UNIVERSITY OF SOUTHAMPTON
Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

A progress report submitted for the award of
MEng Electrical and Electronics Engineering

Supervisor: Dr. Tom Blount
Examiner: TBD

Open-Source Stereo Video Camera System and Software Implementation for Virtual Reality Lifelogging and Content Creation

by Muhammad Hazimi Bin Yusri

April 23, 2024

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND PHYSICAL SCIENCES
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

A project report submitted for the award of MEng Electrical and Electronics Engineering

by Muhammad Hazimi Bin Yusri

NEED REDO This project addresses challenges related to the accessibility and cost-effectiveness of stereoscopic video recording for Virtual Reality (VR) Head Mounted Display (HMD) technologies. The main goal is to democratize VR content creation by developing an open-source, modular stereo video camera system, implementing an efficient video processing pipeline, and creating an intuitive VR software for exploring stereoscopic media. At its core, the project transforms lifelogging into a personalised and artisanal content creation process. The system will incorporates advanced object and scene detection algorithms, allowing custom metadata tagging within captured videos and images. This integration enhances the lifelogging experience, allowing for personalised categorisation and efficient retrieval of specific moments, objects, or scenes. Ultimately, the project aims to provide a practical and accessible solution for stereoscopic lifelogging, empowering users to curate and revisit their experiences immersively in VR.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thanks Dr Tom Blount for his tremendous support, ideas and approval for this project, and also to Godot XR community especially Bastiaan Olij and Malcolm Nixon for help with shaders, stereo video example and relentless guidance.

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 Objectives

This project endeavours to surmount these challenges by:

1. Developing an open-source, cost-efficient stereo video camera hardware system to record stereoscopic content with focus on life-logging.
2. Implementing a stereo processing pipeline to automatically transform captured content to correct format and further processing.
3. Creating intuitive VR software for browsing and viewing captured stereoscopic content.

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.

1.3 Overview

1. **Introduction**: This chapter sets the stage for the entire report by introducing the problem statement, outlining the objectives of the project, and providing a brief overview. It gives readers a clear understanding of what the project aims to achieve and why it's significant.
2. **Background**: The background chapter delves into the context surrounding the project, including relevant literature reviews, conceptual frameworks, and any contextual information necessary to understand the project's foundation. It provides a deeper understanding of the theoretical underpinnings and previous research relevant to the project.
3. **Design**: In the design chapter, the focus is on showcasing the final design decisions made for the project. This includes detailing the requirements analysis, system architecture, and key design decisions that informed the development process. It provides insights into how the project was conceptualized and planned before implementation.
4. **Implementation**: Here, the implementation chapter delves into the details of each part of the project, discussing how the design decisions were put into practice. It covers the development process, component details, hardware development, stereo processing pipeline, and VR software. This chapter provides a comprehensive look at the technical aspects of bringing the project to life.

5. **Testing and Evaluation**: The testing and evaluation chapter discusses the strategies employed to test the project's functionality and performance. It presents the results of these tests and evaluates how well the project meets its objectives. This chapter helps assess the project's effectiveness and identify areas for improvement.

6. **Conclusion**: Finally, the conclusion chapter summarizes the key findings of the project, reflects on its overall success, and discusses potential avenues for future work. It provides closure to the report by reiterating the project's significance and highlighting its contributions to the field.

Chapter 2

Background

2.1 Literature Review

2.1.1 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].

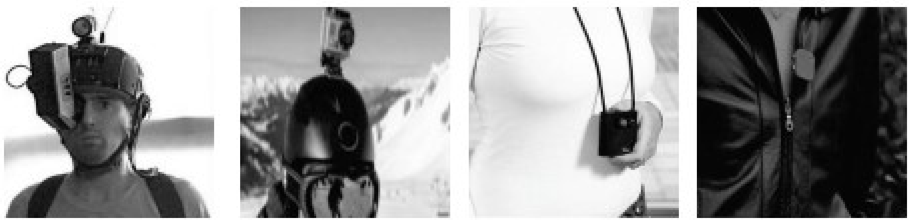

FIGURE 2.1: Evolution of wearable camera technology. From left to right: Mann (1998), GoPro (2002), SenseCam (2005), Narrative Clip (2013), reproduced from [11]

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.

2.1.2 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.

