Chapter 1

Introduction

1.1 Problem Statement

The surge in Virtual Reality (VR) Head Mounted Display (HMD) technologies has opened new creative avenues, offering users immersive experiences [1], [2]. However, the challenges of exclusivity and high costs in existing solutions for recording stereoscopic video limit their potential [3]–[5]. This holds significance for VR's broader adoption, as low user retention is often linked to a lack of exclusive content maximizing the medium's advantages [6]. With the hassles of wearing VR HMDs compared to watching television or using mobile phones, there needs to be a more compelling reason to use VR HMDs for media consumption, thus underlines the need to empower users to effortlessly create their stereoscopic content, enhancing the appeal of VR [7]. Most VR contents that focus only on factors like Field of View (FOV) found on 180 or 360 degree media often results in visually uninteresting videos lacking depth and resolution [7], [8]. The exclusive depth perception feature on VR HMDs and certain autostereoscopic displays has the potential to revive lifelogging, offering individuals an immersive experience to relive memories [9].

1.2 Goals

This project endeavours to surmount these challenges by:

1. Developing an Open-Source, Cost-Efficient Stereo Video Camera Hardware System.
2. Implementing a Video Processing Pipeline.
3. Creating Intuitive VR Software for Browsing and Viewing Stereoscopic Media.

This project not only addresses the critical gaps in hardware accessibility and the video processing pipeline but also represents a significant enhancement to existing stereoscopic video VR software. By fostering an open-source, modular, and cost-effective approach, this initiative strives to democratize VR content creation, making it more widely accessible and fostering innovation in the field.

Chapter 2

Background

2.1 Lifelogging

Lifelogging, a contemporary term encapsulating habitual documentation of one life akin to social media practices, distinguishes itself through its methodical and routine nature. The motivation for lifelogging extends beyond the sporadic capturing of moments to a deliberate effort to systematically record and preserve lifetime memories [10]. From the definition itself, the Lifelog data can be in any format such as texts from diaries, health sensor data or even digital footprint from application use. However, in this project, I will be focusing on visual lifelogging which consists of images and videos [11].

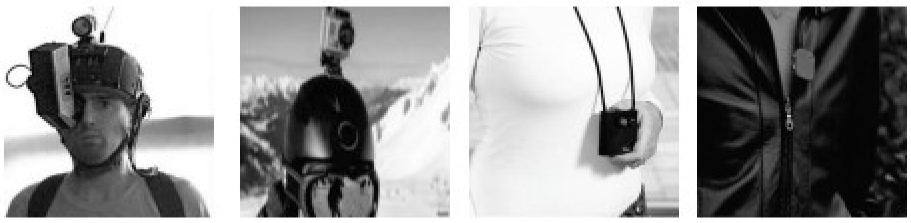

FIGURE 2.1: Evolution of wearable camera technology. From left to right: Mann (1998), GoPro (2002), SenseCam (2005), Narrative Clip (2013), reproduced from [11]

Examples of existing wearable cameras specifics that are used for visual lifelogging are shown in Fig 2.1. However, in recent years with rapid advancement in the miniaturization of consumer products such as smartphones, camera sensors and storage devices, companies like Meta [12] and Snapchat also released their versions of smart glasses, similar to Google Glass [13] released years prior. Although the main appeal is the smart assistance features for convenience, this form factor and placement of the camera is perfect for visual lifelogging aimed for VR userbase especially as already proven by Snapchat Spectacles 3 [3] which can capture 3D photos and videos at 60fps.

2.2 Stereo Camera

Stereo camera refers to a camera with stereoscopic imaging capabilities, this is simply achieved using 2 cameras aligned side-by-side at a distance approximately the same as average human inter-pupillary distance (IPD) to mimic human vision [14], [15]. This gives the content viewed proper depth definition as in real life. This idea can also be extrapolated to higher Field Of View (FOV) content such as 180/360 degrees content but requires either a more complicated setup, computation or both [16], [17]. A higher resolution is also needed as the content is stretched into a larger area [7], this in turn increase storage and processing requirement.

