the final estimate of the unknown position of the object.

3.10 Finding correspondences with epipolar geometry

Epipolar geometry can be used in order to determine the correspondence of points between different camera views as visualized in Figure 3.17 depicting the image planes of a left and a right camera, denoted by l and r respectively.

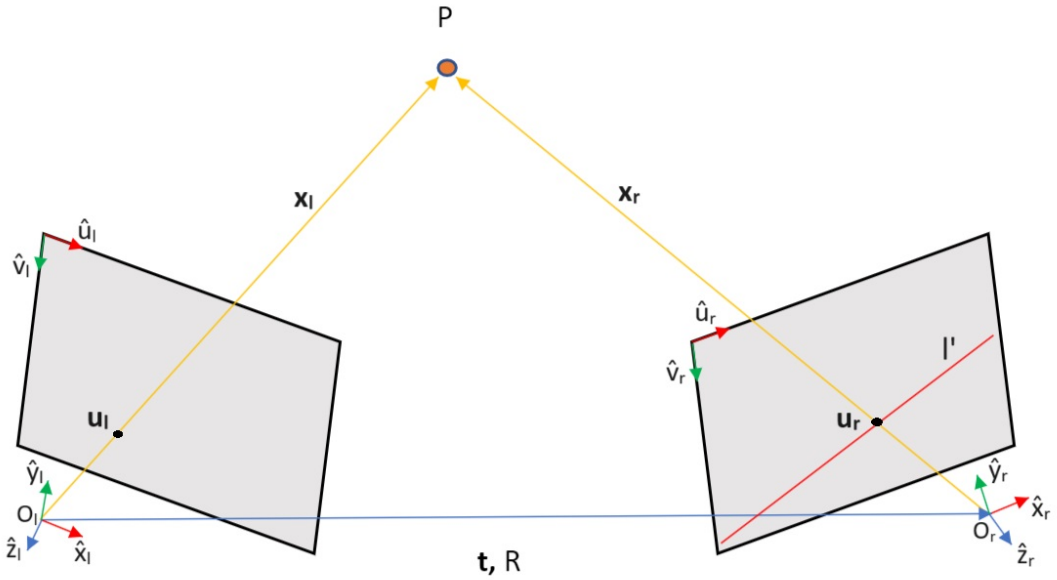


Figure 3.17: Epipolar geometry visualization for two cameras and a single point.

A point P in space will correspond to an image coordinate point, $\boldsymbol{u}_i$, in the image plane of one of the cameras. However, because of the 2D nature of a camera, all points in 3D space along the imaginary line O_lP will be projected onto the same point on the plane [34]. These points form a line l' on the image plane of the other cameras called the epipolar line corresponding to point $\boldsymbol{u}_i$ [31]. The corresponding point in an arbitrary different image frame j, $\boldsymbol{u}_j$, will lie on that frames $\boldsymbol{u}_i$ epipolar line [31].

The epipolar constraint can be used in order to determine if a point in a camera's image frame lies on the epipolar line. For Figure 3.17, where the sought image coordinate point is $\boldsymbol{u_r}$, it can be derived from Equation (3.42) [31]:

$$\boldsymbol{x}_l \cdot \boldsymbol{n} = \boldsymbol{x}_l \cdot (\boldsymbol{t} \times \boldsymbol{x}_l) = 0 \tag{3.42}$$

Where $\boldsymbol{n}$ is a normal to the plane spanned by the points P, O_l and O_r. $\boldsymbol{x_l}$ is the coordinates of point P in the left cameras coordinate frame. The dot product equals zero since $\boldsymbol{n}$ and $\boldsymbol{x_l}$ are perpendicular. The vector $\boldsymbol{t}$ is the translation vector for the position of the right camera in the left camera's coordinate frame. Equation (3.42)

can be rewritten in matrix-vector form as Equation (3.43) [31].

$$\begin{bmatrix} x_l & y_l & z_l \end{bmatrix} \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} = 0 \tag{3.43}$$

The vector $\boldsymbol{x}_l$ can be written in terms of $\boldsymbol{x}_r$ as in Equation (3.44)[31].

$$\begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} = R \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} + \boldsymbol{t} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.44}$$

R is the rotaion matrix of the left camera in image of the the right camera frame. Combining equations (3.43) and (3.44) result in Equation (3.45) [31].

$$\boldsymbol{x_l}^T E \boldsymbol{x_r} = \begin{bmatrix} x_l & y_l & z_l \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = 0 \tag{3.45}$$

The matrix E is called the essential matrix and relates the scene point with respect to the coordinate frames of the two cameras and is given by Equation (3.46) [31].

$$E = T_x R = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.46}$$

The matrices T_x and R correspond to the the translation and rotation of the two cameras. Equation (3.45) can be rewritten in terms of image coordinates by rewriting and substituting equations (3.12) and (3.13) into the equation, resulting in Equation (3.47) [31].

$$\begin{bmatrix} u_l & v_l & 1 \end{bmatrix} {K_l^{-1}}^T \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} K_r^{-1} \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = 0 \tag{3.47}$$

Where K_l and K_r are the camera matrices containing the intrinsic parameters of each camera [31].

$$K_l = \begin{bmatrix} f_{x,l} & 0 & o_{x,l} \\ 0 & f_{y,l} & o_{y,l} \\ 0 & 0 & 1 \end{bmatrix}$$

$$K_r = \begin{bmatrix} f_{x,r} & 0 & o_{x,r} \\ 0 & f_{y,r} & o_{y,r} \\ 0 & 0 & 1 \end{bmatrix}$$

The essential matrix multiplied with the camera matrix is called the fundamental matrix, F, containing both the intrinsic parameters and the relative positions of the two cameras [31].

$$F = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} = {K_l^{-1}}^T \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} K_r^{-1} \tag{3.48}$$

Which gives the short expression for Equation (3.47) as

$$\tilde{\boldsymbol{u}_l}^T F \tilde{\boldsymbol{u}_r} = 0 \tag{3.49}$$

However, in practice, this will not be equal to 0, although quite close. The value called epipolar value e_p is defined as

$$e_p = \tilde{\boldsymbol{u}_l}^T F \tilde{\boldsymbol{u}_r} \tag{3.50}$$

The closest point to the epipolar line in the image plane of one of the cameras formed by a known image point in another image plane can be found. It is done by looping through a list of image coordinate points, given by the different markers position in the image plane, and comparing the resulting epipolar values. The point with the lowest value is the closest to the epipolar line and is therefore most likely to correspond to the image point in the other camera, ultimately meaning that the cameras are viewing the same point in space.

As described above, the calculated value should ideally be equal to zero. However, due to imperfect calibration, camera poses estimation, intrinsic parameters, as well as non perfect distortion compensation, this value will be nonzero. For the system to be able to determine if two points on different cameras correspond to each other, a threshold value must be set. If the calculated value is less than the threshold, the points are considered to correspond to the same object. If the threshold is set too low, true points will be missed, that is a false negative. If the threshold is too high, points not corresponding to each other will be considered the same, that is false positives. To determine the threshold, a single marker was tracked with the four cameras, and the epipolar value for all points was calculated and plotted in a histogram to decide the most appropriate value.

When four cameras are used, the system should be able the detect and group together points in the images that correspond to the same object. Furthermore, the system should be able to handle occlusion, that is if one or two cameras are blocked and out of sight of the object to track, the system should still be able to estimate the position, as long as at least two cameras see the object. Still, if more than two cameras have the object in the line of site, the information from all cameras could be used to estimate the positions. This is done in an iterative process, where all points on the cameras are compared and evaluated systematically to group points in the images that correspond to the same object.

Initially, the points on the first camera are individually compared to all the points of the other cameras. The epipolar constraints are evaluated between the point on the first camera and all points on the next camera. If the best matching point has an epipolar constraint lower than the threshold, and if the point of the first camera is the best match to the point on the second camera, the points are considered to correspond to the same object. Thereafter, the same process is used on all other cameras with all points on the first camera. After that, the points on the second camera that have not yet been grouped are tested on the other cameras with the same process. This function groups together two, three, or four images point that correspond to the same object. These points are later used to estimate the position of the object.

4
System evaluation method

After development, the system is tested to evaluate how it performs in comparison to the pre-determined specifications and goals.

4.1 Maximum detectable marker range

To test the range the system is able to detect, a single camera was attached to a mount approximately 1.5 meters of the floor with the camera view perpendicular to the floor. A marker was then placed in front of the camera and moved further away from the camera until the system reached a point where the marker could not be detected.

4.2 Measurements of accuracy and performance

Several accuracy parameters have been tested to understand the system's potential and limitations. To evaluate the accuracy for these parameters, two main methods for testing have been used in accordance with the specifications in Section 1.1. Firstly, these two test methods previously outlined are explained. Later, test cases for evaluating the accuracy dependency on different parameters are presented.

The accuracy of the system is tested to determine if requirement 2 in Section 1.1 is met. The two sub-requirements 2.a and 2.b are evaluated separately. To evaluate 2.a, that is, if the RMS error of a measured point concerning the true position is less than 10 cm, a stationary marker will be placed at a specific point with known coordinates relative to the room. The marker will be placed on a tall pole with a known height, and the position will be accurately measured with a ruler. By measuring different positions in the room, the RMS error can be calculated as in Equation (1.1). An illustration of the proposed measuring pole is presented in Figure 4.1.

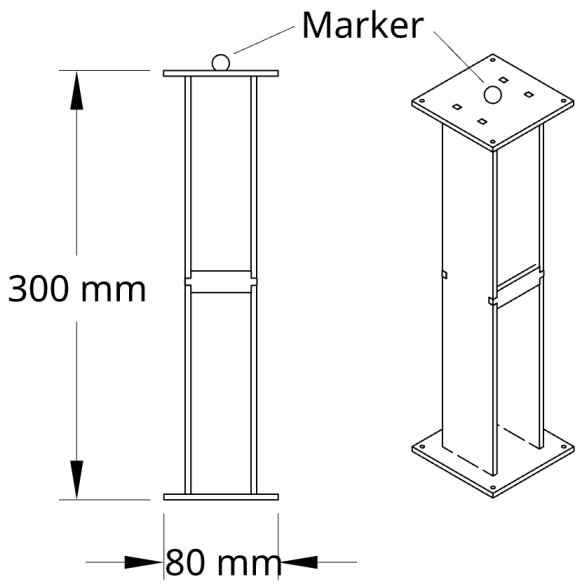


Figure 4.1: Test equipment for measuring the RMS error of a measured position relative to the true position.

A similar measuring pole was made with the height of 10 cm. Three different heights were used during the test, one with the 10 cm pole, one with the 30 cm pole, and a final one with two poles on top of each other, resulting in a height of 60 cm.

Secondly, to test requirement 2.b, the error of the distance between two measured positions is calculated. Two markers will be placed on a long rod 58.9 cm apart from each other as in Figure 4.2. By moving the rod to different positions in the room, the relative RMS error can be calculated as in Equation (1.2).

