Master's Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

Photographic camera device from the camera for computer vision

Bc. Tomáš Reinhold

Supervisor: prof. Ing. Vlastimil Havran, Ph.D.
Field of study: Open Informatics
Subfield: Computer Graphics
May 2024

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Reinhold** Jméno: **Tomáš** Osobní číslo: **485125**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Specializace: **Počítačová grafika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Fotografický přístroj z kamery pro počítačové vidění

Název diplomové práce anglicky:

Photographic camera device from the camera for computer vision

Pokyny pro vypracování:

Pro kameru určenou pro počítačové vidění firmy SVISTEK SHR411-CXGE se sensorem14192 x 10640 pixelů navrhněte zapojení celkovou koncepci přístroje, elektroniku s vestavným počítačem a vhodným způsobem ovládání tak, aby výsledná kombinace hardware a software byla použitelná jako fotografický přístroj. Výsledný přístroj musí umožňovat nastavení délky expozice, mít náhledový displej velikosti alespoň 3.5 palců a musí umožnit pomocí univerzální desky připojení velkoformátových objektivů různé ohniskové vzdálenosti a splňovat funkci jako fotografický přístroj. Kromě toho navrhněte další důležité prvky přístroje z hlediska uživatelského ovládání a použití v praxi jako je např. bezdotykové ovládání spouště či podporu pro manuální ostření podle obdélníku v obraze. Kritéria pro návrh přístroje popište detailně ve specifikace požadovaných vlastností přístroje a diskutujte je se zadavatelem.
Napájení přístroje může být vnitřní i vnější. Využijte platformu Raspberry Pi 4 či jinou vhodnou platformu vestavných mikropočítačů po dohodě s vedoucím práce. Zdroj pro provoz přístroje musí být formou akumulátoru LIFEPO4 či jiné formy lithiových akumulátorů s vysokou objemovou a hmotnostní hustotou energie. Tělo fotografického přístroje bude navrženo jako 3D model pro výrobu na 3D tiskárně.
Navržený fotografický přístroj vyprodukujte jako funkční vzorek s využitím 3D tiskárny včetně osazení elektroniky a realizace ovládacího software. Otestujte přístroj z hlediska uživatelského ovládaní nejméně na nezávislých třech uživatelích. Dále pak změřte další technické vlastnosti jako je rychlost snímání, výdrž akumulátoru v praxi, rychlost demontáže a montáže průmyslové kamery do přístroje, atd.
Na základě provedeného testování funkčního vzorku navrhněte změny hardware i software a následně produkujte finální prototyp, který otestujte alespoň na jednom uživateli.

Seznam doporučené literatury:

1) https://www.svs-vistek.com/en/industrial-cameras/svs-camera-detail.php?id=shr411CXGE, RGB sensor 53.36 x 40.01 mm
2) Josef Průša: Základy 3D tisku, 62 stran, Praha 2019.
3) Alex Burke: Choosing a 4x5 Camera and Lenses
https://www.alexburkephoto.com/blog/2018/2/13/choosing-a-4x5-camera-and-lenses

Jméno a pracoviště vedoucí(ho) diplomové práce:

prof. Ing. Vlastimil Havran, Ph.D. Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **15.02.2024** Termín odevzdání diplomové práce: **24.05.2024**

Platnost zadání diplomové práce: **21.09.2025**

prof. Ing. Vlastimil Havran, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.	
Datum převzetí zadání	Podpis studenta

Acknowledgements

Firstly, I would like to thank my supervisor prof. Ing. Vlastimil Havran, Ph.D., for his invaluable guidance and support throughout this project. His expertise and insights were crucial, especially while designing and working on the hardware part. I would also like to thank all the test participants who contributed to the final version. Finally, I would like to thank my family for supporting me during the studies.

Declaration

Abstract

This thesis focuses on designing and manufacturing a photographic camera device using a computer vision camera, the SVS-VISTEK SHR411CXGE, and a single-board computer. Both are enclosed into a 3D-printed body together with additional components such as a battery, display, and other elements typically found in photographic cameras. This combination brings state-of-the-art camera technology closer to the standard user. Based on preliminary analysis and performed tests, this idea is plausible and the currently best-fitting single-board computer for this application is Raspberry Pi 5 equipped with 5Gbit/s USB 3.0 to Ethernet adapter. For internal power supply, affordable 18 or 20 V power tool batteries can be used which provide more than 100 minutes of continuous operation. Usability testing revealed that the solution is functional and can be used as a photographic camera even by an inexperienced user.

Keywords: Raspberry Pi 5, SVS-VISTEK SHR411CXGE, 3D printing, photographic camera, computer vision camera

Supervisor: prof. Ing. Vlastimil Havran, Ph.D.

Abstrakt

Tato práce se zaměřuje na návrh a výrobu fotografického přístroje s použitím kamery pro počítačové vidění SVS-VISTEK SHR411CXGE a jednodeskového počítače. Obě zařízení jsou uzavřena do těla vytištěného na 3D tiskárně spolu s dalšími komponenty, jako jsou baterie, displej a další, které se běžně nacházejí na fotoaparátech. Tato kombinace přibližuje nejmodernější technologie kamer pro počítačové vidění běžnému uživateli. Na základě předběžné analýzy a provedených testů je tato kombinace možná a v současnosti je nejvhodnějším jednodeskovým počítačem pro tuto aplikaci Raspberry Pi 5 vybavený 5Gbit/s USB 3.0 na Ethernet adaptérem. Pro interní napájení lze použít cenově dostupné 18 nebo 20 V baterie do elektrického nářadí, které zajistí více než 100 minutový nepřetržitý provoz. Testování použitelnosti ukázalo, že řešení je funkční a může ho používat i nezkušený uživatel.

Klíčová slova: Raspberry Pi 5, SVS-VISTEK SHR411CXGE, 3D tisk, fotoaparát, kamera pro počítačové vidění

Překlad názvu: Fotografický přístroj z kamery pro počítačové vidění

Contents

Figures

Tables

Chapter 1
Introduction

Computer vision cameras typically incorporate state-of-the-art camera technology, but are not intended for photography. These cameras are controlled from a computer and do not provide common elements found on photographic cameras, such as a display, battery, or trigger button.

This thesis aims to bridge this gap between computer vision cameras and photographic cameras by selecting suitable hardware and implementing the necessary software. Ideally, the final solution could be used by someone who has no prior experience with computer vision cameras. The idea is based on a combination of a computer vision camera and a small single-board computer (SBC), both integrated into a single 3D-printed body.

To achieve this goal, we study the fundamentals of photography and analyze the different SBCs, electrical, and mechanical components that may contribute to the proposed solution. This is followed by the design and manufacturing of the first prototype, whose functionality is verified through usability testing. After usability testing, an improved final version is manufactured.

In the second chapter, we provide a brief overview of photography, its evolution, fundamentals, and currently used technologies. This will be especially useful in the upcoming chapters, where we mention different photography terms.

In the third chapter, we verify our idea of combining SBC with a computer vision camera, followed by an analysis of different SBCs for our specific use case. For this purpose, a simple testing application is also implemented.

In the fourth chapter, we state our hardware and software requirements for the camera. This will be helpful in the process of selecting individual components and designing the solution.

In the fifth chapter, we analyze and specify the hardware components, such as the display or battery, that will be used in the first prototype.

In the sixth chapter, we describe how we manufactured and assembled the first prototype.

In the seventh chapter, we focus on the usability testing of the first prototype and provide responses and suggestions for improvements from individual participants.

In the eighth chapter, we present the final prototype.

Chapter 2

Photography

The desire to capture reality is an ancient desire of a mankind. It all started with simple paintings that later evolved after understanding the principles of perspective. The first device resembling photography was a camera obscura, see Figure 2.1. It was a device that projected the first images of the world and can be understood as the predecessor of the camera, as we know it today. The principle of its projection is often referred to as a pinhole camera.

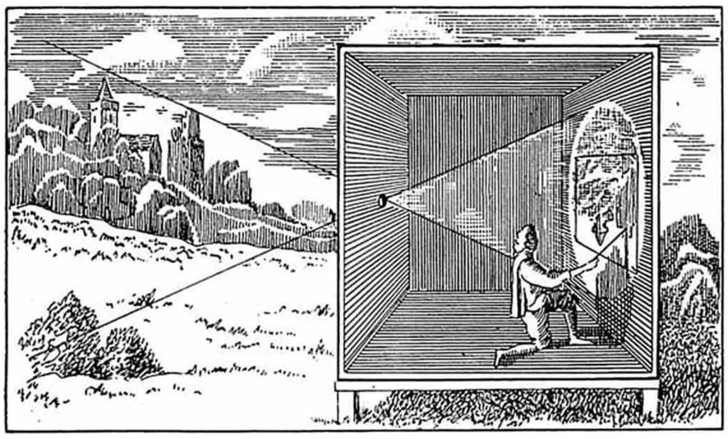


Figure 2.1: Camera obscura, image taken from [1]

To permanently preserve the projection, the painter had to capture it using paint. This changed dramatically in the 19th century when the process of preserving the image using light-sensitive material was discovered. This method is known as analog imaging and uses chemical processes to permanently preserve the captured image.

In the 20th century, the light-sensitive material was replaced with an image sensor, an array of electronic photodetectors that, in combination with an analog-to-digital converter, producing a discretized image. This method is known as digital imaging and is the method still used today. Summarized from [2, pp. 147–152] and [3, pp. 1–2].

Although technology has changed, the core principles remained the same for both methods. In the next section, we briefly explain the basics of the imaging chain.

2.1 Fundamentals

The inverted image is formed when light enters the camera through the lens, where it is focused on the image plane. Depending on the imaging method, the image sensor or light-sensitive material lies in the image plane. See the simplified image in Figure 2.2. From now on, we will use the image sensor, but it can be interchanged with the light-sensitive material.

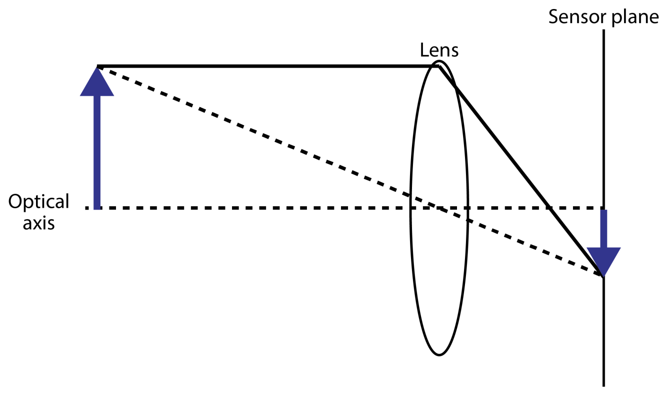


Figure 2.2: Simplified image formation

2.1.1 Camera lenses

Camera lenses are precisely manufactured optical devices that play a very important role in image acquisition. Modern lenses are much more complex than single lens and often consist of multiple elements [3, pp. 184–194]. The combination of elements are designed to reduce different aberrations, some are mentioned below.

Each lens is suitable for different applications, and there does not exist one that does it all. Photographers typically own a variety of lenses with different attributes. The most common categorizing for prime lenses (with fixed focal length, as opposed to zoom lenses) is done by focal length - fisheye, wide angle, standard, telephoto. Ranging from smallest to largest focal length. Different focal lengths allow for different viewing angles. When the viewing angle is equal to the size of the image sensor, the viewing angle is 90°.

Aperture

To control the amount of light that passes through the lens, an aperture is used. It is located inside the lens and physically blocks the incoming light by changing the diameter of the entrance pupil. An example of an iris-type aperture can be seen in Figure 2.3. However, in photography, aperture usually refers to the relative ratio with respect to the focal length, and not to the physical device itself. The f-number N is defined by Equation 2.1, where f is the focal length and d is the diameter of the entrance pupil.

$$N = \frac{f}{d} \tag{2.1}$$

The bigger the f-number, the smaller the diameter, thus reducing the amount of light that passes through. More information on apertures can be found in [2, pp. 160–163] or [3, p. 121].

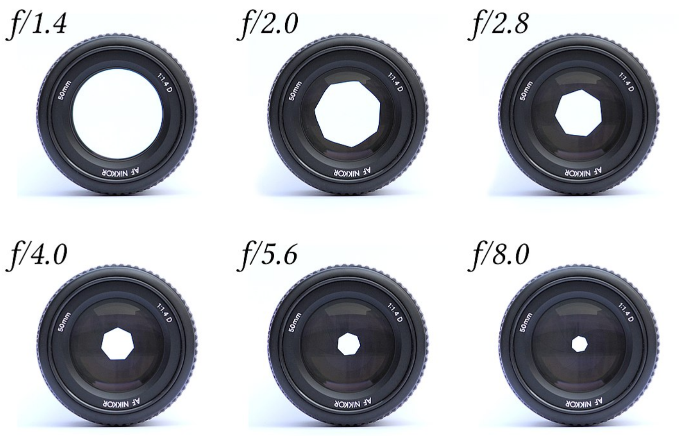


Figure 2.3: Iris aperature with changing f-number, image taken from [4]

Focal length

It is a distance, typically measured in millimeters, between the optical center of the lens and the focal point, the point at which the incoming parallel rays converge when passing through the lens. Together with sensor size, affects the field of view. The smaller the focal length, the wider the field of view, see Figure 2.4.

Depth of field

There is a point in the object scene that is optimally focused. Around this point is an area that appears to be acceptably focused. This area is called the depth of field (DOF). DOF is influenced by focal length, distance to the focused object, aperture, and sensor size [3, p. 5].

In Figure 2.5, we can see the starting point of the black lines around the object plane representing the optimally focused point. The starting points of green and blue lines around the object plane represent the edges of the area that is acceptably focused.

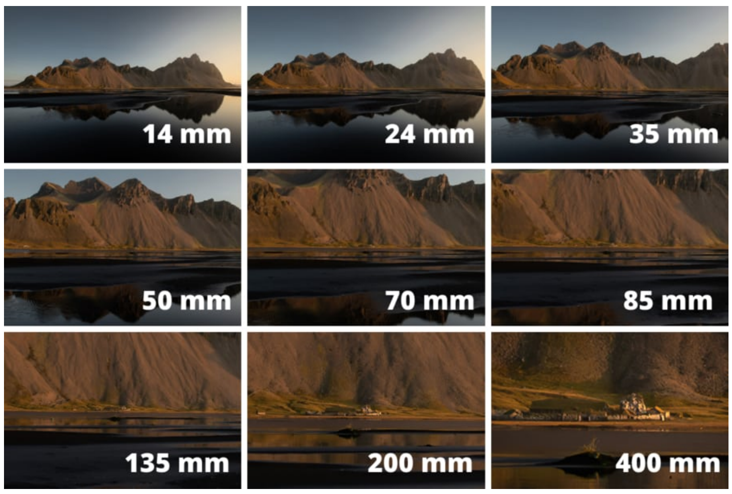


Figure 2.4: Different focal length and how it affects the image, image taken from [5]

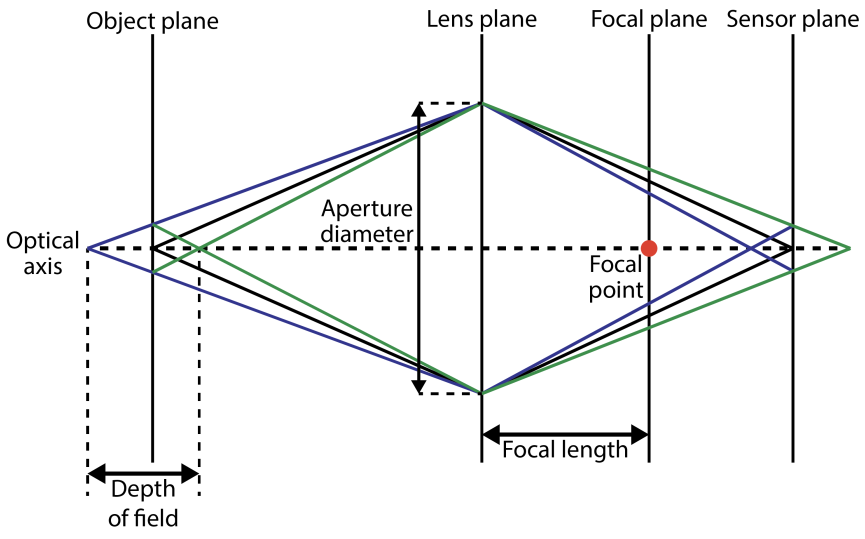


Figure 2.5: Depth of field

Aberrations

Despite the precision of manufacturing, lenses still have limitations that affect the resulting image. The most common aberrations are chromatic and spherical, see Figure 2.6.

Chromatic aberration is noticeable at edges, where the edge seems to split into individual colors. This is caused by air-glass refraction, which slightly depends on the wavelength of light passing through (Snell's law). The focus of blue light is closer to the lens than the focus of red light. Some lenses reduce or eliminate this problem by using two or more different glass elements, separated or connected together. These elements then cancel each other out.

Lenses are typically spherical. It is easy to manufacture, but it has its own problems. Spherical aberration is caused by the fact that refraction depends on the angle of incidence and refractive index of the elements. Therefore, parallel rays refract differently going from the center of the lens to the edge. Again, this can be overcome by using a special shape or two elements (convex and concave), which cancel this effect out. More details on lens abberations are given in [3, pp. 175–180].

2.1.2 Shutter

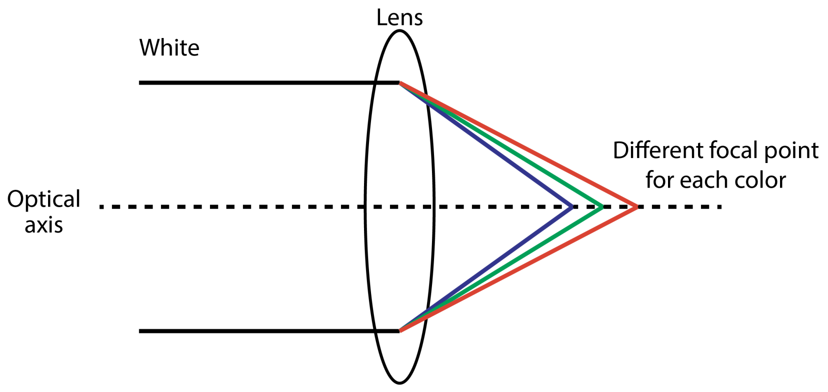
Until now, we have only discussed the reduction of incoming light using an aperture. There is also a second element, called the shutter, which, in contrast to the aperture, completely blocks incoming light. This is done either mechanically or electronically. Analog cameras implement a mechanical shutter. Smartphones and compact digital cameras usually implement the electronic approach. Some cameras, such as digital single-lens reflective cameras (DSLRs), implement both.

The mechanical shutter works by blocking the path to the image sensor. There exist multiple mechanisms that achieve this functionality (e.g., leaf shutter, focal-plane shutter, diaphragm shutter). The vertical focal-plane shutter will be described in more detail. The focal-plane shutter works by moving two so-called blinds (curtains) in the same direction with the desired pause between them. The first curtain starts to move down, revealing the image sensor from top to bottom. After some time, the second curtain starts to move from top to bottom, blocking the image sensor. In this way, each part of the image sensor is exposed for the same time.

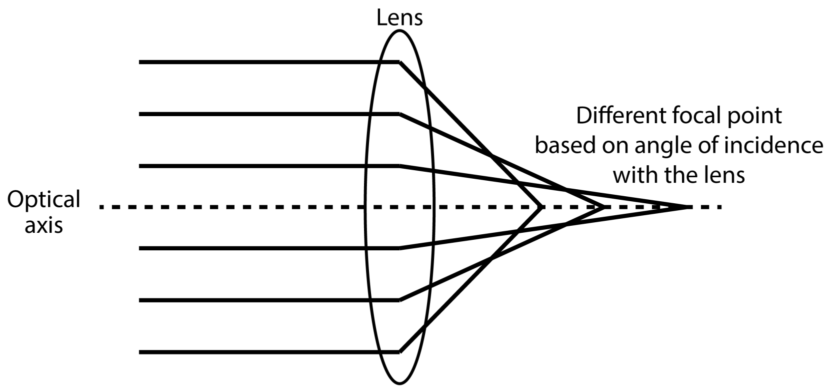
The electronic shutter functions by enabling one or more rows of the sensor at the time while reading their values. There is no mechanical blocking, just switching rows of the sensor on and off. This approach is much faster than the mechanical one. Some image sensors also implement a global shutter, which means that all the values are read at the same time.

Advantages of the electronic shutter over the mechanical:

- faster operation,
- silent,



(a) : Chromatic



(b) : Spherical

Figure 2.6: Lens abberations

- stability,
- less wear.

Disadvantages of the electronic shutter over the mechanical:

- rolling shutter (due to reading line by line, moving objects can be distorted),
- flickering (again due to reading line by line, light source that works by switching rapidly on and off will result in noticable stripes in the acquired image).

Exposure time

It represents the duration for which the shutter remains open during the exposure process. Cameras often call it shutter speed and represent it as a fraction of a second (e.g., exposure time of 500 ms would be 1/2 shutter speed). The bigger the exposure time, the brighter the image will be, but also blurrier, if capturing moving objects.

2.2 Digital imaging

In this section we focus more on digital cameras, mainly their image sensor, how they achieve acquisition of color images, and how we can store the acquired images.

2.2.1 Image sensors

We briefly describe the two predominant sensor types: charge-coupled device (CCD) and complementary metal oxide semiconductor (CMOS). Both exploit the photoelectric effect to capture light via semiconductors.

The CCD sensor is the older of the two. It was invented in 1969 and also got awarded with the Nobel prize in 2009. It is still used today, but in consumer electronics, CMOS has slowly replaced it. The sensor is made up of photosites (metal oxide semiconductor capacitors in this case) arranged in a grid. When these photosites are exposed to light, an electronic charge is generated proportional to the incoming light. The charge is then transferred from one pixel to the next one. In one place (either the whole row or column at the edge of the sensor), the charge is transferred off the sensor, amplified, and converted to a digital value using an analog-to-digital converter.

The CMOS sensor is widely used in smartphones and compact cameras, but also in DSLRs and more professional equipment. It is cheaper than the previous chip. The sensor is again made of photosites, but these photosites are more complicated. They consist of their own circuitry that perform amplifications, readouts, and resets. Alongside the photosites, there is a readout circuitry that switches between rows and columns. In this way, each photosite can be addressed. The values are then read out per the whole column or row. This amount of circuitry caused noise compared to CCD, but with development, this disadvantage slowly disappeared. The main advantage of CMOS over CCD is less heat generation, less power consumption, and cheaper production.

A more in-depth explanation of the two image sensors can be found in [3, pp. 155–171], from which we took inspiration for this text.

2.2.2 Color image

In the previous subsection, we have briefly explained how the image sensor works, mainly that it captures light intensity. To capture color, there are mainly two possible approaches. Both separate light into three components (red, green, and blue) [3, pp. 171–173].

