|  |  |
| --- | --- |
| School of Electronic Engineering and Computer Science | **Programme of study:**  Computer science  **Project Title:**  3D-CAD application for VisionPro  **Supervisor:**  Shanxin Yuan  **Student Name:**  Hayri Olcay  Date: 29/04/2024 |
| Final Year  Undergraduate Project 2023/24 |
| QMLogo |

Abstract

This dissertation explores the development of a 3D CAD (Computer-Aided Design) application tailored specifically for the Apple Vision Pro, a cutting-edge mixed reality device. The study delves into the unique design challenges and opportunities presented by the Vision Pro's immersive capabilities, aiming to create an app that enhances both professional and creative workflows. We integrate human-computer interaction (HCI) theories, including the Technology Acceptance Model and Cognitive Load Theory, to design an intuitive, user-friendly interface. The dissertation further examines inclusive design principles, ensuring the app is accessible to users with varying abilities, and incorporates participatory design methods, involving direct input from users to shape the final product. Through this holistic approach, the study aims to deliver a versatile, accessible CAD application that leverages the Apple Vision Pro's potential, advancing both the design industry and inclusive technology.

**C**

**ontents**

[Chapter 1: Introduction 4](#_Toc165279878)

[1.1 Background 4](#_Toc165279879)

[Chapter 2: Literature Review 7](#_Toc165279880)

[Chapter 3: Specification 16](#_Toc165279881)

[Chapter 4: Source code and Design 19](#_Toc165279882)

[4.1 19](#_Toc165279883)

[Chapter 5: Evaluation 28](#_Toc165279884)

[Chapter 6: Conclusion 30](#_Toc165279885)

[References 32](#_Toc165279886)

[Appendix A – risk assessment 34](#_Toc165279887)

[Appendix B-timeline 35](#_Toc165279888)

# Introduction

## Background

The Vision Pro Apple is a mix-up reality headset developed by the Apple Inc., announced in 2023 and scheduled for a release in early 2024 in the US, with international availability later that year. This device represents the first foray of Apple into a new major product category since introducing the Apple Watch in 2015.

Major features of the Apple Vision Pro include:

\* Design and Display: Boasting a slick design, incorporating eye and hand tracking capabilities, the headset features sharp micro-OLED displays, providing an ultra-high-resolution display system with 23 million pixels across two screens. This setup ensures that every experience feels immersive and real, as if happening right in front of the user's eyes.

\* Spatial Computing and Application Integration: The Vision Pro is called Apple's first special computer, offering a unique user experience where applications are not confined to a traditional screen but can be arranged and scaled in the user's physical space. This creates an 'infinite canvas', transforming how users interact with their favourite apps while staying in touch with the world surrounding them.

\* Multifunctionality and Content Creation: The headset can be used in various ways, including as an immersive video and photo camera. It's capable of both augmented reality (AR) and virtual reality (VR), providing a modern take on content creation similar to a 21st-century stereoscopic camera or View-Master. This multifunctionality makes it perfect for creating immersive videos, photos, and interactive mixed reality experiences.

Problem Statement

1.2 Aim

To develop a cutting-edge 3D Computer-Related Design (CAD) application optimized for the Apple Vision Pros, leveraging its mixed reality capabilities to provide an immersive, intuitive, and highly interactive design experience. This application will harness the advanced display, eye and hand tracking features, and spatial computing power of the Vision Pros to enable designers, architects, and engineers to create, view, and manipulate 3D models in real time within a virtual environment. The goal is to revolutionize the CAD industry by introducing a new level of precision, efficiency, and user engagement in 3D design processes, making the most of the Vision Pros' capabilities to transcend traditional design boundaries.

1.3 Objectives

Objectives for Developing a 3D CAD Application for Apple Vision Pros:

1. Optimization for Mixed Reality: Ensure the CAD application is fully optimized for the mixed reality environment of the Apple Vision Pros. This involves developing a user interface and experience that leverages the headset’s immersive capabilities.

2. Intuitive Design Interface: Create an intuitive and user-friendly interface that allows users to navigate and manipulate 3D models using the Vision Pros' eye and hand tracking features easily.

3. Real-time Interaction and Visualization: Implement real-time interaction capabilities for creating and editing 3D models, ensuring that changes are visualized instantly in the mixed reality space.

4. Spatial Computing Utilization: Utilize the spatial computing abilities of the Vision Pros to enable users to place and scale 3D models in the real world, enhancing the perception of depth and space in design.

5. Advanced Modelling Tools: Incorporate advanced 3D modelling tools into the application, allowing for detailed and precise design work suitable for professional architectural, engineering, and design projects.

6. Collaboration Features: Integrate collaboration tools that allow multiple users to work on the same 3D model in a shared virtual space, regardless of their physical location.

7. Performance Optimization: Ensure the application runs smoothly on the Vision Pros' hardware, optimizing for the high-resolution displays and the custom Apple silicon for seamless performance.

8. User Training and Support: Develop comprehensive training materials and user support systems to assist users in adapting to this new way of 3D modelling and design.

9. Feedback and Iteration: Establish a system for gathering user feedback to continuously improve the application, ensuring it meets the evolving needs of the CAD community.

10. Compliance and Compatibility: Ensure the application complies with industry standards and is compatible with existing CAD file formats and systems for seamless integration into current workflows.

11. Accessibility Adaptation: Implement comprehensive controller support to accommodate mobility-impaired individuals. This should include customizable control schemes, sensitivity adjustments, and adaptive input methods that can interface with assistive technologies, ensuring that the software is accessible and functional for users with a wide range of physical abilities.

1.4 Research Questions

How does the application of crossed reality technology via Apple Vision Pros revolutionize the traditional 3D Computer-Aided Design (CAD) workflow in terms of efficiency, precision, and user experience? To critically assess the transformative effects of integrating mixed reality technology into 3D CAD processes through Apple Vision Pros, the research will focus on the following key aspects:

Efficiency:

\* Measure the time-efficiency improvements in conceptualization, design iteration, and finalization stages using mixed reality.

\* Evaluate the integration of real-world data and virtual modelling in a mixed reality environment for streamlined workflows.

Precision:

\* Compare the granularity and refinement of 3D models produced with mixed reality technology versus conventional CAD systems.

\* Examine how the depth perception and spatial awareness provided by Apple Vision Pros enhance precision of 3D constructs.

User Experience:

\* Conduct surveys and user tests to gauge satisfaction, ergonomic comfort, and overall engagement with mixed reality CAD process.

\* Analyse the cognitive load and intuitive interaction compared to mouse-and-keyboard CAD environments.

Collaborative Work:

\* Assess the impact of mixed reality on teamwork dynamics, remote collaboration, and real-time stakeholder engagement.

\* Compare communication efficacy and project alignment with conventional collaboration methods in CAD systems.

Innovation in Design Process:

\* Identify novel design approaches and creative solutions facilitated by the immersive experience of mixed reality.

\* Document case studies where mixed reality in CAD leads to breakthroughs in design complexity and problem-solving.

Adaptation and Learning Curve:

\* Study the transition process for professionals moving from traditional to mixed reality CAD environments.

\* Identify the necessary training and support systems that aid in skill acquisition for efficient use of the new technology.

By methodically exploring these areas, the research intends to articulate the benefits and hurdles of applying Apple Vision Pros in 3D CAD, thereby informing future technological advancements and adoption strategies in design and engineering domains.

# Literature Review

2.1 About visionOS

This section will provide an understanding of what visionOS is, the architecture of the system, the digital distribution platform it utilizes, and will present the technical details for the project at hand.

2.1.1 What is visionOS?

visionOS is the operating system designed for Apple's Vision Pro headset, representing a significant stride in Apple's software development. It integrates elements from macOS, iOS, and iPadOS but introduces a specialized "real-time subsystem" to process interactive visuals, which enables immersive experiences not possible on other Apple hardware. The system facilitates a diverse range of applications from animated environments and 3D videos to avatars captured with depth sensors of the Vision Pro headset. Apple (2023).

Based on the image and the information gathered, let's delve into the architecture of visionOS with more detail, considering each layer's role and functionality:

2.1.2 Detailed visionOS Architecture

Figure 2-1 illustrates a structured tier of visionOS, representing its sophisticated architecture. Each layer is intricately designed to contribute to the overall functionality and efficiency of the system, ensuring an immersive augmented reality (AR) experience.