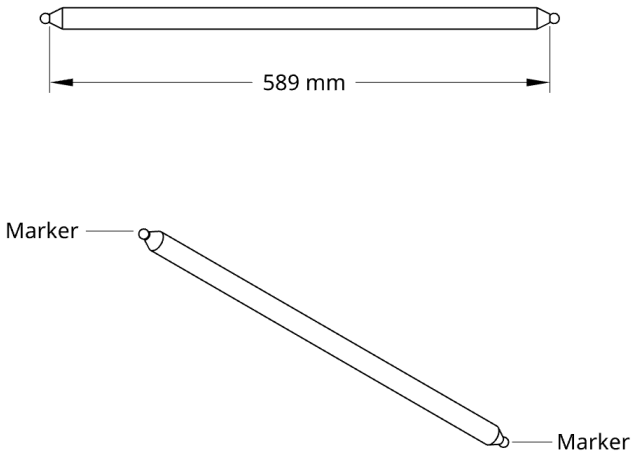


Figure 4.2: Test equipment for measuring the RMS error of the distance between positions of two markers.

Using these two methods, several test cases can be performed and evaluated to understand the accuracy and influence of several parameters. The test cases are presented and explained in the list below.

1. **Relative positional error between measured point and origin**

During the test, cameras were positioned in a square approximately three meters apart and at a height of approximately 2 meters. Origin of the measured area was then calibrated using the calibration routine. The marker was placed on a structure with known height, as shown in Figure 4.1 and the true x and y coordinates of the marker were then measured using a tape measure. Multiple test frames were then recorded and the RMS value between the true and the system's relative coordinates for these frames were then calculated with Equation (1.1). The test was then repeated for several different x and y coordinates together with variable heights.

2. **Relative distance error between two measured points**

The system was calibrated and positioned as described in the previous test. The relative RMS value between two coordinates was measured using a rod with known dimension and a marker at each end, illustrated in Figure 4.2. The tool was moved around by hand in the testing area while data was collected. The RMS value was then calculated with Equation (1.2).

3. **Distance error between two points, dependent on height (z-coordinate)**

The middle point of the two measured points was calculated as Equation (4.1)

$$\frac{\boldsymbol{P_1} + \boldsymbol{P_2}}{2} \tag{4.1}$$

Where $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ are the first and second measured points. The distance error between the two point was then plotted as a function of the height of the rods center-point (z-coordinate). The height h is visualized in Figure 4.3 A linear regression was made to better understand the variation. In addition, the errors was sorted by height, and several histograms were made in order to better understand the performance of the correspondence function, that is the system's ability to determine if points from different cameras corresponds to the same object or not.

4. **Distance error between two points, dependent on polar distance from origin**

Similarly to the previous test case, the center-point of the rod was calculated as in Equation (4.1). The polar distance between the middle point of the two measured positions and the origin of the world coordinate system is the distance in the xy-plane and is calculated as

$$\text{polar distance} = \sqrt{P_x^2 + P_y^2} \tag{4.2}$$

The error was plotted as a function of the polar distance and a linear regression was made to better understand the variation. In addition, the errors were then sorted by the polar distance and several histograms were made to understand the performance of the correspondence function. The polar distance d are shown in Figure 4.3

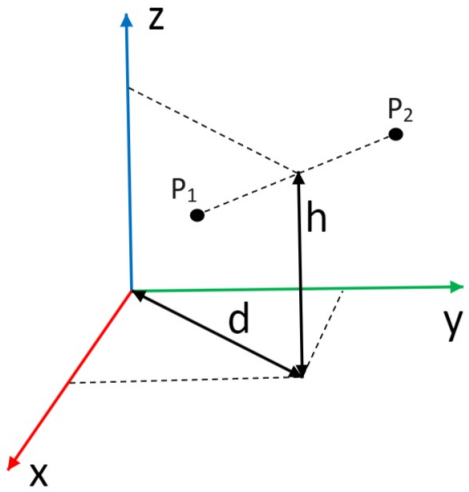


Figure 4.3: Visualization of height h and polar distance d of the middle point of two measured positions P_1 and P_2

5. **Minimum detectable and distinguishable distance between two points**

The system was calibrated identically to test 2, and two markers where then put on the floor in the tracked area 50 cm apart. The markers where then moved closer together until the system could not differentiate them anymore.

6. **System update rate**

While tracking multiple markers, the system should not do any live visualizations nor do anything else than estimate the position and store the time it took to do so in a log file. After the test, the inverse of the average time between each frame was calculated, which resulted in the average update rate.

4.3 Cost

To get the overall cost of the system, the price of all the bought components where combined. This does not however include the cost of the components borrowed during the project or and other components not listed were obtained for free.

4.4 Portability

The following questions were considered during testing in order to determine if the decided requirements for the portability of the system, described in Section 1.1 were met:

a) Does the system require permanent changes to the environment?
b) Does the system require any external measurement and calibration during setup?
c) Do the markers have a diameter of less than five centimeters and weigh less than 10 grams?

5
Results

This section presents the results gathered during the project. Initially results and measurements regarding the hardware developed are presented and followed by simple tests. After this the epipolar parameters used calculating the point correspondences are shown and the following section presents performance and accuracy of points tracked in 3D-space. Finally the cost of constructing the system and the result regarding the goals of portability are shown.

5.1 Hardware related measurements

The camera shutter timings were measured in accordance to the testing scheme introduced in section 3.5 and the following results shown in Table 5.1 were obtained. For all further tests the exposure level -13, corresponding to a shutter speed of 137 µs was used since the brightness of the markers observed at this exposure level was deemed sufficient. The maximum achievable frame rate at this shutter speed was measured to be 92 Hz.

Table 5.1: Table with camera shutter timings for two different exposure levels.

Exposure level	Shutter delay	Shutter speed
-13	788 µs	137 µs
-12	788 µs	265 µs

The oscilloscope trace in Figure 5.1 shows the measured synchronization input and output signals. Moreover, the voltage across the LED-array and current flowing though the LEDs are also shown. All channels except for the current measurement at channel four have the units of volts with their respective voltage per division shown at the bottom of the figure. The current clamp at channel four was set to the 20 A range and outputs 200 mV over the full range, resulting in 10 mV per ampere. The voltage was raised until the average current drawn during the pulse was measured to reach the target value of five ampere previously outlined. The voltage supplied to the driver to reach this current was observed at 14.0 V. For this reason the cutoff voltage for the MCU to stop switching the LED-array was set to 14.0 V. All further tests were carried out at a voltage of 13.8 V to remain below the cutoff limit.

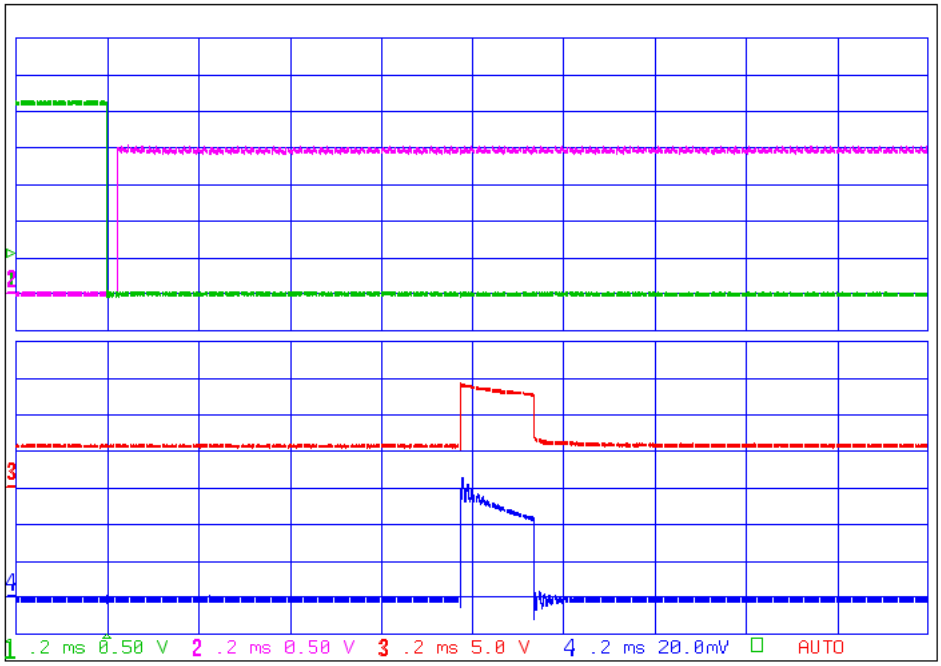


Figure 5.1: Oscilloscope trace of measurement on illuminator driver board. Channel one in green is the incoming synchronization signal and channel two in magenta is the synchronization signal output from the illuminator driver board. Channel 3 in red is the differential probe measuring the voltage over the illuminator LEDs and channel four in blue is the current probe measuring the current passing though the LEDs.

5.2 Maximum detectable marker range

The distance for a stable and reliable detection of a marker was measured and the results are shown in Table 5.2. Since the system could not handle the range required for the originally planned $6 \times 6 \times 3$ meter setup, a camera configuration of approximately $3 \times 3 \times 2$ meters was chosen to perform all following performance and accuracy tests.

Table 5.2: Maximum detectable range by the system for two different marker types. The range given is right before the point detection started to become unreliable.

Marker type	Maximum range
12.6 mm retroreflective sphere	3.2 m
14 mm circular retroreflective fabric	5.7 m

5.3 Epipolar value

To determine the threshold of the epipolar value, e_p defined as in Equation (3.50), a single marker was tracked. During the test, 2850 datapoints of the epipolar value were calculated according to (3.50), logged, and plotted in the histogram shown in Figure 5.2.

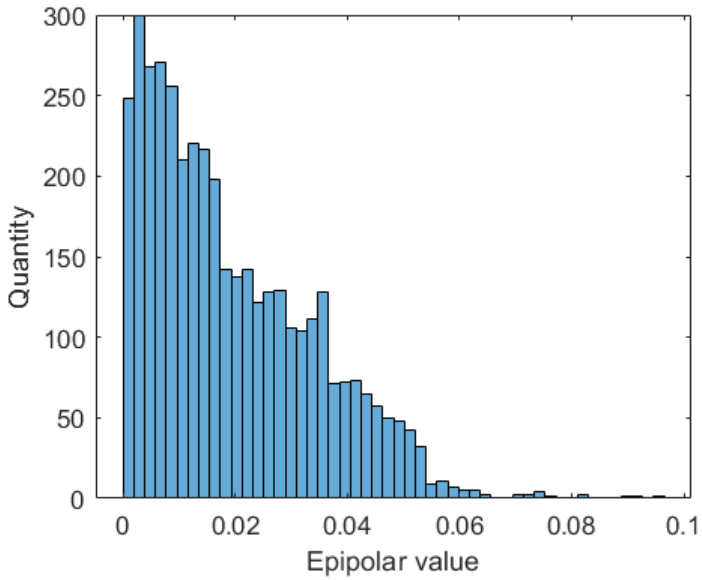


Figure 5.2: Histogram of the epipolar value for a single marker tracked by four cameras.

To optimize for a low false positive and true negative the value was set to 0.055 for the following tests.

5.4 Measurement of accuracy and performance

Following below, the results from all test cases described in Section 4.2 are provided.