The first approach is to use color filters. The selected color filter is placed somewhere in front of the image sensor. Therefore, the stored light intensity corresponds to the intensity of the color that was not filtered. By sequentially changing the filters, the color components can be obtained. To obtain the image, the color components are combined. This approach is not widely used mainly because sequential acquisition is suitable only for capturing static objects. However, a different type of color filter can be used, namely a color filter array (CFA), which filters colors per pixelsite rather than per whole area of the sensor. The CFA is located just above the image sensor. The most widely used CFA is the Bayer pattern. Bayer pattern, see Figure 2.7, prefers the green component over the other two (there are two times more pixelsites that capture green compared to red and blue), due to the peak sensitivity of the human eye to the green component. Some processing has to be done to estimate the pixel colors in between. This process is called demosaicing or debayering.

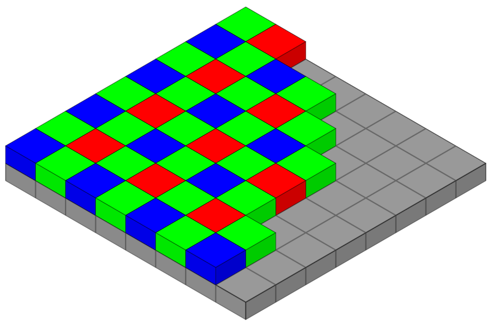


Figure 2.7: Bayer pattern, image taken from [6]

The other approach also uses color filters. However, to avoid sequential acquisition, instead, three image sensors are used. The incoming light from the lens is split by a beam-splitter prism. This approach using three CCD sensors can be seen in Figure 2.8.

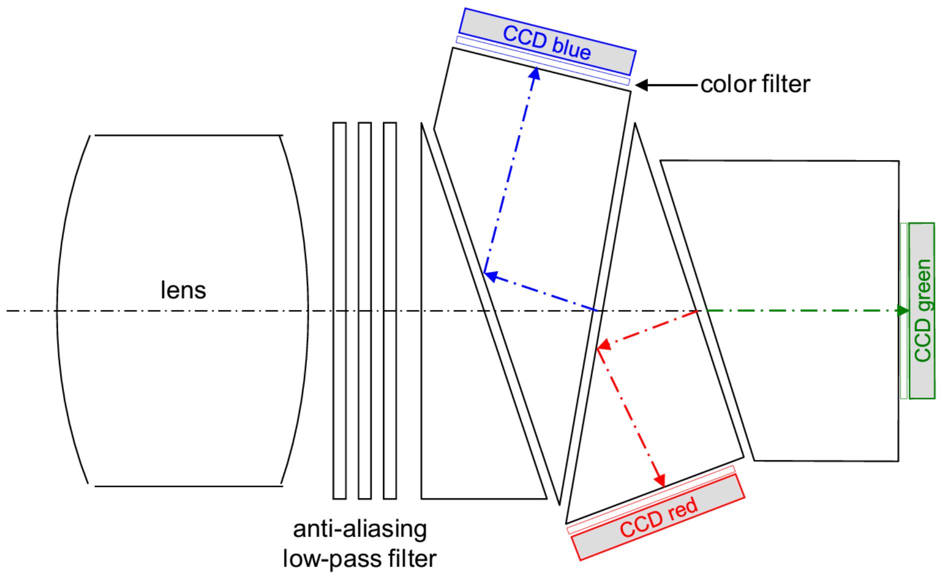


Figure 2.8: Three-sensor camera, image taken from [7, p. 177]

2.2.3 Image formats

Assuming that the camera temporarily stores the three components (color channels) in memory, the image data has to be stored permanently. Before exploring different image file formats, we need to mention bit depth and compression.

Bit depth

The bit depth represents all possible intensity values that one color channel can store in one pixel. When the image is 8-bit, that means 256 (2^8) possible values can be stored inside that pixel. Sometimes 8-bit is also referred to as 24-bit (because of 3 channels per 8-bit). The more bits per channel, the more accurate the captured color is. However, this comes at an image file size cost.

Compression

To overcome a large image file size, image compression is used. There are more compression methods available, usually bound to the specific file format. The image compression methods can be divided into lossless and lossy methods.

Lossless compression methods keep all the original values, usually exploiting the repeating values of the neighboring pixels. Examples of lossless methods are Huffman encoding or dictionary encoding such as LZW.

Lossy compression methods discard data to achieve a smaller file size. The discarded data are precisely selected so that the human visual perception of the image remains almost the same. An example of a lossy compression method is the discrete cosine transform, used in JPEG.

A nice explanation of the image compression methods mentioned above can be found in [8, pp. 61–71].

RAW

The simplest format of all is RAW. Manufacturers typically implement their own proprietary RAW format variants, but in essence, all of them store the unprocessed values that the sensor captured with additional metadata about the current acquisition settings. Some variants may also implement compression. The existence of this format is primarily for postproduction. Photographers can use different demosaicing algorithms or color presets to achieve the desired result. It provides great flexibility; however, this format is typically only intermediate, and the final image is converted to other formats.

PNG

PNG stands for portable network graphics. It is an open specification developed primarily for the world wide web use and as an intermediate format for image editing [9]. It utilizes the DEFLATE compression method, which is a combination of LZ77 (dictionary type) and Huffman encoding. Furthermore, it supports different filters that can be used before compression to optimize the data so that the compression is most effective. It also supports color depth of up to 16-bit per channel.

The file itself starts with a header followed by chunks. The chunks can be divided into two types, critical and ancillary. The critical chunks contain information needed to decode the image so that it can be shown to the user. The ancillary contains additional data that not all image viewers support.

JPEG

JPEG format is the most used in photography. Despite its use of lossy compression, it still produces perceptually lossless-looking images of small data size, if the compression rate is selected appropriately. It relies mainly on discarding color data rather than light intensity, which is more sensitive to the human eye. Standard JPEG supports only 8-bit color depth per channel. Since HDR imaging is on the rise, improved versions of JPEG, such as JPEG-HDR, are necessary. A nice overview can be found in [10].

Chapter 3

Preliminary analysis

In this chapter, we analyze whether the idea of using a single-board computer with a computer vision camera is possible. First, we introduce the camera, followed by possible SBC options. Next, we develop a simple testing application that will test the capabilities of each single-board computer. Based on the results, we select one SBC that will be further used.

3.1 SVS-VISTEK SHR411CXGE

It is an industrial camera for the computer vision application that will be used, as stated in the assignment. It utilizes a large-format sensor camera with an outstanding resolution of 151 Megapixels. Such a resolution comes at a cost - data transfer limitations and following image processing. The camera can be seen in Figure 3.1. For reference, the dimensions of the body are approximately 80 mm for each side.

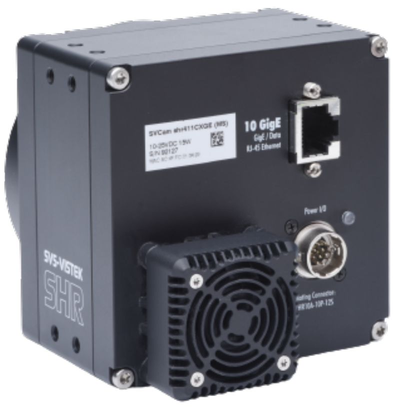


Figure 3.1: SVS-VISTEK SHR411CXGE, image taken from [11]

The connection between the computer and the camera is established using 10 gigabit Ethernet. Such a high-speed standard is not yet implemented into many personal computers, nor is it implemented into single-board computers, which are discussed in Section 3.2. The key features of this camera are provided in the list below. The whole specification can be seen in Appendix B.

- Sensor type: CMOS
- Shutter: electronic
- Sensor size: 53.36×40.01 mm
- Resolution: 151 Mpx (14192×10640 px)
- Pixel size: 3.76×3.76 μm
- Frame rate: 6.1 fps
- Interface: 10GigE RJ-45
- Pixel format: bayer8, bayer12, bayer16
- Lens mount: $M72 \times 0.75$
- Power supply: 10–25 V (DC)
- Power consumption: 19 W

3.1.1 Software

To control the acquisition, the manufacturer provides a software package called SVCamKit. The package is provided for Windows (x64 and x86), Linux (AMD64, ARM, and x86), and macOS. Having such wide OS support will not be limiting when choosing hardware in later steps. The package consists of a software solution called SVCapture and a C++ API.

SVCapture is a complete application that provides control over the acquisition and its parameters. It mainly consists of 2 parts. One part is just for a preview and the other is for controls and parameters in a tree-like structure following the GenICam 3.0 standard developed by the European Machine Vision Association [12].

The API opens possibilities for more specific use cases, such as ours. Camera control can be integrated into the application so that we can change parameters, start acquisition and control the trigger from code. Changing parameters requires knowing the previously mentioned GenICam standard and how it was implemented for our specific camera model. The API comes with detailed documentation for SVS-VISTEK cameras. However, to find the exact capabilities of our model, we had to look in SVCapture.

3.2 Single-board computers

To operate the camera, we need to select the platform that will run our developed software. SBCs fit well into our solution because of their small size and low power consumption. However, there are many boards, and we need to select the ideal fit. For this reason, we provide a brief list of commonly used SBCs with their specifications.

3.2.1 Raspberry Pi 4B

This SBC was released on 24 June 2019. It features a Broadcom BCM2711 1.5 Ghz quad-core Cortex-A72 processor with up to 8 GB of RAM (1, 2, 4, 8 available). It has wide connectivity options with 2x USB 2.0 ports, 2x USB 3.0 ports, gigabit Ethernet, Bluetooth 5.0, and Wi-Fi. The main storage provides a MicroSD card slot, which is also used to load the operating system. It is powered either via a USB-C connector, GPIO header, or PoE (with a specific HAT). It requires a 5 V/3 A DC power supply. The complete specifications can be found at [13].

The official supported operating system is the Raspberry Pi OS (formerly Raspbian), which is based on Debian, as the former name suggests. It is easily installed using their tool called the Raspberry Pi Imager. Other popular Linux distributions are also available, e.g. Ubuntu, Armbian.

There is also a system on board (SOM) version similar to this board called Compute Module 4 [14]. The main difference is that the Compute Module is integrated into a larger electronic system, whereas the Raspberry Pi 4B is a standalone computer. SOMs in general are more suited for embedded applications, allowing their users to specify the needed interface for their intended function and choose or develop relevant base-boards. This approach is used mainly in large productions, as it minimizes size, cost, and power consumption [15]. The difference between the two can be seen in Figure 3.2.

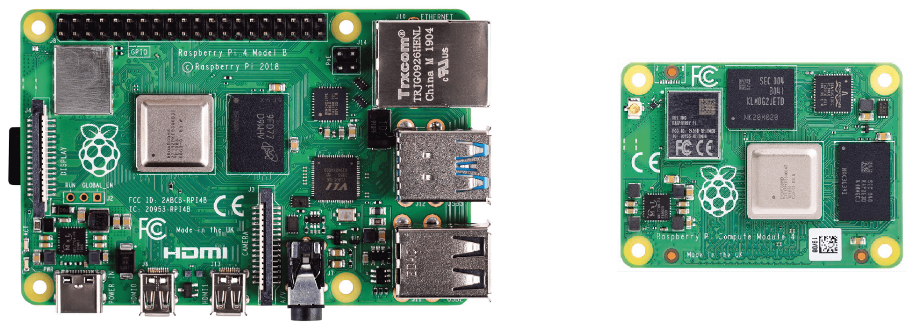


Figure 3.2: Raspberry Pi 4B (left) and Compute Module 4 (right), images taken from [13] and [16] respectively

3.2.2 Raspberry Pi 5

This SBC was released while working on this project, exactly on 23 October 2023, making it the newest board of all listed. It is the successor to the popular model 4B, described in Section 3.2.1. It comes with an upgraded Broadcom BCM2712 2.4 GHz quad-core 64-bit Cortex-A76 CPU with 4 or 8 GB of RAM. Connectivity remained very similar, 2x USB 2.0 ports, 2x USB 3.0 ports, gigabit Ethernet, Bluetooth 5.0, and Wi-Fi. However, unlike its predecessor, it provides a PCI-E interface that, in combination with a specific additional board, allows for more fast-speed components, such as an NVMe drive. Its default storage remained the MicroSD card, but the NVMe drive can be used to load the operating system as well [17]. The upgrade in performance led to increased power usage. Therefore, this model requires 5 V/5 A and is again powered through either a USB-C connector, GPIO header, or PoE (with a specific HAT). The complete specification can be found at [18].

Similarly to model 4B, its official supported operating system is the Raspberry Pi OS. Despite its recent release, other Linux distributions are also available, e.g. Ubuntu, Armbian, Kali.

At the time of writing, the SOM version of this model has not yet been announced. The standard model can be seen in Figure 3.3.

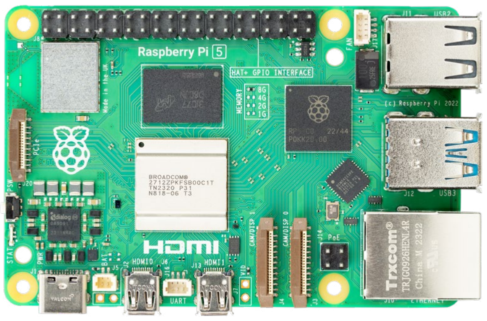


Figure 3.3: Raspberry Pi 5, image taken from [18]

3.2.3 Banana Pi BPI-M5

This SBC was announced in 2020 but became widely available in 2022. It comes with an Amlogic S905X3 2.0 GHz quad-core 64-bit Cortex-A55 CPU with 4 GB of RAM. Connectivity is provided by 4x USB 3.0 ports and a gigabit Ethernet. Additional options, such as Bluetooth or Wi-Fi, are available only using additional boards. It features an onboard eMMC with 16 GB of storage and a MicroSD slot. It requires a 5 V/3 A DC power supply with a USB-C connector. The SBC can be seen in Figure 3.4.

Complete specifications, tutorials, and operating systems are available on the Banana Pi wiki [19]. The official supported operating systems are Android and Linux (Ubuntu and Debian), whose ISO images are available for download.

3.2.4 NanoPi M4V2

This SBC was released in September 2019. It comes with Rockchip RK3399 which combines two CPUs, 2.0 GHz dual-core Cortex-A72 and 1.5 GHz quad-core Cortex-A53, and 4 GB of RAM. Connectivity is provided by 4x USB 3.0 ports, gigabit Ethernet, Bluetooth 4.1, and Wi-Fi. For storage, it is equipped with a MicroSD card slot and an eMMC socket. Optionally, the

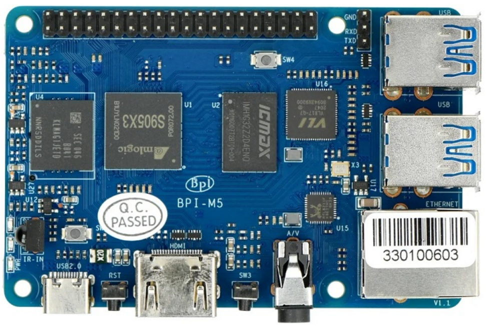


Figure 3.4: Banana Pi BPI-M5, image taken from [19]

official NVMe HAT can be connected to the PCI-E interface, so the SSD can be used for storage and booting. It requires a 5 V/3 A DC power supply with a USB-C connector. The SBC can be seen in Figure 3.5.

The official supported operating systems are Linux (Ubuntu, Debian, and a few more) and Android. They are available on the Friendly ELEC wiki [20] as ISO images, together with complete specifications and tutorials.

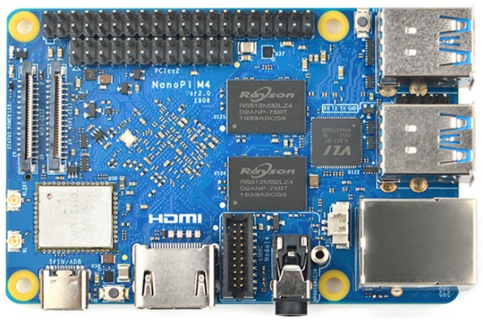


Figure 3.5: NanoPi M4V2, image taken from [21]

3.2.5 Khadas VIM4

This SBC was released on 10 May 2022, making it the second newest on the list. It comes with two CPUs, 2.2 GHz quad-core Cortex-A73 and 2.0 GHz quad-core Cortex-A53 CPU with 8 GB of RAM. Connectivity is provided by only 1x USB 3.0 and 1x USB 2.0, Bluetooth 5.1 and Wi-Fi. For storage, it is equipped with a MicroSD card slot and on-board 32 GB eMMC. Optionally, the official NVMe HAT can be connected to the PCI-E interface on the back side of the board. However, the SSD can only be used for storage at the moment. This SBC has two revisions, the only difference being in the neural processing unit (NPU), which is irrelevant to us. It requires a 12 V/2 A DC power supply with a USB-C connector or a 9-20 V DC power supply using the VIN port. The complete specification can be found at [22]. The SBC can be seen in Figure 3.6.

Installation of the operating system is easily done using the OOWOW embedded service, which requires connecting the board to the Internet. After that, the user selects the desired OS and it is automatically downloaded and installed. Android and Linux (Ubuntu, Debian, Armbian, and some others) are available.

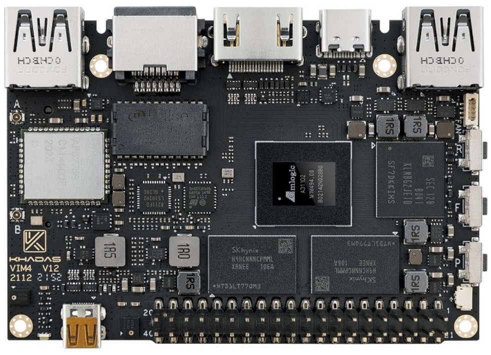


Figure 3.6: Khadas VIM4, image taken from [23]

3.2.6 ODROID-XU4

This SBC was released in 2015, making it the oldest on the list. It comes with Samsung Exynos5422 which combines two CPUs, 2 GHz quad-core Cortex-A15 and 1.3 GHz quad-core Cortex-A7, and 2 GB of RAM. Connectivity is provided by 2x USB 3.0 ports, 1x USB 2.0 port and gigabit Ethernet. Bluetooth is available by using an additional module. For storage, it is equipped with a microSD card slot and an eMMC socket. It requires a 5 V/4 A DC power supply with a barrel jack connector. The SBC can be seen in Figure 3.6.

The complete specification, tutorials, and OS images are available on the ODROID Wiki [24]. The official supported operating systems are Linux (Ubuntu) and Android. However, many other 3rd party options are available.

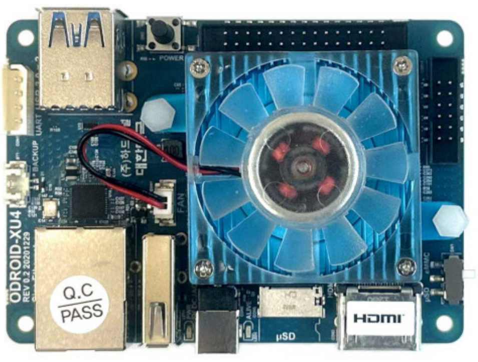


Figure 3.7: ODROID-XU4, image taken from [25]

3.2.7 Overview

All SBCs listed above were selected for additional testing. Namely Raspberry Pi 4B and 5, Banana Pi BPI-M5, NanoPi M4V2, KHADAS VIM4, and

ODROID-XU4. For easier comparison, we provide a table of hardware specifications, see Table 3.1.

Specification	RPi 4B	RPi 5	BPI-M5	M4V2	VIM4	XU4
Architecture	Arm64	Arm64	Arm64	Arm64	Arm64	Arm32
# cores	4	4	4	6 (2+4)	8 (4+4)	8 (4+4)
# threads	4	4	4	6	8	8
Freq. [GHz]	1.5	2.4	2.0	2.0, 1.5	2.2, 2.0	2.0, 1.3
RAM [GB]	8	8	4	4	8	2
# USB 3.0	2	2	4	4	1	2
PCI-E slot	N/A	2.0 x1	N/A	2.0 x1	2.0 x1	N/A
eMMC [GB]	N/A	N/A	16	socket	32	socket
PS voltage [V]	5	5	5	5	12	5
PS current [A]	3	5	3	3	2	4

Table 3.1: Single-board computers hardware overview

3.2.8 USB 3.0 to Ethernet Network Adapter

The camera used in our solution comes with 10 Gbit/s Ethernet and since all listed SBCs have only 1 Gbit/s Ethernet, we had to consider using a USB 3.0 adapter to increase the data transfer rate from the camera to the SBC. The USB 3.0 has transmission speeds of up to 5 Gbit/s, so our choice is either 2.5 Gbit/s or 5 Gbit/s. An adapter is also needed to change the maximum transmission unit (MTU) size. The camera manufacturer recommends jumbo frames of size 9000 bytes, and not all the onboard Ethernet controllers are capable of such an MTU size.

At first, we wanted to use the 5 Gbit/s adapter, however, many of the existing products are discontinued without any new releases. A nice overview of such adapters can be seen in [26]. All of these adapters are based on the Aquantia AQC111U chipset and should be compatible with both Linux and Windows. The only adapter available that we might get later is QNAP QNA-UC5G1T [27].

In the end, we used a 2.5 Gbit/s adapter Axagon ADE-25R USB-A [28], which is based on the Realtek RTL8156B chipset. There are more manufacturers making adapters that use this chipset. We ended up using this adapter temporarily, ideally upgrading to the 5 Gbit/s adapter in the final design.

3.2.9 Other

There are also other SBC options worth mentioning that were not included in our candidate list due to unavailability, excessive performance, or large format.

The first being Nvidia with its Jetson series [29]. For example, Nvidia Jetson Orin Nano or Jetson Nano modules with relevant board are powerful machines of small sizes but focus primarily on AI computing and might be excessive for our intended application. They also have higher power consumption needs.

The other viable option is Orange Pi 5 plus [30], which incorporates two 2.5 Gbit/s Ethernet on board but comes in a larger format. At the time of writing, this board was still unavailable on the market.

If we were forced to use a different architecture than ARM, there are also available x86 SBCs, such as LattePanda Sigma [31]. These are very powerful units, but they are either too big or have high power consumption compared to ARM SBCs.

3.3 Testing application

In this section, we introduce our testing application and its main components. This application will serve as a basis for the final application that will enable the user to control the camera. At this point, the application does not provide any acquisition control, other than switching between pixel bit depth and saving the next incoming image.

3.3.1 GUI

The GUI was implemented using Dear ImGui [32], which is a lightweight C++ library. Using this library, we were able to quickly implement some control buttons and live preview, which can be seen in Figure 3.8. This library will also be used in the final solution.