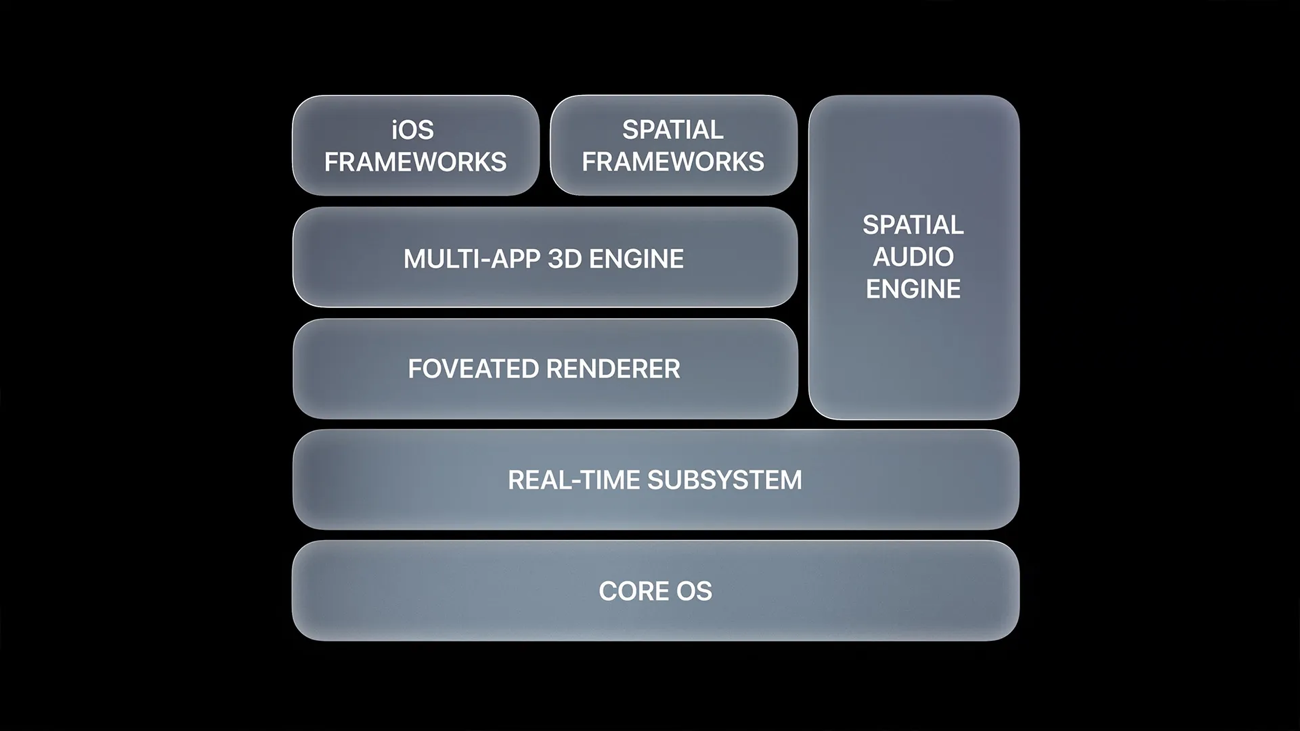


Figure 2-1: Showing VisionOS architecture from Apple (2023).

- Core OS: This foundational layer is the bedrock of the system, responsible for managing the hardware and providing the essential services upon which all other software layers are built. It incorporates the kernel that handles low-level tasks, including security, networking, and device management.

- Real-time Subsystem: Situated above the Core OS, the real-time subsystem is a pivotal component that processes high-priority tasks. This subsystem is engineered to handle interactive visuals and experiences with minimal latency, crucial for maintaining the fluidity and responsiveness expected in AR environments.

- Foveated Renderer: The foveated renderer is an advanced visual processing technique that optimizes graphics rendering based on the user's gaze. By focusing detail where the user is looking and reducing it peripherally, it greatly reduces computational load, resulting in smoother graphics and lower power consumption.

- Multi-App 3D Engine: A level above, the multi-app 3D engine enables the concurrent running of multiple applications in a three-dimensional space. It allows users to seamlessly interact with a multitude of apps within the AR experience, promoting multitasking and fluid navigation between tasks.

- iOS Frameworks: At this layer, visionOS incorporates the proven frameworks of iOS, facilitating app development and providing a vast array of tools and services that developers can leverage to create applications that are consistent with the broader ecosystem of Apple products.

- Spatial Frameworks: Specialized for spatial computing, these frameworks support the development of applications that are aware of and can interact with three-dimensional space and objects. They provide APIs for spatial tracking, scene understanding, and AR content rendering.

- Spatial Audio Engine: On the topmost tier lies the spatial audio engine, which provides an immersive sound experience by simulating how audio plays in a three-dimensional space. This layer handles sound directionality and movement, enhancing the realism of virtual environments.

Developers creating applications for visionOS must engage with these layers via the provided SDKs and APIs. By utilizing these tools, developers can ensure their applications take full advantage of the cutting-edge capabilities of visionOS, from immersive audio-visual experiences to sophisticated interaction models enabled by eye tracking and spatial awareness.

In the context of this structured architecture, developers are encouraged to adhere to the high standards of performance and integration set forth by visionOS. By doing so, applications not only deliver exceptional user experiences but also contribute to the robust and dynamic ecosystem that defines visionOS.

2.1.3 What is the visionOS App Store?

The visionOS App Store is a digital distribution platform created for the Apple Vision Pro headset, which facilitates the browsing and downloading of apps specifically designed or compatible with visionOS. Launched with over 600 apps crafted to leverage the unique capabilities of the Vision Pro, the store quickly grew to host more than 1,000 spatial apps. By early 2024, the visionOS App Store had expanded to include over 1.5 million apps, with the vast majority being iPad apps running in compatibility mode (Apple, 2023).

2.1.4 visionOS Applications

visionOS applications utilize frameworks and technologies like RealityKit and ARKit to present 3D content, animations, and interact with the user’s environment. RealityKit allows for stunning visual effects by automatically adjusting to physical light conditions, while ARKit understands surroundings to make apps more interactive (murata.eetrend.com, n.d.).

Developers can create accessible apps for visionOS using the same techniques and tools as on other Apple platforms. Xcode is the primary development environment, providing the necessary visionOS SDK. There's also Reality Composer Pro within Xcode, designed for preparing 3D content for visionOS apps, and Unity for bringing existing projects into the visionOS ecosystem (Apple Newsroom, n.d.).

2.1.5 visionOS as an Open Platform

visionOS embodies the open platform ethos by making its app store available to a variety of developers and ensuring backward compatibility with existing iOS and iPadOS apps. While the platform was launched with apps specifically optimized for Vision Pro, it also supports iPhone and iPad apps in a 2D environment, allowing users to continue using familiar apps even if they have not been updated for a 3D interface (Apple Newsroom. , n.d.).

2.1.6 visionOS Application Development

Development for visionOS starts with familiar tools but is tailored to the immersive experiences possible with Vision Pro. Third-party developers have been actively encouraged to build or adapt their apps for visionOS, and as a result, a number of productivity, gaming, and entertainment apps have been made available. Notable apps at launch included popular productivity tools like Webex, Zoom, and Microsoft Teams, and games on Apple Arcade that have been specifically created or adapted for spatial experiences (murata.eetrend.com, n.d.).

Additionally, apps like Disney+ offer unique features for visionOS users, such as selecting themed backgrounds for watching content, enhancing the user experience.Developers are provided with comprehensive support through various Apple programs to assist them in creating apps for visionOS, paving the way for innovative and engaging apps that push the boundaries of what is possible in AR and VR (Create With Swift, 2023)

2.1.7 Concluded Technical Project Details for 3D-CAD Application on visionOS

For the development of a 3D-CAD (Computer-Aided Design) application tailored to the Apple Vision Pro, it’s crucial to consider the technical capabilities and constraints of the visionOS platform to determine the range of features and the supported devices. Given the sophisticated nature of visionOS and the powerful hardware specifications of the Vision Pro headset, the application will leverage the full suite of spatial computing features available.

Device Specifications and OS Capabilities

The Vision Pro headset offers a high-resolution display and advanced spatial awareness, both of which are key to a 3D-CAD application. The device supports eye-tracking, hand gesture controls, and a high degree of environmental interaction, as detailed by the available developer documentation (Create With Swift, 2023).

Supported visionOS Versions

As with any platform, ensuring compatibility with the current version of visionOS is essential. At the time of development, visionOS is in its initial major release, and the application will be optimized for the latest update available to ensure stability and access to all current features.

Development Environment and Tools

Development will commence using Xcode with the visionOS SDK, utilizing tools such as Reality Composer Pro for 3D content preparation, and Unity for additional support in creating immersive 3D environments. The application will exploit features such as ARKit’s Scene Reconstruction and RealityKit’s 3D rendering capabilities for a seamless and interactive 3D-CAD experience (Apple,2023).

Testing Devices

Testing will be conducted on a MacBook with an Apple silicon chip exclusively, as it is the only other device supporting visionOS at this stage. Given the sophisticated AR features and the high-fidelity sensors built into the hardware, it is imperative to ensure that the 3D-CAD application fully utilizes these capabilities.

Market Coverage and User Base

Given that the Vision Pro headset is the sole device for visionOS, the market coverage for a 3D-CAD application is effectively 100% of the visionOS user base. The aim is to cater to professionals in the design and engineering industries who seek an immersive 3D-CAD experience with the freedom to design in a spatial computing environment.

Concluding Note

In summary, the 3D-CAD application will be a pioneering design tool specifically crafted for visionOS and the Vision Pro headset, representing the next generation of CAD software. It will enable professionals to create, manipulate, and analyse their designs in a fully immersive 3D space, leveraging the unique capabilities of visionOS and the Apple Vision Pro hardware.

2.2.1 Pre-existing 3D-CAD VR Software

With foundational knowledge of spatial computing on visionOS and the functionality of 3D-CAD applications, the next step in my research involves a detailed examination and comparison of existing 3D-CAD VR applications. This analysis is based on feature sets, usability, and performance within a virtual reality environment.

Shapr3D has been developed as a professional-grade tool with a keen focus on usability and precision. It is particularly beneficial for professionals accustomed to traditional CAD environments, as it bridges the gap between desktop and virtual reality design. The app takes advantage of the tactile precision of the Apple Pencil and the immersive display capabilities of visionOS, enabling designers to work with a level of detail that's typically challenging to achieve in a VR space. The user interface is tailored to minimize distractions and streamline the design process, from concept sketching to export-ready 3D models. Shapr3D for visionOS currently stands as a front-runner with its comprehensive design and collaboration tools. It offers immersive real-time collaboration, which allows multiple users to interact with the 3D models within the same environment. The ability to import custom materials and leverage an adaptive UI for in-canvas controls positions Shapr3D as a leader in precision and efficiency. However, the necessity for an Apple Pencil for certain functions could be a limiting factor for users preferring hand gestures as their primary mode of interaction. (Shapr3D, 2024)

Gravity Sketch

Gravity Sketch's strength lies in its focus on freeform design and collaboration. It eschews the conventions of desktop CAD for a more organic design process. Designers can create within a virtual space as if they were sculpting or drawing in the real world, which can significantly enhance creativity and spatial understanding. This application is particularly suited for automotive, industrial, and fashion design, where the spatial proportion and ergonomic considerations are paramount. Collaboration features within Gravity Sketch allow multiple users to work on the same design simultaneously, even if they are located in different geographical locations. It excels in providing an intuitive interface that mimics the natural drawing and sculpting experience in a 3D space. Its strength lies in the organic design process and the ability to import reference images, crucial for industries like automotive and fashion design. One potential drawback is the learning curve associated with mastering its unique spatial tools for users transitioning from traditional CAD software (GravitySketch 2024).