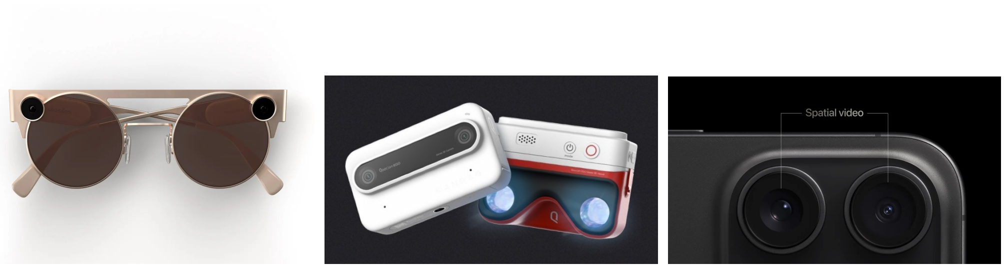

FIGURE 2.2: Commercially available consumer-level stereo cameras. From left to right: Snapchat Spectacles 3[3], Kandao Qoocam Ego [4], iPhone 15 Pro/Pro Max Spatial Video[18]

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.

2.1.3 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]).

2.1.4 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.

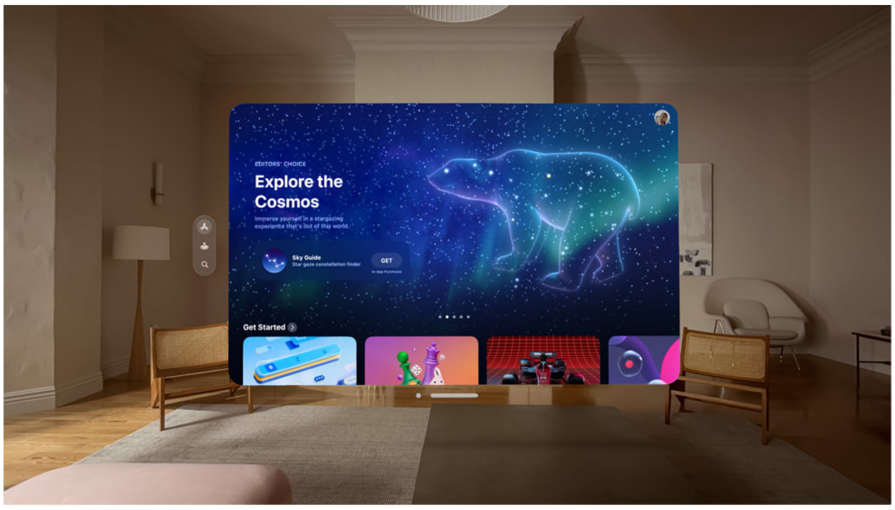

FIGURE 2.4: Apple's visionOS innovative gesture and eye tracking UI navigation, making full use of 3D space for windows elements with variable sizes.[2]

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.

2.2 Conceptual Framework

Present a conceptual framework or theoretical basis for your project, explaining how it fits within existing knowledge or research.

2.3 Context

Provide context for your project by discussing any industry trends, technological advancements, or societal implications.

Chapter 3

Design

3.1 Requirements Analysis

The process of gathering requirements is carried out through the User Stories format, involving the following stakeholders: **VR Users**, **Stereo Content Producers**, **Developer** and **Researchers** specializing in the fields of life-logging and VR:

- **User Story 1:** As VR life-logger **developers**, we need access to an open-source stereo video camera hardware system built using off-the-shelf components. It should be cost-effective and readily available, providing a reliable solution for capturing stereoscopic content.
- **User Story 2:** As **stereo content producers**, we require an automated stereo processing pipeline tool for stitching two mono stills and videos into stereoscopic Side-By-Side (SBS) format for seamless sharing. The image and video quality should be at least in HD (720p) at 30 fps minimum to avoid blurry and nauseating footage.
- **User Story 3:** As **VR users**, we seek an intuitive VR software application for easy file browsing and content viewing. It should leverage the VR medium, allowing us to relive and interact with our stereoscopic memories effortlessly.
- **User Story 4:** As VR life-logger **researchers**, we require access to Free and Open-Source Software (FOSS) processing tools capable of automatically tagging scenes and objects within captured stereoscopic images and

videos. These tools should facilitate efficient organization and categorization of large datasets, enabling us to conduct in-depth analysis and research on life-logging experiences in virtual reality environments.

These requirements are then rewritten in the form of functional and non-functional requirements for better planning and evaluation at the end.

NEED REVISION!

3.1.1 Functional Requirements (FR)

FR1:

3.1.2 Non-Functional Requirements (NFR)

3.2 System Architecture

The system architecture comprises three primary components: **hardware**, **stereo processing pipeline**, and **VR software application**. Each component plays a crucial role in capturing, processing, and presenting stereo media.