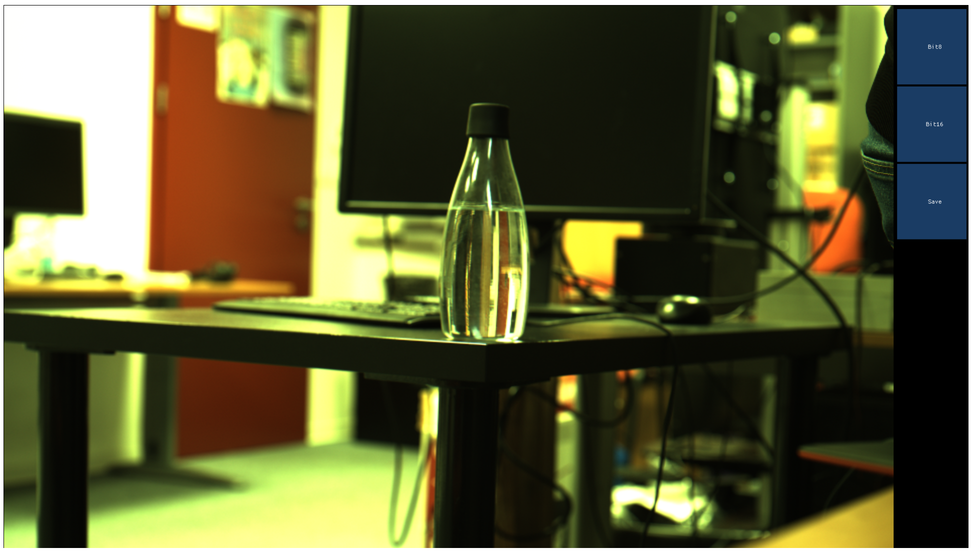


Figure 3.8: Camera preview

3.3.2 Camera control

To control the camera, we used the provided API library. The library implements functionality that handles camera detection, connection, acquisition control, and a few image processing functions, such as saving the image in BMP or PNG format. During the library installation, we had some problems, which were resolved after a bad filepath was found in the provided installation configuration file.

3.3.3 Image processing

After we acquire the raw image in Bayer pixel format from the camera, we need to perform some processing to show a live preview or save the image. For live preview, we downscale the image using pixel binning and convert it to RGB using demosaicing. To save the image, we perform demosaicing and save the image data in the desired image format, currently only supported format is PNG.

Demosaicing

To update or save the image, we need the RGB pixel format. The conversion from Bayer pixel format (or any other color filter) to RGB pixel format is called demosaicing, which was already mentioned in Subsection 2.2.2. Since each pixel of the image sensor captures only one color, the remaining colors must be interpolated from neighboring pixels. We used the implementation from libdc1394 library available at [33].

Pixel binning

It is a technique that averages the values of the corresponding pixels together, resulting in so-called superpixels. This effectively reduces noise at the sacrifice of spatial resolution. In our solution, this sacrifice is beneficial, since the demosaicing algorithm would complete faster. See Figure 3.9, where 2×2 pixel binning is performed.

We implemented 8×8 pixel binning using ARM NEON SIMD instructions. After the pixel binning, the original image of size 14192×10640 is downscaled to 1774×1330. This resolution, mainly its width, had some problems with demosaicing. Demosaicing works with a specific grid that was suddenly not compatible and resulted in shifted image rows. To overcome this issue, we ignore the first 16 pixels of each row in the original image row, resulting in a resolution of 1772×1330.

Image saving

We currently save the images in PNG format with zero compression using the libpng library [34]. We are able to save both 8-bit and 16-bit pixel depth variants. However, this process is rather slow. Therefore, we consider using different libraries or implementing other image formats.

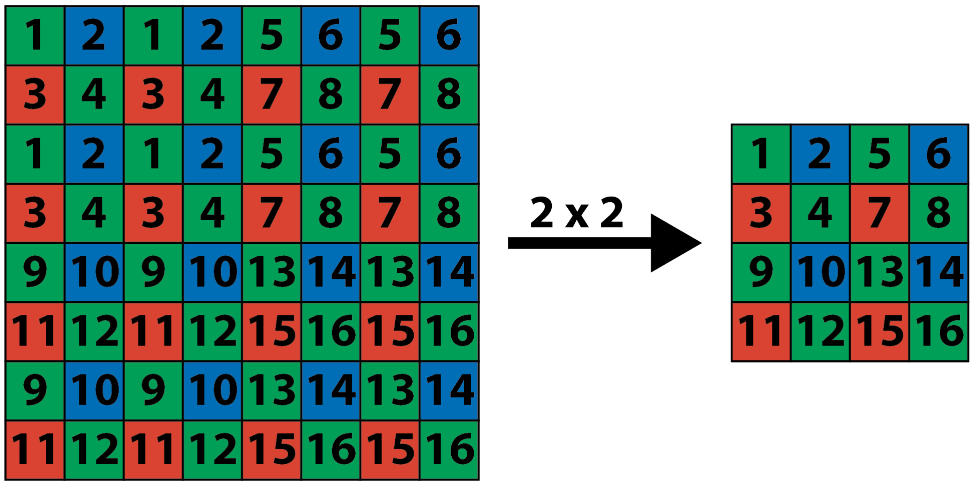


Figure 3.9: Pixel binning 2×2

3.4 Testing

In this section, we describe the tests that were performed on selected SBCs using our developed testing application 3.3. We also choose the final SBC that will be used in our solution and the reasoning behind the choice. To ensure stability in the measured properties, we ran every test 5 times and took an average of the values. Figure 3.10 shows our testing setup and Table 3.4 shows the final results. For complete results from individual runs, see Appendix C.

3.4.1 Prerequisites

Before individual testing, we had to install the OS and our testing application with its dependencies on each board. After that, we had to make sure, that the USB 3.0 to Ethernet adapter was using the correct driver (Realtek r8152) with the required transmission speed. However, there were some problems during this step, which are described below.

Faced issues

We tried to install only supported operating systems on each board. This was not fully successful with NanoPi M4V2, Raspberry Pi 4B, and unsuccessful with ODROID-XU4. The operating systems installed can be seen in Table 3.2.

On NanoPi M4V2 we successfully installed Ubuntu 20.04 on eMMC, however, the driver for the Ethernet adapter was not working at all. We tried updating the driver, and after failure, we even tried installing Debian 11 Desktop, but the driver was still not working. So we moved to the third-party Armbian 23.11.1, which resulted in a working driver.

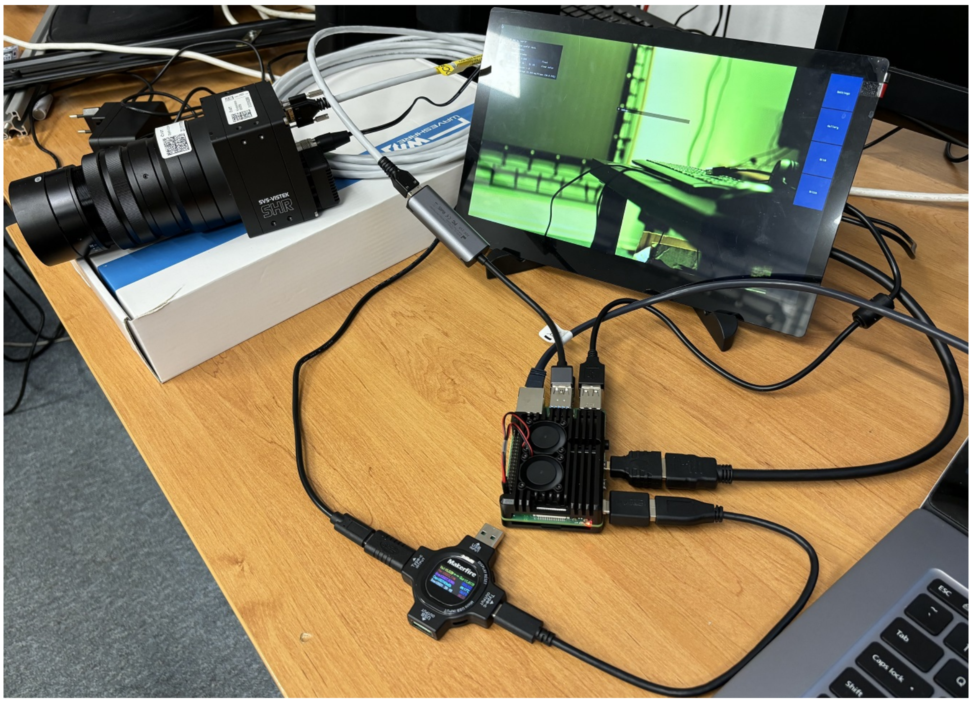


Figure 3.10: Testing setup

Raspberry Pi 4B had a problem with the driver loading at boot. The driver started working correctly only after the Ethernet adapter was removed and plugged back in. This problem occurred only with the Raspberry Pi OS, not with Ubuntu 22.04. However, testing revealed that Ubuntu performs worse than the Raspberry Pi OS in the case of this SBC. The measured values for Ubuntu can be seen in Table C.1.

With ODROID-XU4, we had problems installing Ubuntu 20.04 and 22.04. We tried both eMMC and microSD card as boot devices, however, the boot always got stuck. There is a known procedure that must be followed before installing these Ubuntu versions, which we tried but had no success. Therefore, due to limited time and this SBC having only 2 GB of RAM, we removed it from testing.

SBC	Operating system
RPi 4B	Raspberry Pi OS (64-bit) 2023-10-10
RPi 5	Raspberry Pi OS (64-bit) 2023-12-06
BPI-M5	Ubuntu 20.04 MATE (64-bit)
M4V2	Armbian 23.11.1 (64-bit)
VIM4	Ubuntu 20.04 Gnome (64-bit)
XU4	N/A

Table 3.2: Installed operating systems on SBCs

3.4.2 Power consumption

Since we want our solution to be portable, we aim for the lowest possible power consumption. To measure consumption, we used the Makerfire N03 USB tester [35] which was inserted between the power supply and the SBC. During the measuring, each SBC had an active cooling attached to it, which was connected to the board. The measuring device can be seen in Figure 3.11. We measured the following properties:

- Startup power consumption, the peak power drawn during operating system startup.
- Idle power consumption, the peak power drawn while idle on the desktop, measured for 1 minute after startup.
- For 8-bit and 16-bit images separately:
  - preview peak power consumption, measured for 1 minute,
  - preview average power consumption, measured for 2 minutes,
  - saving peak power consumption.

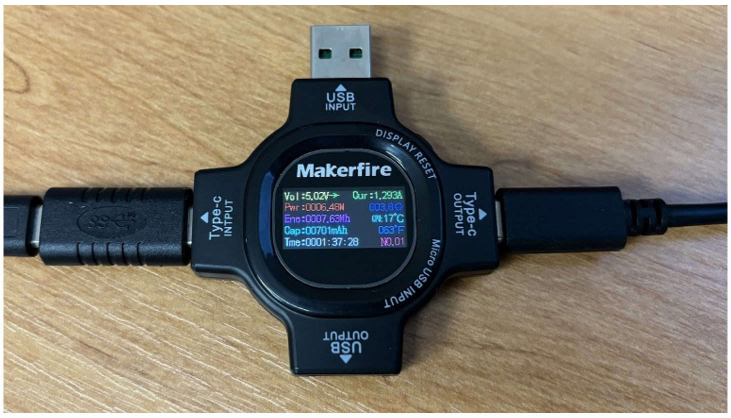


Figure 3.11: USB testing tool Makerfire N03

3.4.3 Preview processing

During the image preview, our application acquires the image data from the camera to a buffer. After that, it downscales the image using pixel binning, and performs demosaicing from Bayer pixel format to RGB so that the preview image is prepared for the texture update. To compare how quickly each SBC is, we measured the following properties for 8-bit and 16-bit pixel depths. We measured 50 of such cycles and calculated the average.

- Image processing, the time required for pixel binning and demosaicing, represents the delay our application has, the time between acquiring the image and showing it on display (note that the time required to update the texture is excluded).

- Frame time, the time between each update of the preview image. Note that camera features were always the same. Mainly exposure time, which was set manually.

3.4.4 Saving images

The other very important aspect to consider is image saving. Image saving depends on the compute performance and write speeds. We have tried multiple storage devices. The saving time includes the duration of the "fsync" call. Again, we measured separately for the 8-bit and 16-bit pixel depth.

- MicroSD, the card used for the test was Kingston SDCIT2/32GB [36] (measured speed: read 80 MiB/s, write 30 MiB/s).
- eMMC, either onboad eMMC or 16 GB module for NanoPi m4v2 [37].
- USB 3.0 SSD, the used external SSD was Verbatim 53234 [38] (measured speed: read 440 MiB/s, write 400 MiB/s).

3.4.5 CPU usage

We measured CPU usage to evaluate how demanding the various tasks are for the specific SBC. Measurement was carried out using the "top" program [39], which collected statistics approximately every 3 seconds. From each run, the average was calculated and normalized. The following tasks were measured for the 8-bit and 16-bit pixel depth.

- Preview, measured for 2 minutes.
- Saving.

3.4.6 Other

We also measured properties that are independent of our testing application. Namely:

- Boot time, the interval between providing power source and landing on desktop, measured manually using a stopwatch.
- Shutdown time, the interval between calling immediate shutdown and zero storage activity indicated by the LED, measured manually using a stopwatch.
- Write speed, measured time to copy a 1 GB file of random values to external SSD, including synchronization. From the measured time, the speed was calculated.
- Disk eject, the time required to eject the disk. It was performed after the write speed test.

3.5 Results

To select the ideal SBC for our solution, we took primarily power consumption, bootup/shutdown time, image processing, and saving into account. The other measurements were more informative (e.g. disk ejection, CPU usage) or very close.

We chose to use a ranking system. The average values were sorted and SBCs were assigned points (1-5). For preview average power consumption, we averaged the 8-bit and 16-bit variants.

The ranking results can be seen in Table 3.3. According to our tests, Raspberry Pi 5 is the best fit for our solution.

Measurement	RPi 4B	RPi 5	BPI-M5	NPi M4V2	VIM4
Boot	4	5	1	2	3
Shutdown	2	5	3	2	4
Power consumption					
Preview avg.	4	3	5	1	2
Saving	4	3	5	1	2
Preview processing					
Image proc. 8-bit	3	5	1	2	4
Image proc. 16-bit	3	5	1	2	4
Saving images					
Saving 8-bit ex. SSD	4	5	1	2	3
Saving 16-bit ex. SSD	3	5	1	2	4
Total points	27	36	18	14	26

Table 3.3: Ranking of SBCs

3.5.1 Alternative storage devices

We also tested the performance of RasPiKey [40], which is an eMMC module that plugs into the microSD card slot. The tests were carried out on Raspberry Pi 4B (model 5 was not supported). The gain in speed performance was insignificant. The comparison with the MicroSD card can be seen in Table 3.5. The tests were performed on identical 64-bit Raspberry Pi OS.

For Khadas VIM4, we tested the NVMe M.2 SSD drive using their official additional board. The SBC can use this drive only for storage, so we compared it with the external SSD used in previous tests. The results can be seen in Table 3.6. In this case, saving is not limited by the speed of the storage device, but by the computational speed.

We also intended to test NVMe M.2 SSD on Raspberry Pi 5, however, the ordered additional board arrived after the tests were performed.

Measurement	RPi 4B	RPi 5	BPI-M5	NPi M4V2	VIM4
Boot [s]	36	21	39	38	37
Shutdown [s]	14	2	9	14	3
Write speed [MiB/s]	216	251	211	223	209
Disk eject [ms]	439	78	388	407	277
Power consumption [W]					
Startup	5.91	6.32	5.27	8.97	7.15
Idle	4.99	4.63	3.81	5.11	4.88
Preview 8-bit	7.67	8.16	5.62	12.63	9.36
Preview 16-bit	7.77	8.63	5.68	12.50	9.45
Preview 8-bit avg.	6.90	7.20	5.10	10.38	8.64
Preview 16-bit avg.	6.90	7.26	5.40	10.20	8.70
Saving 8-bit	7.43	7.66	5.79	12.68	9.57
Saving 16-bit	7.60	8.63	5.76	13.00	9.72
Preview processing [ms]					
Image proc. 8-bit	76	42	289	114	51
Image proc. 16-bit	125	61	161	134	67
Frame time 8-bit	726	709	748	732	793
Frame time 16-bit	1611	1582	1669	1620	1728
Saving images [s]					
Saving 8-bit microSD	24.53	11.86	35.34	27.28	19.64
Saving 16-bit microSD	38.70	21.66	68.80	41.45	32.47
Saving 8-bit eMMC	N/A	N/A	34.89	16.56	14.82
Saving 16-bit eMMC	N/A	N/A	72.07	33.25	31.24
Saving 8-bit ex. SSD	14.40	7.53	33.69	16.38	14.77
Saving 16-bit ex. SSD	31.77	14.94	69.83	35.32	30.25
CPU usage [%]					
Preview 8-bit	48.90	45.10	85.40	38.60	63.10
Preview 16-bit	46.80	44.25	84.75	35.53	64.63
Saving 8-bit	51.45	52.70	86.30	38.43	71.98
Saving 16-bit	55.45	60.25	80.75	39.73	73.55

Table 3.4: Comparison of measured values in SBC testing

Measurement	RasPiKey	MicroSD
Boot [s]	37	37
Shutdown [s]	7	14
Saving images [s]		
Saving 8-bit	24.07	24.52
Saving 16-bit	39.07	38.70

Table 3.5: Comparison of RasPiKey to MicroSD card on Raspberry Pi 4B

Measurement	external SSD	NVMe SSD
Write speed [MiB/s]	210	300
Saving images [s]		
Saving 8-bit	14.77	15.12
Saving 16-bit	30.25	29.88

Table 3.6: Comparison of external SSD to NVMe M.2 SSD on Khadas VIM4

Chapter 4

Camera requirements

In this chapter, we state our requirements for the camera that we will be designing. We draw ideas from existing camera devices (e.g. Canon EOS 700D [41]) to provide the user with a familiar experience.

We first state all the software features that will determine which hardware we need to include with its specific requirements.

4.1 Software

The most important feature of the camera is the preview. It is typically realized using a combination of a display and a viewfinder. The preview helps the user see the composition, adjust the acquisition parameters, and achieve focus. For these reasons, a preview will be incorporated as it is an essential feature. To further improve the experience, a zoomed preview will also be implemented, which is especially useful for precise manual focus.

As stated in the previous paragraph, the user must have the ability to adjust the acquisition parameters. The experienced user may want to set all the parameters manually, whereas the inexperienced user may rely on the camera's automatically determined parameters. Considering computer vision camera functions, the ability to change exposure time, gain, and white balance will be implemented. Furthermore, switching between pixel bit depth and multiple-exposure burst will also be implemented.

The next common feature is the flash, which the user should be able to turn on and off and adjust its timing.

Different trigger behaviors will be implemented, such as a self-timer and a time-lapse. These are commonly found on conventional camera devices.

Multiple image file formats will be available for captured images, one of which will be PNG and one RAW.

Following image capture, metadata is typically stored alongside the image data, either embedded in the image format or as a separate file. The image metadata will store acquisition settings, time, and location on Earth (latitude, longitude and altitude). Location estimation using the global navigation satellite system (GNSS) is usually found on smartphones; however, some digital cameras also have this feature.

The user will be able to browse captured images, see the metadata, and manage them (delete or export, ideally in bulk). Export will be available to the external drive formatted in the ext4 file system.

The camera will be powered by both external and internal power sources. Therefore, when operating using the internal power source, the estimate of the remaining operating time should be visible.

To assist the user in composing the scene, there will be an option to calculate the depth of field, object detail, and object distance based on the user input.

There will also be system settings, allowing the user to change the date and time, format storage, or restart the application.

To summarize all the software requirements, we provide a list:

- camera preview with ability to zoom,
- changing acquisition settings:
  - exposure time (manual/auto),
  - gain (manual/auto),
  - white balance (manual/auto),
- multiple exposure burst,
- selectable pixel depth:
  - 8-bit,
  - 16-bit,
- flash,
- different trigger behaviour:
  - immediate,
  - self-timer,
  - time-lapse,
- selectable image file format:
  - PNG,
  - RAW,
- image metadata:
  - time,
  - acquisition settings,
  - location,
- browse images:
  - see metadata,

  - delete,
  - export,
- manual computing:
  - depth of field from object distance and detail size,
  - detail size from object distance and depth of field,
  - object distance from depth of field and detail size,
- system settings:
  - change date and time,
  - format the internal storage,
  - restart application,
- battery remaining time estimation.

4.2 Hardware

In this section, we focus on the hardware and its parameters. The camera and the SBC have already been chosen. Furthermore, we add more detailed requirements for these two and for other components.

The camera should be easy to disassemble from the construction, allowing us to use the camera for other purposes. The construction will have the option of mounting adapters for different lenses, using a universal mounting plate.

Due to the large image file sizes, the Raspberry Pi will have at least 1 TB of storage, utilizing the PCI-E interface which can theoretically reach 500 MB/s of throughput, so the drive read/write speed will exceed that. For this reason, the M.2 adapter board will be added. Communication with the camera will be done using a USB 3.0 to Ethernet adapter, either 2.5 Gbit/s or ideally 5 Gbit/s. To export data from the internal storage a USB 3.0 port will be used, so the user can plug in his own device. To preserve the date and time after shutdown, a real-time clock battery will be added. To maintain stable operation, an active or passive cooler will be added.

The display will be at least 5" and use AMOLED technology with high brightness. It will also be touch-sensitive, so the user can use it as an input device.

To capture an image, the user will have multiple options. The default hardware trigger button and a touchless sensor on the construction. Compared to the hardware trigger button, the touchless sensor should eliminate the camera shake when activated. There will also be a remote trigger option.

The power source will be both internal and external. The device should be able to maintain at least 90 minutes of operation using the internal power source. It will also disconnect itself from the power source when turned off to prevent undercharge.

The location data will be obtained using the GNSS module. The GNSS module should be either optional to mount or have the ability to be powered off to save energy when not needed.

Lastly, a USB audio adapter will be added since the Raspberry Pi 5 does not provide any audio interface. The speaker will be used to play some notification sounds and the microphone can be optionally used too.

To summarize all the hardware requirements, we provide a list:

- SVS-Vistek SHR411CXGE:
  - easy to disassemble from construction,
  - universal lens mounting plate,
- Raspberry Pi 5, 4 or 8 GB:
  - M.2 adapter with at least 1 TB SSD (500MB/s or faster),
  - 2.5 or 5 Gbit/s ethernet adapter,
  - USB 3.0 extension,
  - RTC battery backup,
  - cooler,
- 5" or larger AMOLED display with touch capabilities,
- trigger options:
  - physical,
  - touchless using IR sensor,
  - remote (available from at least 4 different sides),
- power source:
  - turning on/off,
  - battery:
    - 90+ minutes of continuous operation,
    - disconnected when device turned off,
  - external power source,
- disconnectable GNSS module,
- USB audio adapter.

Chapter 5

Analysis and proposed design

To fulfill all the requirements stated in the previous Chapter 4, we need to analyze and design different parts that will contribute to the proposed solution. First, we want to clarify the naming. The camera stands for the SVS-VISTEK SHR411CXGE, the camera casing for the 3D-printed body, and the box camera for the combination of both as a whole. We have already done some analysis in Chapter 3, mainly on SBCs. We have chosen to use the Raspberry Pi 5 to control the camera. In this chapter, we analyze the remaining electrical and mechanical components. Next, we examine possible layouts of the components. Based on the layout, we model the construction that will be manufactured using a 3D printer.