ShapeLab VR

ShapeLab VR is geared towards education and accessibility, offering a gateway for students and hobbyists into the world of 3D design. With a simplified interface and a collection of easy-to-use tools, it allows users to quickly grasp the concepts of 3D modelling without the steep learning curve often associated with professional CAD software. The app provides an engaging platform for interactive learning, making it an attractive choice for schools and workshops that aim to introduce participants to the principles of 3D design and virtual reality. The app is accessible for novices, with an emphasis on education and ease of use. While it's user-friendly and excellent for teaching the fundamentals of 3D modelling, professionals may find it lacking in the advanced features necessary for complex project development (ShapeLab, 2024).

MakeVR Pro

Designed for the seasoned professional, MakeVR Pro delivers a comprehensive suite of CAD tools within a VR environment. It aims to replicate the experience of using a high-end CAD system but with the added benefit of a natural, immersive interface. Key features include precision alignment, scaling, and the ability to manipulate complex geometries with ease. MakeVR Pro's sophisticated toolset is matched by its export capabilities, enabling seamless integration with established engineering and design workflows. The app provides a highly precise and professional CAD experience, replicating high-end desktop software in a VR setting. It is tailored for users requiring detailed geometrical manipulation and offers advanced CAD functions. Its interface, while powerful, might be overwhelming for beginners or those looking for a more streamlined VR design experience (MakeVR, 2024).

Holo-Light's Holo-Space

Holo-Space by Holo-Light is tailored for enterprise applications where interacting with detailed, life-sized models is essential. Its ability to import and visualize CAD data in AR/VR makes it a compelling tool for review and validation processes in sectors such as construction, mechanical engineering, and architecture. Holo-Space allows for collaborative sessions where stakeholders can interact with the virtual model in real-time, facilitating decision-making processes that traditionally rely on physical prototypes. It is suited for enterprise-scale applications, particularly in visualizing large models. It emphasizes secure data handling and real-world scale visualization. The complexity of Holo-Space makes it incredibly powerful but might also necessitate a longer familiarization period for new users (Holo-Light, 2024).

SkyReal

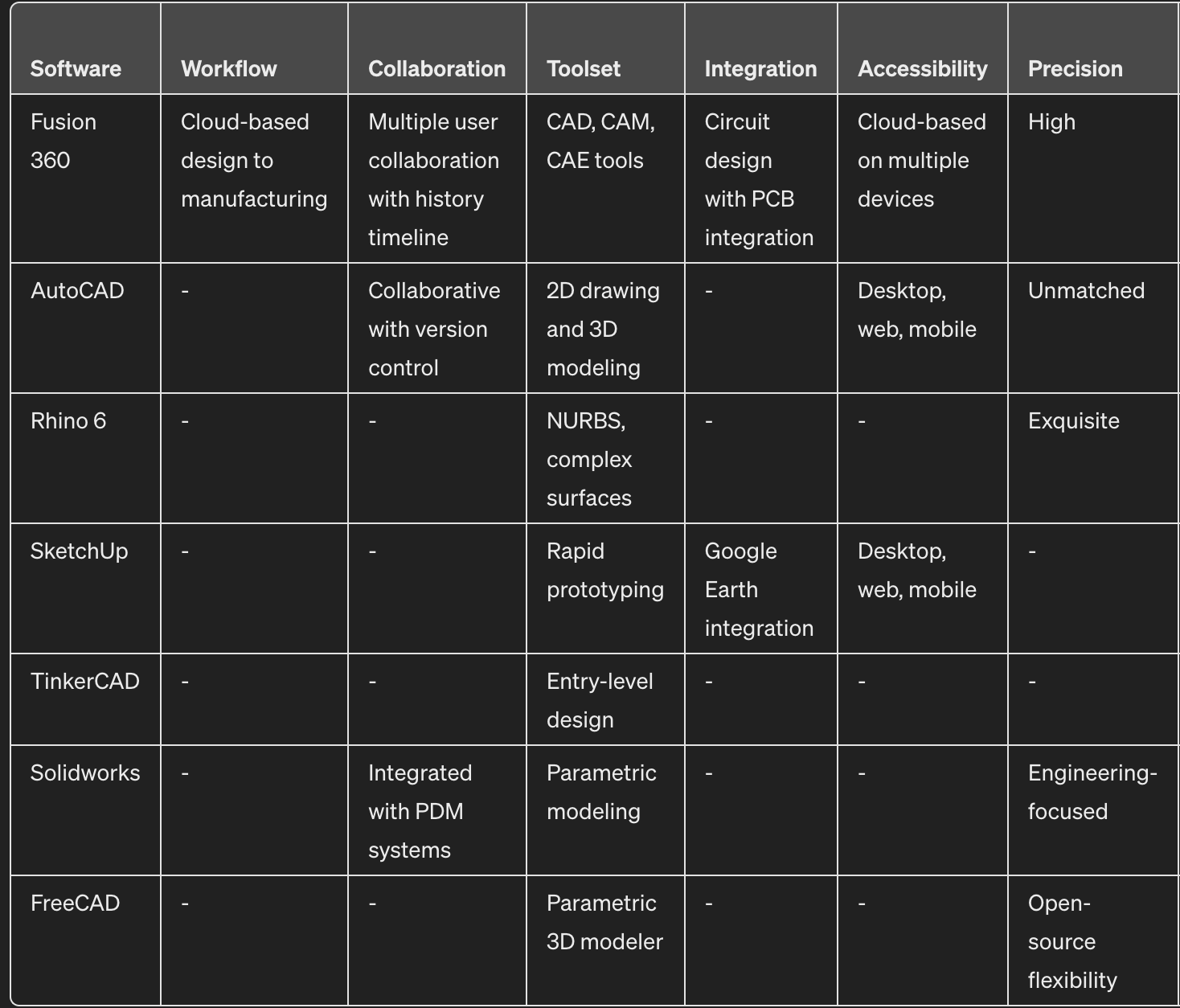
SkyReal provides a distinct advantage for industries that rely on simulations to validate designs. It enables real-time interactions with CAD models, allowing for virtual prototyping and testing. This application shines in scenarios where understanding the mechanical function and performance of a design under various conditions is crucial. With SkyReal, manufacturers can iterate faster, identify design flaws early, and optimize performance before committing to the cost-intensive physical prototyping stage. The app stands out with its VR simulation capabilities that allow for virtual prototyping and real-world interaction testing. The focus on simulation is a significant advantage, but the application's specialized nature means it might not be as well-rounded for general 3D modelling purposes (SkyReal, 2024).

Steseal

Spatial Painting is an app designed for Apple Vision Pro, transforming augmented reality into a creative playground. It lets users wave their hands to paint in the air, bringing a magical touch to art creation with vibrant gradient colours. Steseal is praised for its intuitive and user-friendly interface, making 3D drawing accessible to artists of all skill levels. The application combines fun, creativity, and technology, encouraging users to explore and innovate in a world where their imagination sets the limits. This app is part of a burgeoning field where art intersects with cutting-edge AR technology, expanding the traditional boundaries of how and where we create art. For a detailed look at Steseal's features, visit the FrameSixty website.

Each of these applications addresses a unique aspect of the design and modelling process, collectively broadening the horizons of what's possible within the virtual design space. They represent the forefront of 3D-CAD VR technology, continuously pushing the envelope in terms of immersive user experience, collaborative capabilities, and integration into traditional design workflows. (Steseal, 2024)

2.2.2 Non-VR 3D-CAD Applications



Each of these applications targets specific segments of the CAD market, from individual hobbyists and educators to professional engineers and designers, with a range of functionalities suited to different project needs and industry requirements.

2.3 Research conclusion and Critical Review of 3D-CAD Applications on visionOS

Building upon my understanding of visionOS development and the functionalities of Apple's M1 chip, I delve into the analysis of 3D-CAD applications suitable for visionOS. These apps will be rigorously tested on an Apple MacBook powered by the M1 chip, chosen for its graphic prowess and computational efficiency, crucial for CAD operation. The chosen MacBook with M1 chip will run the latest macOS that supports visionOS, providing an ideal platform for assessing the performance of 3D-CAD applications and the assessment will focus on the integration with visionOS, the application's ability to harness the M1 chip's power, and the implementation of visionOS's augmented reality functionalities.

2.4 Inclusive Design and HCI Theories

Emphasizing inclusivity within 3D CAD applications involves applying Human-Computer Interaction (HCI) theories to understand and cater to diverse user needs. Incorporating 'Design for All' principles ensures that the applications developed for platforms like Apple Vision Pro are not only technically advanced but also accessible to users with varying abilities. By adhering to these principles, 3D CAD applications can become more than just tools; they can serve as enablers that offer equal opportunity for all users to engage in design processes.