1. **The relative error in position between the measured point and the origin**

Multiple points were measured by a ruler in the measurement setup and the system estimated the position. For each measurement, the true position measured with a ruler, x, y, z, the estimate by the system $\hat{x}, \hat{y}, \hat{z}$, and the calculated measurement error is shown in Table 5.3 below. All points together give an RMS error of 2.51 cm

Table 5.3: Systems estimate of known positions and the errors

x [cm]	y [cm]	z [cm]	$\hat{x}$ [cm]	$\hat{y}$ [cm]	$\hat{z}$ [cm]	Error [cm]
4.5	4.5	10.16	3.63	4.1	9.74	1.05
3.5	3.5	30.16	2.28	3.1	29.7	1.36
3.5	3.5	60.16	0.84	2.26	60.0	2.79
104.5	4.5	10.16	103.47	5.64	10.35	1.55
103.5	3.5	30.16	101.76	4.79	30.09	2.17
103.5	3.5	60.16	101.87	4.85	61.26	2.38
4.5	104.5	10.16	1.23	103.6	9.88	3.41
3.5	103.5	30.16	0.42	103.5	30.22	3.08
3.5	103.5	60.16	0.06	103.6	59.71	3.47

2. **The relative error of distance between two measured points**

The pole with two markers at both ends was tracked through different positions in the room. The positions of both points were measured by the system and the calculated distance between them was compared to the known true distance of 58.9 cm. The relative error was calculated, and is presented in Figure 5.3.

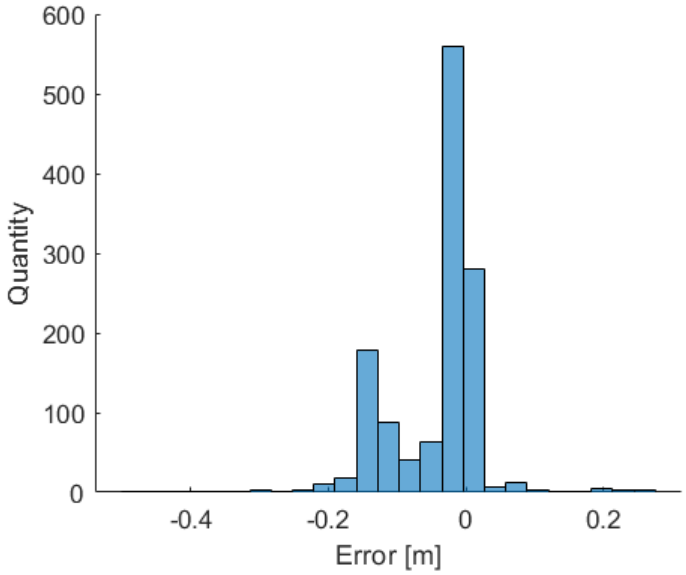


Figure 5.3: Histogram showing the relative error between two points separated by a fixed distance of 58.9 cm when moved through the tracking space.

The implementation for finding correspondences often returns the wrong image points, resulting in an incorrect position. In the following example the two points were correctly identified during the first frame but incorrectly identified during the frame after. In Figure 5.4 the green line between the two points is correct and has

an accurate distance, while the blue line has a wrong lower point and a too-short distance. The red line is the expected measurement of the two points during the second frame to illustrate the error of the blue line. This is calculated by translating the lower point in the same way as the upper. Note that this estimation does not correspond to the correct position of the second point but is used to easily interpret the error of the measured position.

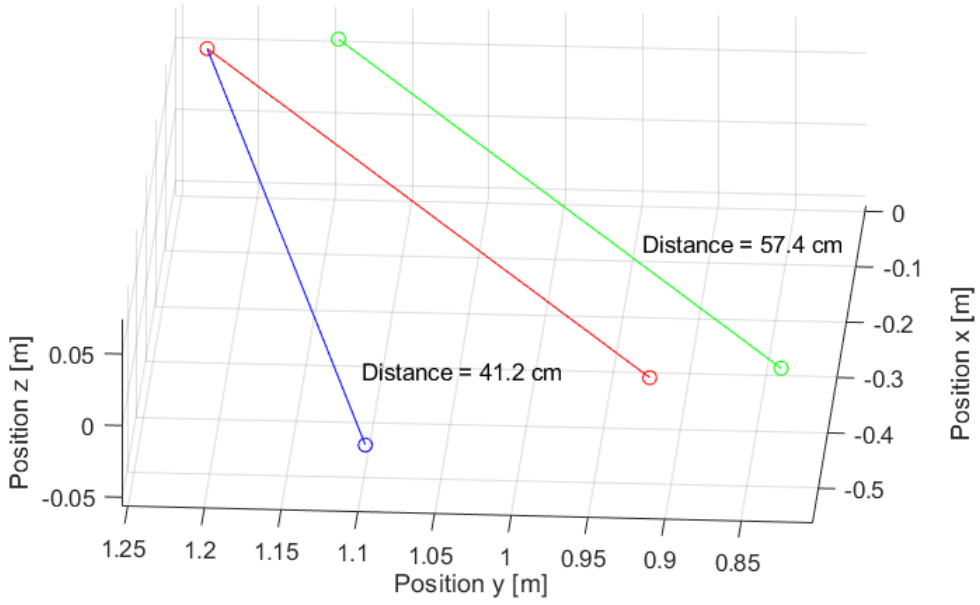


Figure 5.4: Graph of two measurements green and blue, of two points, where the low blue point deviated from the expected red position.

A better understanding of the accuracy of the system is given by removing all points where the error is more than 8.0 cm as these are considered to be due to a bad implementation of the correspondence function and not lack of real accuracy. The value 8.0 cm is chosen as it is in the middle between the two maximums, and are slightly conservatively chosen, meaning it could probably be chosen lower. However, a larger value was chosen to ensure a result that is not lower than the actual accuracy of the system. A histogram with the outliers removed is presented in Figure 5.5.

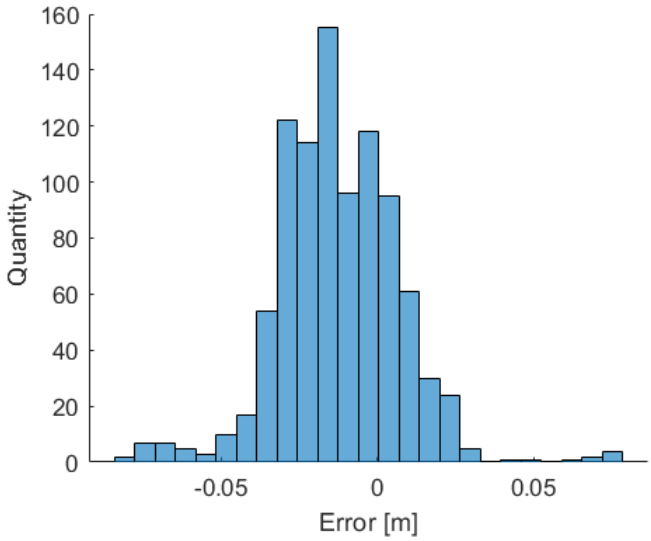


Figure 5.5: Histogram showing the relative error of the measured distance between two points separated 58.9 cm with errors greater than 8.0 cm subtracted from the dataset.

The RMS error of the measured distance between the two points was calculated using two both all points and points with outliers larger than 8.0 cm removed. The two different errors are presented in Table 5.4.

Table 5.4: The RMS error of the measured distance between the two points using all point and all errors lower than 8.0 cm

Measurement	rms error [cm]
Error between two points, all points used	7.79
Error between two points, outliers larger than 8.0 cm removed	2.28

3. **Error of distance between two points, dependent on height (Z-coordinate)**

The relative errors of the distance between the points was sorted by the height of the measurements poles center-point. The errors are plotted below as histograms in Figure 5.6.

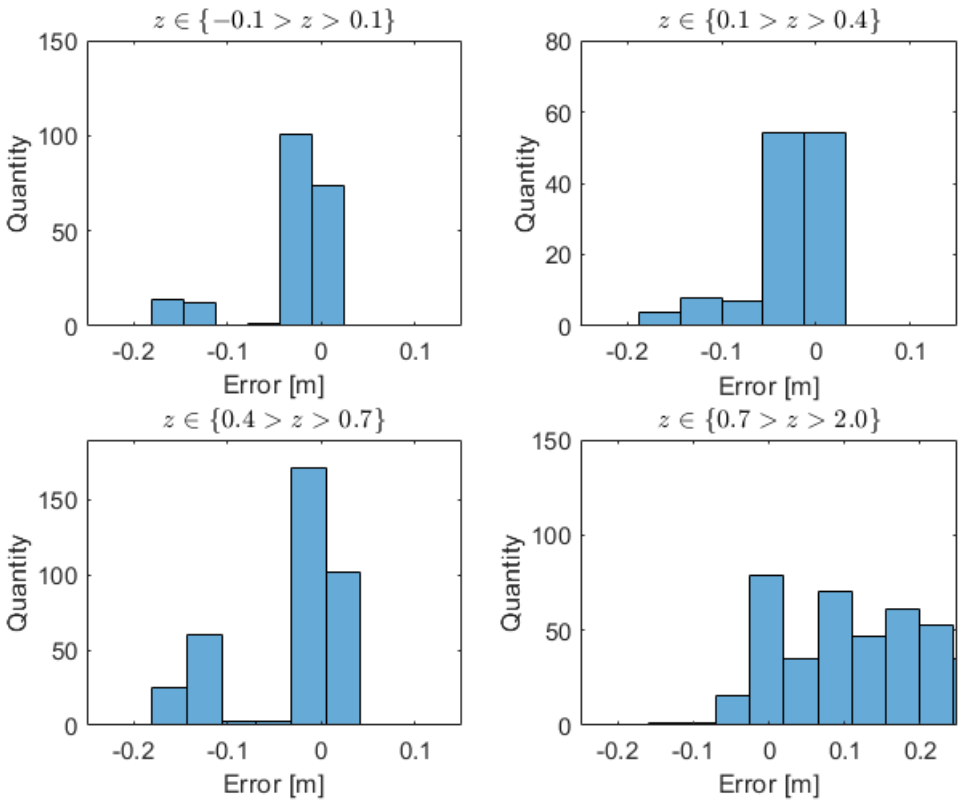


Figure 5.6: Histograms showing the relative error in meters (x-axis) and number of data points (y-axis) grouped by the height of the middle-point.

All errors with outliers larger than 8.0 cm removed, are plotted in Figure 5.7 as a function of the height (z-coordinated) of the measurement poles center point. Thereafter, a linear regression fit was also made.

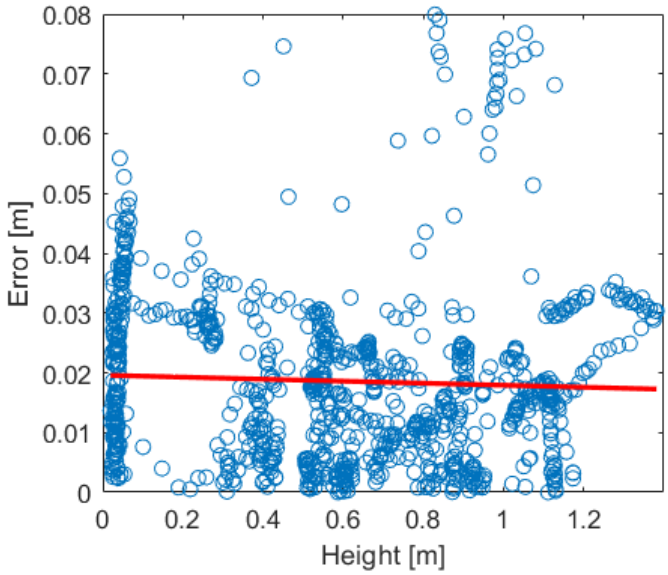


Figure 5.7: The y-axis is the relative error between two fixed points on the measurement pole and the x-axis is the height above the ground plane at which the center-point of the measurement pole lies. All measurements with error above 8.0 cm are regarded as erroneous datapoints and are not plotted. The red line is a linear regression fit.