5.1 Components

In this section, we present and discuss all the components that will be used. For the battery and display, we also present more options and the justification why we chose that particular one. For the rest of the components, we either had them from previous projects or they were not that important for further investigation.

5.1.1 Raspberry Pi 5

The Raspberry Pi will be equipped with passive or active cooling. Initially, we will use passive cooling. If it proves insufficient, active cooling will be used instead in the final design.

To utilize the PCI-E interface for the NVMe drive, we will use an additional board, the Pineboards HatDrive! Bottom [42]. The chosen NVMe drive is the Lexar NM710 [43] with a storage capacity of 2 TB, selected based on recommendations and proven compatibility with the board.

To maintain the date and time after shutdown, we will use the Raspberry RTC battery (Panasonic ML-2020), which connects to the designated connector on the board.

Given that the Raspberry Pi 5 lacks an audio interface, we will use a USB sound card, the Axagon ADA-12 USB [44]. This will be paired with a 0.3 W amplifier module and a 2 W, 8 ohm, of 40 mm diameter speaker. Additionally, a small 10 mm microphone will be also connected to the sound card.

To improve the accessibility of the USB 3.0 and RJ-45 ports on the Raspberry Pi, we will use the extension cables.

We will also replace the 2.5 Gbit/s Ethernet adapter used in Subsection 3.2.8 with the 5Gbit/s QNAP QNA-UC5G1T, which was mentioned there.

5.1.2 Display

For display, we wanted a display of size between 5" and 6". During the analysis, we found two candidates, both from the Waveshare company. Other manufacturers did not provide AMOLED technology.

The first of the two considered is a Waveshare 5inch capacitive touch AMOLED display (SKU: 19299) [45]. The resolution of the display is 960×544 px, with a screen size of 110.44×63.02 mm. The display data are transferred via HDMI, while powered on using a micro USB.

The second option is a Waveshare 5.5inch capacitive touch AMOLED display (SKU: 16103) [46]. The resolution is higher 1920×1080 px, with a screen size of 121.76×68.70 mm. Similarly to the previous one, the display data are transferred via HDMI, while powered on using micro USB.

Both displays can be powered on and transfer touch data using the same connection, but they also support using two separate cables. To decide which one to use, we drew the dimensions of the correctly scaled preview window and looked at how much space would remain for other elements such as control buttons, see Figure 5.1.

The preview size and the remaining space were almost identical. Both results were satisfactory, so we picked the one with the higher resolution.

5.1.3 Battery

We estimated the power consumption of the box camera to be 32 W. The camera operates at 10-25 V while consuming around 19 W. The Raspberry Pi 5 operates at 5 V, while the measured consumption was around 8 W, based on the SBC testing in Section 3.4. The remaining 5 W was left for the display, a different USB to Ethernet adapter and other components that were not used while testing the power consumption. To satisfy our requirement of 90 minutes of continuous operation, we had to choose a battery with a capacity of at least 48 Wh.

Since both devices operate at different voltages, we decided to use a higher voltage battery and include a step-down adapter that can convert 10-25 V to 5 V/5 A. The exact model chosen is the XY-3606 Module, available at [47].

Concerning the battery, we were looking for an assembled battery pack of LiFePo4 cells. However, we were only able to find batteries with a large capacity of extremely heavy weights or small batteries with insufficient capacity. To use a LiFePo4 battery, we would have to make our own battery pack, probably 5 cells connected in series using a cell balancer. Each cell would have 3.2 V and at least 3000 mAh. In this way, we would assemble a 16 V

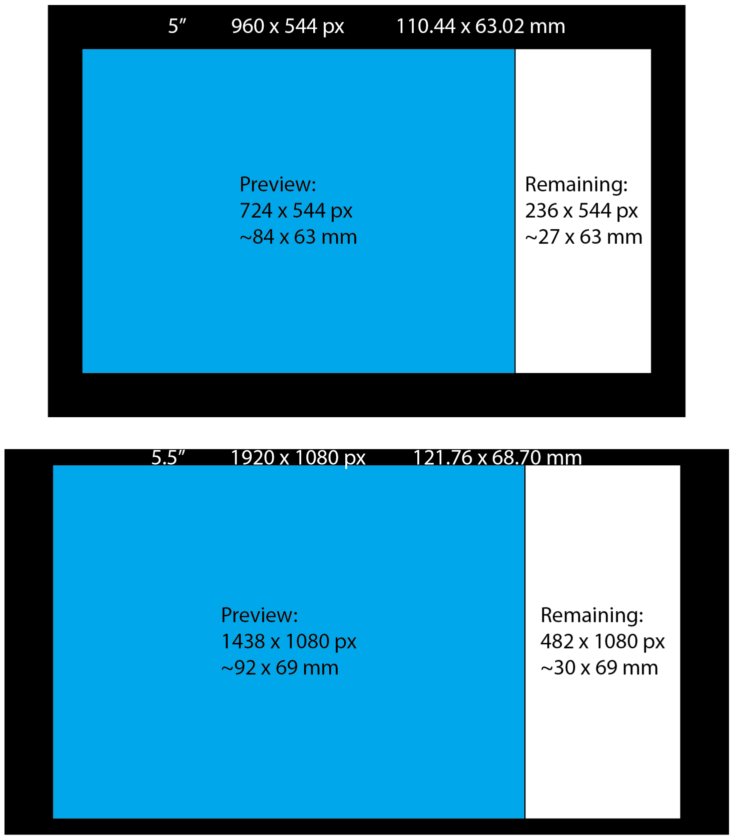


Figure 5.1: The comparison of 5" and 5.5" inch displays

battery with capacity of 48 Wh. We quickly discarded this idea, as it is not easily reproducible, and we would have to deal with designing a charger.

The next idea that seemed more reasonable and easily applicable was to use batteries from power tools used in workshops. The disadvantage of these batteries is the Li-Ion technology, which is older than LiFePo4. Here is a short comparison of the two technologies [48]:

- Li-Ion
  - higher energy density,
  - widely used in consumer electronics, therefore cheaper,
- LiFePo4
  - better thermal stability,
  - longer cycle life.

However, these power tool batteries are specially designed to be ejected from the tool and charged, which quietly suits our needs. They typically come in 12 V, 18 V, 20 V, or 40 V variants. The 12 V is right on the edge of the requirement of the camera (10-25 V), when discharged it might be even lower, so 12 V cannot be used. The 40 V exceeds the camera requirement, so it cannot be used either, or two step-down converters would have to be used. We were left with 18 V and 20 V batteries. Some manufacturers also use the 20 V label despite the battery being only 18 V. In the end, we were choosing between BOSCH ProCore 18V 4.0 Ah [49] and Parkside 20V 4.0 Ah PAP 20 B3 [50]. There are also other companies that produce similar batteries such as Makita, Ryobi, DeWALT, Einhell, and many more.

	BOSCH 18 V	Parkside 20 V
Capacity [Wh]	72	80
Weight [g]	520	750
Dimensions W x L x H [mm]	80 x 150 x 90	86 x 130 x 77
# cells	5	10
Price [EUR]	74	35
Cost per Wh [Price/Capacity]	1.03	0.44

Table 5.1: Battery comparison

For our later implementation, we chose the Parkside battery, primarily because of its accessibility. However, if we were aiming for the lowest weight possible, we would prefer BOSCH.

5.1.4 Power input board

To prevent the battery from undercharging, we designed a circuit to disconnect the battery from other components on a button press. The simplest solution would be to include a physical slide switch that disconnects the battery. However, the user would have to switch it every time.

We took it one step further and designed a circuit that starts the box camera on a button press and measures the voltage of the battery and external power source during runtime. The same button is then used to perform a soft shutdown when the box camera is powered on. We did not design it from scratch. Instead, we found a similar circuit [51] already designed and modified it to our needs. It utilizes MOSFET to switch common ground. When the Raspberry Pi is powered on, it maintains the circuit connection via the MOSFET. The circuit scheme can be seen in Figure 5.2.

5.1.5 Connector board

For easier debugging and an overview of all the components during assembly, we designed a connector board. Most components will plug into this board using KK254 2.54mm pitch PCB connectors of 2,3,4,5 or 6 pins. The board will then be connected to the 40 pin GPIO header on the Rasberry Pi. The board will also protect the Raspberry Pi since the resistors are positioned between connections. The scheme of the board can be seen in Figure 5.3.

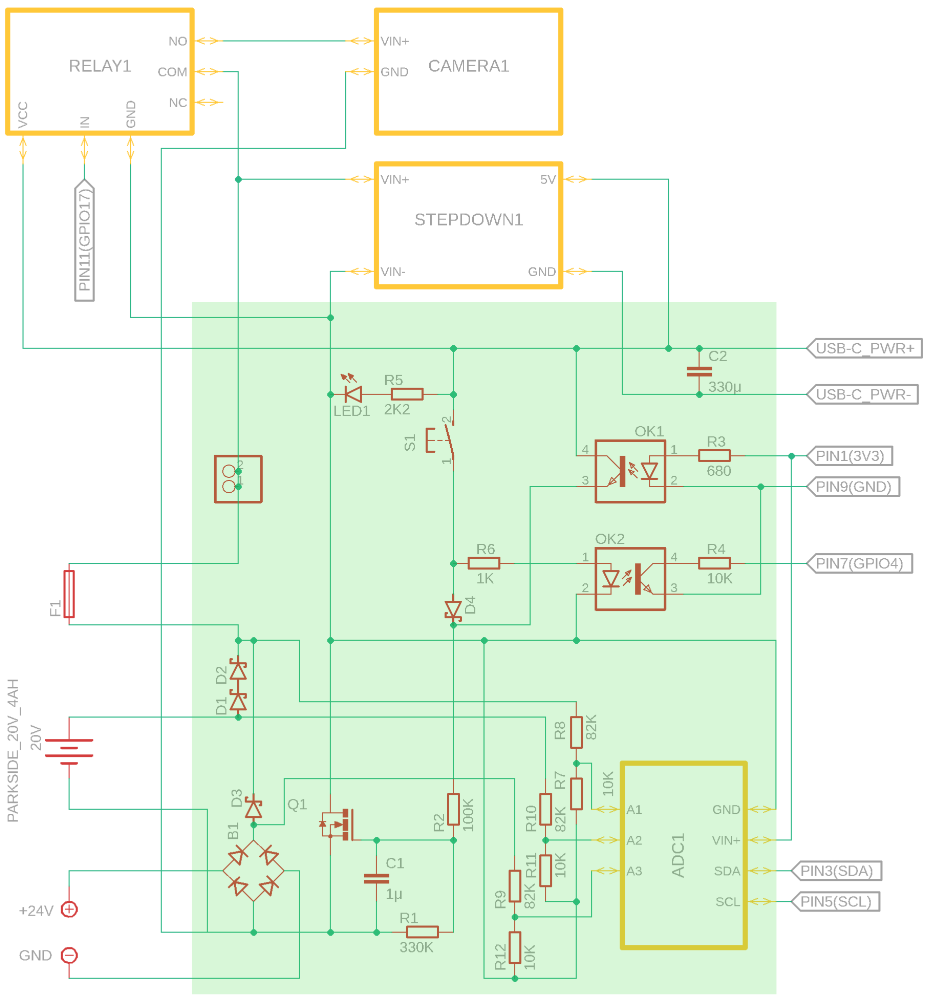


Figure 5.2: Power input board scheme

5.1.6 GNSS module

There are many compatible GNSS modules with the Raspberry Pi, we will use the EZ-0048 [52] based on the supervisor's recommendation. The only requirement for the module is that it can use UART for communication and a separate power supply rather than a USB since we would have to use a USB hub. The selected module is supplied with 5 V DC. This is potentially risky since Raspberry Pi 5 uses 3.3 V, however, we measured the output of the module and it never exceeded 3.3 V.

5.1.7 Input

In addition to the touch screen, we will add a rotary encoder to allow user settings changes. This will be especially useful for precise control, e.g. when setting precise values. It is also common among professional digital cameras.

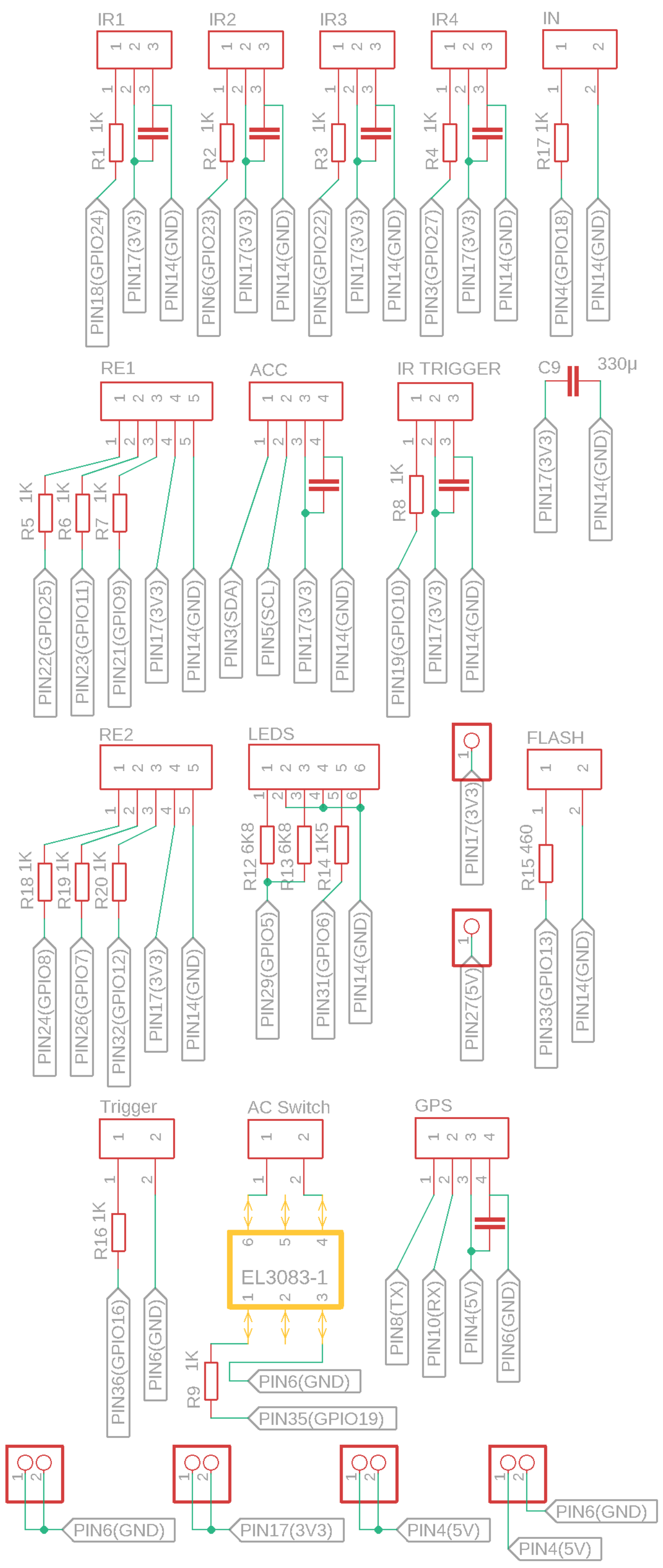


Figure 5.3: Connector board scheme

For the trigger, we will use a regular push button, an IR reflective sensor for the touchless trigger, and a Canon RC6 remote controller paired with four IR receivers. We will also add an accelerometer so that the camera trigger can be interrupted when the camera is moving.

5.1.8 Other

The external source will be connected to the camera with a DC Barrel Jack power connector. For the flash, we will use a pre-made flash mount assembly. The flash itself will be triggered using a relay.

To support various use cases of the camera, we will also add a 9-pin D-Sub connector that will expose 3.3 V, 5 V, 20 V (the power source), GPIO used for input on the Raspberry Pi, and camera input and output (see Figure 5.4). The PIN 4 is separated from the Raspberry Pi using an opto-isolator.

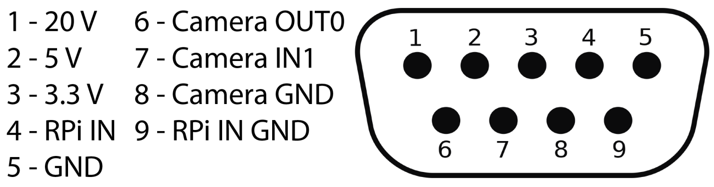


Figure 5.4: 9-pin D-Sub connector

5.2 Layout

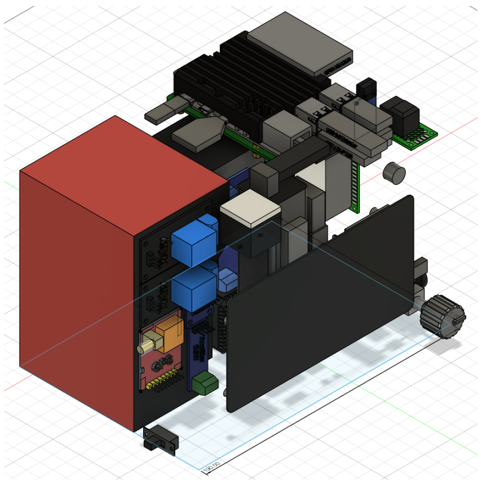
After selecting all the necessary components, we started creating component layouts. The first step was to model all the components in the 3D CAD software, for which we used Adobe Fusion [53]. The camera and display 3D models were provided by the manufacturers. Everything else we had to precisely measure and model (we still used some pre-modeled parts, e.g. the universal PCB). Of course, not every part of the component was modeled in detail, only important parts like mounting holes, longest dimensions, or connector sizes.

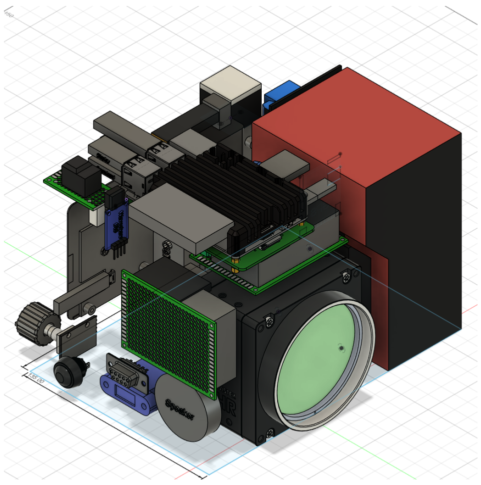
The biggest components ended up being the camera, battery, and Raspberry Pi (due to the size of the connectors). These mainly affect the dimensions of the camera casing and the placement of other components. We were also limited by the maximum print size of most common 3D printers, which does not exceed 200 × 200 × 200 mm, so we wanted to fit every printed part in these dimensions. Furthermore, we wanted to put the Raspberry Pi and camera right on the side or top wall, to allow better cooling. Due to many limitations, this task was quite challenging and had to be well thought out for later manufacturing. In the end, we came up with three poossible layouts to be evaluated.

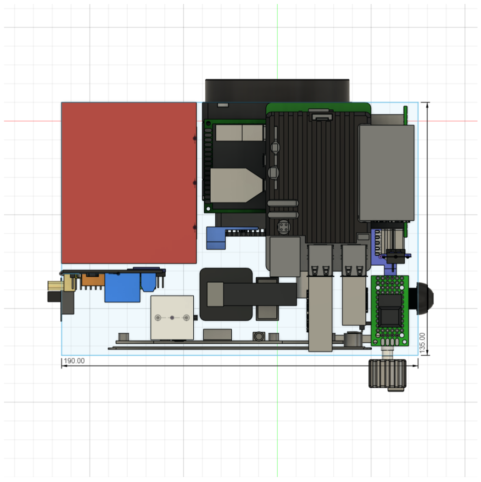
5.2.1 Layout 1

This layout was designed around the idea of positioning the battery vertically side by side with the camera. The Raspberry Pi is placed at the top, allowing good ventilation of warm air from the cooler. Most of the components are placed so that they can be mounted on the inner side of the outer walls or inner walls. The battery slides from the top inside the camera casing. The size of the layout is approximately 190 x 135 x 145 mm (3.72 liters). See the layout in Figure 5.5.

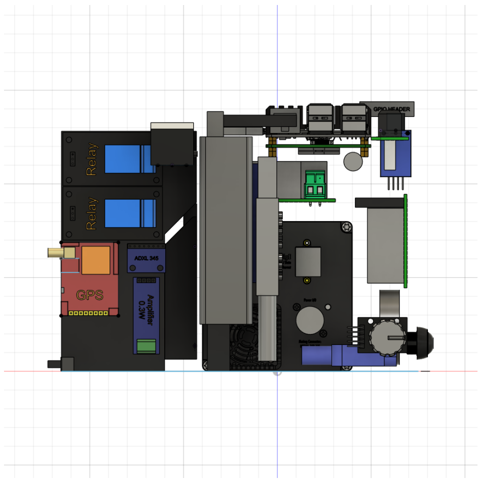
The main disadvantage of this layout is its longest side, 190 mm plus the printed parts would be right on the edge of the 200 mm limitation of a 3D printer.


(a) : Back left view


(b) : Front right view



(c) : Top view

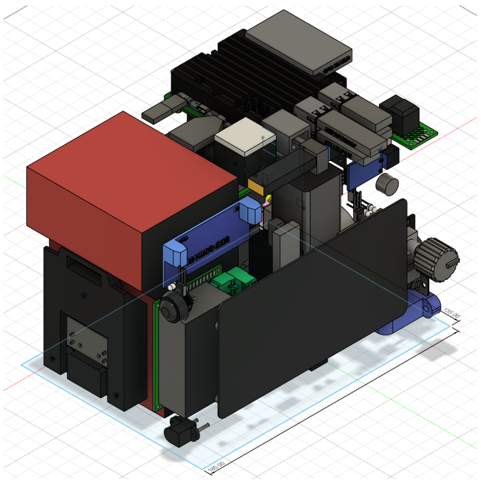


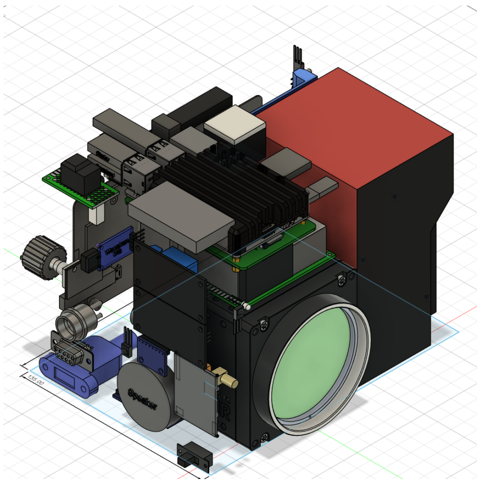
(d) : Back view without display

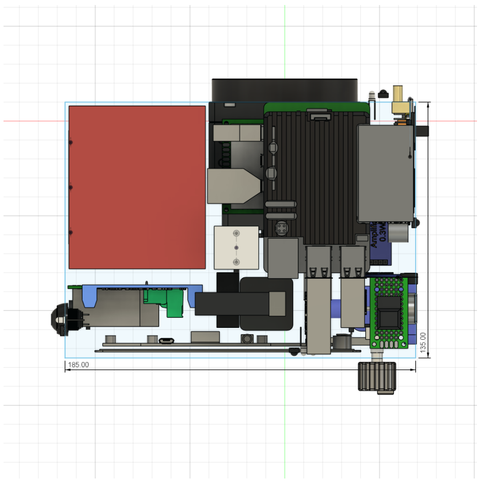
Figure 5.5: Layout 1

5.2.2 Layout 2

This layout is an iteration of the previous one. Some components are slightly rearranged for easier mounting and the total volume is reduced, see Figure 5.6. The size of the layout is approximately 185 x 135 x 145 mm (3.62 liters).