1. **Hardware:** The selected hardware includes a Raspberry Pi 5 8GB running Raspbian Lite OS, 2 Raspberry Pi Camera Module 3 (Wide), 2 20cm mini CSI-MIPI cables, and a USB microphone. This configuration enables Full-HD (1920x1080) resolution stereo image and video capture at 30fps. Despite some compromise on audio quality to meet the budget**??** constraint of under £150, the chosen components provide satisfactory performance. Additionally, a custom enclosure is designed and 3D printed**??** to ensure precise alignment of stereo pairs. This is then mounted on a headwear to get desired POV**??**. A python script is used to control the algorithm behaviour for the media capturing process.**??**

   IMAGE OF COMPLETED HARDWARE PROTOTYPE,

2. **Stereo Processing Pipeline:** The stereo processing pipeline runs on the main PC that hosts the files and VR software. An FTP [footnote] through SSH[footnote] setup facilitates media transfer to the main PC for processing

and storage via FileZilla[footnote]. Python scripts is used with PILLOW for image stitching and subprocess to invoke FFMPEG**<empty citation>** for video manipulation tasks such as audio mixing and stereo pair stitching. Furthermore, JSON metadata files containing scene recognition predictions generated by the MAX-SCENE Classifier**<empty citation>** API from locally ran docker service.

IMAGE OF FLOW CHART SHOWING DATA FLOW

3. **VR Software Application:** The VR software application is developed using Godot, specifically version 4.2.1. It immerses the user in a virtual environment, providing a large 2D UI window for browsing and viewing stereo images and videos. Custom shaders ensure accurate depth effects, while interactive 3D elements also available as alternative UI control

   IMAGE OF SOFTWARE APP SS

3.3 Design Decisions

The design decisions underlying the stereo camera system encompass hardware selection, stereo processing methodology, and VR software development.

3.3.1 Hardware Selection

The stereo camera hardware is engineered to capture stereo images and videos effectively. Key considerations include the processing unit, cameras for stereo pair formation, and audio capture capability. Initially, candidates such as the Raspberry Pi Pico microcontroller, OV5640 Camera module, and Electret microphone were evaluated. However, limitations such as slow SPI connection bandwidth results into high capture latency and image quality issues that led to the adoption of the Raspberry Pi 5. Despite being costlier and bulkier, the Raspberry Pi 5 offers superior performance with its faster processor and larger memory. Additionally, both combination of USB webcams and CSI MIPI Rasberry Pi Camera Module 3 (Wide) was evaluated to assess quality and cost-effectiveness. To ensure precise stereo image alignment, custom enclosures were designed and 3D printed. RTC clock is added for accurate filename timestamp even without being connected to the internet for a long time during boot. A python script is used to configure the behaviour of the capturing algorithm so systemCTL[footnote] functionalities were incorporated for seamless execution on startup.

(IMAGES of raspi pico, pi 5, cameras, soldered pico and 3D printed webcam setup?)

3.3.2 Stereo Processing

Stereo processing is performed on the main PC to leverage its robust processing power. File transfer via FTP through SSH using FileZilla simplifies data transfer, as it is more straightforward and easier to setup than a webserver, and more robust than manual transfer via SD Card that can lead to file corruption [footnote ref]. Python is used due to its ease of use for scripting and automation, this is proven with its wealth of library including ones used such as PILLOW and subprocess to invoke FFMPEG commands. FFMPEG, a powerful multimedia processing tool, efficiently stitch together the stereo pair videos to form a Full Side-By-Side (SBS) format video, suitable for playback on most VR video player, including my VR software application built for it. Furthermore, automated metadata generation utilizing the MAX-SCENE classifier API from a locally ran Docker image enhances organizational efficiency by generating JSON predictions for scene recognition that is also utilised by the VR software application. Other processing that arise from case-by-case basis such as conversion from mjpeg to mkv and 180 degree rotation is also written on Python utilising similar PILLOW and FFMPEG methods, thus easily scalable and extended to fit more custom needs in the future.

3.3.3 VR Software Development

The VR software application is developed using the Godot game engine, chosen for its adherence to open-source principles and robust scripting capabilities with GDScript. While not as feature-rich as Unity or Unreal Engine, Godot meets project requirements and fosters community-driven development. Custom shaders enable accurate depth effects for immersive viewing experiences, enhancing emotional engagement with stereo media. Additionally, VR controls offer intuitive interaction, enhancing user immersion and interface effectiveness.