2.4.1 TAM in the Context of Assistive Technologies

The Technology Acceptance Model (TAM) is particularly relevant in assessing the adoption of assistive technologies in mixed reality for 3D CAD. It can predict user acceptance by evaluating the perceived ease of use and usefulness, key factors that can influence the successful adoption of the Apple Vision Pro’s assistive features. When applied, TAM can help developers optimize the design of mixed reality tools to be more intuitive and effective for users with disabilities. (Davis, F.D.,1989). An avenue to consider may be controller support for the paralysed as has been demonstrated by popular streamer “RockyNoHands” who currently holds the Guinness World record for most eliminations in a Fortnite Battle Royale using a QuadStick mouth-operated joystick. (Guinness World Records,2019)

2.4.2 Evaluate through Lens of A for Dance Equilibrium Theory

Design assistant features in mixed party devices can be evaluating using A for Dance Equilibrium Theory. Understanding the relationships between the user's intentions and the device's capabilities, a set of tools can create naturally align with user interactions. Features like voice control, gesture recognition, and haptic feedback, if implemented well, can enhance the user experience significantly in CAD applications, mainly in the sophisticated ecosystem like the Apple Vision Pro.

2.4.3 Applet of Cognitive Load Theory

Cognitive Load Theory can give insights into how mixed reality use influences users' cognitive processing while performing complicated tasks. The immersive environment of the Apple Vision Pro has the potential to minimize extraneous cognitive load, enabling users to focus more effectively on the intrinsic aspects of their CAD tasks. This could result in better learning outcomes and a more user-friendly experience for individuals with inabilities. (Sweller, J, 2011)

2.4.4 Adaptor Diffusion Theory

The Adaptor Diffusion Theory can be adapted to comprehend the factors affecting the upswing of the Apple Vision Pro compared to other systems. By considering aspects like relative advantage and compatibility, developers can gain insights into how to position their assistive CAD applications in the market and encourage broad adoption among potential users. (Rogers, E.M.,2003).

2.4.4 Universal Design and Social Model of Inability

Creating an accessible 3D CAD application for the Apple Vision Pro should be guided by the principles of Universal Design and Social Model of Inability. These frameworks advocate for removing societal barriers and designing technology that accommodates all users, despite their abilities. The application should challenge conventional design processes by creating solutions that are universally accessible. (Mace, R, 1985).

2.4.5 Employing Participatory Design Principles

In line with participatory design principles, involving users with inabilities in the design process of the 3D CAD application should ensure the product meets their needs and preferences. This approach fosters an inclusive design philosophy that recognizes the value of diverse user input in creating solutions that are genuinely user-centric.

This theoretical and practical approach ensures that assistant technologies within 3D CAD applications are not just designed with kind intentions but are founded on well-established research and proven frameworks. These underpinnings ensure that the resulting products are capable of delivering a meaningful and inclusive user experience across a spectrum of abilities.

**2.5.1 summary**

Anchored in HCI principles and inclusive design, we assess the current scope of assistive features in CAD and explore the transformative potential of the Apple Vision Pro for users with disabilities. By examining the theoretical underpinnings of the Technology Acceptance Model, affordance theory, and Cognitive Load Theory, we gain insights into user interaction, cognitive processing, and the adoption of new technologies. The review contrasts the Apple Vision Pro with similar systems, considering Rogers' Adoption Diffusion Theory to understand market adoption patterns. It advocates for a user-centric design philosophy guided by the principles of Universal Design and the Social Model of Disability, emphasizing the removal of societal barriers rather than focusing solely on user impairments. Participatory design principles also play a crucial role, advocating for direct involvement of users with disabilities in the development process. This holistic approach ensures that the proposed 3D CAD application is not only accessible and functional but also aligns with broader educational and professional impacts, fostering innovation and inclusivity in design disciplines.

# Specification

This chapter outlines the specified requirements for the envisioned 3D-CAD application on visionOS. These requirements have been distilled from comprehensive analysis and comparative studies of existing applications as detailed in sections 2.2 and 2.3. Additionally, the non-functional requirements have been extrapolated from the characteristics of visionOS, highlighted in section 2.1. This ensures that the application is not only functional but also delivers an enhanced user experience.

3.1. Primary Functional Requirements

The primary functional requirements constitute the fundamental features that the application must possess to perform its intended function. These form the essential feature set needed to meet the project's objectives.

Design Interface: The design mode should lead users to a fully functional 3D-CAD environment tailored for the Vision Pro headset. This mode must function independently but may incorporate educational aspects. It should feature an interactive workspace with tools for creating, editing, and viewing 3D models.

Reference Library: This section should provide comprehensive resources, including detailed guidelines on 3D-CAD principles, to equip new users with the necessary knowledge base to utilize the application effectively.

- Educational Features: Embedded within various modes, these features will offer:

* Textual representation of design information.
* Visual cues for precision in modelling.
* Accessibility options for users with varying abilities.

-Spatial Computing Tools: Utilizing the spatial capabilities of visionOS, these tools will support users in engaging with 3D designs in an AR environment, including model manipulation and environment interaction.

-Safety and Confirmation Mechanisms: To prevent inadvertent actions, critical functions such as application exit or significant changes to designs will require user confirmation.

3.2. Secondary Functional Requirements

These features serve as enhancements to enrich user experience and will be implemented subsequent to the primary requirements.

* Customization Settings: Users should be able to adjust settings for visual preferences, haptic feedback, and system performance to tailor the experience to their needs.
* User Profile and Analytics: The application will track user engagement and progress, offering insights into usage patterns and design efficiency.
* Sensory Feedback: Incorporating auditory and tactile feedback will make the application more immersive and engaging.
* Dynamic Animations: To create a lively and intuitive interface, animations will illustrate actions such as model assembly, transformation, and real-time collaboration.
* Help and Support: This section will detail the app's development process, offer troubleshooting guidance, and provide copyright and licensing information.
* Optimization for Vision Pro: While ensuring high-quality graphics, the application should maintain performance across the Vision Pro's range of capabilities.

3.3. Non-Functional Requirements

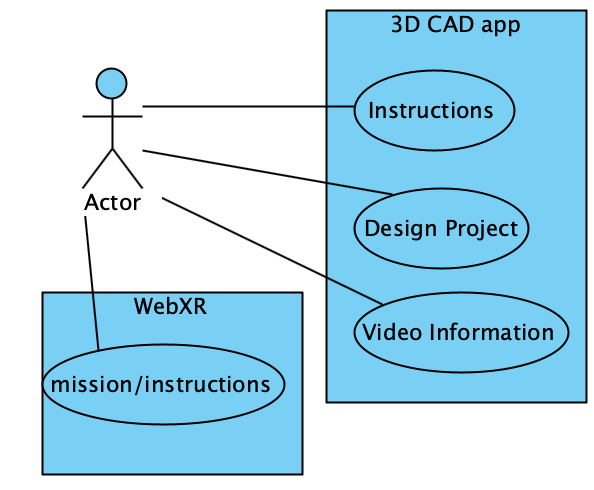
These requirements are critical for a comprehensive user experience and include:

* Cross-Compatibility: The application must function flawlessly on visionOS while retaining compatibility with other devices that can support the application in a more limited capacity.
* Adaptability: The architecture should follow a modular design to facilitate future expansions and adaptations to other AR environments or updates.
* Performance: The application should respond swiftly to user input, adhering to the performance standards established for visionOS applications. Decisions for rendering and processing should be nearly instantaneous.
* Storage and Memory Efficiency: Given the complex nature of 3D-CAD files, the application should maximize data storage and memory usage, aiming to keep the application size within the standard limits for similar software.
* Security and Privacy: Ensuring user data is handled with the highest security standards is crucial, especially when dealing with proprietary design files.

By meeting these requirements, the 3D-CAD application on visionOS will not only stand as a highly functional tool for design professionals but also as an exemplar of the advanced capabilities of AR technology in complex applications.

3.4 Use Cases

Figure 3-1: Interaction use case diagram for the AR 3D CAD Application

The AR 3D CAD application offers an immersive and interactive design experience facilitated by augmented reality technology. Users can engage with the application in two primary ways:

1. Learning the Interface and Tools: newcomers to the AR 3D CAD application can navigate to a learning module from the the webXR QR code. The learning module is rich with textual content, and visual aids that elucidate the functions and capabilities of the AR 3D CAD tools. This learning phase is bolstered by real-time AR visualizations that teach the user how to interact with 3D models and the physical space around them, offering a hands-on approach to mastering the application. Once users have grasped the basics, they can proceed to a sample design process, enabling them to apply their newfound knowledge in a controlled environment.

2. Executing Design Projects: For those familiar with AR 3D CAD applications, the user can jump straight into a new design project by selecting the appropriate option from the main menu. In this mode, users leverage AR to place and manipulate 3D models within a real-world context, refining their design skills through practice. As the user interacts with the system, they can visualize changes and additions in real-time through their AR interface. The session continues until the design project reaches its completion criteria. Afterward, users can access a summary of project metrics and achievements.

Through these use cases, the AR 3D CAD application ensures that both novices and experienced designers have a clear path to learning and applying their skills in an augmented reality environment, enhancing both their understanding and their practical design capabilities.

3.5 Evaluation

This chapter outlines the requirements identified for the application, detailing both primary and secondary functional requirements. In the latter part of the chapter, I discuss the non-functional requirements, which, while not critical for the application's basic operations, are crucial for enhancing user experience.

# Source code and Design

## 

This chapter will delve into the design and implementation of the 3D CAD application for visionOS, utilizing Swift and RealityKit for an augmented reality (AR) environment on devices with the M1 chip. This narrative encompasses the foundational concepts behind the design patterns employed, and the technical structure of the application packages.

4.2. Development Environment

The development was conducted in Xcode, Apple's integrated development environment (IDE), which is specifically designed for Swift, the programming language chosen for its modern syntax and performance efficiency. Swift provided the robustness needed for intricate CAD modelling tasks, while also enabling the use of SwiftUI, Apple’s innovative framework for designing and structuring user interfaces.