4. **Error of distance between two points, dependent on polar distance from origin**
The relative error was sorted by the distance from the tracking system's origin and are shown in Figure 5.8.

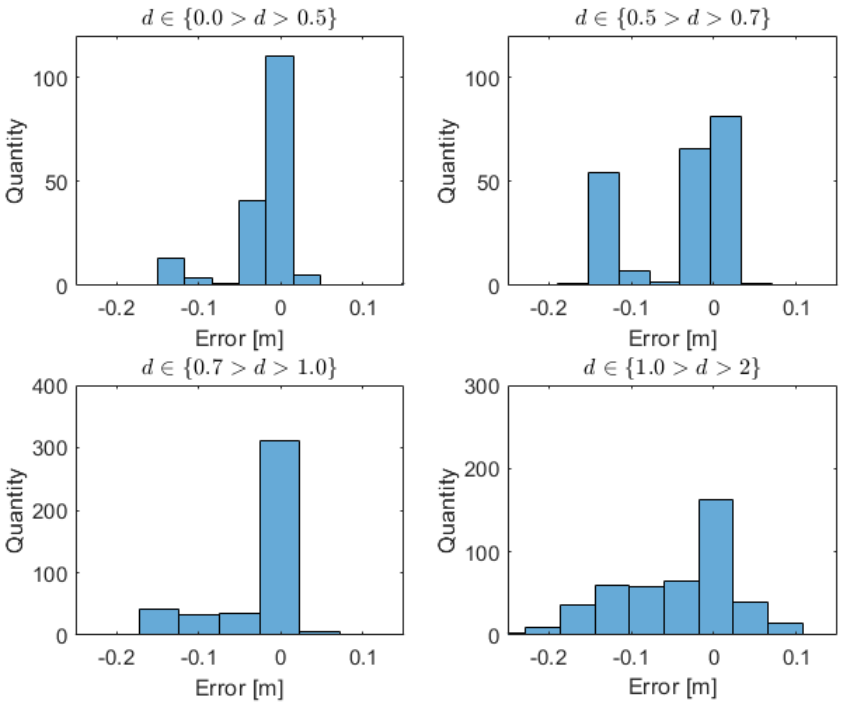


Figure 5.8: Histograms showing the relative error in meters (x-axis) and number of data points (y-axis) grouped by the polar distance from the origin to the the center-point of the measurement pole.

All errors with outliers greater than 8.0 cm removed are plotted as a function of the polar distance between the center-point of the measurement pole and the origin. Thereafter, a linear regression was calculated and plotted in the same figure. The result is given in the figure below.

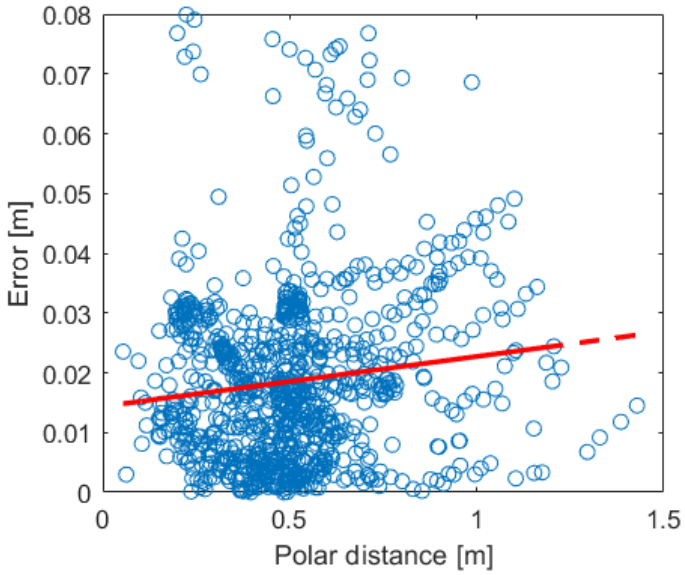


Figure 5.9: The y-axis is the relative error between two fixed points on the measurement pole and the x-axis is the distance from the origin to the center-point of the measurement pole. All measurements with error above 8.0 cm are regarded as erroneous datapoints and are not plotted. The red line is a linear regression fit.

5. **The minimum distance between two points the system can detect and distinguish**

Two markers were placed on the floor and tracked by the system. They were then pushed together until the system was no longer able to distinguish them. The test was done in multiple positions and orientations in the room.
During all test cases, the system was able to detect and distinguish the points until they where put in contact with each other. The closest detectable distance was then 2 cm as that is the radius of two markers.

6. **The update frequency the system is able to achieve**

The average update frequency during the test was 20.9 Hz, which is slightly less than the goal in section 1.1.

5.5 Cost

Table 5.5 below presents the total cost for the system.

Table 5.5: Table showing the total costs for the system. All items not listed were obtained free of charge.

Equipment	Quantity	Cost [SEK]
Active USB 2.0 cable [10 m]	2	422
USB 3.0 cable [5 m]	2	247.46
USB splitter	2	197.6
Camera [Arducam B0332]	4	2702.17
PCB	5	540
PCB components	-	312.73
Total	-	4421.96

5.6 Portability

The system does not require outside measurements during setup and calibration. It also does not require permanent installation or changes to the rooms structure to set up. The diameter of the markers were measured to be 12.6 mm and their weight was measured to be less than one gram. The time it took for two people to setup the system was timed at seven minutes.

6
Discussion

The resulting performance and accuracy is discussed, and potential improvements and areas of further potential or work are suggested. Furthermore, the cost and portability are analysed, followed by a section regarding ethics and societal aspects.

6.1 Maximum detectable marker range

One reason for the limited range of the system compared to the initial estimates could be attributed to the fact that the cameras used only provide a compressed video stream. Image compression was a point of discussion early on in the project when multiple cameras and bandwidth limitations were first brought up. The frames produced by the cameras consist mostly of black or dark pixels and are therefore suitable for a lot of compression. By dynamically compressing only the dark parts of the frame and keeping the fidelity of the markers, a lot of bandwidth and marker resolution could have been saved.

Unfortunately, that is not how the compression of the camera module used operates and as such the previously mentioned limitations were imposed upon our system. The problem that arose during range testing was that the markers suddenly disappeared once a certain distance from the camera was reached. Since the markers appeared as almost completely white up until the point of disappearance, it is assumed that the compression algorithm used on the camera interprets the small dot as noise and completely removes it from the picture. This problem could be solved by using cameras that are capable of transmitting the uncompressed data stream, this however could increase the cost of the system significantly.

Assuming that compression is not the range limiting factor, there are other ways that the range of the system could be increased, for instance by using active light emitting markers. Recalling the inverse square law from Equation (2.1), active markers with the same source strength as the infrared LEDs used for the system would appear four times brighter since the light travels half the distance. The drawbacks of using active markers are, for example, that they have to be powered in some way, usually by batteries and need to be equipped with required electronics which increases the weight that is put on the tracked object.

Another way that the range of the system could be improved is to increase the size of the markers. The main drawback is that the system would then more easily

mix up the different markers when multiple points are being tracked. In turn, that would reduce the short range accuracy of the system and limit its capabilities of tracking small or complex objects.

6.2 Accuracy and performance

While the requirement for the RMS value concerning errors in respect to room coordinate outlined in Section1.1 is notably surpassed, the second RMS goal relative two markers is not achieved when all data is used. Notably, by excluding the outliers exceeding 8 cm this criterion is also met.

To improve the accuracy of the system even further, a filter algorithm could be implemented in future studies. This filter could reduce possible outliers and also increase the accuracy of the system. Besides removing the outliers, a filter could also decrease the white noise and make the output more reliable.

While the filter could result in precision improvements, it could lead to other disadvantages. Depending on the filter used, a time delay due to an increase in computational complexity would most likely be introduced into the system. This could further increase calculation time which does decrease the system's update rate. While there are some disadvantages to consider when implementing a filter it could likely result in improvements in accuracy and reliability outweighing the drawbacks and making it a worthwhile effort to explore.

The performance in terms of update rate were satisfied from a hardware standpoint but lackluster from the software side. The software struggled had to skip frames in order to stay real-time and could not track point at the sought after rate. The main reason for this is that the software ran single-threaded on a single core. Efforts were made during the project to utilize multi-core processing and it was found to give a performance increase. This was however not implemented in the final codebase because of complexity but it would probably bring a noticeable performance increase. Still, even when running tests on multiple cores, it was found that only the image processing steps using OpenCV took too long for the system to be able to run in real time. Switching away from Python would most likely not yield huge performance increases since OpenCV uses C++ bindings as previously discussed. A solution could be running the software on a more powerful system or integrating the image processing in hardware. The second option would however require major developments and engineering efforts. If degraded accuracy is allowable, the cameras could run at a lower resolution, speeding up the image processing steps.

6.3 Finding corresponding point in different images

The method for finding correspondences across different images with multiple cameras and with the ability to handle occlusion turned out to be much harder than expected. As previously stated, by calculating the epipolar constraint, a value close to zero is given, but it is never equal to zero. If the threshold is set too high, points not corresponding to each other are given and when it is set too low, valid points are not accepted. In addition, if a valid pair of points is not considered to correspond to each other, the system will try to match the point with other points on other cameras, and that sometimes leads to the false detection of a point.

As in Figure 5.4, point correspondences are sometimes thoroughly wrong. From our testing, it is clear that this error is due to wrong correspondence and not bad accuracy. During testing, points could flicker back and forth between correct and completely wrong positions. In addition to this, too many points could be register as some cameras correctly detected an object, while others got the wrong correspondences and assumed extra points.

As seen in Figure 5.3, it results in a second order maximum where multiple points are registered at an invalid position. Even if this is a major problem that future work must take care of, it is still of interest to understand the accuracy of the correctly identified positions and ignore the outliers.

Since the number of points in the dataset is too large for manually finding erroneous detections, calculations were made for all errors less than 8 cm. This allowed us to obtain as many of the correctly identified points as possible, and as few of the incorrectly identified points as possible. The value was selected by observing the histogram of all captured points in Figure 5.3 and making the judgement that the local minimum between the two peaks of point detections is a good middle ground. From the figure it is worth noticing that no *major* maximums are present. Secondly the distribution seems to be of Gaussian nature and therefore it could be assumed that most of the correctly identified points are captured.

6.4 Accuracy depending on height and polar distance

According to Figure 5.9, the error increases when the object tracked is far away from the origin of the coordinate system. This is as expected, as the system is calibrated at the origin. The fact that the error increases with distance to the origin leaves us with a very broad set of variables to investigate but it is entirely possible that that the system itself isn't perfectly calibrated, contributing to projection errors near the fringes of the tracked volume.