(a) : Back left view


(b) : Front right view


(c) : Top view

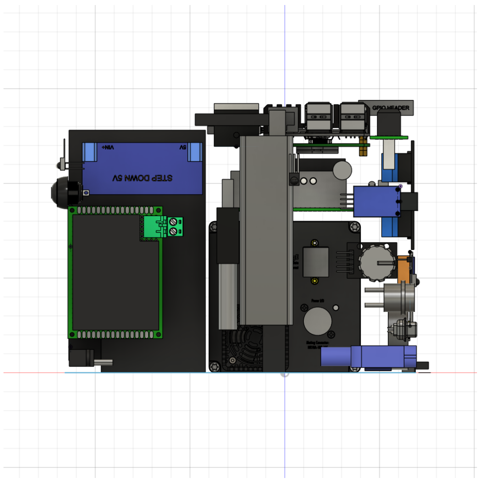
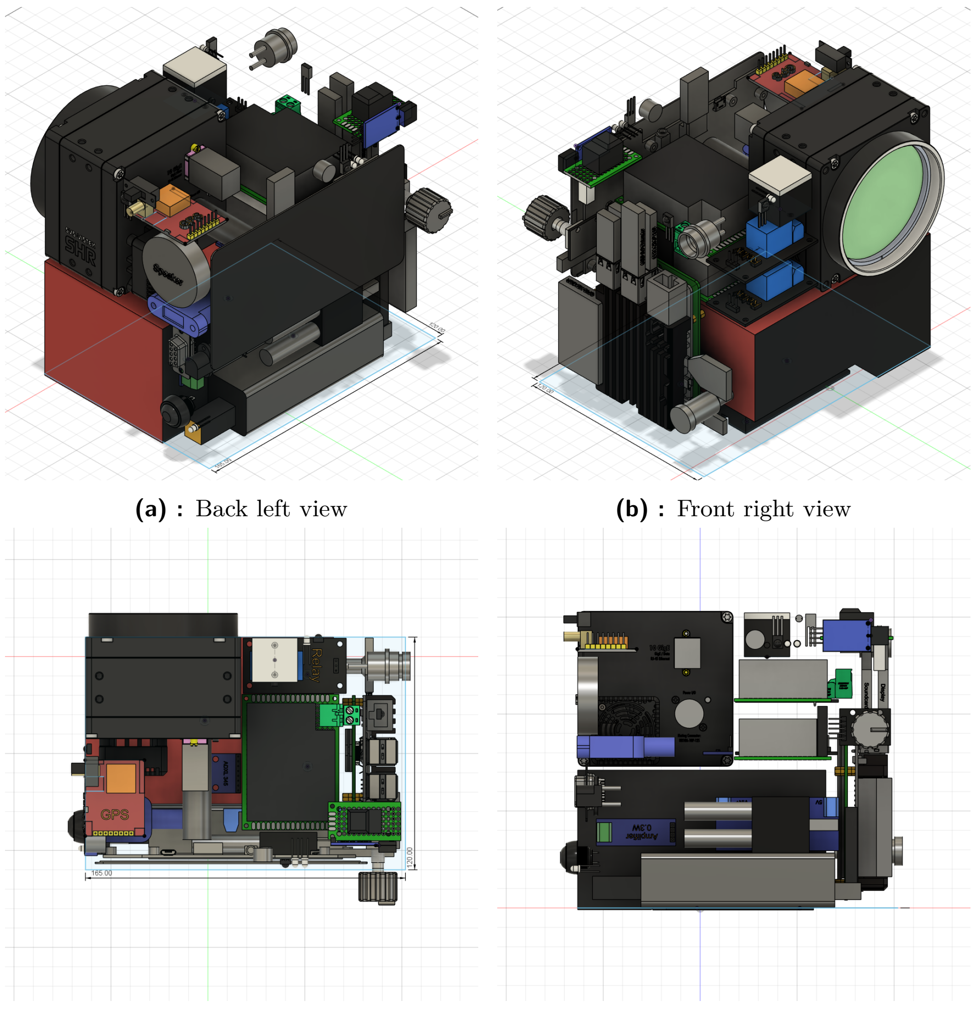

(d) : Back view without display

Figure 5.6: Layout 2

5.2.3 Layout 3

This layout is completely different from the previous two. The camera is above the battery, which is positioned horizontally and slides from the side. The Raspberry Pi is on the side wall rather than the top, see Figure 5.7. The size of the layout is approximately 165 x 120 x 155 mm (3.07 liters). This is the layout with which we chose to continue. We thought of other possible battery and camera mutual placements, but found them not meaningful.



(a) : Back left view

(b) : Front right view

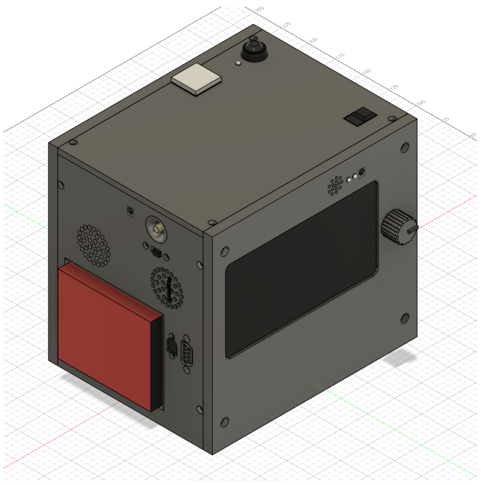
(c) : Top view

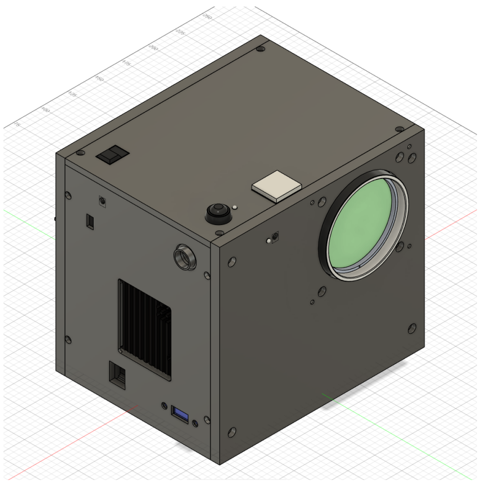
(d) : Back view without display

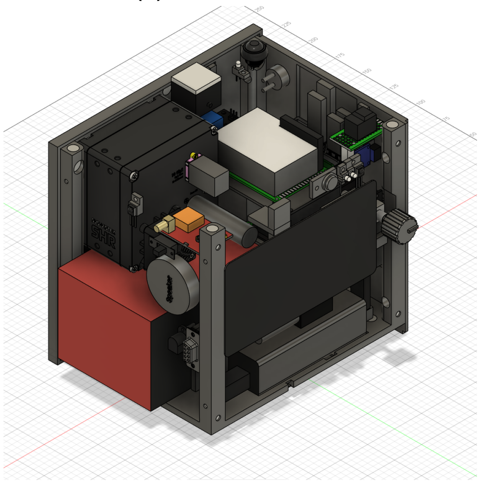
Figure 5.7: Layout 3

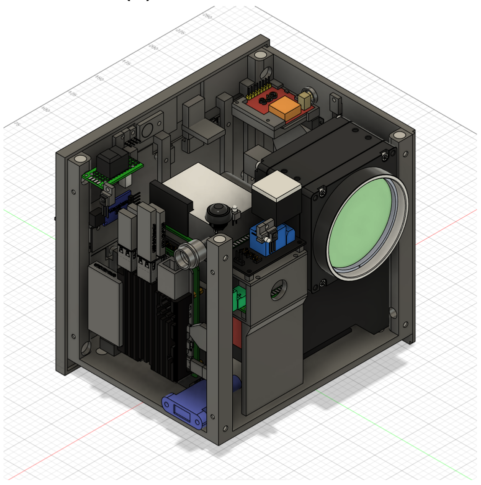
5.3 Camera casing

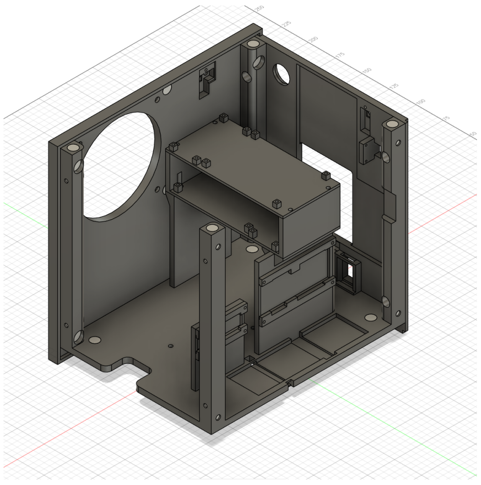
After we selected the layout, we needed to model the body that would be possible to print and assemble. We chose the approach of having one base plate with walls that attach to it. The order of the mounting walls during assembly will be important, as some walls will interlock with the previous ones, improving the overall rigidity. The walls will be mounted with M3 or M4 screws and threaded inserts. The camera will be attached to the front wall, so it can be easily removed, using only eight M4 screws. During the modeling, we had to move or add some components, so it slightly differed from the original layout. The proposed box camera design can be seen in Figure 5.8.


(a) : Back left view


(b) : Front right view


(c) : Back left view without left, back and top wall


(d) : Front right view without right, front and top wall


(e) : Back left view without components and left, back and top wall

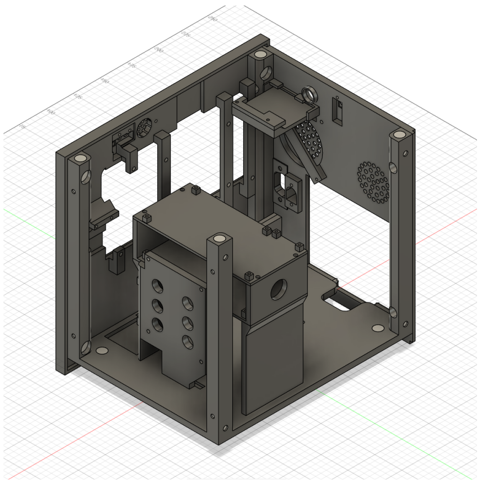

(f) : Front right view without components and right, front and top wall

Figure 5.8: Proposed box camera design

Chapter 6
Implementation

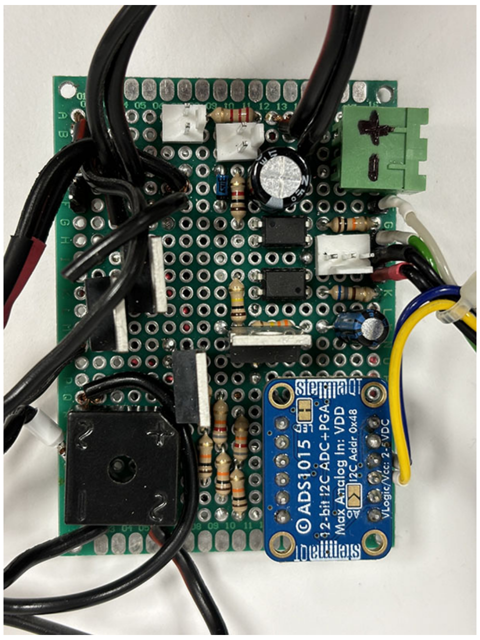
In this chapter, we first present the custom circuit boards that were made followed by the assembly of the box camera. In the last section, we briefly explain how the GUI application was implemented.

6.1 Circuit boards

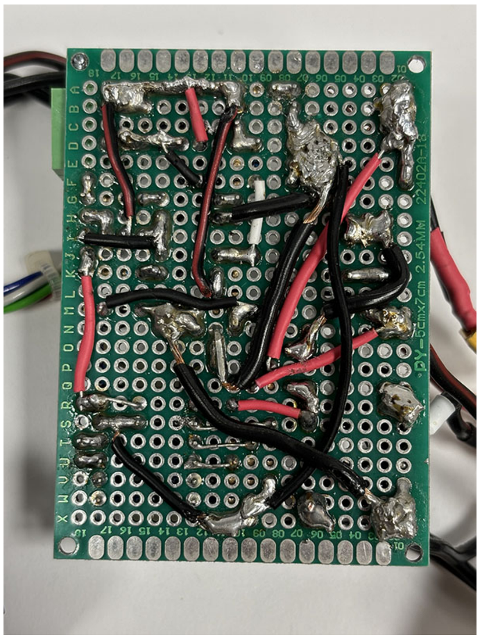
We made two custom boards based on the previously designed schematics, see Figures 5.2 and 5.3. For both, we used 70×50 mm general-purpose perforated circuit boards. The completed boards can be seen in Figure 6.1.

We started with the power input board. At first, we tested the schematic function on the breadboard. We found that the circuit does not work as intended unless at least 17 V is provided. The higher voltage is needed at the start, and after the MOSFET completes the circuit, the lower voltage will keep it working. This ended up being an advantage because the undercharged battery would not start the box camera, as we have the option to measure the voltage only after the Raspberry Pi boots and starts reading voltages using ADC via I^2C. However, if someone were to replicate this board, the battery and external power supply would have to be chosen appropriately. We also had some problems with the Raspberry Pi 5 8 GB version. The SBC never booted up, therefore we only used the 4 GB version which worked. We also planned to use the camera relay in a normally closed configuration, so the camera would start at the same time as the Raspberry Pi. This also did not work. The camera had trouble starting up. We had to connect it directly to the power source ground in a normally opened configuration, instead of the MOSFET controlled common ground. In this way, the camera starts only after the Raspberry Pi boots up and triggers the relay. Unfortunately, due to a lack of experience and knowledge of electronic circuits, we were unable to solve these issues and continued with the schematic as is shown in Figure 5.2.

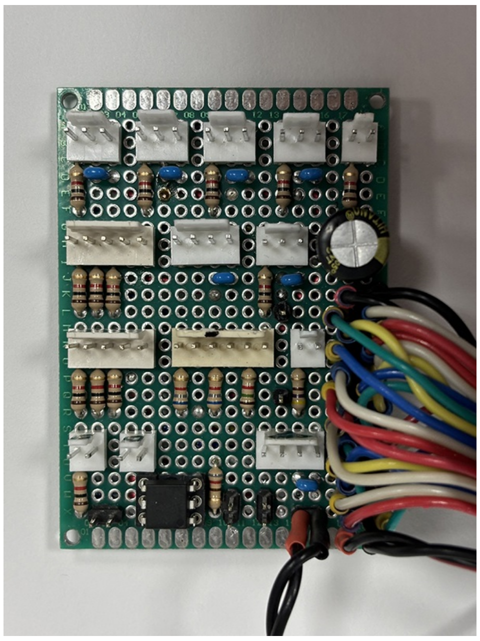
Making the connector board was quite tricky due to the very close soldered connections. After completion, we double-checked that there were no unwanted connections or short circuits. Only after we plugged it into the Raspberry Pi and checked the functionality of each connected element separately.



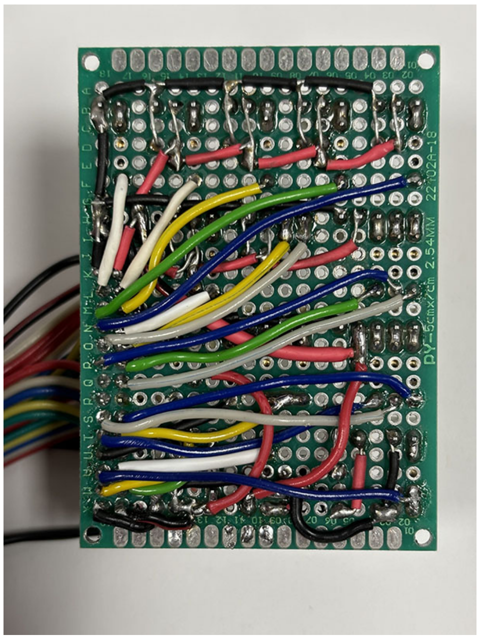
(a) : Power input board front



(b) : Power input board back



(c) : Connector board front



(d) : Connector board back

Figure 6.1: Custom circuit boards

6.2 Printing parts of the camera body

We manufactured the designed parts onto the Prusa MK3.5S using a black PETG filament. We printed on a textured steel sheet so that the outer walls would have the same texture. All parts were printed with 0.4 mm nozzle, 0.2 mm layer height, and 10 % infill.

6.3 Assembly

Before assembly, the threaded inserts needed to be pressed into the 3D-printed parts. The components and their connectors were designed mostly so that they could not be interchanged. If some connectors were the same, we labeled them for distinction. The components are mostly attached using 2.2×6.5 mm pan head screws. There are four IR receivers in total.

The assembly then starts with the base plate part, where the accelerometer, the battery adapter, the Raspberry Pi, the amplifier, the step-down converter, the USB to Ethernet adapter, and the sound card are mounted in this exact order. Afterward, the mounting plate with the power input board and the camera relay is mounted. Lastly, the mounting plate with the connector board and flash relay is mounted.

The assembly continues with the right wall. Before attaching it, the USB 3.0 extension cable, the RJ 45 socket, the IR reflective sensor, the IR receiver, and the AC connector must be mounted. Then the wall is attached to the base plate using four M3 screws.

Switching to the left wall, the external power input DC socket, the 9-pin D-Sub, the GNSS module, the IR receiver, and the speaker must be mounted to the wall first. The same as for the right wall, four M3 screws are used for attaching.

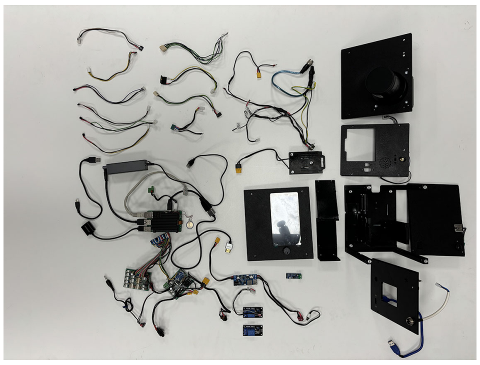
Before continuing, we tested the power input and connector boards, to ensure no harm was done during the assembly.

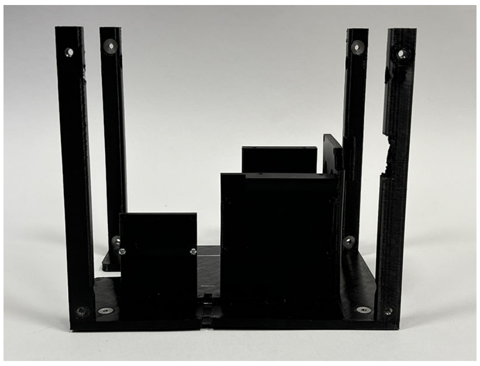
The next in the process is the top wall. The trigger button, the flash mount, the power button, and the LED are mounted to the wall. As for the previous two, the top wall is attached using four M3 screws.

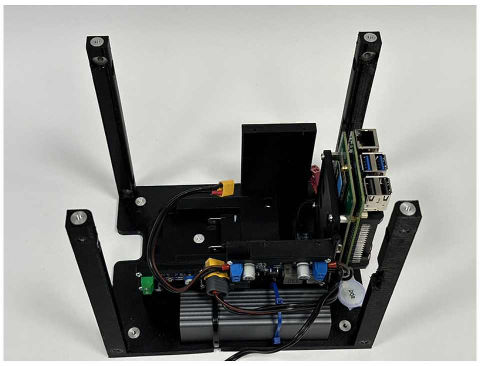
Now, either the front or the rear wall can be mounted. In Figure 6.2, we continue with the front wall. The camera is attached (using four M4 screws) to the wall together with the LED and IR receiver. Then the wall is attached using four M4 screws. The edge of the wall should overlap the side and top walls.

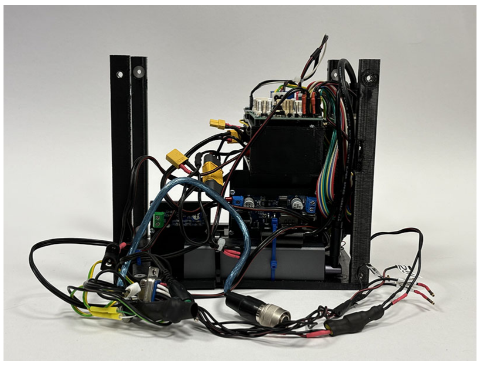
Lastly, the display, the rotary encoder, two LEDs, and the IR receiver are mounted to the back wall. Then the wall is attached similarly to the front wall, with four M4 screws with overlapping edges.

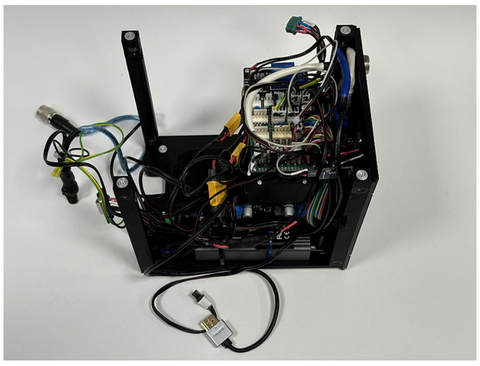
More photos of the process are available in the `images/firstDesign` directory on the attached media.