Chapter 4

Implementation

4.1 Development Process

The development process for implementing the system involved a combination of methodologies and tools to ensure efficiency and organization. Git version control was utilized for managing project files and ensuring backups, providing a structured approach to development. Additionally, Gantt charts were employed for project planning and tracking progress over time.

Project files were organized into categorized folders to maintain clarity and ease of access. For example, the ”Hardware_mount” folder contained STL files exported from the 3D enclosure designed using Onshape for 3D printing. Similarly, the ”Software_Algorithm” folder housed the ”run_me.py” code containing algorithms for lifelogger behavior, while the ”Software_Game” directory stored all source code for the Godot project files. The ”Software_Processing” folder contained Python scripts related to stereo processing tasks. Every single one of the folders is stored on both OneDrive and GitHub to ensure data redundancy and expedite synchronization.

The development process commenced with hardware development, concurrently with a preliminary feasibility study of Godot. This involved experimenting with custom shaders for side-by-side (SBS) viewing and implementing basic stereo screens within a 3D game world. Once hardware development reached completion and data acquisition was feasible, focus shifted to perfecting the stereo processing pipelines for immediate testing with captured media.

Stereo processing tasks included automating file crawling, stereo pair stitching, audio mixing, and metadata tagging for JSON generation. The resulting output was systematically transferred to designated folders for categorization based on the week, month, and year.

Subsequently after basic testing of stereo media successful, further software development for the VR application commenced. Initially, a simple 2D UI was developed before transitioning to a VR version. This involved integrating scene filters and refining UI/UX elements, including interactive 3D elements for enhanced user engagement. Throughout the development process, iterative improvements were made to ensure the functionality and usability of the VR application.

4.2 Component Details

Following is total cost breakdown of final prototype:

COST TABLE HERE

4.2.1 Processing Unit (Raspberry Pi 5 and Pico)

The Raspberry Pi 5 is a single-board computer, while Raspberry Pi Pico is a microcontroller board, both are developed by the Raspberry Pi Foundation. While they share the Raspberry Pi branding and are designed for a range of applications, they serve different purposes within the system architecture.

The Raspberry Pi 5, the latest iteration of the Raspberry Pi series, is a powerful and versatile single-board computer equipped with a quad-core ARM Cortex-A72 processor and up to 8GB of RAM. It offers significant improvements in performance compared to its predecessors, making it well-suited for tasks requiring intensive computation, such as image and video processing. With its Full-HD resolution image and video capture capabilities at 30 frames per second (fps), the Raspberry Pi 5 provides the processing power necessary for capturing high-quality stereo footage.

In contrast, the Raspberry Pi Pico is a microcontroller board based on the Raspberry Pi Foundation's RP2040 microcontroller chip. It is designed for embedded applications and projects requiring real-time control and low-power operation. While the Raspberry Pi Pico is more affordable and energy-efficient than the

Raspberry Pi 5, it has limitations in terms of processing power and connectivity. The SPI connection bandwidth and processing capabilities of the Raspberry Pi Pico are insufficient for handling the demands of stereo image and video capture at Full-HD resolution.

Ultimately, the Raspberry Pi 5 was selected over the Raspberry Pi Pico for its superior performance and capability to meet the requirements of the stereo processing pipeline. Its higher processing power and Full-HD resolution image and video capture capabilities ensure smooth operation and high-quality output for the system's stereo footage capture and processing tasks.

4.2.2 Cameras

4.2.2.1 ArduCAM 5MP OV5642

The ArduCAM 5MP OV5642 camera module features a 5-megapixel image sensor and is capable of capturing high-resolution images. It offers a wide range of configuration options for adjusting settings such as exposure, white balance, and focus. While the OV5642 module provides excellent image quality, compatibility issues arose when attempting to integrate it with the Raspberry Pi Pico. These compatibility challenges, coupled with concerns about SPI connection bandwidth limitations, led to the exploration of alternative camera options.

4.2.2.2 Microsoft LifeCAM HD-3000 USB Webcams

USB webcams, such as the Microsoft LifeCAM HD-3000, were considered for their plug-and-play compatibility and affordability. USB webcams are widely available and offer ease of use with a simple plug-and-play setup. However, while USB webcams provide convenience, they may lack the image quality and configurability compared to dedicated camera modules.

4.2.2.3 Raspberry Pi Camera Module 3 (Wide)

The Raspberry Pi Camera Module 3 (Wide) is a dedicated camera module designed specifically for use with Raspberry Pi single-board computers. It features a wide-angle lens and is capable of capturing Full-HD resolution stereo footage at 30 frames per second. The Camera Module 3 offers seamless integration with the

Raspberry Pi 5, providing compatibility and optimal performance for the stereo processing pipeline. Its compact form factor and flexible mounting options make it well-suited for embedded applications requiring stereo image and video capture.