For the AR functionality, RealityKit was the framework of choice due to its high-performance 3D simulation and rendering capabilities, which is crucial for a 3D CAD application. It offers a rich set of tools and a library of pre-built assets that expedite the development process, allowing for more focus on the application's core features such as immersive interactivity and spatial understanding.

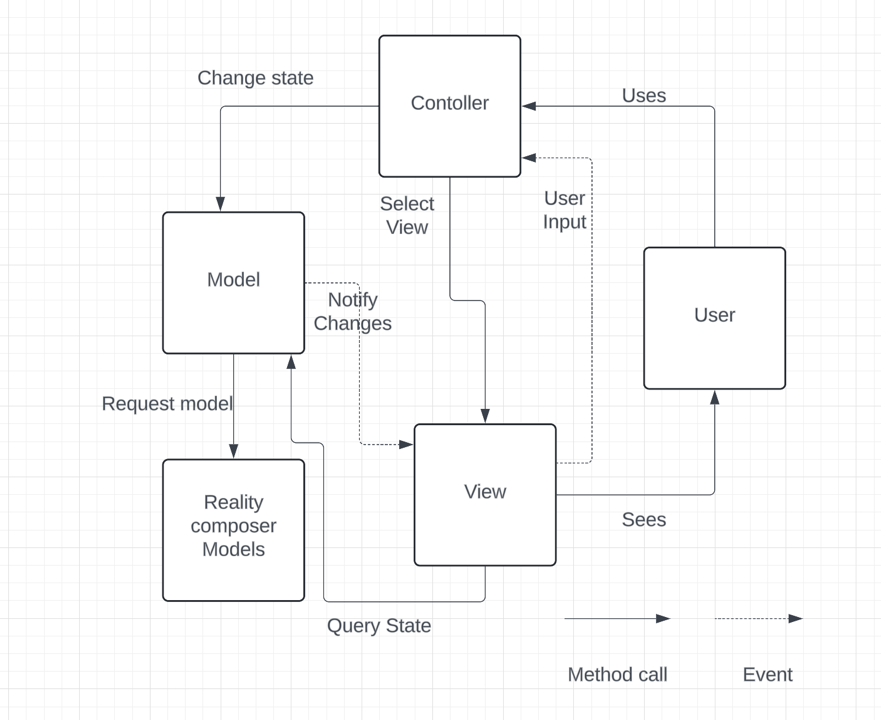


Figure 4.1: MVC graph

User Interface and Interaction

The application's UI was designed using SwiftUI, chosen for its declarative syntax that makes UI development straightforward and its inherent compatibility with RealityKit. By blending SwiftUI with RealityKit, we could craft immersive AR experiences with an interface that is not only aesthetically pleasing but also provides a seamless user experience.

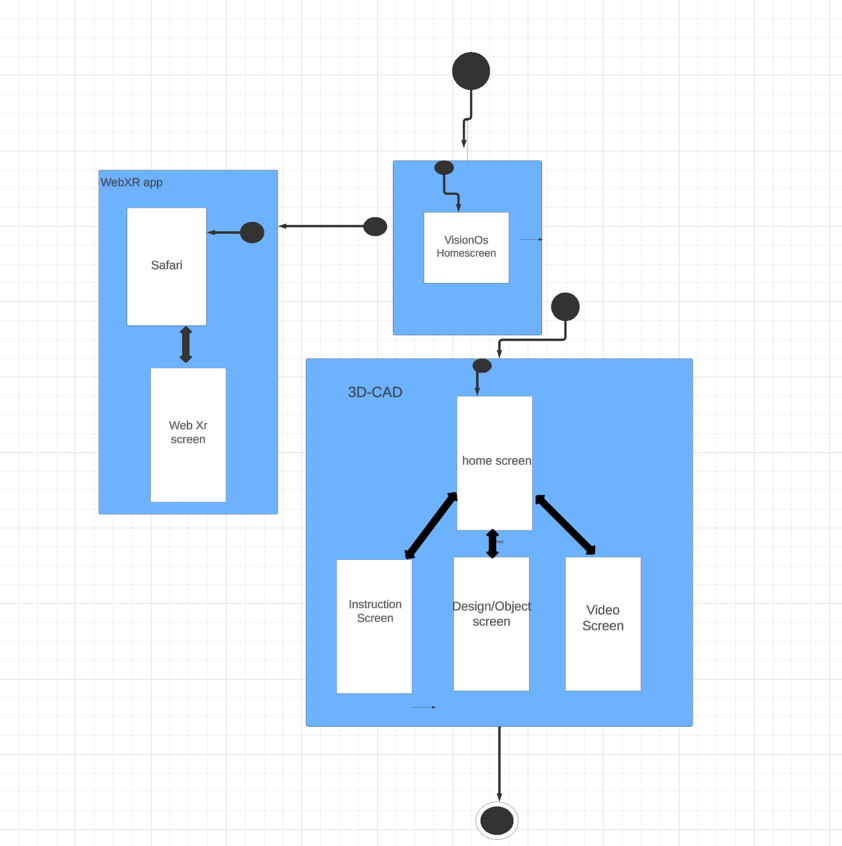
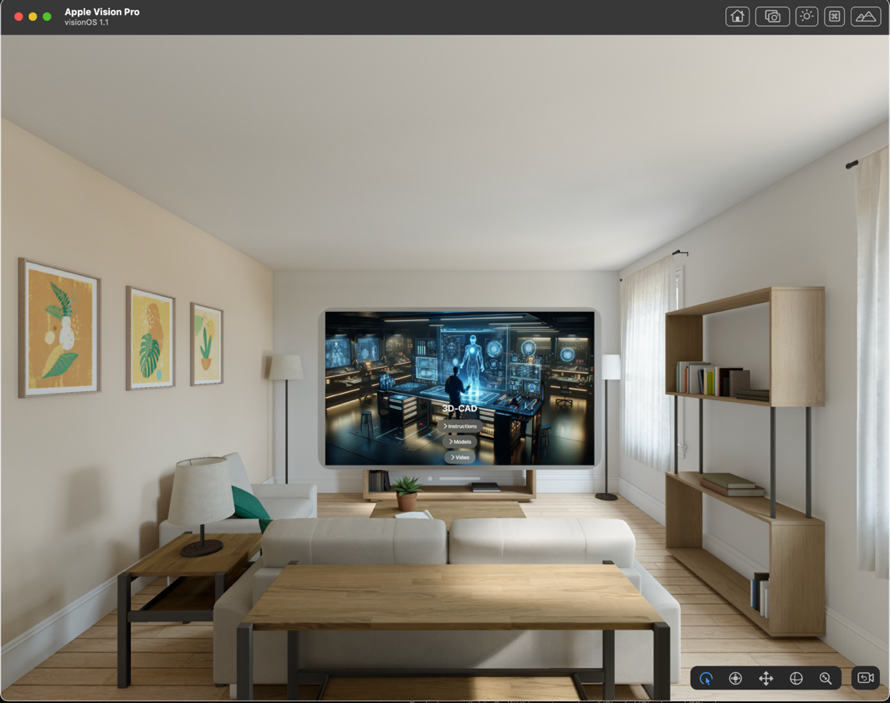
Custom gestures and controls were a focal point in ensuring a natural user interaction within the AR space. We integrated support for common gestures like tap, swipe, pinch, and rotate through RealityKit's built-in gesture recognizers. For more complex interactions, custom gestures were implemented using `UIGestureRecognizer` to bridge UIKit functionalities with the SwiftUI-based app, such as a long-press gesture to delete 3D models within the scene.

Figure 4.2: Screenflow diagram

Performance Optimization Given the resource-intensive nature of AR applications, optimizing performance was paramount. This involved streamlining the 3D models to be performance-friendly by reducing polygon count, employing efficient texturing, and optimizing scene lighting. The AR experiences crafted were thus not only realistic and immersive but also responsive and smooth on devices powered by the M1 chip.

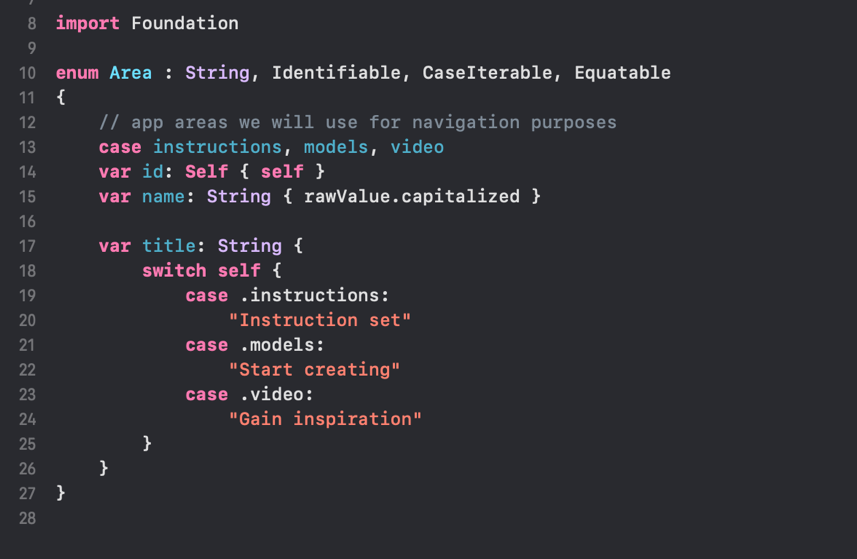


Figure 4.3: Homepage and relevant code

4.3. How the Homepage functions

The code requires imports the Foundation framework, which is commonly used in Swift programs for various foundational functionalities.

1. enum Area: String, Identifiable, CaseIterable, Equatable {

Here, an enumeration named Area is defined, inheriting from several protocols:

String: Indicates that the enum's raw values are of type String.

Identifiable: Allows each instance of Area to have a unique identifier.