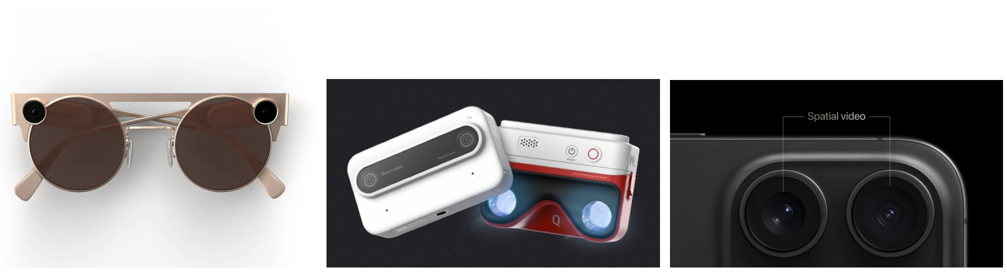

FIGURE 2.2: Commercially available consumer-level stereo cameras. From left to right: Snapchat Spectacles 3[3], Kandao Qoocam Ego [4], iPhone 15 Pro/Pro Max Spatial Video[18]

Examples of consumer-level stereo cameras commercially available to buy right now as shown in Fig 2.2 are Snapchat Spectacles 3 [3], Kandao QooCam Ego [4] which includes an integrated stereo/VR viewer and technically, iPhone 15 Pro/Pro Max [5] with version iOS 17.2 and above [18] which had Spatial Video recording enabled. This move is obviously to entice consumers using Apple products to buy their upcoming Apple Vision Pro VR HMD [2] which will make stereo camera and content more mainstream.

2.3 Virtual Reality (VR)

Virtual Reality (VR) transcends the conventional by offering users a virtual 3D environment, or an immersive emulation of real life overlaid with digital objects which usually referred to as Mixed Reality (MR). The emotional attachment fostered by VR content stems from its heightened realism thanks to proper 3D depth perception, creating a sense of physical presence within the virtual space [7]. This emotional resonance distinguishes VR content from traditional media, providing a compelling reason for its adoption in content consumption including lifelog data.

FIGURE 2.3: Example of VR HMDs. Meta Quest 3[1] and Apple Vision Pro[2]

Contrary to fully immersive Free Viewpoint Video (FVV) [19] or scene reconstruction [20], the adoption of the SBS format is grounded in the current limitations of VR Head-Mounted Display (HMD) hardware. While FVV offers unparalleled immersion and depth perception through 6DoF [21], practical constraints dictate a compromise. This decision is further supported by the spatial video capabilities of the iPhone 15 Pro, which, despite being 1080p at 60fps, aligns with the current standards for windowed style viewing. The emphasis here is not on resolution but on perceived Pixel Per Degree (PPD) which is higher as the virtual window/screen is further, and lower FOV needed to stretch out across [7]. Choosing windowed viewing over panoramic alternatives (180/360 degrees) acknowledges the balance between realism and current technological constraints.

A poignant illustration of this concept is evident in the Black Mirror episode which explores the immersive preservation of memories in an eye-camera format [22]. This format aligns with human visual perception, eliminating the necessity for constant 3 Degrees of Freedom (3DoF) as the human gaze is not constantly omnidirectional. Although a higher FOV would enhance the lifelogging experience, it remains cost-prohibitive at present (full human FOV is approximately 200 to 220 degrees [23]).

2.4 User Interface in VR

Integrating VR software for content browsing amplifies the lifelogging experience. Leveraging the full potential of 6DoF through innovative UI design, expanded screen space [24], and interactive features such as timeline shelves, this approach redefines how users engage with their memories. The incorporation of hand tracking and eye tracking further enhances the intuitive and immersive nature of content navigation within the VR environment as shown in visionOS demo of Apple Vision Pro in Figure 2.4.

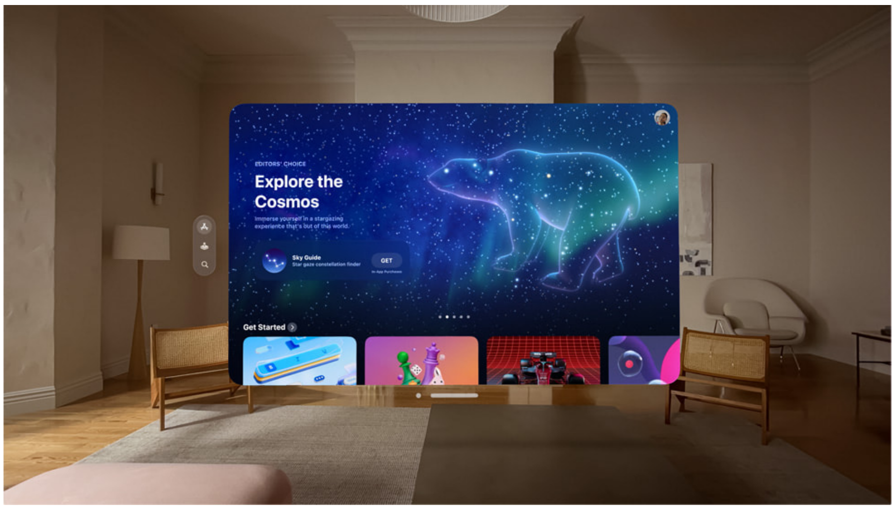

FIGURE 2.4: Apple's visionOS innovative gesture and eye tracking UI navigation, making full use of 3D space for windows elements with variable sizes.[2]