Ultimately, the Raspberry Pi Camera Module 3 (Wide) was chosen as the preferred camera option for its compatibility with the Raspberry Pi 5 and ability to meet the requirements of the stereo footage capture task.

4.3 Hardware Development

Following the selection of components, further refinement was essential to ensure acceptable quality and streamline the process. This involved the creation of a custom enclosure to house the components effectively, with careful consideration given to the mounting location for the camera. Additionally, the development of an onboard script was necessary to implement the lifelogging algorithm.

4.3.1 Custom Enclosure and POV Considerations

Achieving precise alignment between the stereo camera modules is crucial for generating accurate stereo images and videos. To ensure proper alignment, a custom 3D-printed enclosure was designed using Onshape. This enclosure allows for the precise mounting of the camera modules at a fixed distance and orientation relative to each other, minimizing any discrepancies in the stereo pair.

IMAGES OF 3D DESIGN ONSHAPE

Additionally, considerations regarding the point-of-view (POV) were taken into account during the design process. Initially, a design resembling Ray-Ban Meta and Snapchat Spectacles glasses was considered to capture footage closely resembling human vision. However, practical constraints led to the adoption of a head-mounted design. Although the POV is slightly higher than eye level in this configuration, the movement of the cameras still follows the wearer's head movement, resulting in a realistic POV perspective as seen in Figure 4.1. This compromise between POV and practicality ensures an immersive viewing experience while maintaining ease of use and wearability.

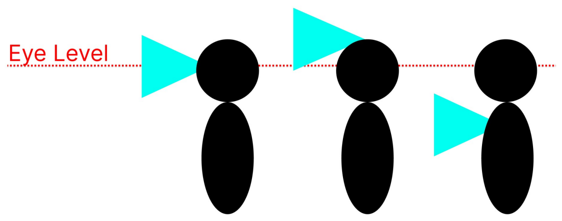

Figure 4.1: Designs and POV Example. Left to right: Glasses, Head-Mounted and Chest-Mounted

4.3.2 Onboard Lifelogging Algorithm

The onboard lifelogging algorithm script orchestrates the frequency and method of image and video capture within the system. It incorporates optimizations such as utilizing a Real-Time Clock (RTC) to ensure accurate timestamping even when not connected to the internet, and employing systemctl[footnote] to automatically execute the script on boot. This ensures seamless operation and consistent data capture regardless of network connectivity.

The script, which runs on the Raspbian operating system, allows for remote access via SSH, enabling control and configuration adjustments without the need for additional interfaces or servers. By configuring the script parameters, users can easily modify the frequency and duration of image and video capture sessions to suit their specific requirements. This simplicity and flexibility make it well-suited for the scope of the project, facilitating efficient lifelogging operations.

To further enhance efficiency, the script leverages optimized FFmpeg[footnote] and rpicam[footnote] commands for video recording and image capture. It also includes conditional logic to select between USB webcams and the Raspberry Pi camera module based on user preferences, ensuring compatibility and optimal performance.

CODE EXAMPLE

When using FFMPEG and USB webcam combination, an issue arises with the Raspberry Pi overheating during successive video recordings, potentially leading to performance degradation. It is observed that despite the overheating, there are no noticeable frame drops. To address this, a trial is planned for using the "very fast" preset instead of "ultrafast" to assess if it mitigates heat throttling. However, uncertainties remain regarding whether the choice of preset, or indoor/low-light conditions contribute to stuttering and flickering, warranting further investigation.

Additionally, the "-ss 1" parameter, implemented to skip the first frame for video thumbnails, is noted as necessary for better thumbnail quality.

Considering the continuous capturing of images and videos during the user's waking hours from 0800 to 2400, which spans approximately 16 hours each day, the estimated data usage can be calculated. Assuming an infinite battery life, the system is expected to capture 12 images and record two 30-second videos every hour.

This results in a data consumption of approximately 12MB for images and 720MB for videos per day, totaling to approximately 11532 MB (11.532GB) over the 16-hour period. Extrapolating this usage over a week, the cumulative data requirement amounts to approximately 80.724GB.

Given this estimated data consumption, implementing a weekly backup and data transfer strategy becomes imperative to ensure data integrity and manage storage capacity effectively. This approach aligns with the project's objectives of maintaining a consistent and reliable lifelogging system while managing data storage efficiently.