CaseIterable: Provides functionality for iterating over all cases in the enum.

Equatable: Allows for comparisons between instances of the enum.

1. case instructions, models, video

These are the individual cases defined in the enum, each with a corresponding raw string value: "instructions", "models", and "video".

1. var id: Self { self }

var name: String { rawValue.capitalized }

id: A computed property that returns self, allowing instances to act as their own identifiers.

name: A computed property that returns the capitalized form of the enum's raw value. For example, if the raw value is "instructions," name would return "Instructions."

1. var title: String {

switch self {

case .instructions: "Instruction set"

case .models: "Start creating"

case .video: "Gain inspiration"

}

}

This property uses a switch statement to return a specific title string based on the current enum case:

If self is .instructions, it returns "Instruction set."

If self is .models, it returns "Start creating."

If self is .video, it returns "Gain inspiration."

Summary:

The Area enum defines various sections or areas of an application, each associated with a specific title for display purposes. The combination of protocols, computed properties, and raw values makes it convenient to manage different application sections. This setup useful for navigation menus since it is managing state across different screens



Figure 4.4 Instructions and relevant code

* 1. Instructions Page

This also utilises an enumeration (enum) and various properties and methods related to the enum very similar to what was shown in figure 4.3.

1. enum Instructions: String, Identifiable, CaseIterable, Equatable {

An enumeration named Instructions is defined, with each case having a string raw value. The enum conforms to the following protocols:

* Identifiable: This provides a unique identifier for each case.
* CaseIterable: This allows for iteration over all cases of the enum.
* Equatable: This provides comparison capabilities between different enum cases.

1. var id: Self { self }

var name: String { rawValue.capitalized }

id: A computed property that returns self, allowing the enum case to serve as its own identifier.

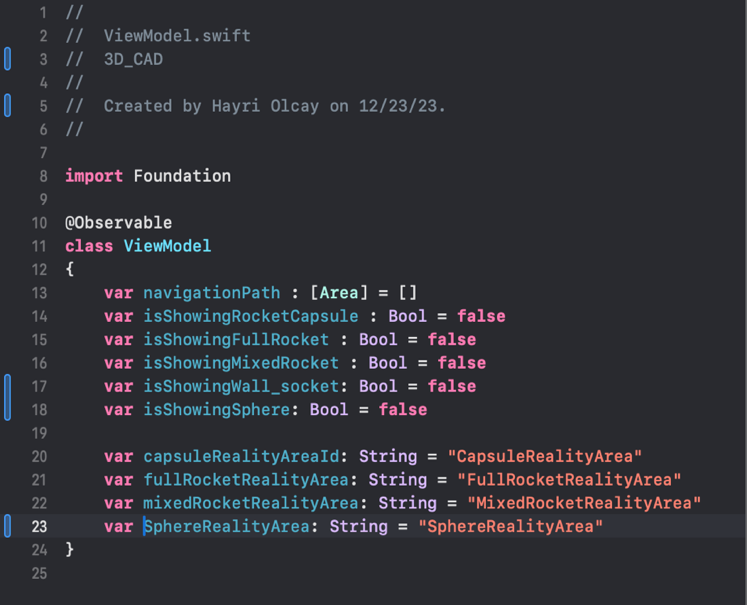
name: A computed property that returns the enum's raw value in a capitalized format. For example, if the raw value is "start," this property returns "Start."

This code defines a clear and well-structured enumeration, which represent stages in an app's functionality. Each enum case corresponds to a specific stage or step, with associated properties and information. This structure is highly suitable for managing various stages of an application, providing a coherent design for navigation, descriptions, and more

Figure 4.5. Design Menu

Observable Protocol: The ViewModel class conforms to the Observable protocol, which allows its properties to be observed for changes, making it ideal for managing application state in a SwiftUI app.

Properties:

navigationPath: An array of Area instances, likely used to manage navigation within the app or different AR experiences.

Boolean Flags: Several Boolean properties manage the visibility state of different components or areas:

isShowingRocketCapsule

isShowingFullRocket

isShowingMixedRocket

isShowingWall\_socket

isShowingSphere

These flags are all initialized to false, indicating that the corresponding areas or components are not shown initially.

Identifiers: The class also defines several string identifiers, each corresponding to a different "Reality Area":

capsuleRealityAreaId = "CapsuleRealityArea"

fullRocketRealityArea = "FullRocketRealityArea"

mixedRocketRealityArea = "MixedRocketRealityArea"

sphereRealityArea = "SphereRealityArea"

Usage: The ViewModel's properties and methods can be used to dynamically control and monitor which AR or 3D content is visible, making it suitable for integration with RealityKit or other frameworks that handle AR content.

In summary, this code defines a ViewModel class tailored to manage an AR or 3D content app. It includes properties for navigation and state management, allowing seamless transitions between different content areas or scenes, thus enhancing the user experience.



@Environment(ViewModel.self) private var model

This gets the viewmodels and this process is called projecting. The access to the environment information which is then injected into the actual 3D-CAD app. Once injected, the system has access to any environment information within all the views (this is how you access the data stored in ViewModel)

@Environment(\.openWindow) private var openWindow

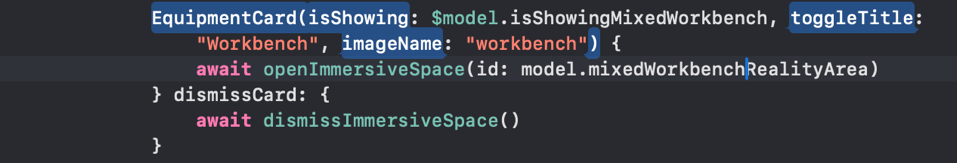
@Environment(\.dismissWindow) private var gvghhv dismissWindow @Environment(\.openImmersiveSpace) private var openImmersiveSpace.

@Environment(\.dismissImmersiveSpace) private var dismissImmersiveSpace

These handle the windows that bind the models and allow the tabs to be open and closed

VStack{Image("sphere") -🡪 .glassBackgroundEffect() This whole body of code is what determines the specifications for the generated model (in this case the sphere) and the height and width as clearly defined when it is first launched (this is resizable as seen above). In this example, the sphere is spawned in with a toggle button which is a ternary operator. “isOn” reads the bindable property defined above it and and then $model.isShowingSphere) designates that it is binding to that property. Then the button binds to the onChange which commits the activity based on what is currently active.

4.5.1 Workbench that becomes part of the Environment



EquipmentCard Function:

The function takes several parameters to configure its behavior:

isShowing: A Boolean flag tied to the isShowingMixedWorkbench property of a model object, determining if the card is visible.

toggleTitle: A string value for a title, set to "workbench" here.

imageName: Another string value set to "workbench," referring to an image resource associated with the card.

Content Block:

Inside the function, an asynchronous block of code performs actions related to the card's behavior:

openImmersiveSpace: This function is called with an identifier id: set to model.mixedWorkbenchRealityArea. This indicates an action to transition to an immersive AR space or environment identified by model.mixedWorkbenchRealityArea.

Dismiss Block:

The card also defines a dismissCard block, which when triggered, calls the dismissImmersiveSpace function asynchronously to close the immersive AR space or environment.

In summary, this code configures an EquipmentCard UI component, managing its visibility and transition to or from an immersive AR experience. The integration of RealityKit or a similar framework is implied, making this code suitable for managing both UI elements and AR content transitions smoothly. The asynchronous handling ensures non-blocking updates, which is crucial for maintaining responsiveness in such applications.

Figure 4.5.1: View of Workbench

Figure 4.6: Object manipulation



CapsuleRealityArea Struct: This struct conforms to the View protocol, making it a valid view component within a SwiftUI application. It's designed to manage AR content specifically.

ViewModel and State Properties:

@Environment(ViewModel.self) injects a ViewModel instance into the struct, allowing access to shared app state or other environment-specific properties.

@State private var capsule: Defines a capsule variable of type Entity, which holds the primary AR object or scene to be managed.

attachmentID: A constant attachmentID is defined, likely to reference specific attachments or components in the AR scene.

RealityView: Within the body property, a RealityView is constructed, managing both content and attachments.

Entity Loading: The code attempts to load an Entity named "Scene" from a content bundle. If this fails, a fatal error is triggered to stop execution.

**Setting Up the Capsule:**

The loaded entity is assigned to the capsule variable.

The capsule's sunlight intensity is set to 13.

An OrbitComponent is assigned to the capsule, with specific parameters like radius, speed, and orientation rotation.

Attachments:

The code attempts to fetch an entity attachment based on attachmentID.

If successful, this attachment's position and rotation are modified directly using the SIMD3 and simd\_quat types, indicating precise transformations.

Update and Attachments Sections:

An update block responds to changes in the RealityView, printing a log message.

An attachments block handles different capsule details, including:

Turning off or on sunlight.

Managing orbit speed and state for the capsule's orbit component.

onDisappear: Defines a cleanup action for when the view is dismissed, updating isShowingRocketCapsule to false in the shared model.

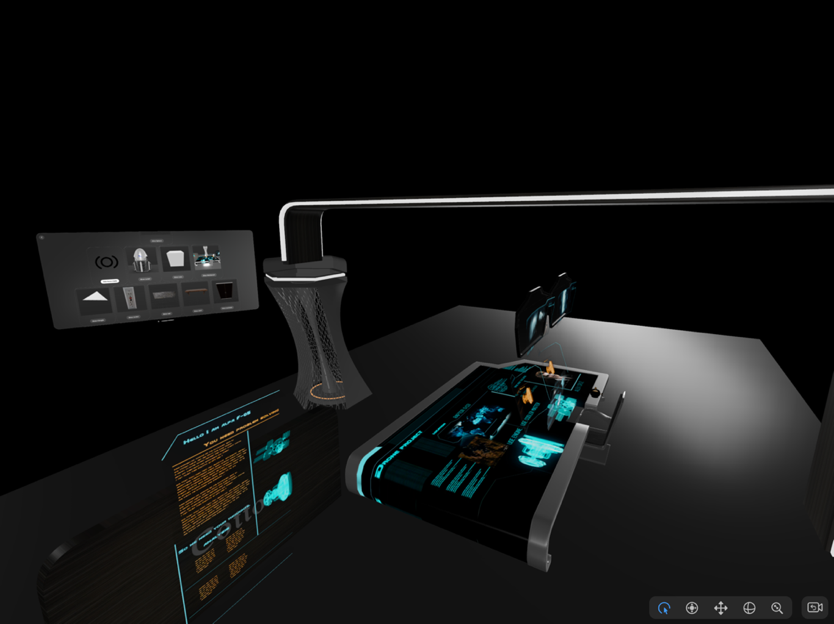
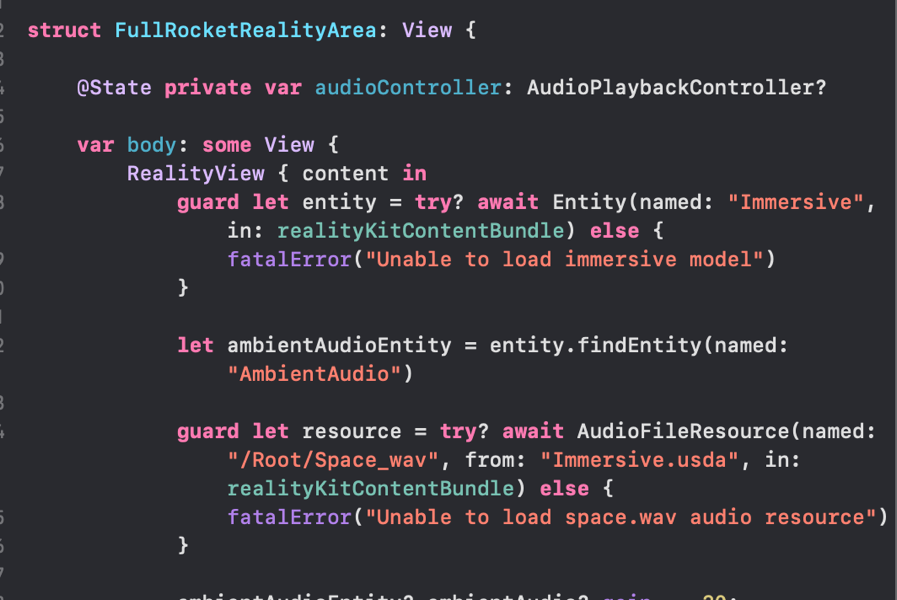
4.7.Focus Mode

Figure 4.7: Focus mode in App



This functions very similary to the workbench however it is given immersive properties. It additionally has ambient music. The @State keyword is used to define a audioController variable of type AudioPlaybackController. This suggests that the view's state is tied to the control of audio playback.

Entity Loading: The try? await construct is used to attempt to load an Entity named "Immersive" (which is an immersive environment tied to the Holotable) from a specified content bundle. The guard statement ensures that if this loading fails, a fatal error is thrown, stopping execution.

Audio Entity: Once the main Entity is loaded, the code retrieves an Entity named "AmbientAudio," likely intended to provide background sound for the AR experience.

Audio Resource: Another guard statement attempts to load an audio resource from a file named "Space.wav," specifying the path and bundle to load it from. If this fails, another fatal error is triggered.

4.8. Web XR missions

There is also a web Xr application that can be run by scanning a QR code however this did not require any coding. This application works for all VR headset however it should be noted that the max file size is only 50mb so a full-fledged cross platform application was not possible. When the user opens the app, they are greeted with this screen and are simply given text instructions on how the app works alongside a challenge.

# Evaluation

5.1. Portability and Cross-Platform Compatibility

Portability testing focused on leveraging the native capabilities of iOS devices equipped with the M1 chip. SwiftUI's responsive design interface allows the 3D CAD application to adapt to various screen sizes and resolutions natively. The AR functionalities, powered by RealityKit and ARKit, were rigorously tested across different visionOS versions (1.0 and 1.1) to ensure consistent behaviour in AR sessions, taking into account the diverse input methods and system gestures specific to Apple devices

5.2. Application Stability and Load Testing

- Load Testing: The application’s stability was tested by simulating extended AR sessions, given the lengthy processes CAD applications often involve, such as rendering and simulations. This ensured that the application's rendering pipeline and computational logic could handle prolonged use without performance degradation

- Thread Management: Swift's concurrency model and RealityKit's performance optimization tools help manage threading, ensuring efficient handling of multiple tasks without overloading the system. This is crucial for maintaining stability in an AR environment where multiple entities interact dynamically within the same scene.

- Entity Behaviour: Stability also required testing the behaviour of individual entities within the AR scene, ensuring they interacted cohesively and that no unexpected behaviours disrupted the user experience. Entity tests examined interactions, physics simulations, and user-triggered actions to ensure consistency and reliability.

- AR Session State: RealityKit's management of AR sessions ensures smooth transitions between states, preventing session drops or unexpected behaviour. This helped maintain the application's stability, allowing users to navigate between design tasks seamlessly without experiencing interruptions.

- Resource Management: Memory and resource management strategies were implemented to prevent crashes or slowdowns due to resource-intensive tasks. This included optimized memory allocation for entities, asset bundling, and careful session state handling to keep runtime memory usage efficient.

5.3. User Acceptance Testing

User Acceptance Testing was conducted with a focus on intuitive interaction within the AR space. Real users engaged with the CAD application’s AR features, providing feedback on its usability and functionality. SwiftUI’s integration with RealityKit allowed for an immersive and interactive experience, with a particular emphasis on the application's ease of use and the natural feeling of the AR interactions, leveraging both SwiftUI views and RealityKit entities for an immersive experience.

5.4. Performance and Efficiency Testing

Performance and efficiency are critical facets of the AR 3D CAD application's success, especially given the real-time nature of CAD tasks in an augmented reality environment. Here's a closer look at how performance was optimized:

- Frame Rate and Rendering: Ensuring smooth frame rates and efficient rendering was a key priority. This was achieved by leveraging RealityKit's rendering pipeline, which seamlessly integrates with the M1 chip's hardware acceleration capabilities. The Universal Render Pipeline (URP) was employed, allowing for streamlined graphics rendering and foveated rendering, which optimizes visual fidelity by focusing resources where the user's gaze is directed, thus reducing unnecessary GPU workload.

- Memory Management: AR applications, particularly those involving complex 3D models, can quickly become memory-intensive. Therefore, the CAD application employed strategies such as efficient texture management and object pooling. This not only reduced memory consumption but also helped to prevent runtime memory issues and ensured smooth user experiences.

- Performance Profiling: Xcode's built-in performance profiling tools were used to monitor CPU and GPU usage, memory allocation, and other performance metrics during testing. This allowed for identification and resolution of bottlenecks, resulting in a seamless AR experience without lag or frame drops.

- Input Responsiveness: The application's input responsiveness was another focal point, as users interact with the AR environment through various gestures and controls. SwiftUI's integration with RealityKit facilitated instant feedback for gestures such as tapping, swiping, and pinching, ensuring smooth and intuitive navigation.

5.5. Memory Management and Application Size

Given the sophisticated nature of CAD applications, memory management and file size are paramount considerations:

- Memory Footprint: To maintain a lightweight application, the CAD software implemented techniques such as texture compression and asset bundling. This ensured that the memory footprint remained low, even with complex CAD functionalities and 3D models integrated into the AR environment.

- Storage Efficiency: The final application package was designed to be compact, ensuring it did not exceed a reasonable file size. This was achieved by minimizing redundant assets and implementing efficient data structures, allowing for the inclusion of necessary CAD functions without bloating the storage requirements.

- Session Management: RealityKit's efficient management of AR sessions played a pivotal role in keeping the application's runtime memory usage under control, preventing memory leaks and minimizing resource usage during extended sessions

In conclusion, the technical testing of the AR 3D CAD application using SwiftUI and RealityKit was comprehensive, addressing the specific demands of the visionOS platform and the M1 chip's capabilities. The focus was on ensuring that the application was intuitive, efficient, and stable, providing a seamless user experience that fully leverages the advantages of Apple’s augmented reality frameworks.

# Conclusion

This chapter offers insights into my journey through the development of an innovative AR 3D CAD application using the RealityKit Xcode editor, detailing the learning curves, triumphs, and setbacks. I reflect on what was accomplished, the hurdles overcome, and how I would approach things differently in the future.

6.1. What I Learned and Achieved

Embarking on the creation of an AR 3D CAD application represented a significant personal milestone in software development, especially within the realm of augmented reality and RealityKit programming. I take immense pride in the robust functionality and the user-friendly interface of the application.

The project served as an exceptional opportunity to apply theoretical knowledge from my Computer Graphics and User Interface Design courses to a practical scenario. The M1 chips powerful rendering capabilities were instrumental in achieving high-quality visuals, and its compatibility with AR tools allowed for a seamless integration of virtual designs with the real world.

I was particularly pleased with the application's intuitive design, which proved to be adaptable and precise on a variety of devices, maintaining a professional aesthetic consistently. The use of swifts versatile scripting and prefabs led to a modular design, which could potentially serve as a scaffold for future CAD applications.

The application not only meets all functional and non-functional requirements but does so with a finesse that I believe marks a pinnacle in my academic and developmental pursuits.

6.2. Extended Challenges Faced

The journey of developing an AR 3D CAD application using the Unity engine was paved with a unique set of challenges that tested my technical prowess, problem-solving skills, and creative capabilities. Unity, while robust and supportive for a variety of development needs, came with its complexities.

One of the first hurdles was managing performance for the AR functionality, ensuring that the application ran smoothly across various devices without draining the battery or causing lag. Xcode's rendering system demanded a careful balance between quality and performance, necessitating the use of techniques like occlusion culling, LOD (Level of Detail), and careful management of draw calls to maintain that balance.

Creating an intuitive and responsive user interface (UI) for the CAD application was another significant challenge. SwiftUI's UI tools, though flexible, are not as developed or standardized as one might find in other engines or native development environments like the normal Swift used in Xcode for iOS. This meant that for more complex or scalable solutions, a lot of custom work was necessary, sometimes requiring the creation of custom plugins or scripts to achieve the desired functionality.

Ensuring clean code and architecture was a task in itself. Currently SwiftUI's environment does not inherently promote best programming practices, which can lead to disorganized project structures and suboptimal architectural decisions. Avoiding pitfalls such as relying too heavily on static classes, and ensuring a maintainable codebase through the use of design patterns and Swift's best practices, was essential for the long-term sustainability and expandability of the application.

Working with the new SwiftUI language and its garbage collection also posed challenges. Avoiding memory leaks and ensuring efficient memory management was crucial for maintaining performance, especially given the complexity and resource-intensive nature of CAD applications. This entailed avoiding unnecessary allocations in update loops and being mindful of the Swift language constructs that could impact performance.

Furthermore, Xcode's multi-threading limitations meant that optimizing for heavy computational tasks, like those found in CAD applications, required a judicious use of Coroutines and a deep understanding of Xcode’s SceneKit and how it interacts with the AR components.

Lastly, as the application evolved, the importance of rigorous testing became evident. Manual testing, while necessary, could become a tedious and time-consuming process. Hence, developing a suite of automated tests to streamline this process and ensure the reliability of each application feature became a significant focus.

In hindsight, these challenges were learning opportunities. They underscored the importance of thorough planning, continuous learning, and adapting to the unique demands of AR and Xcode development. As the application took shape, each challenge surmounted contributed to a stronger, more resilient final product.

6.3. What Would I Add or Do Differently

In retrospect, the application's success does not overshadow the potential for improvement. Given more time and resources, several enhancements could be implemented:

- Enhanced Collaboration Tools: Inspired by the multiplayer capabilities of gaming in Unity, I would integrate more robust collaboration features that allow multiple users to work on a single design in real time.

- Advanced Geometric Algorithms: To push the boundaries of what the application can do, I would like to incorporate more sophisticated algorithms for geometry processing and computational design.

- Comprehensive User Testing: While I conducted initial testing, a broader testing phase could provide invaluable feedback for refining user interactions and workflows.

- Greater Customization: Introducing more customization options for users to tailor the workspace to their preferences could significantly enhance user experience.

6.4. Summary

Reflecting on the development of this AR 3D CAD application, I am filled with a sense of accomplishment and a recognition of the growth I've experienced as a developer. This project has not only tested my technical abilities but has also taught me the value of persistence and adaptability. The potential for future expansion and the foundation laid for subsequent innovations make this endeavor a defining feature of my academic and professional portfolio.

References

* Apple (2023). *Apple Vision Pro*. [online] Apple. Available at: <https://www.apple.com/apple-vision-pro/>.
* DPReview. (n.d.). *Apple Vision Pro: What it means for content and photography*. [online] Available at: https://www.dpreview.com/articles/0778031862/apple-vision-pro-what-it-means-for-content-and-photography [Accessed 26 Nov. 2023].
* Digital Trends. (2023). *Apple Vision Pro vs. Meta Quest Pro: will Apple hold up?* [online] Available at: https://www.digitaltrends.com/computing/apple-vision-pro-vs-meta-quest-pro-best-premium-vr-headset/#:~:text=Apple%E2%80%99s%20new%20Vision%20Pro%20is [Accessed 26 Nov. 2023].
* Apple Newsroom. (n.d.). *Introducing Apple Vision Pro: Apple’s first spatial computer*. [online] Available at: https://www.apple.com/newsroom/2023/06/introducing-apple-vision-pro/#:~:text=The%20breakthrough%20design%20of%20Vision [Accessed 26 Nov. 2023].
* DPReview. (n.d.). *Apple Vision Pro: What it means for content and photography*. [online] Available at: https://www.dpreview.com/articles/0778031862/apple-vision-pro-what-it-means-for-content-and-photography.
* murata.eetrend.com. (n.d.). *Apple Vision Pro的空间计算与主要芯片 | 电子创新元件网*. [online] Available at: https://murata.eetrend.com/blog/2023/100571619.html [Accessed 29 Apr. 2024].
* Guinness World Records. (2019). *The gamer who plays Fortnite with his mouth and smashes records*. [online] Available at: https://www.guinnessworldrecords.com/news/book/2019/9/rockynohands-the-disabled-fortnite-gamer-who-uses-a-mouth-operated-joystick-to.
* Moments Log. (No Date) Working with RealityKit: Building Interactive AR Scenes in Swift. Available at: https://www.momentslog.com (Accessed: 29 April 2024).
* Create With Swift. (2023) Creating an augmented reality app in SwiftUI using RealityKit and ARKit. Available at: https://www.createwithswift.com/creating-an-augmented-reality-app-in-swiftui-using-realitykit-and-arkit/ (Accessed: 29 April 2024).
* I can provide a list of sources for the theories mentioned, formatted in Harvard style. Note that this list is based on common references available for each theory:
* Davis, F.D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, 13(3), pp. 319-340.
* Sweller, J. (1988). Cognitive Load During Problem Solving: Effects on Learning. Cognitive Science, 12(2), pp. 257-285.
* Sweller, J., Ayres, P., & Kalyuga, S. (2011). Cognitive Load Theory. Springer.
* Rogers, E.M. (2003). Diffusion of Innovations. 5th ed. Free Press.
* Mace, R. (1985). Universal Design: Housing for the Lifespan of All People. Design Quarterly, 134, pp. 32-34.
* Oliver, M. (1996). \*Understanding Disability: From Theory to Practice\*. Palgrave.
* Norman, D.A. (1988). The Design of Everyday Things. Basic Books.
* Create With Swift. (2023) Adding custom gestures to an AR application with SwiftUI. Available at: https://www.createwithswift.com/adding-custom-gestures-to-an-ar-application-with-swiftui/ (Accessed: 29 April 2024).
* Unity. (2024) PolySpatial XR Documentation. Available at: https://docs.unity3d.com/Packages/com.unity.polyspatial.visionos@1.0/manual/index.html (Accessed: 29 April 2024).
* Unity Blog. (2024) Start building spatial apps for Apple Vision Pro with Unity. Available at: https://blog.unity.com/ (Accessed: 29 April 2024).
* Moments Log. (No Date) Working with RealityKit: Building Interactive AR Scenes in Swift. Available at: https://www.momentslog.com (Accessed: 29 April 2024).
* Gravity Sketch. (n.d.). *Product*. [online] Available at: https://www.gravitysketch.com/products [Accessed 29 Apr. 2024].
* Shapelab. (n.d.). *Home*. [online] Available at: https://shapelabvr.com/ [Accessed 29 Apr. 2024].

Appendix A – risk assessment

| **Description of Risk** | **Description of Impact** | **Likelihood** | **Impact** | **Impact Rating** | **Prevention Actions** |
| --- | --- | --- | --- | --- | --- |
| Technological Compatibility | Inadequate integration with Vision Pros leading to limited functionality | Medium | Moderate | Medium | Rigorous testing and early prototyping on the Vision Pros platform. |
| User Adoption | Low usage due to cost or aversion to new technology | High | High | High | Market research, develop compelling USPs, and effective marketing strategies. |
| Development Costs | Overspending without guarantee of return on investment | High | Severe | High | Secure funding, budget management, and phased development. |
| Technical Performance | Poor performance leading to negative user experience | Medium | High | High | Performance optimization and ongoing benchmarking against user expectations. |
| Health and Safety | Health issues like VR sickness from prolonged use | Low | Moderate | Medium | Health guidelines in-app and features to reduce discomfort. |
| Accessibility | Exclusion of users with disabilities | Medium | High | High | Adherence to accessibility standards and inclusive design testing. |
| Intellectual Property | Legal disputes or loss of proprietary advantage | Low | Severe | High | IP review, secure patents, and monitor for infringement. |
| Market Competition | Loss of market share to competitors | High | High | High | Market trend analysis, continuous innovation, and differentiation. |
| Regulatory Compliance | Non-compliance leading to legal and financial repercussions | Low | Severe | High | Regular compliance audits and legal consultations. |
| Data Security | Breach leading to loss of user trust and legal issues | Medium | Severe | High | Robust cyber-security protocols, data encryption, and user education. |

Appendix B-timeline

| **Task Name** | **Start Date** | **Due Date** | **Duration (weeks)** | **Description** |
| --- | --- | --- | --- | --- |
| Market Analysis | 2023-11-26 | 2023-12-24 | 4 | Assess market demand, identify target users, and analyse competitors. |
| Conceptual Design | 2023-12-25 | 2024-01-22 | 4 | Outline the application’s features, user interface, and user experience. |
| Technical Feasibility Study | 2024-01-23 | 2024-02-20 | 4 | Evaluate the technical requirements and ensure compatibility with Apple Vision Pros. |
| Prototype Development | 2024-02-21 | 2024-03-21 | 4 | Develop an initial prototype of the application to test core functionalities. |
| User Testing & Feedback | 2024-03-22 | 2024-04-19 | 4 | Conduct testing with target users and gather feedback to refine the application. |
| Final Development & Launch | 2024-04-20 | 2024-04-29 | 1.5 | Implement feedback, finalize the application, and prepare for launch. |