Other than that, the existence of a window or virtual screen to host the content will cause less motion sickness as users have virtual anchors to the virtual space [25], [26]. Lower FOV helps as the experience will be just like watching movies in 3D cinema[27]–[29].

The combination of lifelogging and VR represents a novel and niche subset within both domains. As technology advances and user preferences evolve, this integrated approach is poised to become more mainstream, particularly in the realm of personal videos, memories, and photos. The synergy between lifelogging and VR anticipates a future where individuals seamlessly capture, relive, and share their most cherished moments in a more immersive and engaging manner.

Chapter 3

Design and Progress

3.1 User Stories

User Story 1 (Hardware Development): As a VR lifelogger, I must have access to an open-source stereo video camera hardware system using off the shelf components. It should be cost-effective and easily obtainable, providing a reliable solution for capturing stereoscopic content.

User Story 2 (Video Processing Pipeline): As a content creator, I must have the option to use a video processing pipelined service or scripts for stitching two mono stills and videos into stereoscopic Side-By-Side (SBS) format. Mixing audio files is also needed for a stereo soundscape during playback. These should be automated.

User Story 3 (VR Software Application): As a VR enthusiast, I should have access to an intuitive VR software application for intuitive file browsing and content viewing. It should take advantage of VR medium, allowing me to relive and interact with my stereoscopic memories effortlessly.

3.2 MoSCoW Requirement

Thus, the project is divided into 3 distinct parts, following the MoSCoW requirement framework (M: Must-haves, S: Should-haves, C: Could-have, W: Wont-have).

3.2.1 Hardware Development

- **M:** Develop an open-source, modular stereo video camera system with the Raspberry Pi Pico microcontroller. Ensure it is low-cost and easily accessible.
- **S:** Consider additional features or improvements based on feasibility, such as using an onboard rechargeable Li-Po battery circuit instead of a power bank to power it.
- **C:** Explore advanced features like wireless connectivity or additional sensor integration, time permitting and if resources allow.
- **W:** Exclude features or components that are deemed impractical or beyond the scope of the project, such as higher resolution or different video format (180/360).

3.2.2 Video Processing Pipeline

- **M:** Implement a video processing pipeline for transforming mono stills and video into stereoscopic Side-By-Side (SBS) format. Synchronize audio files for an immersive surround soundscape.
- **S:** Explore additional video processing features, such as metadata tagging through object and scene detection using existing libraries and tools.
- **C:** Automated video stabilization or advanced filtering options, based on available resources and time constraints.
- **W:** Exclude overly complex video processing tasks that may hinder the project timeline or exceed available resources, such as 3D depth reconstruction.

3.2.3 VR Software Application

- **M:** Develop an intuitive VR software application for seamless file browsing and content viewing. Ensure compatibility with the stereo video format.
- **S:** Implement innovative UI designs for enhanced interaction within the VR environment.

- **C:** Explore the integration of hand tracking, and eye tracking, if resources and time allow.
- **W:** Exclude overly ambitious features that may compromise the core functionality or extend the project beyond feasible timelines, such as a personal AI assistant.

3.3 Complete System Flow

The algorithm in the first part of the project, 3.4 runs onboard the microcontroller, the second part, 3.5 is designed to run on a machine running Windows 10 while the last one 3.6 has options to run on any OpenXR [30] capable devices, however, PCVR support is prioritized. Figure 3.1 shows a flowchart representing the flow of different parts of the project.

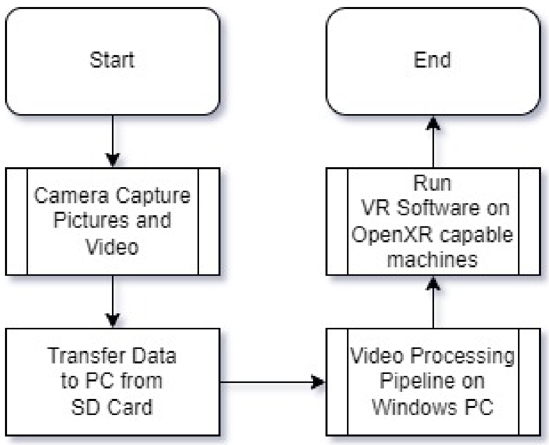
Start
End
Camera Capture Pictures and Video
Run VR Software on OpenXR capable machines
Transfer Data to PC from SD Card
Video Processing Pipeline on Windows PC

FIGURE 3.1: Overall flow of the project

3.4 Hardware System

3.4.1 Cost Analysis

The budget provided for the project is £150, and the aim is to get all components under £100, making it a low-cost solution compared to alternatives that usually cost around £300 [3], [4] or more [5]. The most expensive part, as expected, is the camera modules, which are around £35 each. However, they boast a 5MP sensor and are capable of taking 1080p60fps video, justifying their cost. The exact component details and costs can be seen in Appendix A.

3.4.2 Components

After comparing costs, availability, ease of use, and hardware constraints, the choice of the microcontroller is Raspberry Pi Pico due to its low cost, cohesive documentation, compatible hardware, and, most importantly, buffer size, which is important to get high enough resolution images/video to prevent motion sickness.

To achieve an immersive experience, audio is also an important variable. Thus, the use of 2 independent electret microphone modules is added to work in tandem with the camera. This, in theory, should make it possible to achieve stereo sound channels for each ear. An SD card extension board is also added to host the SD card that holds all the data.

For the complete system, one of the MCUs will act as the main MCU, it also has a status LED to show when it is recording or taking pictures. It sends signals through GP18 to the other two mono camera systems when the button is pressed to synchronise the capture timing. Trying to make it power-efficient might prove challenging and needs to deal with additional circuitry, so for now, the system will be powered with a power bank to the micro-USB on the main Pico board. The circuit schematics for mono camera setup and complete system can be seen in Figure 3.2.

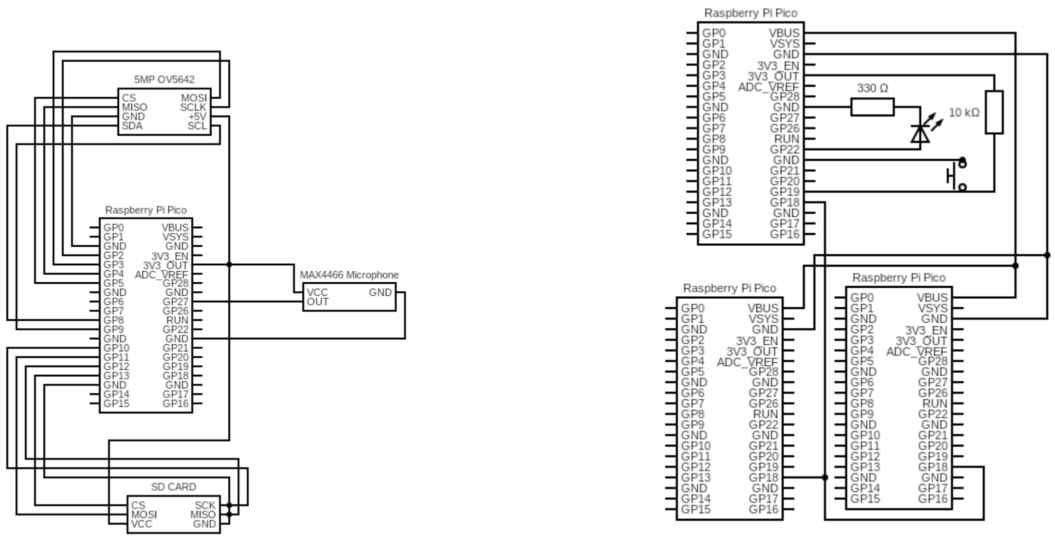

FIGURE 3.2: Mono Camera and Complete System Prototype Circuit

The components testing has been performed with success and further information can be seen in the Appendix C.

3.4.3 POV Considerations

The initial draft design is to have both cameras mounted on the side of eyeglasses, akin to Ray-Ban Meta and Snapchat Spectacles glasses. However, after acquiring the camera modules and other components, it is deemed too unwieldy to fit the electronics into a small form factor. The main reason for this choice is to capture the footage as close as how humans see the world, and mounting the camera close to the eye would achieve that compared to a chest mount design.

To compromise, a head-mounted design is chosen, where the POV is higher than eye level, but the camera movement/rotation will still follow the head movement and should give a realistic enough POV as seen in Figure 3.3.

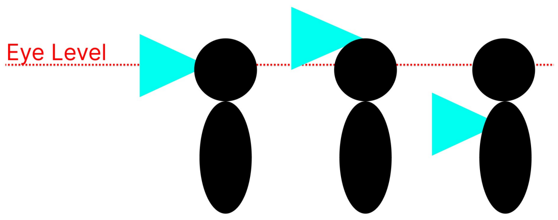

FIGURE 3.3: Designs and POV Example. Left to right: Glasses, Head-Mounted and Chest-Mounted

3.4.4 Onboard Embedded Software Algorithm

In normal lifelogging mode, the cameras would take stereo still images on a timed interval, which will be decided through trial and error and optimization depending on how much storage the images occupy and the power efficiency of the algorithms. However, to get a more immersive experience, video is also needed, and a button can be used to manually start and stop recording, with an LED being an indicator when it's recording. The saved files should be aptly named for easier processing later, in a standardized DATE_NUMBER format with L or R prefixes for the left and right camera respectively. The proposed algorithm can be seen in Figure 3.4

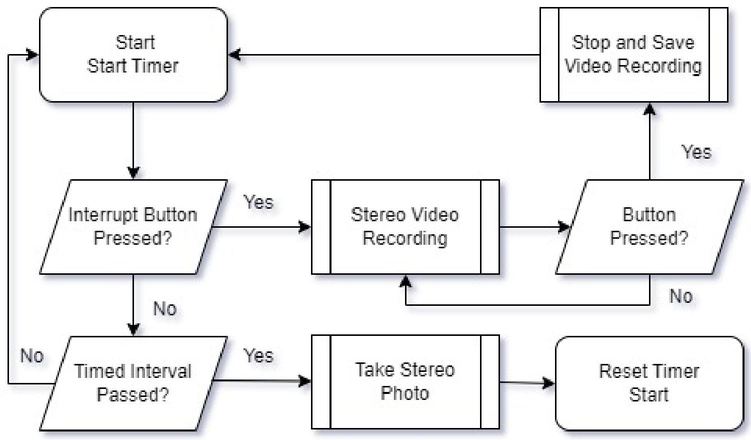

FIGURE 3.4: Flow Chart for Onboard Embedded Software Algorithm

3.5 Video Processing Pipeline

3.5.1 Justification

The justification behind implementing a pipelined process for video processing is to offload the work to a more capable machine, this stems from the acknowledgement that attempting to execute this intricate task directly on the Raspberry Pi Pico would introduce unnecessary complications exceeding its memory and processing power limitations. This is particularly crucial for the generation of Side-By-Side (SBS)/3D format from two mono/2D stills using FFmpeg. Additionally, the inclusion of a custom metadata tagging system for object and scene detection is pivotal. This customized metadata enhances the overall VR experience by enabling advanced filtering and indexing, thereby optimizing the video and image browsing experience within the VR application.

3.5.2 FFmpeg for stitching

The utilization of FFmpeg [31] for mono to stereo stitching serves as a cornerstone in the pre-processing pipeline. FFMPEG, a powerful multimedia processing tool, efficiently transforms mono/2D videos and images into the desired stereoscopic Side-By-Side (SBS)/3D format as seen in Fig 3.5. This process is instrumental in creating a lifelike and immersive visual experience for VR content, aligning with the project's goal of democratizing VR content creation. Refer to Appendix D for the automated FFMPEG PowerShell script.

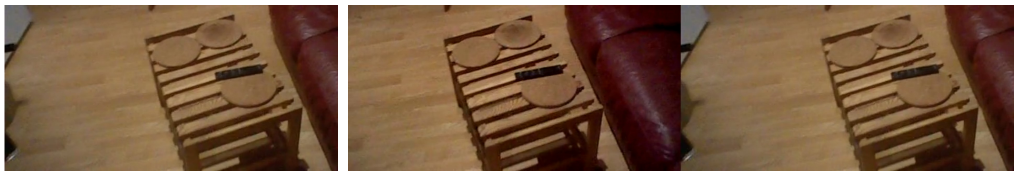

FIGURE 3.5: Examples of Mono (left) and Stereo (right) images that are stitched from two mono stills

3.5.3 Metadata tagging

Incorporating object and scene detection algorithms further enhances the preprocessing pipeline. Utilizing well-established computer vision techniques, this step automatically identifies and tags objects and scenes within the videos and images. This integration not only enriches the visual content but also establishes the groundwork for sophisticated filtering and indexing capabilities within the VR application. The custom metadata tagging component allows for more refined filtering options, enabling a personalized categorization of lifelogging data. This bespoke metadata layer optimizes the VR content browsing experience, ensuring users can effortlessly locate and revisit specific moments within their immersive collection. At the simplest, this will just be .json files corresponding to each media file.

3.6 Video Player VR App

3.6.1 Game Engine – Godot 4.2

The selection of the Godot 4.2 game engine[32] for the development of the video player VR app is rooted in the project's commitment to Free and Open Source Software (FOSS) principles. Unlike more established game engines such as Unity or Unreal Engine, Godot aligns with the FOSS ethos, making it the ideal choice for this project. Despite having fewer documentation and examples compared to its counterparts, this presented an opportunity for active participation in its development. Consultation with the Godot community, particularly through their Discord channel, guided the decision-making process, revealing that utilizing shaders is the most straightforward approach for rendering stereo Side-By-Side (SBS) video playback.

3.6.2 SBS Video Projection

In the implementation of stereo video playback using the Godot 4.2 game engine, shaders play a crucial role in rendering. The process is straightforward, aligning with the Side-By-Side (SBS) format of the video. The left half is rendered for the left camera eye, and the right half for the right eye. The gdshaders code in Appendix E utilized adheres to this logic, ensuring an efficient rendering process.

Moreover, the user interface (UI) for the video player VR app is designed within a 3D space, a common practice for VR applications and games. However, a challenge arises as the video player operates as a 2D screen within this 3D environment. Initial attempts to integrate the shader script within the same scene as the videostreamplayer node, following a tutorial by Malcolm Nixon on YouTube, faced difficulties. Subsequent experimentation and development efforts proved inconclusive.

Fortunately, after seeking guidance from the Discord community, Malcolm Nixon, the tutorial's author, provided a pivotal solution [33]. The revised method involves instantiating a 2D screen videostreamplayer node in the main scene, where the shader is then applied. This modification successfully addresses the challenges encountered during the development process. An example of the app running is depicted in Figure 3.6.

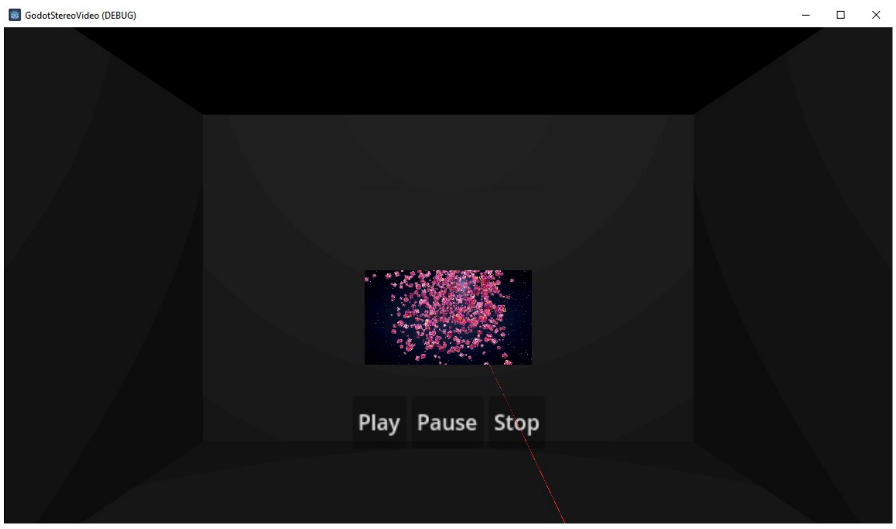

FIGURE 3.6: Images of App Running

3.6.3 File Browsing

While file browsing functionality has not been fully implemented, conceptualization has begun. Proposed ideas include leveraging metadata tagging for specific searches and employing bookshelves or 3D objects as interfaces for navigating between months or weeks, deviating from the conventional 2D screen approach to enhance interactivity.

To optimize browsing efficiency, the screen space may be expanded to resemble an ultra-wide monitor or more, enabling users to view a greater number of files in a single window. This approach capitalizes on the immersive capabilities of VR, utilizing the 360-degree view and 6 Degrees of Freedom (6DoF). Additionally, the implementation of the depth axis remains contingent on contextual considerations and future developments [24].

Chapter 4

Plan of Remaining Work

This project consists of three distinct components, simplified to their basic elements to demonstrate a minimum viable product prototype. Therefore, meticulous planning and initial design drafting are essential, and this has been executed thoroughly, as evidenced. However, the challenging phase lies in translating the bulk of the design into a functional solution, which is where my focus will be directed in the upcoming phases.

As observed in the Gantt charts in Figure 4.1 and Figure 4.2, the proposed timeline has encountered some deviations, primarily due to unforeseen risks such as the microcontroller breaking due to a short circuit during user testing and instability in the connections of the wires provided by manufacturers, thereby causing delays in testing. This had been recorded in the risk assessment in the Appendix B. Nevertheless, all three main components underwent preliminary testing, and initial work was successfully accomplished.

Figure 4.2 displays the executed Gantt chart timeline alongside the planned one, distinguished by varying colour densities. Through this comparison, I recognized the necessity of refining the specificity of tasks in my Gantt chart for future reference.

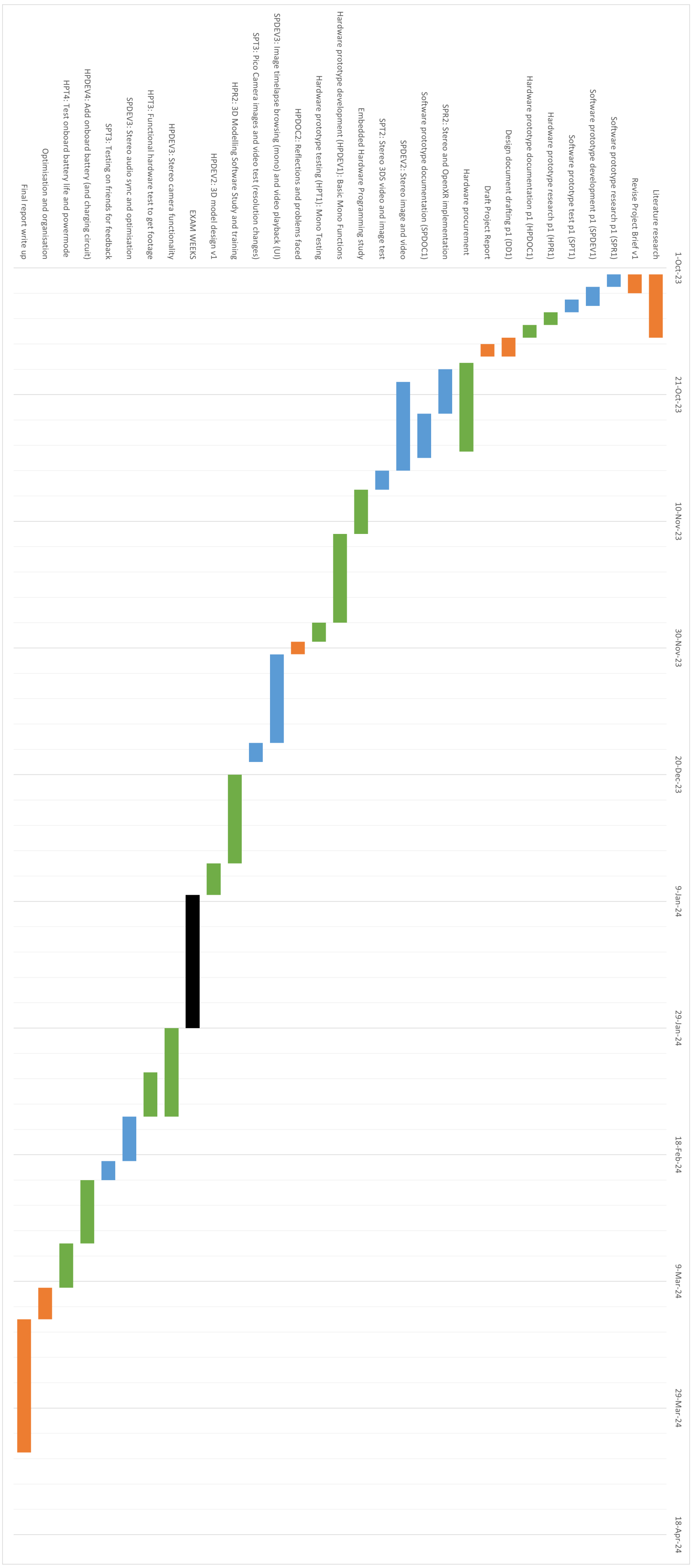

FIGURE 4.1: Planned Gantt Chart. Legend: Orange - Report or writing related task, Blue - Software related task, Green - Hardware related task.
]