The result in Figure 5.8 shows how the system tends to handle multiple tracked points worse at a distance further away from the origin. As a larger amount of points seem to get matched with the wrong point when the distance increases in all cases except for the distances 0.7-1.0m.

An interesting result is that the error does not seem to increase with height (z-coordinate) as shown in Figure 5.9, rather, it seems to decrease slightly. This is in contradiction to the previously stated result in the paragraph above. One explanation could be that as the point gets placed at a higher position, it gets closer to the camera, thus increasing the accuracy of the cameras and the system. Similarly, it is easier for us humans to estimate the distance between two points if they are close to us. This however contradicts the previous hypothesis that the estimates are of good quality close to the origin where the system is calibrated.

However, Figure 5.6 depicting error based on height above the floor, indicated that the system becomes worse at handling multiple points and correlating different points of different images when the height increases. This is most likely caused by a higher projection error when the object is placed further away from the origin of the calibration.

6.5 Bundle Adjustment

Implementing bundle adjustment into the project turned out to be more difficult than anticipated and ultimately was not ready in time for testing. As mentioned in section 2.1, the procedure attempts to simultaneously optimize all parameters camera and point parameters greatly increasing the points of failure. The nature of the transforms made the incorrect results incoherent and hard to interpret even with a live 3D preview and therefore hard to debug.

Regrettably, the feature was correctly implemented shortly after testing was done and time was running short, meaning that the tests could not be repeated with the same setup used for all other tests. Due to this it cannot be presented as a result in this report but is still worth as a discussion aspect.

Bundle adjustment remains fully implemented at this point and showed great promise regarding accuracy during the limited testing it saw. For this reason it is recommended as a great starting point for future development.

6.6 Cost

Some of the equipment used in the project was not bought since it was easily accessible to borrow from various sources. That includes retroreflective markers, camera tripods and a function generator to run the synchronization signal. Some equipment used was also found for free in scrap bins from other projects, including the MCUs, power cables for the cameras and retroreflective tape. Out of the equipment that

was not bought the function generator has the highest cost and its requirement to run the system limits the usability at the same price range. However, since the synchronisation signal only needs to be a square wave at 90 Hz with an amplitude of 5 V peak to peak, a generator for this signal can effortlessly be built with a microcontroller.

6.7 Portability

Portability was a major design goal for both the software and hardware during the project. When it comes to enabling portability through the software, ease of calibration plays a major role and for this reason it was made automatic through camera pose estimation. Because of this implementation moving the cameras poses little to no problem and the user is free to mount it however they like.

Hardware-wise, the steps taken when the system hardware is set up includes to mount the cameras on stands. Since the camera houses are fitted with screw holes for M6 and 1/4-20 UNC threads they can easily be mounted on camera stands or other types of mounts with corresponding threading. This further contributes to the portability of the system.

Currently, the system requires a dedicated function generator to function and it has been operated using a lab bench power supply. Needless to say, this drastically decreases the overall portability of the system. To greatly improve the portability one could implement a dedicated small square wave generator into the system using a inexpensive MCU. A smaller portable power supply or AC adapter could also be used, preferably one with tunable voltage around the range of 10-14 V.

6.8 Society and ethics

The development of a positioning system such as the one discussed necessitates evaluation of certain valid social and ethical concerns. Optical systems and cameras in public spaces, for example security cameras, are a basis for discussions mainly about intrusion on peoples privacy and integrity. Thus, questions about how the recorded information is used and stored are important to assess when proceeding with the development of such systems.

The optical positioning system developed in this project relies on a portable solution and is disassembled after use. The room used when testing should also be cleared and there is therefore a limited risk of recording people. There does however remain a limited risk of privacy intrusion due to the possibility of windows and of people entering the room and camera field of views by mistake.

Furthermore, the computer connected to the cameras will always have the capability to store footage of the rooms and buildings where it has been used. The fact remains that the technique used for tracking involves normal cameras, and therefore

leaves the user with the ability to record and save footage despite this not being an intended use.

The ways to address this privacy issue are very limited as the ability to capture images itself is what gives rise to the problem. Currently there exist commercial systems that aim to mitigate or eliminate the issue, one of which is the infra-red system used by Oculus. Although the system uses infra-red cameras, they are programmed in a way as to not expose them as video devices to the operating system. This approach has the advantage of not adding any hardware complexity, but means the video stream could be accessed by a modified computer driver [35].

In contrast, had the video stream been processed and blobs detected onboard the camera using a suitable processor, there would be no need to send the video stream to a computer but only coordinates of the points detected. This implementation implies that the hardware itself would be designed to make it impossible to save a video and therefore pose little to no threat to privacy.

7
Conclusion

The system is capable of achieving high accuracy and stays well within most of the goals for this project. Although it experiences difficulty in distinguishing multiple points in space from each other which limits the capabilities to track multiple objects at the same time reliably. However, when disregarding the easily identifiable errors it is still able to achieve great accuracy. Furthermore it still accomplishes the goal of being portable, all while staying within the budget of 5000 SEK.

Although the system is able to reach the goals for accuracy, budget and portability, the updating frequency specified in the requirements is not achieved. While the hardware is able to deliver a higher refresh rate, it is limitation within the software and image processing that takes too much processing time.

Furthermore, the system's performance is subjected to limitations imposed by the detection range of the markers. While the intended coverage area was envisioned at 6x6 m, the actual trackable volume is constrained to a more confined space of 3x3 m.

Bibliography

[1] W. Kang and Y. Han, "SmartPDR: Smartphone-Based Pedestrian Dead Reckoning for Indoor Localization," *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2906–2916, May 2015. DOI: 10.1109/JSEN.2014.2382568. visited on 2024-04-07.

[2] S. Shang and L. Wang, "Overview of wifi fingerprinting-based indoor positioning," *IET Communications*, vol. 16, no. 7, pp. 725–733, Apr. 2022. DOI: https://doi.org/10.1049/cmu2.12386. visited on 2024-03-15.

[3] Z. Min, D. Zhu, and M. Q.-H. Meng, "Accuracy assessment of an n-ocular motion capture system for surgical tool tip tracking using pivot calibration," in *2016 IEEE International Conference on Information and Automation (ICIA)*, Aug. 2016, pp. 1630–1634. DOI: 10.1109/ICInfA.2016.7832079. visited on 2024-03-05.

[4] Lianjun Zhang and Lifang Zhao, "Research of ultrasonic distance measurement system based on DSP," in *2011 International Conference on Computer Science and Service System (CSSS)*, IEEE, Jun. 2011, pp. 2455–2458. DOI: 10.1109/CSSS.2011.5974489. visited on 2023-03-08.

[5] F. Ijaz, H. K. Yang, A. W. Ahmad, and C. Lee, "Indoor positioning: A review of indoor ultrasonic positioning systems," in *2013 15th International Conference on Advanced Communications Technology (ICACT)*, Jan. 2013, pp. 1146–1150. [Online]. Available: https://ieeexplore.ieee.org/document/6488379 visited on 2024-04-22.

[6] C. Medina, J. Segura, and De la Torre, "Ultrasound Indoor Positioning System Based on a Low-Power Wireless Sensor Network Providing Sub-Centimeter Accuracy," *Sensors*, vol. 13, no. 3, pp. 3501–3526, Mar. 2013. DOI: 10.3390/s130303501. visited on 2024-03-02.

[7] A. Alarifi, A. Al-Salman, M. Alsaleh, *et al.*, "Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances," *Sensors*, vol. 16, no. 5, p. 707, May 2016. DOI: 10.3390/s16050707. visited on 2024-03-02.

[8] Optitrack, "Flex 3 - In Depth". [Online]. Available: https://optitrack.com/cameras/flex-3/ visited on 2024-02-06.

[9] H. Ishii, "Augmented reality: Fundamentals and nuclear related applications," *International Journal of NUCLEAR SAFETY AND SIMULATION*, vol. 1, p. 320, Dec. 2010.

[10] A. V. Arecchi, T. Messadi, and R. J. Koshel, *Retroreflectors*. [Online]. Available: https://spie.org/publications/spie-publication-resources/optipedia-free-optics-information/fg11_p31-32_retroreflectors visited on 2024-05-24.

[11] V. Petrović and J. Popović-Bozovic, "Towards real-time blob detection in large images with reduced memory cost," Jun. 2016, p. 1.

[12] M. Maheepala, A. Z. Kouzani, and M. A. Joordens, "Light-Based Indoor Positioning Systems: A Review," *IEEE Sensors Journal*, vol. 20, no. 8, pp. 3971–3995, Apr. 2020. DOI: 10.1109/JSEN.2020.2964380.

[13] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (slam) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, Jun. 2001. DOI: 10.1109/70.938381.

[14] D. Doody, *Basics of spaceflight*. 2017, ch. Electromagnetics. [Online]. Available: https://science.nasa.gov/learn/basics-of-space-flight/chapter6-1/ visited on 2024-05-10.

[15] K. Sadekar, "Understanding Lens Distortion", LearnOpenCV, Oct. 2020. [Online]. Available: https://learnopencv.com/understanding-lens-distortion/ visited on 2024-05-09.

[16] The MathWorks Inc., *What Is Camera Calibration? - MATLAB & Simulink - MathWorks Nordic*. [Online]. Available: https://se.mathworks.com/help/vision/ug/camera-calibration.html visited on 2024-05-09.

[17] Y. Chen, Y. Chen, and G. Wang, "Bundle Adjustment Revisited," Dec. 2019, arXiv:1912.03858 [cs]. [Online]. Available: http://arxiv.org/abs/1912.03858 visited on 2024-04-29.

[18] C. Stachniss, "The Basics about Bundle Adjustment (Cyrill Stachniss)," *Youtube*, Sep. 26, 2020. [Online]. Available: https://www.youtube.com/watch?v=sobyKHwgB0Y visited on 2024-04-29.

[19] X. Huang and R. Qin, "Multi-view large-scale bundle adjustment method for high-resolution satellite images," 2019. DOI: 10.48550/ARXIV.1905.09152.

[20] E. R. Eiríksson, *Understanding Reprojection Error*, Aug. 2022. [Online]. Available: https://calib.io/blogs/knowledge-base/understanding-reprojection-errors visited on 2024-04-29.

[21] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[22] C. R. Harris, K. J. Millman, S. J. van der Walt, *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[23] Arducam, *Ov9281 global shutter - arducam wiki*. [Online]. Available: https://docs.arducam.com/UVC-Camera/Appilcation-Note/External-Trigger-Mode/OV9281-Global-Shutter/ visited on 2024-04-29.

[24] National Aeronautics and Space Administration, Science Mission Directorate, "Visible Light ", 2010. [Online]. Available: https://science.nasa.gov/ems/09_visiblelight/ visited on 2024-05-10.

[25] Newopto, "Infrared light emitting diode," XYC-HIR76C-LX0 datasheet, Edition. A0, Sep. 2021.

[26] Compaq, Hewlett-Packard, Intel, *et al.*, *Unicersal serial bus specification*, 2000.

[27] Infineon, "High and low side gate driver," IRS2181/IRS21814(S)PbF datasheet, Jun. 2006.