(a) : All 3D-printed parts and components


(b) : Base plate


(c) : Base plate fitted with components without custom boards and relays


(d) : Base plate fitted with components


(e) : Base plate and right wall fitted with components

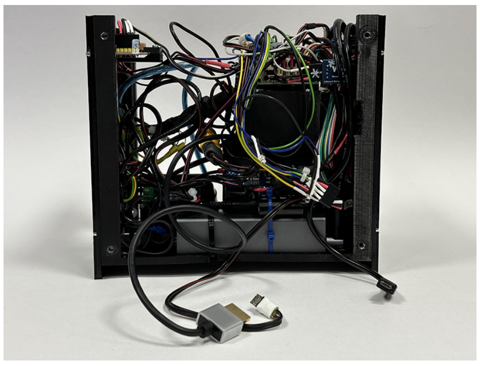
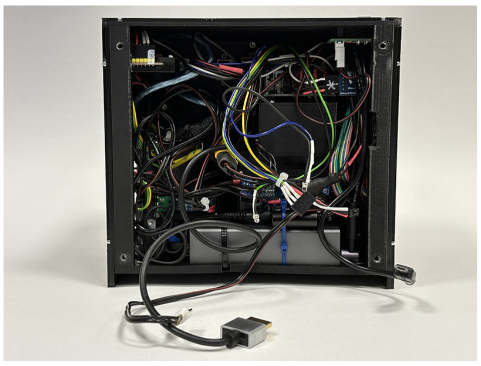
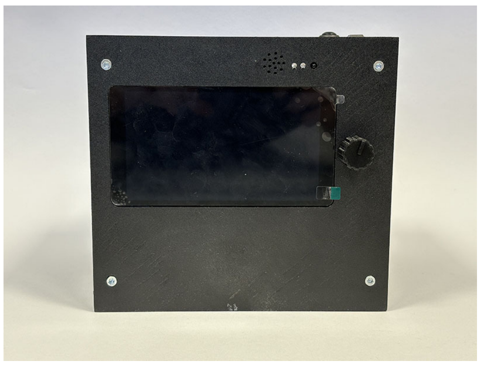

(f) : Base plate, right and left wall fitted with components

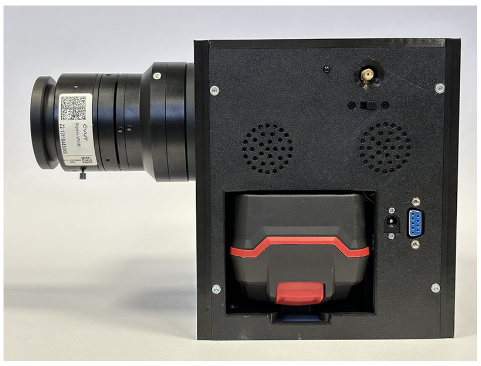
Figure 6.2: Assembly of the first design



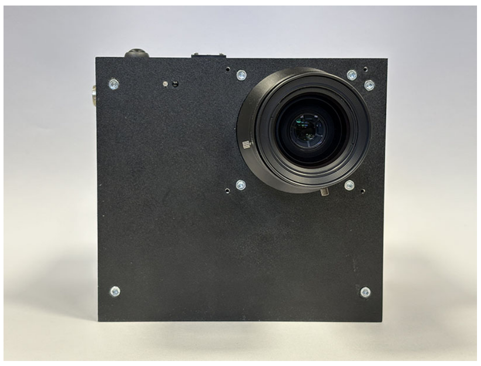
(g) : Base plate, right, left and top wall fitted with components



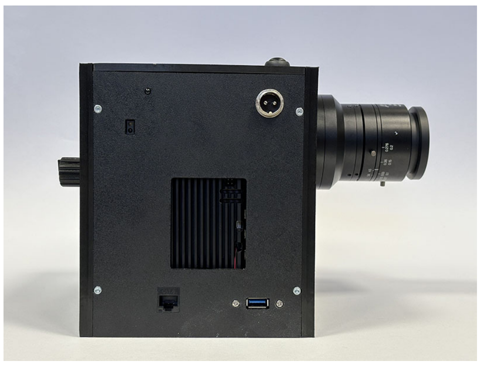
(h) : Back view of the assembled box camera



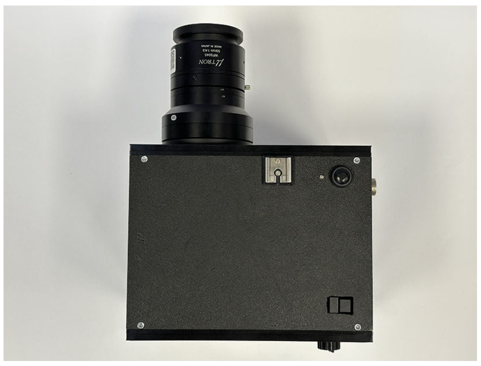
(i) : Left view of the assembled box camera



(j) : Front view of the assembled box camera



(k) : Right view of the assembled box camera



(l) : Top view of the assembled box camera

Figure 6.2: Assembly of the first design (cont.)

6.4 GUI application

The application was based on the Testing application 3.3. The codebase consists of 44 header and 38 source code files (exluding libraries). Describing the software implementation in depth is outside the scope of this thesis. For more information, the entire documented code can be found in `application` directory on the attached media, where the doxygen documentation is also generated. Screenshots of the application can be found in `images/gui` directory on the attached media. In the following, we provide a brief overview of how the application works.

The application consists of two important threads (GUI thread and image data thread) and some helper threads. The first thread (GUI) draws the menu, live preview, and handles user input. The second thread (image data thread) acquires new image data and prepares it (binning and demosaicing) for the first thread to display. This follows the producer-consumer pattern. The helper threads are used for handling GPIO events, triggering the flash, saving the images, GNSS communication, etc.

The camera API library fills the prepared buffers with new image data based on trigger events. The acquisition of images is performed sequentially. The trigger event to acquire new image data is only called after the image data from the previous call arrives. When the user requests to save the image, the first image data with a timestamp greater than the request timestamp is temporarily put aside, until the image save thread saves the image on the disk.

Compared to the Testing application we used a different library (fpng [54]) to convert and store RAW images as 8-bit PNG images. Despite the fact that the library was written to utilize x86 SIMD instructions, saving time was reduced significantly from 7 s to 4 s while still utilizing compression. Unfortunately, this library does not support 16-bit yet. In addition to the image, we also save its metadata in a separate JSON file and the preview image.

Chapter 7
User testing

In this chapter, we focus on testing the first prototype. For the test method, we chose usability testing, since we only wanted to test three participants. The goal of this test was to reveal software bugs and inconsistencies but also to get ideas for improvements to our initial design.

7.1 Participants

We chose the participants so that each had a different level of knowledge about cameras in general.

The first participant (A) works in the Image Processing Laboratory (ImproLab) at CTU FIT, so he is in contact with computer vision cameras on a daily basis. He fully understands the imaging chain but is less familiar with personal cameras.

The second participant (B) has experience in photography and in designing and manufacturing custom devices, similar to the box camera.

The third participant (C) has little to no experience with cameras. He represents the standard user, who is mostly familiar with capturing images using a smartphone.

7.2 Tasks

At the start of the test, we introduced the box camera, primarily its hardware features. After a short introduction, the participant was given a list of tasks, see below. The tasks explore almost all the features that the box camera has. During the test, we helped the participant complete each task. In the end, we had a short discussion about their experience and provided a questionnaire to fill in, see Appendix D.

Task list:

1. Insert the battery into the box camera and turn it on.
2. Check the date and time and change it if necessary.
3. Select the object you want to photograph and point the camera at it.

4. Set the gain to manual - 16 dB, the exposure time according to the preview (so you are satisfied) and auto white balance, or manual if you are not happy with auto.
5. Focus the lens on the object.
6. Take a picture.
7. After saving, preview the image.
8. Switch the pixel depth to 16-bit and the save format to RAW.
9. Turn on multi-exposure, 3 frames, 1.5 multiplier.
10. Check the settings for remote trigger (turn it on if necessary).
11. Take a picture using the remote trigger.
12. Find out what exposure time frame 3 had.
13. Export the last 3 images to external media.
14. After exporting, remove the medium.
15. Delete the oldest image.
16. Check the battery status and see how much run time is left.
17. Switch exposure time to manual and set it to 150 ms, gain to 0 dB, bit depth to 8bit.
18. Connect the GNSS antenna and turn on the module.
19. Connect the flash and turn on its use.
20. Check the settings for the touchless trigger (turn it on if necessary).
21. Make sure the GNSS module knows the camera's location and take a picture using the touchless trigger.
22. Turn off the touchless trigger, use of flash (and disconnect it). Set exposure time and gain to auto, bit depth to 8-bit and save format to PNG.
23. Set the time-lapse every 1 minute.
24. Turn on the time-lapse and stop it after 2 frames.
25. Turn off the box camera.

7.3 Results

In this section, we summarize the feedback from each participant separately. Some of the suggested modifications will be implemented in the final design described in Chapter 8. The complete answers (in Czech) can be found in the `questionnaire/results.pdf` on the attached media.

Participant A successfully completed all tasks without any major intervention. During the test, he found some problematic spots. Right at the start when inserting the battery, he noticed that it was hard to pull the battery out. He suggested adding a further hole at the top. During task 4., he had some problems setting the white balance. He wanted to lock the parameters that the auto procedure set. We quickly realized that this feature was missing and that auto white balance works only continuously. Furthermore, he was looking for the histogram while setting the exposure time. During Task 18. he had problems connecting the GNSS antenna and turning the module on. He suggested adding an LED indicating whether the GNSS module is turned on. During the discussion after the test, he further suggested adding active cooling to the box camera, handle, tripod mount, and covering the Raspberry Pi heatsink.

Participant B also completed all tasks. However, the camera became unresponsive during Task 9. The GUI worked, but the acquisition was stuck, so we had to turn off the box camera and turn it on again. During the discussion, he suggested many changes. Here we point out the most important points, the details can be seen in the complete answers.

- Add an option to restart the camera and the application from within the application.
- Allow preview zoom in all menus, not in a special submenu.
- Add feedback to shutdown and trigger press.
- Update the information bars more frequently.
- Allow for zooming in captured images.
- Make the acquisition settings persist after a restart.
- Better status indication during timelapse, self-timer, and multi-exposure.

Participant C also completed all tasks. At first, he had some problems finding the required settings, but after exploring each menu, he quickly gained confidence. He suggested simplification of menus (e.g. hiding gallery and manage images to a submenu, as well as all the settings under one submenu) and rearrangement of the top information bar for better visibility. Even as the least experienced user, he completed the tasks without any major intervention.

Chapter 8

Final design

In this chapter, we describe the modifications made to the first prototype. The modifications were primarily based on the user testing in the previous chapter. Next, we describe the last user test and its result. Lastly, we provide the final specifications.

8.1 Hardware modifications

In this section we describe the modifications to the camera casing and components, see Figure 8.1.

The most noticeable modifications are the rounded corners and the handles. Both of these modifications should improve the experience when manipulating the camera. The position of handles may not be optimal, but we did not want to completely change the layout to have them in a better position (more forward, to accommodate for the heavy camera and battery) since the box camera will mostly be operated on a tripod. For tripod usage, an additional bottom wall was added with standard 1/4" thread used for cameras. Due to added handles, the IR trigger, GNSS power switch, and connectors next to the battery had to be rearranged, so that they do not interfere with the user's hands.

Inside the camera casing, we also made a few modifications, the pillars on the base plate are now removable. In this way, we can print them separately in a horizontal position to improve the strength. We added another mounting point for the left, right, and front walls that attach to the base plate. The inner walls holding the step-down converter and the amplifier were connected, and one inner wall was added between the battery and the Raspberry Pi. These additional walls were added to protect the cables when inserting the battery.

The next change that may not be visible is the different material used. This time we printed all the parts using black ASA filament, which has better mechanical properties, mainly heat and UV resistance. However, printing with ASA is not as straightforward as with PETG. We had problems with large printed models that were deformed during print because the corners lifted off the print bed. Despite the enclosure, different steel sheet (satin-coated), and adhesion promoter, the resulting finish was slightly worse compared to the previous prototype made of PETG. If we were to print it again, we would probably try a special high-temperature resistive PETG or use a heated enclosure.

The AC connector was replaced with an HDMI port because we could not get it to work and HDMI seemed more practical. We were able to complete the AC circuit from the Raspberry Pi, but we could not break the circuit as it stayed connected until it was disconnected. This could probably be resolved by using a different optotriac.

Since the Raspberry Pi thermally throttled (measured over 85 °C) when using a passive cooler, we switched to the active one and added protection against inserting fingers into the fan blades and the hot cooler. To further address the overheating issues, we added two fans that blow air at the camera and inside the box camera. We also added ventilation holes above the fans.

We added a new power switch for the amplifier below the display. Sometimes the speaker made a disturbing buzzing sound when nothing was playing, and this partly addresses this issue. We also added status labels for the GNSS and the amplifier switch.

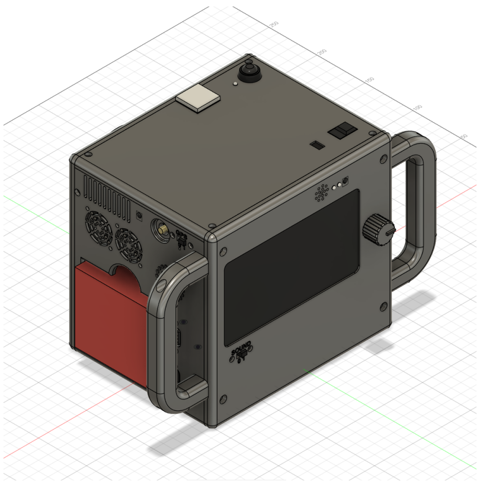
Above the USB port, we added a fuse holder for the cartridge fuses. Before we used the car fuse, which was hidden inside the casing, making it impossible to exchange it without disassembling. It is worth mentioning that we did not need to replace the fuse even once.

We also added a hole above the battery, so it should be easier to remove.

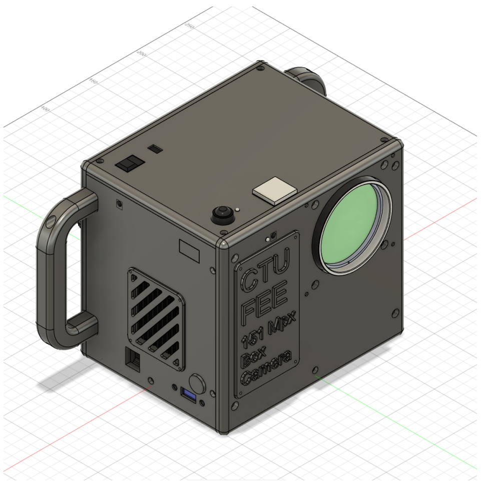
8.2 Software modifications

Here, we provide a list of implemented modifications to the GUI application:

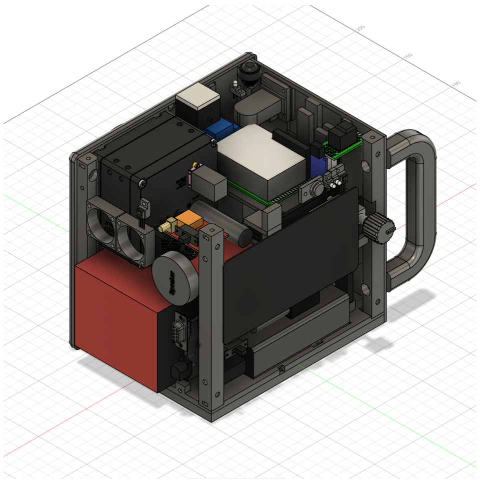
- restart camera,
- show GNSS time,
- renaming:
  - Trigger settings -> Trigger/Flash settings,
  - Edit date and tame -> Change date and time,
  - Timed trigger -> Self-timer/Timelapse
- showing all the exposure times in ms,
- add option to show last captured image in gallery from main menu,
- better status indication during timelapse, self-timer, and multi-exposure,
- audio feedback on trigger press,
- smaller time period between information updates,
- option to zoom preview in other menus,



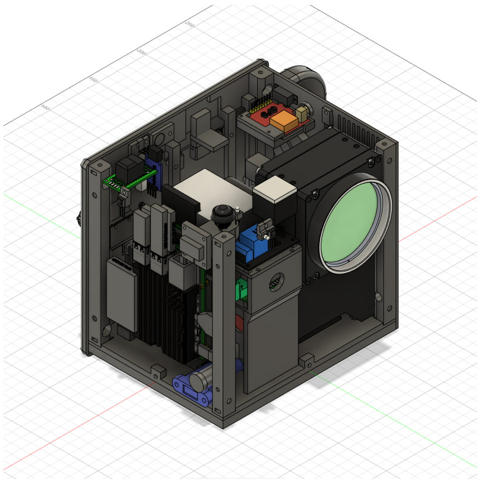
(a) : Back left view



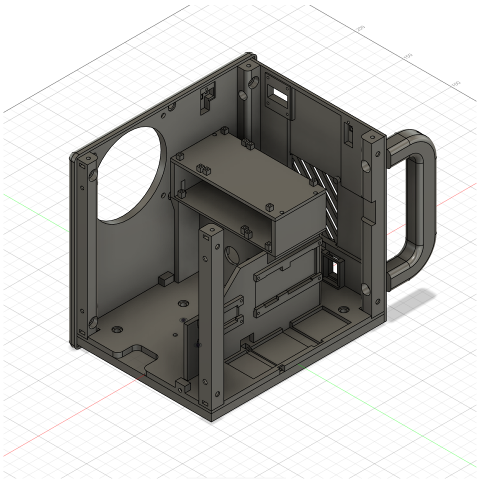
(b) : Front right view



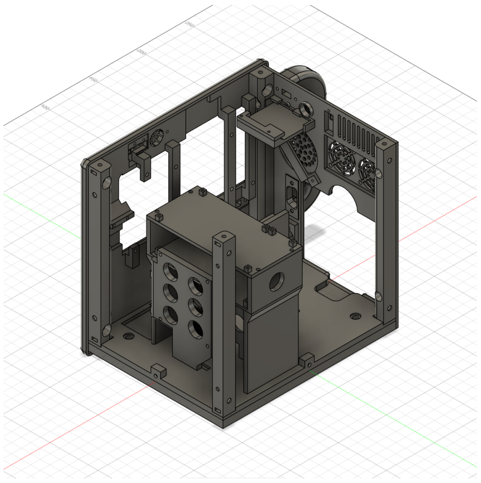
(c) : Back left view without left, back and top wall



(d) : Front right view without right, front and top wall



(e) : Back left view without components and left, back and top wall



(f) : Front right view without components and right, front and top wall

Figure 8.1: Final box camera design

- added page navigation in manage images,
- option to lock auto white balance parameters.

8.3 Assembly

The assembly was very similar to the first prototype. The only difference is the new tripod mount wall, which has to be installed before the other walls. See the process in Figure 8.2. More photos are available in the `images/finalDesign` directory on the attached media.

8.4 Lenses

We have two lenses for the box camera that can be directly attached to the camera using the M72 × 0.75 lens mount. Both are manufactured by Myutron. The first is Myutron WF5045-M [55], which has a focal length of 50 mm and the second is Myutron XLS01-M [56], which has a focal length of 65 mm.

Lastly, we have the Schneider-Kreuznach G-Claron lens [57] designated for a different lens mount. This lens has a focal length of 150 mm. We designed and 3D printed a tubular adapter with variable length. The adapter is attached to the universal mounting plate which we incorporated into the camera casing.

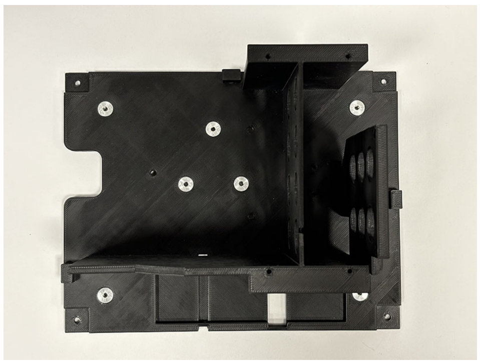
8.5 Final user test

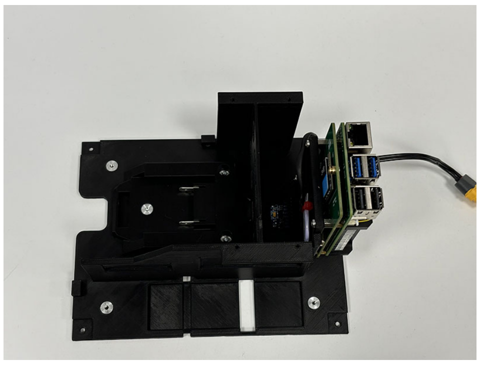
For the final participant (D), we chose the UX/UI specialist from the DCGI at FEE CTU, who is also familiar with cameras. The procedure was the same as in 7.2. This time, the feedback will be only noted and left for possible future improvements.

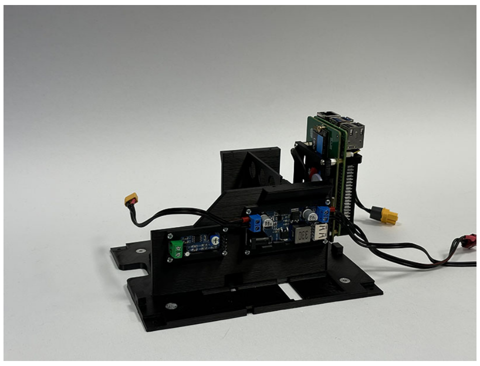
During the test, he had no problems with completing the tasks. However, he noticed a lack of feedback in many places. For example, when ejecting a disk or saving a date/time, the feedback is hardly visible. During Task 4, when changing the exposure time and white balance, the histogram feature was missing. He also suggested improving GNSS module status information. This was already improved based on Participant A's experience by adding a label and status LED. However, the GUI still needs to be improved to distinguish between three states: online, turned on but not receiving, and turned off.

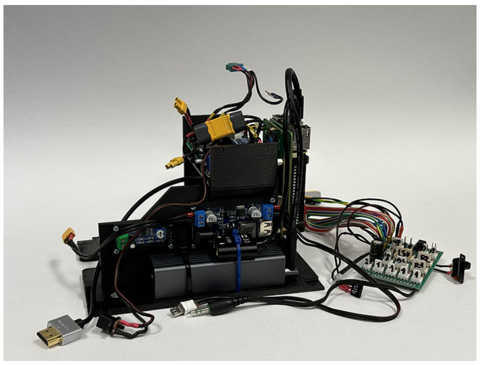
He had no problems with the layout of hardware elements but suggested adding labels to buttons and LEDs, so their function is clear.

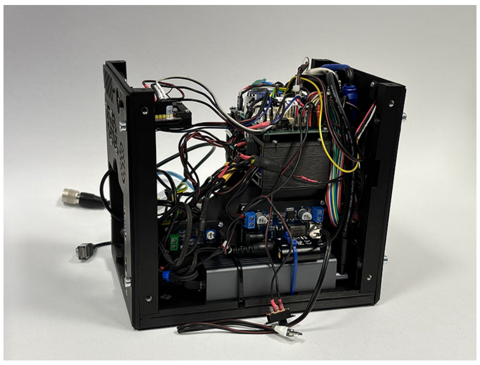
Overall, he was satisfied with the box camera, despite the missing features and feedback.