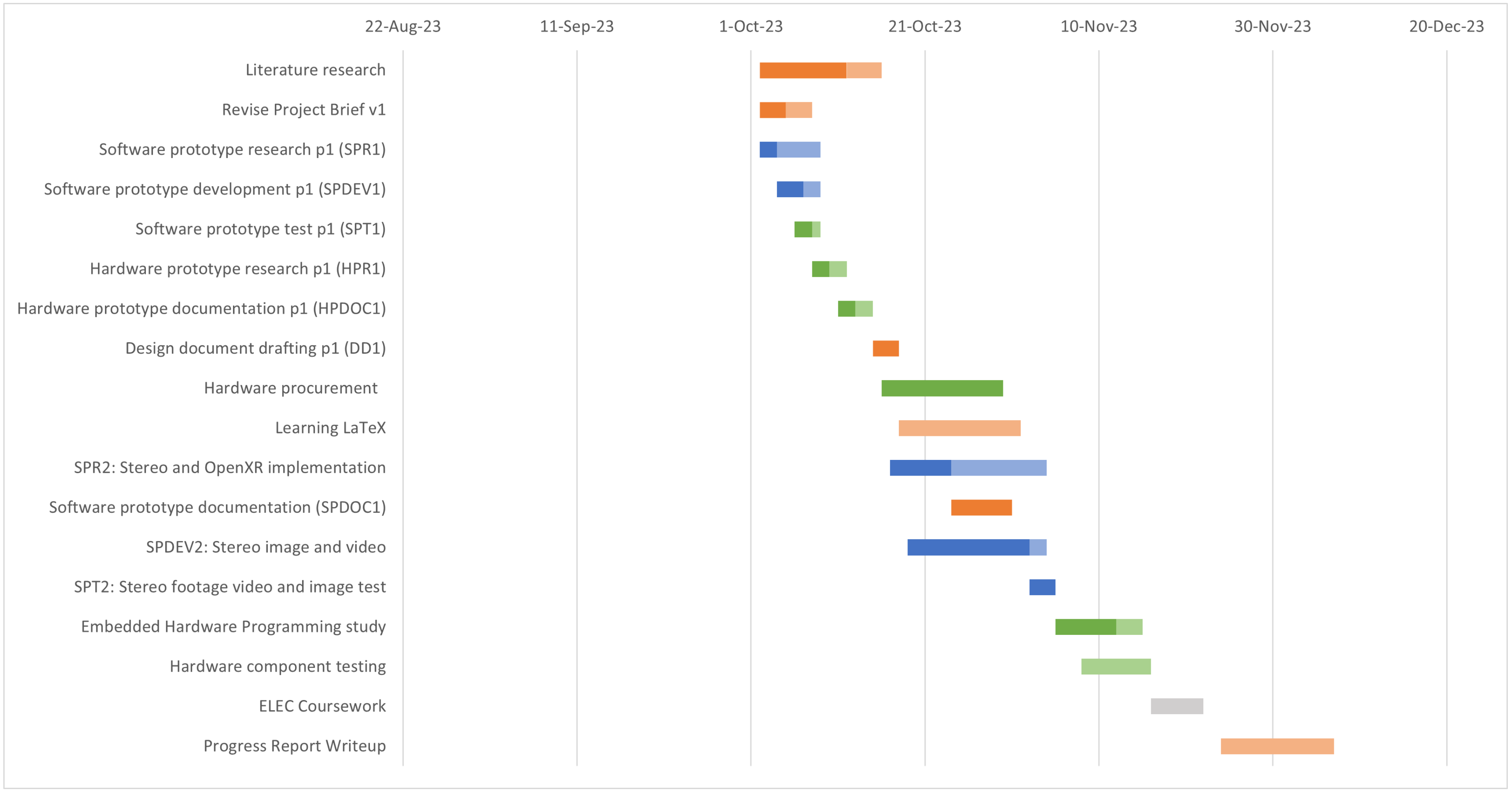

FIGURE 4.2: Combined Gantt Chart. Legend: Orange - Report or writing related task, Blue - Software related task, Green - Hardware related task. Softer color represents tasks over expected time or unexpected tasks compared to the Planned Gantt chart.