[28] OpenCV Foundation, "OpenCV: Contour Features". [Online]. Available: https://docs.opencv.org/4.x/dd/d49/tutorial_py_contour_features.html visited on 2024-05-09.

[29] S. Nayar, "Linear Camera Model | Camera Calibration," *Youtube*, Apr. 18, 2021. [Online]. Available: https://www.youtube.com/watch?v=qByYk6JggQU&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo&index=2&ab_channel=FirstPrinciplesofComputerVision.

[30] S. Nayar, "Computing Depth | Uncalibrated Stereo," *Youtube*, Apr. 26, 2021. [Online]. Available: https://www.youtube.com/watch?v=OYwm4VM6uNg&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo&index=12&ab_channel=FirstPrinciplesofComputerVision.

[31] S. Nayar, "Epipolar Geometry | Uncalibrated Stereo," *Youtube*, Apr. 26, 2021. [Online]. Available: https://www.youtube.com/watch?v=6kpBqfgSPRc&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo&index=9&ab_channel=FirstPrinciplesofComputerVision.

[32] P. S. Foundation, "pickle — Python object serialization — Python 3.7.3 documentation", 2019. [Online]. Available: https://docs.python.org/3/library/pickle.html visited on 2024-04-25.

[33] J. S. Dai, "Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections," *Mechanism and Machine Theory*, vol. 92, pp. 144–152, 2015. DOI: https://doi.org/10.1016/j.mechmachtheory.2015.03.004.

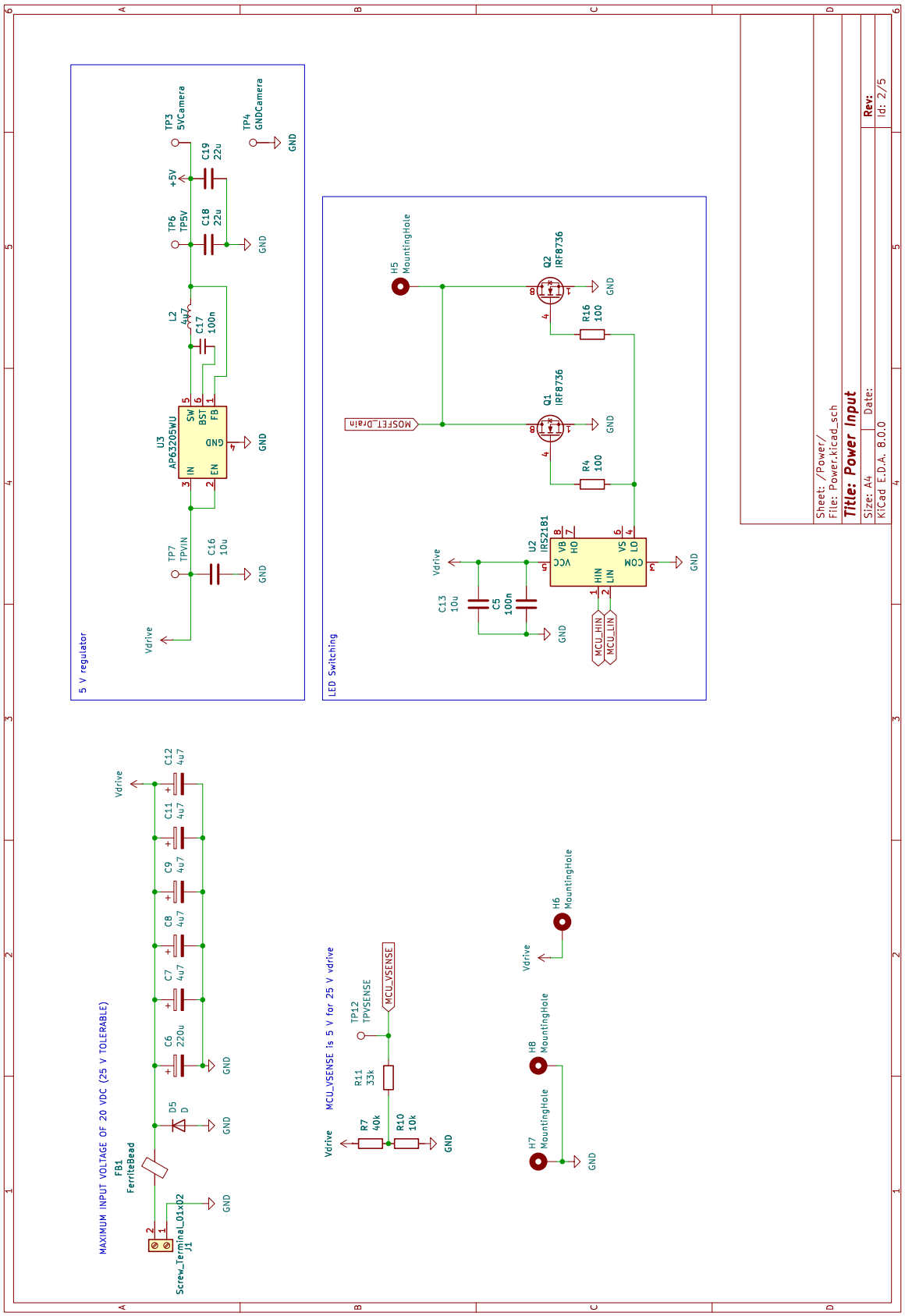
[34] S. Nayar, "Finding Correspondences | Uncalibrated Stereo" *Youtube*, Apr. 26, 2021. [Online]. Available: https://www.youtube.com/watch?v=erpiFudDBlg&list=PL2zRqk16wsdoCCLpou-dGo7QQNks1Ppzo&index=11.

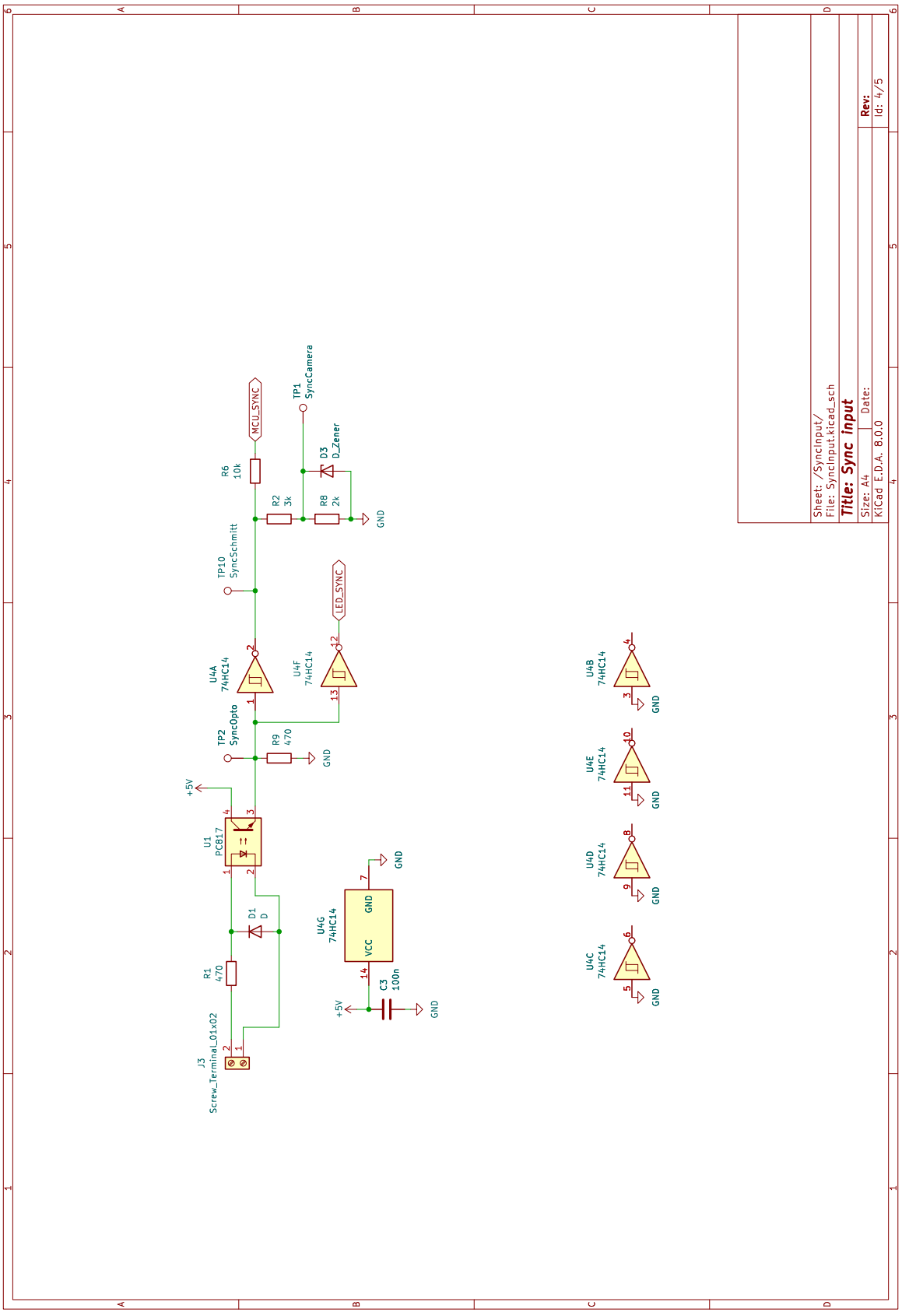
[35] J. Durbin, *Oculus sensors are technically hackable webcams*, UploadVR, Jan. 2017. [Online]. Available: https://www.uploadvr.com/hackable-webcam-oculus-sensor-be-aware/ visited on 2024-02-06.