(a) : Base plate


(b) : Base plate fitted with Raspberry Pi 5 and battery adapter


(c) : Base plate fitted with Raspberry Pi 5, battery adapter, amplifier and step-down converter


(d) : Base plate fitted with components


(e) : Base plate, right and left wall fitted with components

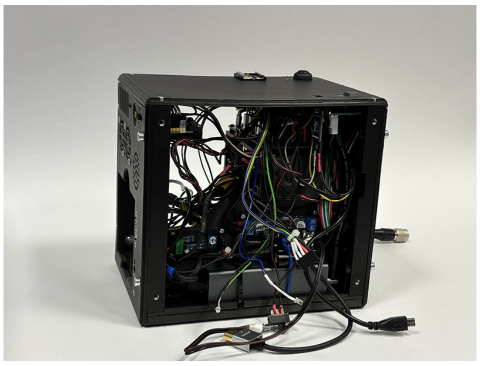
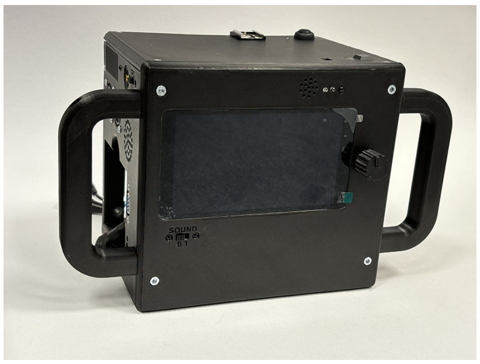
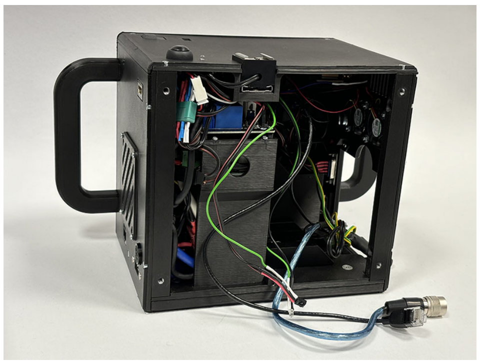

(f) : Base plate, right, left and top wall fitted with components

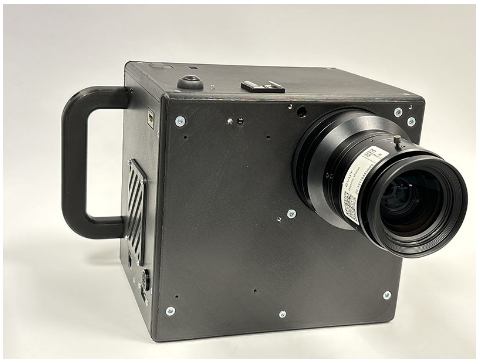
Figure 8.2: Assembly of the final prototype



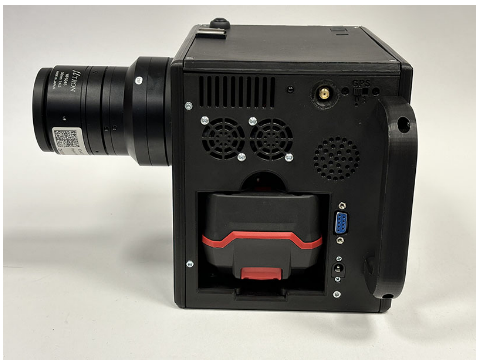
(g) : Base plate, right, left, top and back wall fitted with components (back view)



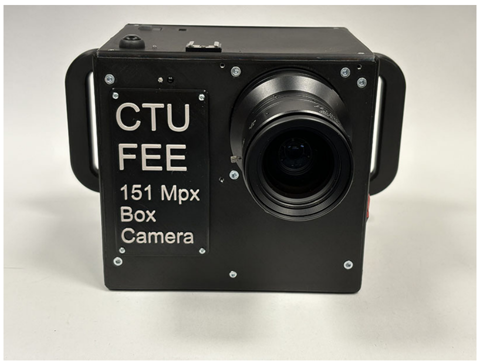
(h) : Base plate, right, left and top wall fitted with components (front view)



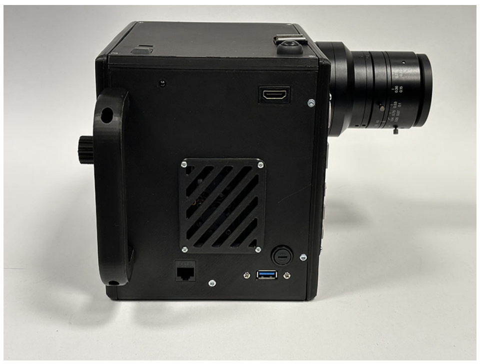
(i) : Front right view of the assembled box camera



(j) : Left view of the assembled box camera

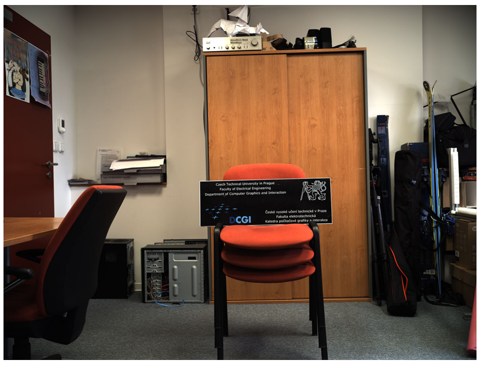


(k) : Front view of the assembled box camera with a sign

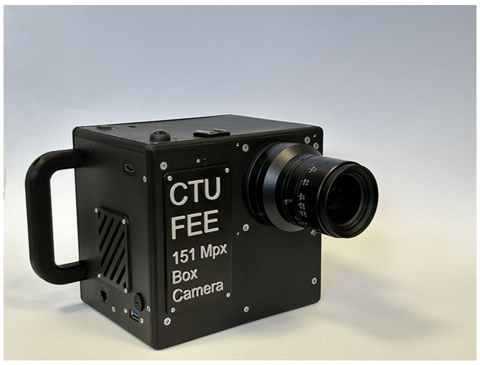


(l) : Right view of the assembled box camera

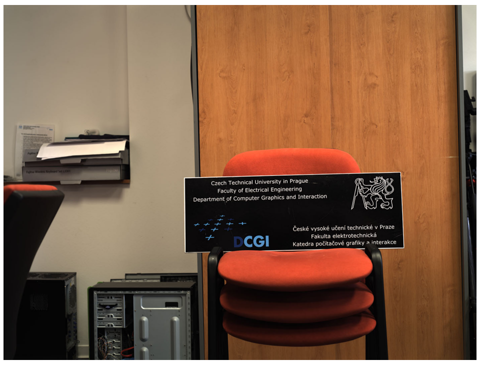
Figure 8.2: Assembly of the final prototype (cont.)



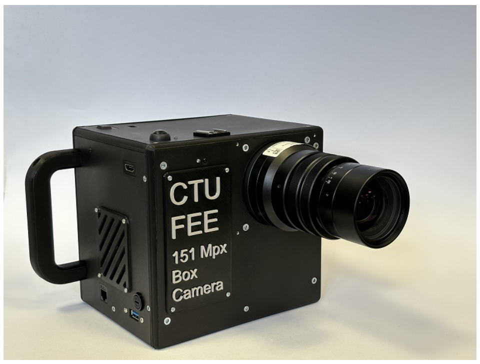
(a) : Image taken with Myutron WF5045-M (50 mm)



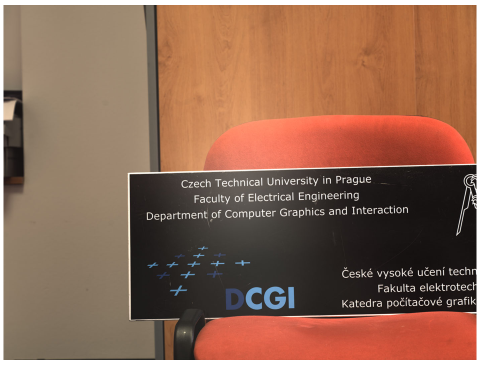
(b) : Box camera with Myutron WF5045-M


Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Graphics and Interaction
DCGI
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

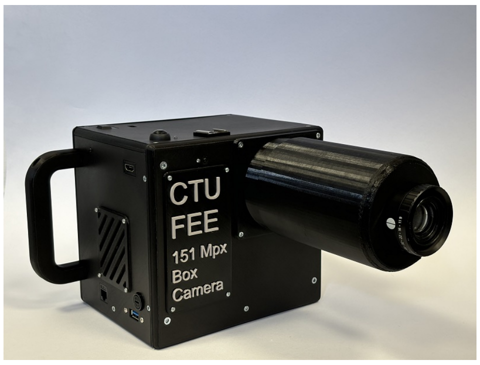
(c) : Image taken with Myutron XLS01-M (65 mm)



(d) : Box camera with Myutron XLS01-M


Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Graphics and Interaction
DCGI
České vysoké učení techn
Fakulta elektrotech
Katedra počítačové grafik

(e) : Image taken with Schneider-Kreuznach G-Claron (150 mm)



(f) : Box camera with Schneider-Kreuznach G-Claron attached to a tubular lens adapter

Figure 8.3: Box camera lenses

8.6 Specification

In this section, we provide the physical and performance specifications of the box camera. In Table 8.2 the boot, time to save, export speed, and preview were averaged from five measurements. The temperatures were measured after 30 minutes of running live preview in a room at a temperature of 26 °C.

Weight [g]	
Casing w/o camera and battery	1627
Casing w/o camera lens	2886
Casing with 50 mm lens	3278
Dimensions [mm]	
Width w/o handles	190
Width	290
Length	152
Height	175
Time [s]	
Remove camera from casing	80
Install camera to casing	160

Table 8.1: Box camera physical specifications

Boot until the application startup [s]	30
Boot until first acquired image [s]	60
Shutdown [s]	5
Battery life [min]	120
Power consumption camera running [W]	40
Power consumption camera standby [W]	29
Power consumption camera turned off (W]	8.6
Time to save image 8-bit, PNG [ms]	4815
Time to save image 8-bit, RAW [ms]	422
Time to save image 16-bit, PNG [ms]	17775
Time to save image 16-bit, RAW [ms]	825
Export speed [MiB/s]	65
Preview 8-bit [ms]	596
Preview 16-bit [ms]	1369
Camera temperature [°C]	66
Camera temperature w/o battery [°C]	65
RPi temperature [°C]	75
RPi temperature w/o battery [°C]	72

Table 8.2: Box camera performance specifications

Chapter 9
Conclusion

In this thesis, we focused on creating a combination of hardware and software that would transform the computer vision camera SVS-VISTEK SHR411CXGE into a more traditional personal camera.

We began by studying the possible single-board computers that could be used in combination with the camera. To select the most suitable one, we implemented a simple application. This application served two purposes, to show that it is possible to combine the camera with a single-board computer and as a benchmark tool. The results showed that this combination is indeed possible, and the Raspberry Pi 5 was identified as the best choice.

Following this, we defined all the requirements for the box camera. These requirements guided us in selecting additional components and identifying the software features that needed to be implemented.

Having requirements in mind, we focused on choosing the right electrical and mechanical components, primarily the battery and display. After finalizing the list of necessary components, we proposed three possible layouts and chose one that was used when designing the camera casing.

Once we were satisfied with the design, we manufactured the camera casing on the 3D printer. We assembled the 3D-printed parts together with individual components and then started implementing software features. To identify design flaws, we conducted usability testing with three participants with different level of expertise.

Based on the testing result and our insights, we designed a second iteration of the camera casing, which was also 3D printed and assembled. We also improved the application by implementing suggested features and enhancements. The second iteration was also tested by a single participant.

The output of this thesis is a functional box camera that meets all the assignment requirements as well as our stated requirements.

There is still room for improvement, primarily on the software side. We did not finish all the suggested features from testing, such as showing the histogram or the ability to zoom the captured images in the gallery. We also left some hardware unused, such as a microphone, which may be somehow implemented in the application to annotate the captured images. We also did not utilize the integrated Wi-Fi from the Raspberry Pi. This may be used to host a web server, which could be used to operate the camera or transfer image data.

Bibliography

1. *Camera Obscura* [online]. 2020. [visited on 2024-05-11]. Available from: https://magazine.artland.com/wp-content/uploads/2020/01/Camera-Obscura-1024x609.jpg.
2. FIALA, Josef; BAŘINA, Květoslav; PROČEK, Jiří. *Optickomechanické přístroje*. Druhé vydání. Praha: SNTL - NAKLADATELSTVÍ TECHNICKÉ LITERATURY, 1982.
3. ALLEN, Elizabeth; TRIANTAPHILLIDOU, Sophie. *The Manual of Photography*. 10th edition. Oxford: Focal Press, 2011. ISBN 978-0-240-52037-7.
4. *Different apertures of a lens* [online]. 2001-2024. [visited on 2024-05-11]. Available from: https://upload.wikimedia.org/wikipedia/commons/thumb/f/f8/Lenses_with_different_apertures.jpg/1920px-Lenses_with_different_apertures.jpg.
5. *Understanding focal length and camera zoom* [online]. 2020. [visited on 2024-05-13]. Available from: https://imgcap.capturetheatlas.com/wp-content/uploads/2020/04/understanding-focal-length-and-camera-zoom.jpg.
6. *The Bayer arrangement of color filters on the pixel array of an image sensor* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [visited on 2024-05-13]. Available from: https://en.wikipedia.org/wiki/Bayer_filter%5C#/media/File:Bayer_pattern_on_sensor.svg.
7. LUHMANN, Thomas; ROBSON, Stuart; KYLE, Stephen; BOEHM, Jan. *Close-range photogrammetry and 3D imaging*. 2nd ed. Berlin: De Gruyter, 2013. ISBN 978-3-11-030269-1. Available from DOI: 10.1515/9783110302783.
8. ŽÁRA, Jiří; BENEŠ, Bedřich; SOCHOR, Jiří; FELKEL, Petr. *Moderní počítačová grafika*. 2., přeprac. a rozš. vyd. Brno: Computer Press, 2004. ISBN 80-251-0454-0.
9. RANDERS-PEHRSON, Glenn; BOUTELL, Thomas. *PNG (Portable Network Graphics) Specification* [online]. World Wide Web Consortium (W3C), 2003. [visited on 2024-05-16]. Available from: https://www.w3.org/TR/PNG/. Version 1.2.

10. BANTERLE, Francesco. *Advanced high dynamic range imaging: theory and practice*. 1st edition. Natick: A.K. Peters, 2011. ISBN 978-1-56881-719-4.

11. *Shr411XGE* [online]. 2024. [visited on 2024-01-19]. Available from: https://ads-img.co.jp/mv/wp-content/uploads/2021/03/Foto_SHR411_461_XGE_01_rgb_mini-removebg-preview.png.

12. *GenICam* [online]. 2024. [visited on 2024-01-16]. Available from: https://www.emva.org/standards-technology/genicam/.

13. *Raspberry Pi 4 Model B*. Raspberry Pi Ltd, 2023. Available also from: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf.

14. *Raspberry Pi Compute Module 4*. Raspberry Pi Ltd, 2023. Available also from: https://datasheets.raspberrypi.com/cm4/cm4-product-brief.pdf.

15. *System-on-Modules (SOMs): How and Why to Use Them* [online]. 2024. [visited on 2024-01-17]. Available from: https://www.xilinx.com/products/som/what-is-a-som.html.

16. *RASPBERRY-PI CM4008008* [online]. 2024. [visited on 2024-01-17]. Available from: https://cz.farnell.com/productimages/large/en_GB/3563474-500.jpg.

17. GEERLING, Jeff. *NVMe SSD boot with the Raspberry Pi 5* [online]. 2023. [visited on 2024-01-18]. Available from: https://www.jeffgeerling.com/blog/2023/nvme-ssd-boot-raspberry-pi-5.

18. *Raspberry Pi 5*. Raspberry Pi Ltd, 2023. Available also from: https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf.

19. *Banana Pi BPI-M5* [online]. 2023. [visited on 2024-01-18]. Available from: https://wiki.banana-pi.org/Banana_Pi_BPI-M5.

20. *NanoPi M4V2* [online]. 2023. [visited on 2024-01-19]. Available from: https://wiki.friendlyelec.com/wiki/index.php/NanoPi_M4V2.

21. *NanoPi M4V2* [online]. 2023. [visited on 2024-01-19]. Available from: https://wiki.friendlyelec.com/wiki/images/5/54/NanoPi_M4V2-A01.jpg.

22. *VIM4 Specifications*. Khadas Technology Co., Ltd., 2023. Available also from: https://dl.khadas.com/products/vim4/specs/vim4-specs.pdf.

23. *Khadas VIM4* [online]. 2022. [visited on 2024-01-19]. Available from: https://docs.khadas.com/_media/products/sbc/vim4/hardware/vim4-top-side.webp.

24. *ODROID-XU4* [online]. 2023. [visited on 2024-01-19]. Available from: https://wiki.odroid.com/odroid-xu4/odroid-xu4.

25. *ODROID-XU4* [online]. 2023. [visited on 2024-01-19]. Available from: https://wiki.odroid.com/_media/odroid-xu4/odroid-xu4np.jpg.
26. ROHIT, Kumar. *USB 3.1 Gen1 to 5GbE Network Adapter Guide* [online]. 2021. [visited on 2024-01-21]. Available from: https://www.servethehome.com/usb-3-1-gen1-to-5gbe-network-adapter-guide/.
27. *USB 3.2 Gen 1 to 5GbE Adapter*. QNAP Systems, Inc., 2019. Available also from: https://www.qnap.com/en/product/qna-uc5g1t/specs/hardware/USB%203.2%20Gen%201%20to%205GbE%20Adapter.pdf.
28. *ADE-25R SUPERSPEED USB-A 2.5 GIGABIT ETHERNET* [online]. 2023. [visited on 2024-01-20]. Available from: https://axagon.eu/en/produkty/ade-25r.
29. *Embedded Systems with Jetson* [online]. 2024. [visited on 2024-01-20]. Available from: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/.
30. *Orange Pi 5 plus 32GB* [online]. 2024. [visited on 2024-01-20]. Available from: http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/Orange-Pi-5-plus-32GB.html.
31. *LattePanda Sigma* [online]. 2023. [visited on 2024-01-20]. Available from: https://www.lattepanda.com/lattepanda-sigma.
32. CORNUT, Omar. *ImGui: Bloat-free Immediate Mode Graphical User Interface for C++ with Minimal Dependencies*. 2023. Version 1.90. Available also from: https://github.com/ocornut/imgui. GitHub repository.
33. DOUXCHAMPS, Damien. *libdc1394*. 2018. Available also from: https://damien.douxchamps.net/ieee1394/libdc1394/.
34. GROUP, PNG Development. *libpng: Official PNG Reference Library*. 2023. Version 1.6.40. Available also from: http://www.libpng.org/pub/png/libpng.html.
35. *Makerfire USB Tester Type C Meter Detector Checker DC 0-30V/0-6.5A* [online]. 2024. [visited on 2024-01-18]. Available from: https://shop.makerfire.com/products/copy-of-makerfire-usb-tester-type-c-meter-detector-checker-dc-0-30v-0-6-5a.
36. *INDUSTRIAL microSD*. Kingston Technology, 2023. Available also from: https://www.kingston.com/datasheets/sdcit2_en.pdf.
37. *16GB eMMC 5.1 Module for NanoPi* [online]. 2024. [visited on 2024-01-19]. Available from: https://www.friendlyelec.com/index.php?route=product/product%5C&product_id=240.
38. *VERBATIM Store ´n´ Go Portable SSD 120GB* [online]. 2024. [visited on 2024-01-19]. Available from: https://www.alza.cz/EN/verbatim-store-n-go-portable-ssd-120gb-d5748153.htm.

39. *top Manual Page*. Linux Documentation Project, 2023. Available also from: `https://man7.org/linux/man-pages/man1/top.1.html`.

40. *RasPiKey: Plug and Play eMMC Module for Raspberry Pi* [online]. 2023. [visited on 2024-01-20]. Available from: `https://www.uugear.com/product/raspikey-plug-and-play-emmc-module-for-raspberry-pi/`.

41. *CANON EOS 700D SPECIFICATIONS* [online]. 2024. [visited on 2024-05-15]. Available from: `https://www.canon-europe.com/for_home/product_finder/cameras/digital_slr/eos_700d/specifications/`.

42. *HatDrive! Bottom (NVMe 2230, 2242, 2280 GEN 3) for Raspberry Pi 5* [online]. 2024. [visited on 2024-05-19]. Available from: `https://pineboards.io/products/hatdrive-bottom-2230-2242-2280-for-rpi5`.

43. *Lexar NM710 M.2 2280 PCIe Gen4x4 NVMe SSD* [online]. 2024. [visited on 2024-05-19]. Available from: `https://www.lexar.com/product/lexar-nm710-m-2-2280-pcie-gen4x4-nvme-ssd/`.

44. *ADA-12 USB - CABLE AUDIO* [online]. 2024. [visited on 2024-05-19]. Available from: `https://www.axagon.eu/en/produkty/ada-12`.

45. *5inch HDMI AMOLED* [online]. 2024. [visited on 2024-05-16]. Available from: `https://www.waveshare.com/wiki/5inch_HDMI_AMOLED`.

46. *5.5inch HDMI AMOLED* [online]. 2024. [visited on 2024-05-16]. Available from: `https://www.waveshare.com/wiki/5.5inch_HDMI_AMOLED`.

47. *Modul DC-DC step-down měniče, max 6A* [online]. 2024. [visited on 2024-05-19]. Available from: `https://rpishop.cz/napajeni/2871-modul-dc-dc-step-down-menice-max-6a.html`.

48. *LiFePO4 vs. Lithium Ion Batteries: What's the Best Choice for You?* [online]. 2023. [visited on 2024-05-16]. Available from: `https://blog.ecoflow.com/us/lifepo4-vs-lithium-ion-batteries/`.

49. *PROCORE18V 4.0AH PROFESSIONAL* [online]. 2021. [visited on 2024-05-16]. Available from: `https://www.bosch-professional.com/gb/en/products/procore18v-4-0ah-1600A016GB`.

50. *Parkside 20V Battery 4.0 Ah PAP 20 B3 Li-Ion Battery EU for tools of the Parkside X 20V Family* [online]. 2024. [visited on 2024-05-16]. Available from: `https://www.grizzlytools.shop/Parkside-20V-Battery-40-Ah-PAP-20-B3-Li-Ion-Battery-EU-for-tools-of-the-Parkside-X-20V-Family`.

51. MANGIARI. *Battery powered pi, implement real power button?* [online]. 2019. [visited on 2024-05-16]. Available from: `https://forums.raspberrypi.com/viewtopic.php?t=230195`.

52. *EZ-0048* [online]. 2021. [visited on 2024-05-19]. Available from: `https://wiki.52pi.com/index.php/EZ-0048`.

53. AUTODESK. *Fusion*. 2024. Version 2.0.18961. Available also from: `https://www.autodesk.com/products/fusion-360/`.

54. GELDREICH, Rich. *fpng: Super fast C++ .PNG writer/reader*. 2023. Version 1.0.6. Available also from: `https://github.com/richgel999/fpng`. GitHub repository.

55. *Myutron WF5045-M* [online]. 2023. [visited on 2024-05-23]. Available from: `https://www.sodavision.com/product/myutron-wf5045-m/`.