4.4 Stereo Processing Pipeline

4.4.1 FileZilla FTP via SSH

FileZilla FTP via SSH is utilized for transferring media capture from the Raspberry Pi 5 to the main/host PC for processing and storage. This method provides a secure and efficient means of transferring files over a network connection. By leveraging FileZilla, the system minimizes manual intervention and simplifies the data transfer process, improving overall workflow efficiency.

IMAGE OF FILEZILLA

4.4.2 Python Scripts and FFMPEG

Python scripts play a crucial role in the stereo processing pipeline, facilitating automation and manipulation of captured media. FFMPEG, a powerful multimedia processing tool, is integrated into the pipeline for tasks such as audio mixing and stereo pair stitching. Together, these tools enable seamless processing and preparation of stereo footage for viewing in the VR software application.

EXAMPLE STITCHING CODE

4.4.3 MAX-Scene Classifier (Scene Metadata .json Generation)

The MAX-Scene Classifier**<empty citation>** is instrumental in our stereo processing pipeline, enabling scene recognition and metadata tagging. To integrate this functionality, a Docker image provided by MAX-Scene Classifier was utilized. By deploying this Docker image, a service environment was set up to call the MAX-Scene Classifier API and obtain scene predictions in JSON format. This approach facilitated the generation of metadata for captured images and videos, with the JSON response files saved under the same name as the corresponding media files.

For images, the process is straightforward as only a single frame is analyzed. However, for videos, five evenly spaced frames are extracted and used for scene prediction analysis. Consequently, this results in the generation of five separate .json files for each video file. These .json files contain the scene predictions derived from the sampled frames and are utilized for filtering within the VR software application.

EXAMPLE OF JSON RESPONSE

4.5 VR Software

4.5.1 Godot Game Engine

The choice of the Godot game engine[30] for developing the video player VR app stems from 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. This is also beneficial for our open-source project as it promotes community development and ensures project longevity, as it does not rely on other vendors and is fully customizable. This is evident from the multitude of different kinds of apps created in Godot, which are not exclusive to games but also include apps such as Pixelorama and Godello. Additionally, its lightweight nature makes development and prototyping easier due to faster compile times and hot reload capabilities. Furthermore, its custom language, GDScript, is similar to Python, making it easy to learn and reducing time spent grappling with language intricacies.

The VR software application is built upon the robust foundation of the Godot version 4.2.1 game engine. Godot's intuitive interface and extensive documentation provide a conducive environment for rapid development, facilitating the creation of engaging VR software.

4.5.2 SBS Video Projection

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. 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 **??** 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 [31]. 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 4.2.

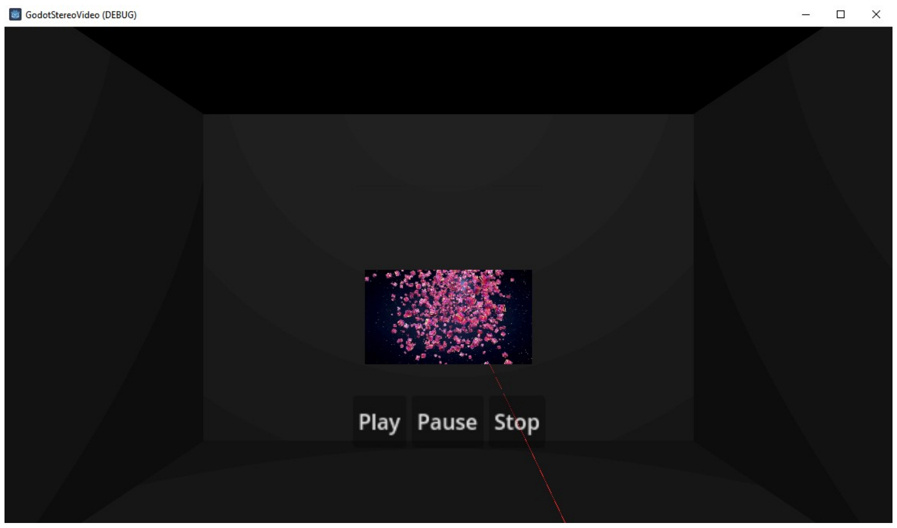

Figure 4.2: Images of App Running

4.5.3 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].

4.5.4 2D UI and VR Port

During the development process, initial emphasis was placed on creating a user-friendly interface (UI) that seamlessly transitions between 2D UI to VR environments, this is important to make sure the barebones functionality of browsing and viewing works before making it more complicated. Other innovative UI design concepts, such as using 3D game elements like bookshelves and tapes for browsing images and videos, to enhance user engagement and interaction are also experimented with. Additionally, the VR software was meticulously ported from a 2D

UI application, ensuring continuity in functionality and user experience across different platforms. This strategic approach allowed for the integration of VR-specific features while maintaining familiarity for users accustomed to traditional 2D interfaces.