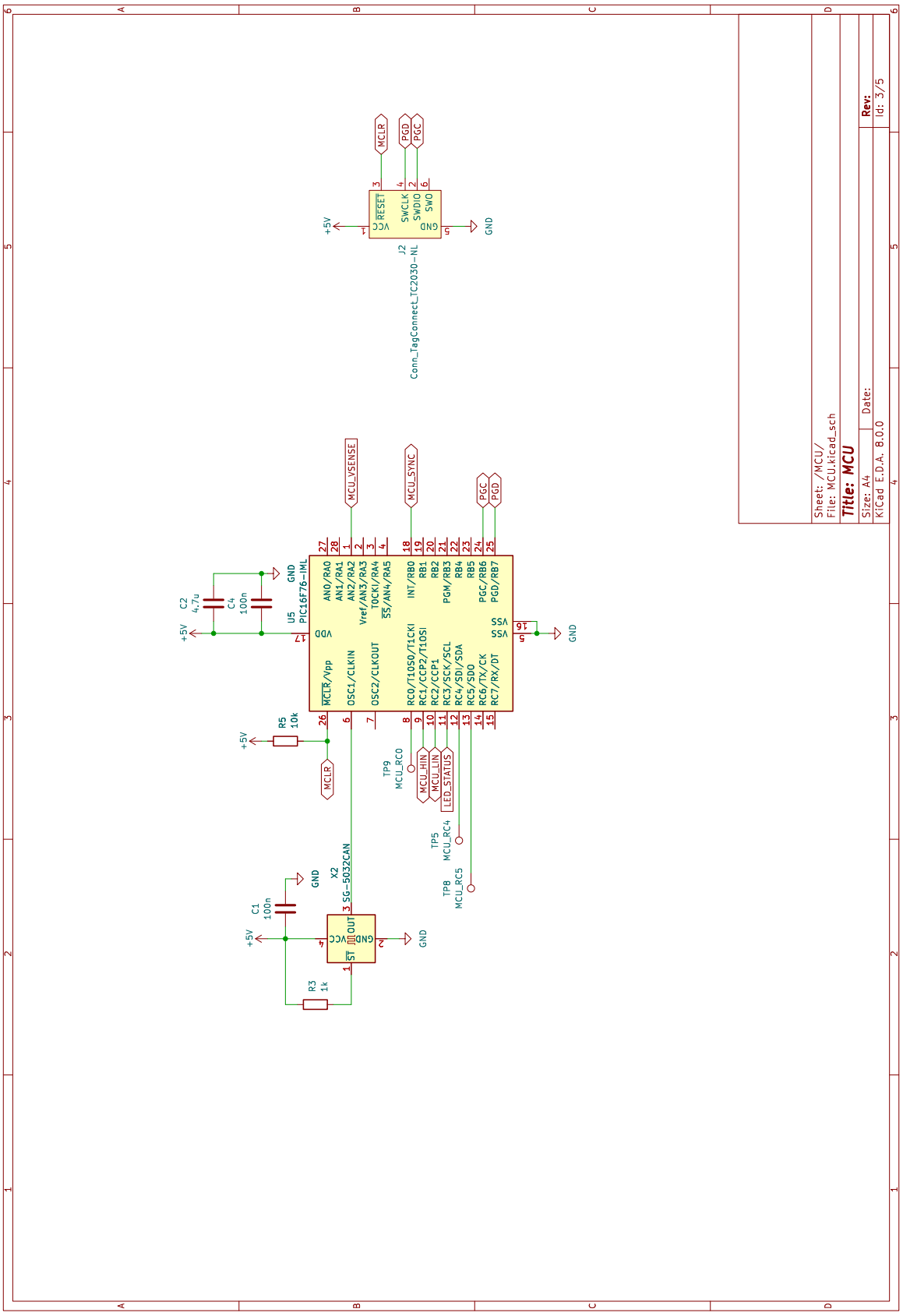
A

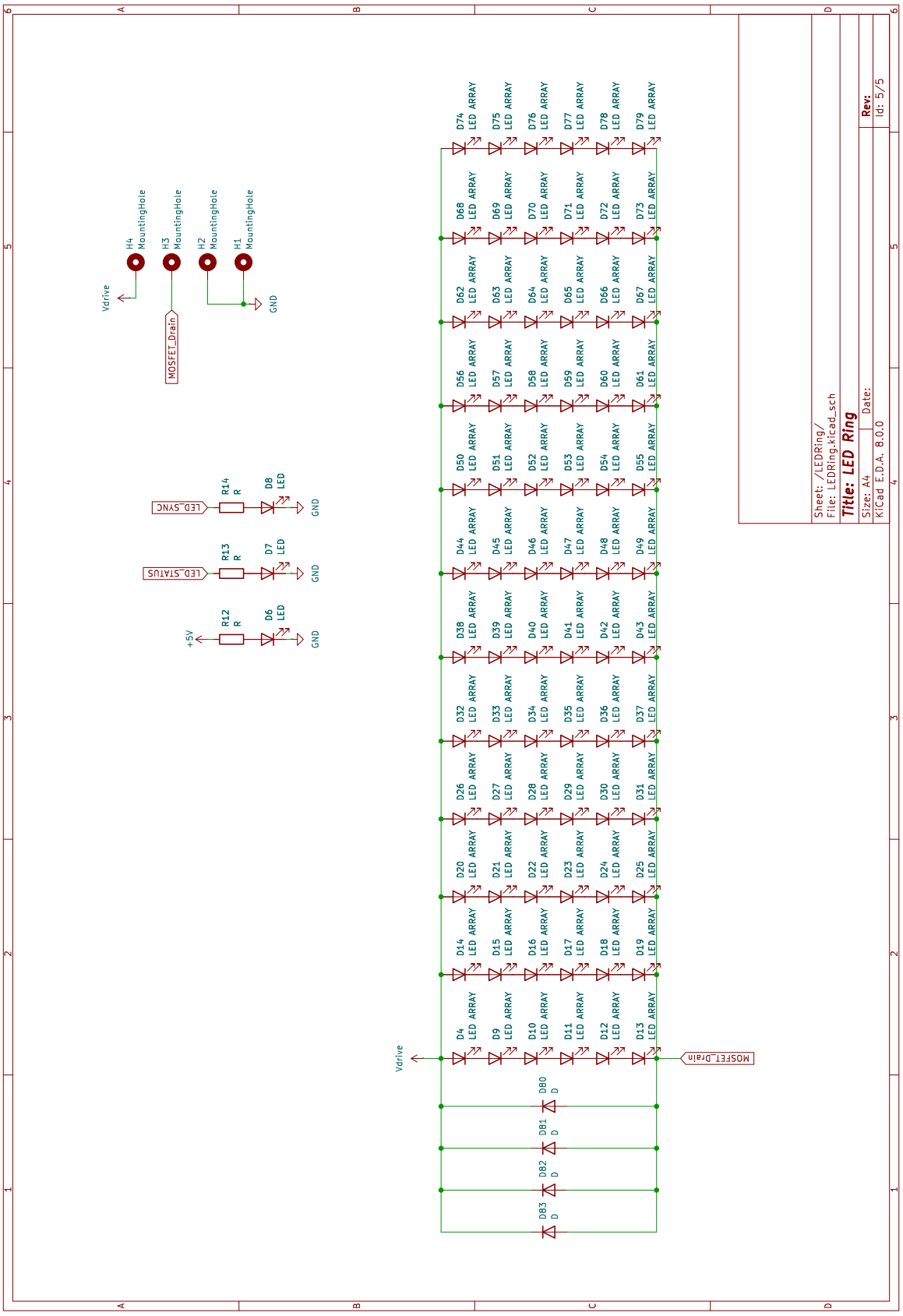
Schematics & PCB layouts.

The following pages contain full schematics and PCB layouts for the illuminator LED-ring and illuminator driver PCBs.

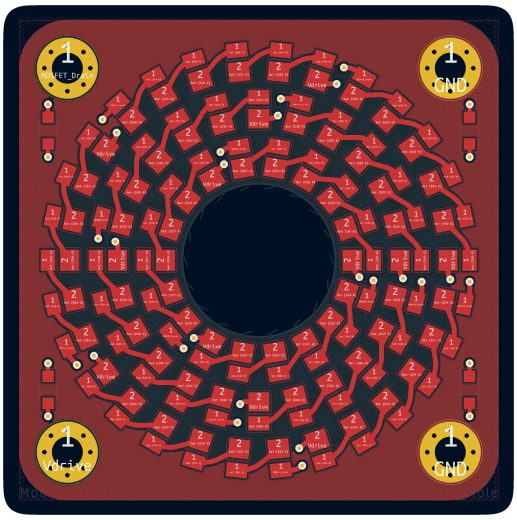

5 V regulator
LED Switching
MAXIMUM INPUT VOLTAGE OF 20 VDC (25 V TOLERABLE)
MCU_VSENSE is 5 V for 25 V vdrive
Sheet: /Power/
File: Power.kicad_sch
Title: Power Input
Size: A4
Date:
Rev:
KiCad E.D.A. 8.0.0
Id: 2/5


Sheet: /SyncInput/
File: SyncInput.kicad_sch
Title: Sync input
Size: A4
Date:
Rev:
KiCad E.D.A. 8.0.0
Id: 4/5
MCU_SYNC
TP1
SyncCamera
R6
10k
D3
D_Zener
R2
3k
R8
2k
GND
TP10
SyncSchmitt
LED_SYNC
U4A
74HC14
U4F
74HC14
TP2
SyncOpto
R9
470
+5V
U1
PC817
D1
D
R1
470
J3
Screw_Terminal_01x02
U4G
74HC14
VCC
C3
100n
U4B
74HC14
U4E
74HC14
U4D
74HC14
U4C
74HC14

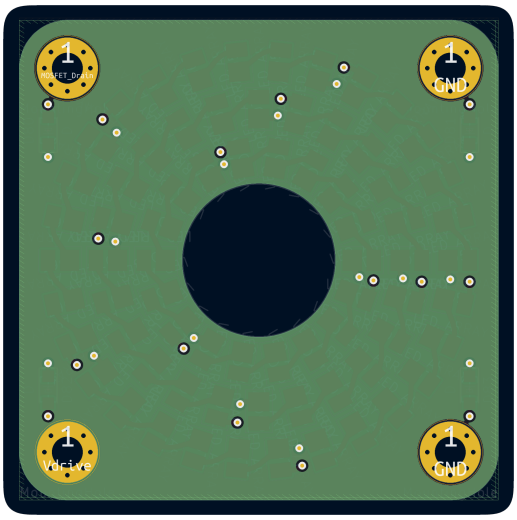

MCLR
PGD
PGC
RESET
SWCLK
SWDIO
SWO
VCC
GND
+5V
GND
J2
Conn_TagConnect_TC2030-NL
MCU_VSENSE
MCU_SYNC
PGC
PGD
U5
PIC16F76-IML
C2
4.7u
C4
100n
R5
10k
TP9
MCU_RC0
MCU_HIN
MCU_LIN
LED_STATUS
TP5
MCU_RC4
TP8
MCU_RC5
X2
SG-5032CAN
C1
100n
R3
1k
Sheet: /MCU/
File: MCU.kicad_sch
Title: MCU
Size: A4
Date:
Rev:
KiCad E.D.A. 8.0.0
Id: 3/5


H4 MountingHole
H3 MountingHole
H2 MountingHole
H1 MountingHole
Vdrive
MOSFET_Drain
GND
LED_SYNC
LED_STATUS
+5V
R14 R
R13 R
R12 R
D8 LED
D7 LED
D6 LED
D4 LED ARRAY
D9 LED ARRAY
D10 LED ARRAY
D11 LED ARRAY
D12 LED ARRAY
D13 LED ARRAY
D14 LED ARRAY
D15 LED ARRAY
D16 LED ARRAY
D17 LED ARRAY
D18 LED ARRAY
D19 LED ARRAY
D20 LED ARRAY
D21 LED ARRAY
D22 LED ARRAY
D23 LED ARRAY
D24 LED ARRAY
D25 LED ARRAY
D26 LED ARRAY
D27 LED ARRAY
D28 LED ARRAY
D29 LED ARRAY
D30 LED ARRAY
D31 LED ARRAY
D32 LED ARRAY
D33 LED ARRAY
D34 LED ARRAY
D35 LED ARRAY
D36 LED ARRAY
D37 LED ARRAY
D38 LED ARRAY
D39 LED ARRAY
D40 LED ARRAY
D41 LED ARRAY
D42 LED ARRAY
D43 LED ARRAY
D44 LED ARRAY
D45 LED ARRAY
D46 LED ARRAY
D47 LED ARRAY
D48 LED ARRAY
D49 LED ARRAY
D50 LED ARRAY
D51 LED ARRAY
D52 LED ARRAY
D53 LED ARRAY
D54 LED ARRAY
D55 LED ARRAY
D56 LED ARRAY
D57 LED ARRAY
D58 LED ARRAY
D59 LED ARRAY
D60 LED ARRAY
D61 LED ARRAY
D62 LED ARRAY
D63 LED ARRAY
D64 LED ARRAY
D65 LED ARRAY
D66 LED ARRAY
D67 LED ARRAY
D68 LED ARRAY
D69 LED ARRAY
D70 LED ARRAY
D71 LED ARRAY
D72 LED ARRAY
D73 LED ARRAY
D74 LED ARRAY
D75 LED ARRAY
D76 LED ARRAY
D77 LED ARRAY
D78 LED ARRAY
D79 LED ARRAY
D80 D
D81 D
D82 D
D83 D
MOSFET_Drain
Sheet: /LEDRing/
File: LEDRing.kicad_sch
Title: LED Ring
Size: A4
Date:
Rev:
KiCad E.D.A. 8.0.0
Id: 5/5