56. *Myutron XLS01-M* [online]. 2023. [visited on 2024-05-23]. Available from: `https://www.sodavision.com/product/myutron-xls01-m/`.

57. *Schneider-Kreuznach 150mm f9 G-Claron - Lens* [online]. 2024. [visited on 2024-05-23]. Available from: `https://kamerastore.com/products/schneider-kreuznach-150mm-f9-g-claron-1`.

Appendix A

List of abbreviations

AC	Alternating current
ADC	Analog-to-digital converter
AI	Artificial intelligence
AMOLED	Active-matrix organic light-emitting diode
API	Application programming interface
ARM	Advanced RISC Machines
ASA	Acrylonitrile styrene acrylate
BMP	Bitmap
CAD	Computer aided design
CCD	Charge-coupled device
CFA	Color filter array
CMOS	Complementary metal–oxide–semiconductor
CPU	Central Processing Unit
DC	Direct current
DOF	Depth of field
DSLR	Digital single-lens reflex camera
eMMC	Embedded multi-media card
FEE	Faculty of Electrical Engineering
FIT	Faculty of Information Technology
Gbit	Gigabit
GNSS	Global navigation satellite system
GPIO	General-purpose input/output
GUI	Graphical user interface
HAT	Hardware attached on top
HDR	High dynamic range
I^2C	Inter-integrated Circuit
IR	Infrared
JPEG	Joint photographic experts group
LED	Light emitting diode
microSD	Micro secure digital
MOSFET	Metal-oxide-semiconductor field-effect transistor
MTU	Maximum transmission unit
NVMe	Non-volatile memory express

OS	Operating system
PCB	Printed circuit board
PCI-E	Peripheral component interconnect express
PETG	Polyethylene terephthalate glycol
PNG	Portable network graphics
PoE	Power over Ethernet
RAM	Random-access memory
RTC	Real time clock
SBC	Single-board computer
SIMD	Single instruction, multiple data
SKU	Stock keeping unit
SOM	System on board
SSD	Solid-state drive
UART	Universal asynchronous receiver-transmitter
USB	Universal serial bus
Wi-Fi	Wireless fidelity

Appendix B

SVS-VISTEK shr411CXGE datasheet

shr411CXGE

SHR 10GigE

The high resolution SHR series offers outstanding image quality. This is made possible by the excellent physical characteristics of large pixels in large format sensors. The thermally highly optimized housing guarantees highest optical precision. The large M72 lens mount can be adapted to any lens. The economical and high-performance 10GigE interface offers high data transfer speeds of up to 1.1 GB/s for these large images. A special frame grabber is not required.

Technical Highlights

> User defined shading correction
> Automatic and user defined pixel correction
> Extremely low noise
> Excellent dynamic range
> high performance 10GigE interface
> Integrated 4-channel LED strobe controller
> Industrial TTL-24V I/O Interface with Safe Trigger, programmable logic functions, sequencers and timers, RS232

SHR Series shr411CXGE

Resolution [MP]	151 MP
Resolution (h x v)	14192 x 10640 px
Frame rate (max.)	6.1 fps
Chroma	color
Interface	10GigE (RJ-45)

Sensor

Sensor	IMX411AQR
Manufacturer	Sony
Sensor type	Area CMOS
Shutter type	rolling shutter
Sensor size (h x v)	53.36 x 40.01 mm
Optical diagonal	66.69 mm
Sensor format	Medium Format
Pixel size (h x v)	3.76 x 3.76 µm

Camera

Exposure modes	MANUAL;AUTO;EXTERNAL
Trigger modes	INTERNAL;SOFTWARE;EXTERNAL
Exposure time (min)	60 µs
Exposure time (max)	1 sec (external ∞)
Pixel format / max	bayer8, bayer12, bayer16 / 16 bit
Gain modes / max	manual, auto / 36 dB
S/N ratio (max)	46.7 dB (dep. on environment)
Dynamic range (max)	82 dB (dep. on environment)
Internal memory	512 MB SDRAM

Feature Set

Manual white balance	yes
Automatic white balance	yes
LUT	yes
Offset	yes
Binning	yes
Image flip	yes
Shading correction	yes (external)
Defect pixel correction	yes
Sequencer	yes
POE	yes (POE+)(optional)

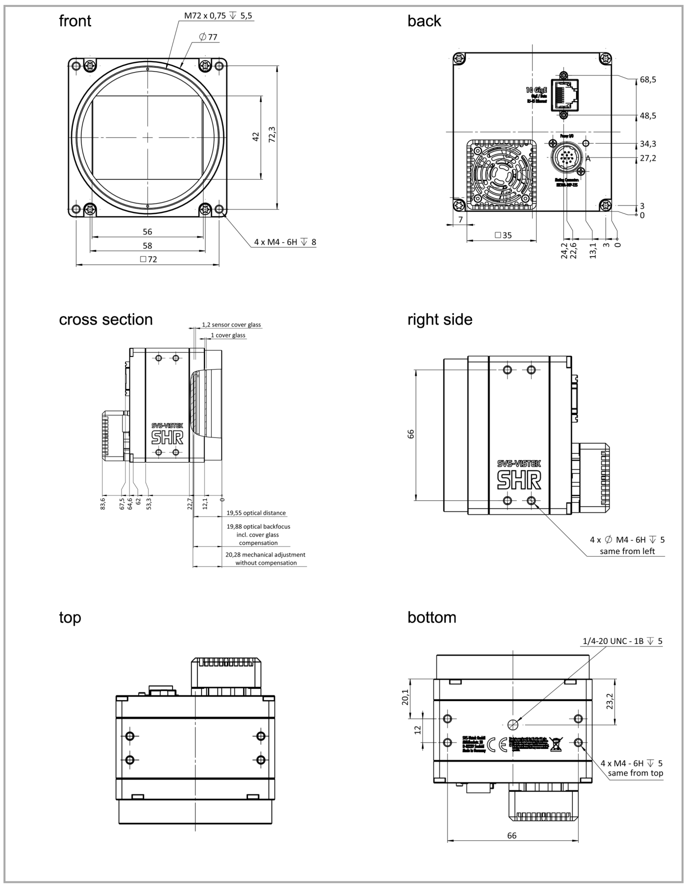
Housing

Lens mount	M72x0,75
Dimensions (w x h x d)	80 x 80 x 83 mm
Weight	580 g
Operating temperature (housing)	-10 to 65 °C
Ambient humidity	10 to 90 % (non-condensing)
Protection class	IP30
Filter-/Coverglass	IR-Cut - 680nm

I/O-Interfaces

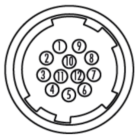
Input up to 24V	2 x
Input OPTO	1 x
Output open drain	4 x
I/O RS-232	1 x
Power supply	10 to 25 V (DC)
Power consumption	19 W (dep. on operating mode)

Dimensions [mm]



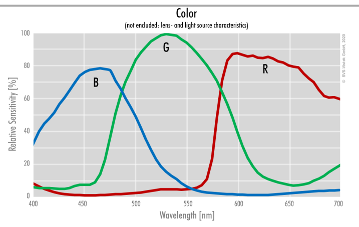
Pinout Mating Connector

Hirose 12 Pin



1	VIN –	(GND)	7	OUT 1	(open drain)
2	VIN +	(10 V to 25 V DC)	8	OUT 2	(open drain)
3	IN 4	(RXD RS232)	9	IN 3 +	(opto In +)
4	OUT 4	(TXD RS232)	10	IN 3 –	(opto In –)
5	IN 1	(0 - 24V)	11	OUT 3	(open drain)
6	IN 2	(0 - 24V)	12	OUT 0	(open drain)

Spectral Response *



* Sensor data – excludes camera cover- or IR-cut filter characteristics

Appendix C

SBC testing results

Measurement	Run 1	Run 2	Run 3	Run 4	Run 5
Boot [s]	52	53	53	52	52
Shutdown [s]	8	8	8	9	8
Write speed [MiB/s]	165	142	145	143	139
Disk eject [ms]	451	286	475	644	490
Power consumption [W]					
Startup	7.57	7.31	7.29	7.51	7.57
Idle	5.69	4.78	4.73	5.89	6.29
Preview 8-bit	8.04	8.00	8.06	8.07	8.10
Preview 16-bit	7.97	8.15	8.13	7.87	8.08
Preview 8-bit avg.	7.50	7.50	7.50	7.50	7.50
Preview 16-bit avg.	7.50	7.50	7.50	7.50	7.50
Saving 8-bit	8.08	8.24	8.18	8.08	8.20
Saving 16-bit	8.40	8.20	8.34	8.27	8.28
Preview processing [ms]					
Image proc. 8-bit	76	76	77	76	77
Image proc. 16-bit	145	146	145	144	145
Frame time 8-bit	740	751	749	735	750
Frame time 16-bit	1642	1655	1650	1648	1662
Saving images [s]					
Saving 8-bit microSD	27.76	27.69	27.76	27.80	27.18
Saving 16-bit microSD	41.81	41.92	41.44	41.90	42.46
Saving 8-bit ex. SSD	18.64	18.13	19.22	19.18	19.32
Saving 16-bit ex. SSD	38.69	38.56	38.54	38.42	38.63
CPU usage [%]					
Preview 8-bit	55.75	55.50	55.50	58.00	57.75
Preview 16-bit	53.50	53.75	52.50	55.00	54.00
Saving 8-bit	57.25	57.50	57.75	58.25	57.00
Saving 16-bit	63.50	62.75	63.25	64.25	63.50

Table C.1: Raspberry Pi 4B (Ubuntu 22.04) measured values

Measurement	Run 1	Run 2	Run 3	Run 4	Run 5
Boot [s]	37	39	38	37	37
Shutdown [s]	10	11	11	15	24
Write speed [MiB/s]	221	213	194	255	197
Disk eject [ms]	628	395	395	397	379
Power consumption [W]					
Startup	5.06	6.20	6.13	6.21	5.93
Idle	4.87	5.25	5.09	4.85	4.89
Preview 8-bit	7.60	7.59	7.73	7.73	7.70
Preview 16-bit	7.90	7.67	7.63	7.82	7.81
Preview 8-bit avg.	6.90	6.90	6.90	6.90	6.90
Preview 16-bit avg.	6.90	6.90	6.90	6.90	6.90
Saving 8-bit	7.43	7.50	7.37	7.33	7.51
Saving 16-bit	7.67	7.66	7.59	7.43	7.67
Preview processing [ms]					
Image proc. 8-bit	75	76	77	75	75
Image proc. 16-bit	124	125	126	125	125
Frame time 8-bit	723	721	734	719	731
Frame time 16-bit	1629	1605	1605	1606	1612
Saving images [s]					
Saving 8-bit microSD	24.67	24.55	24.75	24.178	24.51
Saving 16-bit microSD	36.51	39.51	40.23	37.12	40.14
Saving 8-bit ex. SSD	14.08	14.20	14.43	15.23	14.11
Saving 16-bit ex. SSD	31.63	32.50	31.92	31.39	31.40
CPU usage [%]					
Preview 8-bit	47.50	50.00	49.25	48.25	49.50
Preview 16-bit	48.00	45.75	47.00	47.00	46.25
Saving 8-bit	52.25	50.25	51.00	53.00	50.75
Saving 16-bit	55.50	55.75	55.25	56.00	54.75

Table C.2: Raspberry Pi 4B (Raspberry Pi OS) measured values

Measurement	Run 1	Run 2	Run 3	Run 4	Run 5
Boot [s]	21	21	20	21	20
Shutdown [s]	2	2	2	2	2
Write speed [MiB/s]	248	251	265	248	243
Disk eject [ms]	63	70	88	91	78
Power consumption [W]					
Startup	5.87	6.56	6.35	6.35	6.46
Idle	4.62	4.62	4.61	4.65	4.64
Preview 8-bit	7.71	8.85	8.51	7.96	7.79
Preview 16-bit	7.98	9.15	8.76	8.16	9.11
Preview 8-bit avg.	7.20	7.20	7.20	7.20	7.20
Preview 16-bit avg.	7.20	7.20	7.50	7.20	7.20
Saving 8-bit	7.60	7.68	7.61	7.62	7.80
Saving 16-bit	7.95	8.08	9.33	8.03	7.83
Preview processing [ms]					
Image proc. 8-bit	43	42	41	43	41
Image proc. 16-bit	62	63	61	60	60
Frame time 8-bit	710	708	707	707	711
Frame time 16-bit	1583	1581	1582	1582	1584
Saving images [s]					
Saving 8-bit microSD	11.85	11.00	11.90	11.96	12.61
Saving 16-bit microSD	21.74	21.36	21.76	21.79	21.63
Saving 8-bit ex. SSD	7.56	7.68	7.70	7.51	7.19
Saving 16-bit ex. SSD	15.01	15.21	15.17	14.65	14.63
CPU usage [%]					
Preview 8-bit	44.00	45.50	45.00	45.50	45.50
Preview 16-bit	44.25	44.25	44.25	44.25	44.25
Saving 8-bit	54.00	52.00	52.00	51.75	53.75
Saving 16-bit	59.50	60.00	61.00	60.50	60.25

Table C.3: Raspberry Pi 5 measured values

Measurement	Run 1	Run 2	Run 3	Run 4	Run 5
Boot [s]	39	39	39	39	39
Shutdown [s]	9	9	9	9	9
Write speed [MiB/s]	174	215	221	226	220
Disk eject [ms]	631	280	416	354	261
Power consumption [W]					
Startup	5.10	5.32	5.25	5.41	5.29
Idle	3.82	3.81	3.83	3.81	3.79
Preview 8-bit	5.58	5.63	5.61	5.62	5.65
Preview 16-bit	5.71	5.65	5.66	5.66	5.70
Preview 8-bit avg.	5.10	5.10	5.10	5.10	5.10
Preview 16-bit avg.	5.40	5.40	5.40	5.40	5.40
Saving 8-bit	5.90	5.81	5.76	5.77	5.73
Saving 16-bit	5.90	5.80	5.72	5.69	5.70
Preview processing [ms]					
Image proc. 8-bit	290	291	283	289	294
Image proc. 16-bit	161	165	157	161	159
Frame time 8-bit	754	746	744	744	752
Frame time 16-bit	1660	1672	1677	1669	1669
Saving images [s]					
Saving 8-bit microSD	34.80	34.01	35.38	34.85	37.66
Saving 16-bit microSD	73.59	66.85	68.30	68.38	66.87
Saving 8-bit eMMC	34.88	35.02	35.55	34.64	34.37
Saving 16-bit eMMC	69.99	73.03	72.35	72.53	72.47
Saving 8-bit ex. SSD	33.44	34.20	33.89	32.68	34.26
Saving 16-bit ex. SSD	70.59	71.27	69.28	68.30	69.70
CPU usage [%]					
Preview 8-bit	85.25	84.50	85.00	85.50	86.75
Preview 16-bit	84.75	86.75	84.00	83.75	84.50
Saving 8-bit	86.50	87.25	84.50	86.00	87.25
Saving 16-bit	81.50	83.25	79.00	79.50	80.50

Table C.4: Banana Pi BPI-M5 measured values

Measurement	Run 1	Run 2	Run 3	Run 4	Run 5
Boot [s]	39	39	38	38	38
Shutdown [s]	14	14	14	14	14
Write speed [MiB/s]	240	218	217	219	219
Disk eject [ms]	420	221	459	463	470
Power consumption [W]					
Startup	8.69	8.91	9.11	9.00	9.16
Idle	5.10	5.13	5.13	5.10	5.11
Preview 8-bit	11.78	12.58	13.24	12.40	13.14
Preview 16-bit	11.92	12.56	12.30	12.81	12.90
Preview 8-bit avg.	10.20	10.20	10.50	10.50	10.50
Preview 16-bit avg.	9.90	10.20	10.20	10.50	10.20
Saving 8-bit	12.77	12.09	13.04	13.34	12.15
Saving 16-bit	13.21	12.77	13.03	13.00	13.01
Preview processing [ms]					
Image proc. 8-bit	107	155	105	102	101
Image proc. 16-bit	141	133	130	138	130
Frame time 8-bit	729	745	728	729	730
Frame time 16-bit	1618	1622	1621	1620	1619
Saving images [s]					
Saving 8-bit microSD	27.50	27.32	27.20	27.21	27.18
Saving 16-bit microSD	41.47	41.45	41.39	41.41	41.52
Saving 8-bit eMMC	16.50	16.42	16.54	17.37	15.96
Saving 16-bit eMMC	33.26	32.90	33.66	32.80	33.62
Saving 8-bit ex. SSD	16.61	16.32	16.36	16.38	16.23
Saving 16-bit ex. SSD	34.41	35.42	35.75	35.60	35.45
CPU usage [%]					
Preview 8-bit	37.83	41.17	38.33	37.83	37.83
Preview 16-bit	36.17	33.50	36.00	38.00	34.00
Saving 8-bit	38.67	38.50	38.00	38.17	38.83
Saving 16-bit	39.17	39.83	39.33	41.00	39.33

Table C.5: NanoPi M4V2 measured values

Measurement	Run 1	Run 2	Run 3	Run 4	Run 5
Boot [s]	37	37	37	37	37
Shutdown [s]	3	3	3	3	3
Write speed [MiB/s]	209	199	221	211	204
Disk eject [ms]	265	296	255	282	285
Power consumption [W]					
Startup	7.17	7.17	7.11	7.11	7.17
Idle	4.53	4.62	5.20	4.69	5.35
Preview 8-bit	9.40	9.24	9.18	9.55	9.43
Preview 16-bit	9.49	9.30	9.34	9.54	9.58
Preview 8-bit avg.	8.70	8.40	8.70	8.70	8.70
Preview 16-bit avg.	8.70	8.70	8.70	8.70	8.70
Saving 8-bit	9.87	9.48	9.47	9.52	9.50
Saving 16-bit	9.87	9.50	9.64	9.53	10.07
Preview processing [ms]					
Image proc. 8-bit	51	52	51	52	51
Image proc. 16-bit	68	68	71	71	65
Frame time 8-bit	786	794	792	793	801
Frame time 16-bit	1729	1722	1722	1726	1741
Saving images [s]					
Saving 8-bit microSD	16.46	20.49	20.49	20.38	20.38
Saving 16-bit microSD	32.92	32.25	32.37	32.15	32.63
Saving 8-bit eMMC	17.55	14.22	14.16	14.17	14.02
Saving 16-bit eMMC	30.93	31.81	32.31	29.32	31.84
Saving 8-bit ex. SSD	13.84	13.85	15.59	15.31	15.28
Saving 16-bit ex. SSD	30.22	29.95	30.18	30.53	30.37
CPU usage [%]					
Preview 8-bit	63.38	62.38	64.00	63.13	62.63
Preview 16-bit	64.75	64.13	64.75	64.63	64.88
Saving 8-bit	74.63	71.00	70.38	72.00	71.88
Saving 16-bit	74.25	73.13	72.75	73.63	74.00

Table C.6: Khadas VIM4 measured values

Appendix D
Questionnaire

The usability testing was conducted in the Czech language. Therefore, the questionnaire is in Czech with English translation either below or in parentheses.

Dotazník po práci s fotografickým přístrojem

(Questionnaire after working with box camera)

* Indicates required question

1. Účastník *

   Participant

   Mark only one oval.

   - A
   - B
   - C
   - D

2. Jak často fotíš? *

   How often do you take pictures?

   Mark only one oval.

   - Každý den (Every day)
   - Párkrát týdně (Few times a week)
   - Párkrát ročně (Few times a year)
   - Nefotím (Not at all)
   - Other: ______________________

3. Máš předchozí zkušenosti s fotoaparátem (mimo mobilního telefonu)? *

Do you have previous experience with a camera (outside of a mobile phone)?

Mark only one oval.

- ⬭ Ano (Yes)
- ⬭ Ne (No)

Navigace GUI (GUI Navigation)

4. Bylo rozmíštění a velikost ovládacích a informačních prvků přehledné? (čitelnost textu, velikost posuvníku apod.) *

Was the layout and size of the controls and information elements clear? (readability of text, size of scroll bar, etc.)

Mark only one oval.

- ⬭ Ano (Yes)
- ⬭ Ne (No)

5. Pokud ne, co bys změnil?

If not, what would you change?

____
____
____
____
____

6. Bylo dané nastavení vždy na očekávaném místě? (např. zapnutí blesku, změna expozičního času, export fotografií..) *

Was the setting always in the expected place? (e.g. turning on the flash, changing the exposure time, exporting photos...)

Mark only one oval.

- Ano (Yes)
- Ne (No)

7. Pokud ne, co a kam bys přemístil?

If not, what and where would you relocate?

8. Měl jsi vždy zpětnou vazbu na provedenou akci? (např. věděl jsi, že fotoaparát zareagoval na stisknutí spoušti, změnu parametrů..) *

Did you always have feedback on the action you took? (e.g. did you know that the camera reacted to a trigger, a parameter change...)

Mark only one oval.

- Ano (Yes)
- Ne (No)

9. Pokud ne, kde zpětná vazba chyběla?

If not, where was the feedback missing?

10. Bylo grafické rozhraní konzistentní? *

Was the GUI consistent?

Mark only one oval.

- ( ) Ano (Yes)
- ( ) Ne (No)

11. Pokud ne, kde sis všiml nekonzistentnosti?

If not, where did you notice the inconsistency?

12. Připadaly ti některé úkoly obtížné? Pokud ano, napiš čísla a důvody.

Did you find some of the tasks difficult? If so, write down the numbers and reasons.

Fyzické rozložení prvků Physical layout of the elements

13. Změnil bys rozložení některých prvků na těle fotoaparátu? *

Would you change the layout of some of the elements on the camera body?

Mark only one oval.

- Ano (Yes)
- Ne (No)

14. Pokud ano, co a kam bys přemístil?

If so, what and where would you relocate?

15. Přidal bys nějaký ovládací či informační prvek na tělo fotoaparátu? (madlo, LED diodu, tlačítko, apod.) *

Would you add any controls or information on the camera body? (handle, LED, button, etc.)

Mark only one oval.

- Ano (Yes)
- Ne (No)

16. Pokud ano, co a kam bys přidal?

If so, what and where would you add?

Appendix E

Contents of the attached media

```
application ............................ GUI application source code
├── components.xlsx .................. spreadsheet with used components
├── images
│   ├── components ........................... images of used components
│   ├── finalDesign ............... images of final design and its assembly
│   ├── firstDesign ............... images of first design and its assembly
│   └── gui ......................... screenshots from the GUI application
├── manual.pdf ............................. manual for the box camera
├── models
│   ├── printables. .................. printable 3D models of both designs
│   ├── FinalDesign.f3z.............Fusion project file of the final design
│   └── FirstDesign.f3z.............Fusion project file of the first design
├── questionnaire
│   ├── questionnaire.pdf ...................... printable questionnaire
│   └── responses.pdf...........................participants responses
├── text .............................. LaTeX source codes of the thesis
└── thesis.pdf ......................................... thesis text
```