Chapter 5

Testing and Evaluation

5.1 Testing Strategy

Outline the testing strategy, including unit testing, integration testing, and acceptance testing.

5.2 Test Results

Present the results of testing, including any issues or bugs encountered and how they were addressed.

5.3 Performance Evaluation

Evaluate the performance of the system against predefined criteria or benchmarks.

Chapter 6

Conclusion

6.1 Summary

Summarize the key findings, achievements, and contributions of your project.

6.2 Reflections

Talk about project planning etc. gantt chart

6.3 Future Work

Identify areas for future work, research, or enhancements to the project.

Bibliography

[1] *Meta Quest 3: New Mixed Reality VR Headset*. [Online]. Available: https://www.meta.com/gb/quest/quest-3/.

[2] *Introducing Apple Vision Pro: Apple's first spatial computer*, Jun. 2023. [Online]. Available: https://www.apple.com/newsroom/2023/06/introducing-apple-vision-pro/.

[3] *Spectacles by Snap Inc. Spectacles 3*, 2023. [Online]. Available: https://www.spectacles.com/uk/shop/spectacles-3.

[4] *Kandao QooCam Ego 3D Camera*. [Online]. Available: https://www.kandaovr.com/qoocam-ego/.

[5] *Apple unveils iPhone 15 Pro and iPhone 15 Pro Max*, Sep. 2023. [Online]. Available: https://www.apple.com/newsroom/2023/09/apple-unveils-iphone-15-pro-and-iphone-15-pro-max/.

[6] M. Park, *3 keys to improving user retention in virtual reality*, Oct. 2017. [Online]. Available: https://venturebeat.com/games/3-keys-to-improving-user-retention-in-virtual-reality/.

[7] V. Weis, *3DOF, 6DOF, RoomScale VR, 360 Video and Everything In Between — Blog — Packet39*, 2018. [Online]. Available: https://packet39.com/blog/3dof-6dof-roomscale-vr-360-video-and-everything-in-between/.

[8] E. Callenbach, ": The Five C's of Cinematography: Motion Picture Filming Techniques Simplified . Joseph V. Mascelli.," *Film Quarterly*, vol. 19, no. 4, 1966, ISSN: 0015-1386. DOI: 10.1525/fq.1966.19.4.04a00320.

[9] N. Dodgson, "Autostereoscopic 3D displays," *Computer*, vol. 38, no. 8, pp. 31–36, Aug. 2005, ISSN: 0018-9162. DOI: 10.1109/MC.2005.252. [Online]. Available: http://ieeexplore.ieee.org/document/1492263/.

[10] C. Gurrin, A. F. Smeaton, and A. R. Doherty, "LifeLogging: Personal Big Data," *Foundations and Trends® in Information Retrieval*, vol. 8, no. 1, pp. 1–125, 2014, ISSN: 1554-0669. DOI: 10.1561/1500000033. [Online]. Available: http://www.nowpublishers.com/articles/foundations-and-trends-in-information-retrieval/INR-033.

[11] M. Dimiccoli and P. Radeva, "Visual Lifelogging in the Era of Outstanding Digitization," *Digital Presentation and Preservation of Cultural and Scientific Heritage*, vol. 5, pp. 59–64, Sep. 2015, ISSN: 2535-0366. DOI: 10.55630/dipp.2015.5.4. [Online]. Available: https://dipp.math.bas.bg/dipp/article/view/dipp.2015.5.4.

[12] E. Waisberg, J. Ong, M. Masalkhi, *et al.*, "Meta smart glasses—large language models and the future for assistive glasses for individuals with vision impairments," *Eye*, 2023, ISSN: 1476-5454. DOI: 10.1038/s41433-023-02842-z. [Online]. Available: https://doi.org/10.1038/s41433-023-02842-z.

[13] A. Orchard, M. O'Gorman, C. La Vecchia, and J. Lajoie, "Augmented Reality Smart Glasses in Focus: A User Group Report," in *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, New York, NY, USA: ACM, Apr. 2022, pp. 1–7, ISBN: 9781450391566. DOI: 10.1145/3491101.3503565. [Online]. Available: https://dl.acm.org/doi/10.1145/3491101.3503565.

[14] D. Marr and T. Poggio, "A computational theory of human stereo vision," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 204, no. 1156, pp. 301–328, May 1979, ISSN: 0080-4649. DOI: 10.1098/rspb.1979.0029. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rspb.1979.0029.

[15] M. S. Banks, J. C. A. Read, R. S. Allison, and S. J. Watt, "Stereoscopy and the Human Visual System," *SMPTE Motion Imaging Journal*, vol. 121, no. 4, pp. 24–43, May 2012, ISSN: 1545-0279. DOI: 10.5594/j18173. [Online]. Available: https://ieeexplore.ieee.org/document/7308394/.

[16] K. Matzen, M. F. Cohen, B. Evans, J. Kopf, and R. Szeliski, "Low-cost 360 stereo photography and video capture," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–12, Aug. 2017, ISSN: 0730-0301. DOI: 10.1145/3072959.3073645. [Online]. Available: https://dl.acm.org/doi/10.1145/3072959.3073645.

[17] R. Aggarwal, A. Vohra, and A. M. Namboodiri, "Panoramic Stereo Videos with a Single Camera," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2016-December, IEEE, Jun. 2016, pp. 3755–3763, ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.408. [Online]. Available: http://ieeexplore.ieee.org/document/7780777/.

[18] Umar Shakir, *The iPhone 15 Pro is getting spatial video capture in iOS 17.2*, Nov. 2023. [Online]. Available: https://www.theverge.com/2023/11/9/23954310/apple-iphone-15-ios-17-beta-spatial-video-vision-pro.

[19] M. Mühlhausen, M. Kappel, M. Kassubeck, *et al.*, "Immersive Free-Viewpoint Panorama Rendering from Omnidirectional Stereo Video," *Computer Graphics Forum*, vol. 42, no. 6, 2023, ISSN: 14678659. DOI: 10.1111/cgf.14796.

[20] R. Pagés, K. Amplianitis, D. Monaghan, J. Ondřej, and A. Smolić, "Affordable content creation for free-viewpoint video and VR/AR applications," *Journal of Visual Communication and Image Representation*, vol. 53, 2018, ISSN: 10959076. DOI: 10.1016/j.jvcir.2018.03.012.

[21] A. Serrano, I. Kim, Z. Chen, *et al.*, "Motion parallax for 360° RGBD video," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 5, 2019, ISSN: 19410506. DOI: 10.1109/TVCG.2019.2898757.

[22] J. Armstrong, *"The Entire History of You"*, 2011.

[23] H. Strasburger, *Seven Myths on Crowding and Peripheral Vision*, 2020. DOI: 10.1177/2041669520913052.

[24] A. Duane, B. P. Jónsson, and C. Gurrin, "VRLE: Lifelog Interaction Prototype in Virtual Reality: Lifelog Search Challenge at ACM ICMR 2020," in *LSC 2020 - Proceedings of the 3rd Annual Workshop on the Lifelog Search Challenge*, 2020. DOI: 10.1145/3379172.3391716.

[25] H. Chen, R. Shi, D. Monteiro, N. Baghaei, and H. N. Liang, "VRCockpit: Mitigating Simulator Sickness in VR Games Using Multiple Egocentric 2D View Frames," in *IEEE Conference on Computatonal Intelligence and Games, CIG*, vol. 2022-August, 2022. DOI: 10.1109/CoG51982.2022.9893678.

[26] S. H. Park and G. C. Lee, "Full-immersion virtual reality: Adverse effects related to static balance," *Neuroscience Letters*, vol. 733, 2020, ISSN: 18727972. DOI: 10.1016/j.neulet.2020.134974.

[27] K. Lim, J. Lee, K. Won, N. Kala, and T. Lee, "A novel method for VR sickness reduction based on dynamic field of view processing," *Virtual Reality*, vol. 25, no. 2, 2021, ISSN: 14349957. DOI: 10.1007/s10055-020-00457-3.

[28] A. S. Fernandes and S. K. Feiner, "Combating VR sickness through subtle dynamic field-of-view modification," in *2016 IEEE Symposium on 3D User Interfaces, 3DUI 2016 - Proceedings*, 2016. DOI: 10.1109/3DUI.2016.7460053.

[29] O. M. Zarka and Z. J. Shah, "Virtual Reality Cinema: A Study," *IJRAR-International Journal of Research and Analytical Reviews*, vol. 3, no. 2, 2016.

[30] *Godot 4.2 Game Engine Docs.* [Online]. Available: https://docs.godotengine.org/en/4.2/.

[31] Malcolmnixon, *Malcolmnixon / godot-stereo-video Github Repository*, Nov. 2023. [Online]. Available: https://github.com/Malcolmnixon/godot-stereo-video.