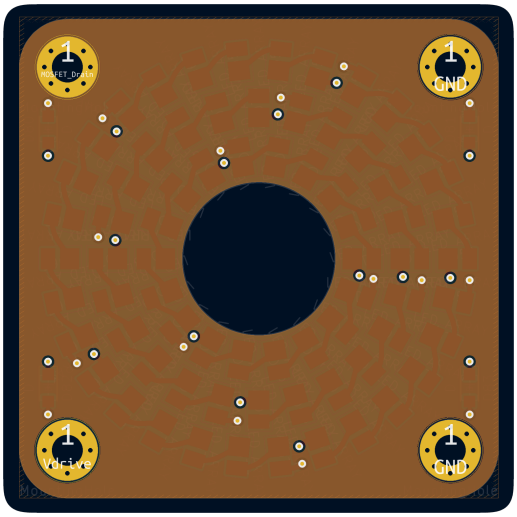
Layout of the four layer illuminator LED-ring PCB.



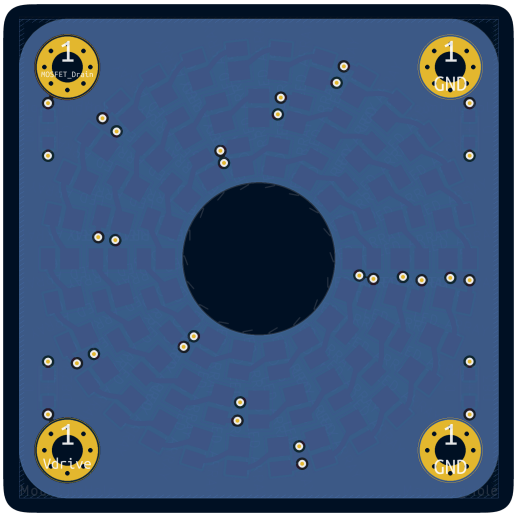
Top copper layer.



First inner copper layer.

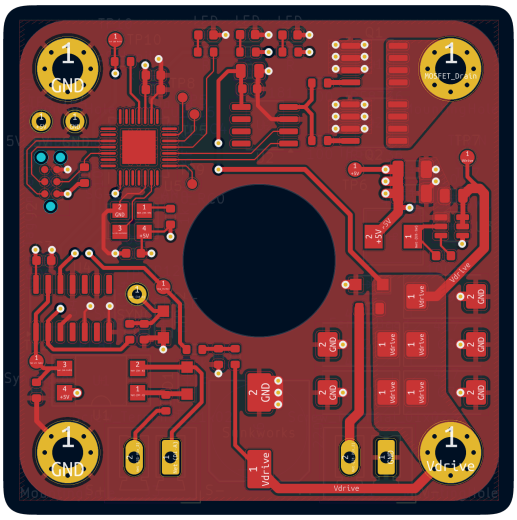


Second inner copper layer.

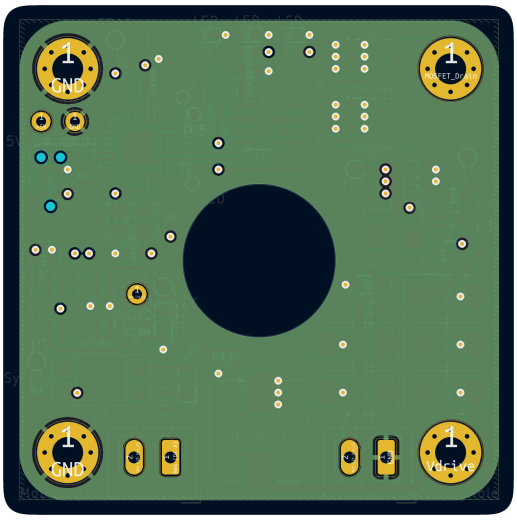


Bottom copper layer.

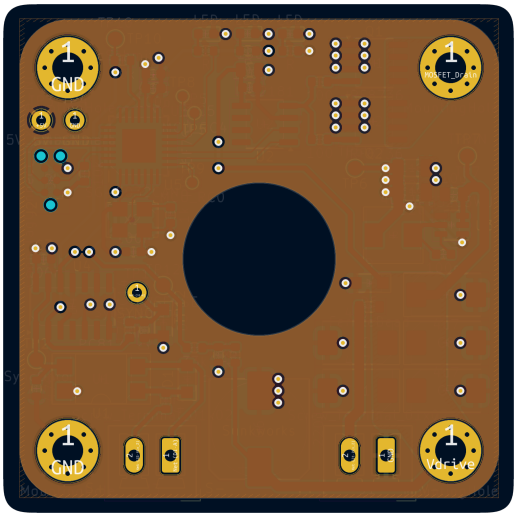
Layout of the four layer illuminator driver PCB.



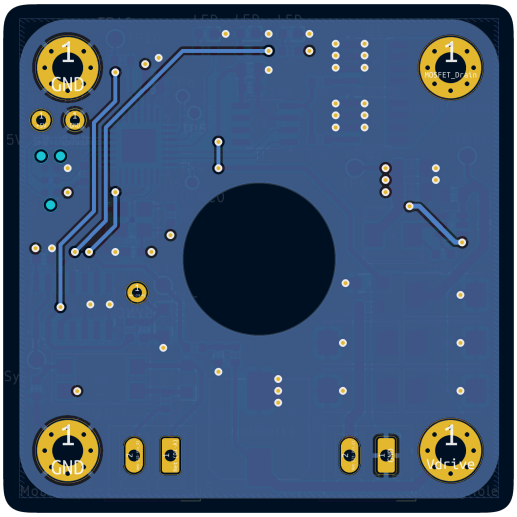
Top copper layer.



First inner copper layer.



Second inner copper layer.



Bottom copper layer.

B

MCU Source code.

```
#include <xc.h>
#include <stdbool.h>

// CONFIG
#pragma config FOSC = HS        // Oscillator Selection bits (RC oscillator)
#pragma config WDTE = OFF        // Watchdog Timer Enable bit (WDT enabled)
#pragma config PWRTE = OFF      // Power-up Timer Enable bit (PWRT disabled)
#pragma config CP = OFF         // FLASH Program Memory Code Protection bit (Code protection off)
#pragma config BOREN = ON       // Brown-out Reset Enable bit (BOR enabled)

// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.

void delay(int j);

int voltageConfidence = 0;
bool turnOnFromOvervoltage = true;
int voltageConfidence_high = 0;

void main(void)
{
    TRISA = 0b00000100;     // Set RA2 to input
    TRISB = 0b00010001;     // Set RB0
    TRISC = 0b00000000;     // Set RC0 to output

    OPTION_REG = 0b11000011; // Interrupt & prescaler setup
    INTCON = 0b11000000;

    T1CON = 0b00110001;

    ADCON0 = 0b10010001;    // 10: Fosc/32, 010: Channel 2 (RA2/AN2), 1: ADON
    ADCON1 = 0b00000000;    // Use VDD as ADC reference.



    INTCONbits.INTE = 1;
    INTCONbits.INTF = 0;

```

```
39      while (1){
40          // Request ADC-read
41          ADCON0bits.GO_DONE = 1;
42          while (ADCON0bits.GO_DONE == 1)      // Wait for ADC-read
    completion
43              delay(1);
44
45          if (ADRES >= 0b10010001){   // Trigger over voltage
    protection at 14 V
46              PORTC = 0b00001000;
47
48              voltageConfidence = 0;
49              INTCONbits.INTE = 0;
50              INTCONbits.TMR0IE = 0;
51              PIE1bits.TMR1IE = 0;
52              turnOnFromOvervoltage = true;
53
54          } else {
55              if (voltageConfidence < 10000)       // Voltage
    confidence allows for reading hysteresis
56                  voltageConfidence += 1;
57              else
58              {
59                      // Turn off red-LED
60                  if(turnOnFromOvervoltage)
61                  {
62                  PORTC = 0b00000000;
63                  INTCONbits.INTF = 0;
64                  INTCONbits.INTE = 1;
65                  turnOnFromOvervoltage = false;
66                  }
67
68              }
69
70          }
71
72      }
73
74  return;
75  }
76
77  // Function for creating delays
78  void delay(int j)
79  {
80      int i;
81      for (i = 0; i < j; i++) {
82
83      }
84      return;
85  }
86
87
88  void __interrupt() isr_ext(void)
89  {
90       if(INTCONbits.INTE == 1 && INTCONbits.INTF == 1) // SYNCPULS
    DETECTED.
```

```
91      {
92          INTCONbits.INTF = 0; // RESETS EXT INTERRUPT FLAG
93          INTCONbits.INTE = 0; // turn off sync pulse detection
    interrupt
94
95          INTCONbits.TMR0IF = 0; // RESETS TIMER0 FLAG
96          INTCONbits.TMR0IE = 1; // Enable timer0
97
98          //Sets delay to turn on gate driver
99          TMR0 = 0b01001000;
100         OPTION_REGbits.PS2=0;
101         OPTION_REGbits.PS1=1;
102         OPTION_REGbits.PS0=1;
103     }
104     if(INTCONbits.TMR0IE == 1 && INTCONbits.TMR0IF == 1) //Turn on
    gate driver
105     {
106         //Reset flag and disable timer0 interrupt
107         TMR0 = 0;
108         INTCONbits.TMR0IE = 0;
109         INTCONbits.TMR0IF = 0;
110
111         //Turn on gate driver
112         RC2 = 1;
113         RC1 = 1;
114
115         //Set delay for turning off gate driver
116         TMR1H = 0b11111111;
117         TMR1L = 0b10110100;
118
119         // Turn on timer1
120         PIR1bits.TMR1IF = 0;
121         PIE1bits.TMR1IE = 1;
122     }
123     if(PIE1bits.TMR1IE == 1 && PIR1bits.TMR1IF == 1) //Turn off
    gate driver
124     {
125         // Turn of gate driver
126         RC2 = 0;
127         RC1 = 0;
128
129         PIE1bits.TMR1IE = 0;  // Disable timer0 interrupt
130         PIR1bits.TMR1IF = 0;  // Resets time0 flag
131
132         //Enable sync pulse detection interrupt
133         INTCONbits.INTE = 1;
134         INTCONbits.INTF = 0;
135     }
136
137 }
```

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS