Bachelor Project

Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

New techniques for selecting, manipulating and placing objects for world editing inside VR

Dmytro Kadar

Supervisor: Ing. Uršuľa Žákovská
Field of study: Open Informatics
Subfield: Computer Games and Graphics
May 2023

BACHELOR'S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Kadar Dmytro**	Personal ID number:	**496001**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Computer Graphics and Interaction**		
Study program:	**Open Informatics**		
Specialisation:	**Computer Games and Graphics**		

II. Bachelor's thesis details

Bachelor's thesis title in English:

New techniques for selecting, manipulating and placing objects for world editing inside VR

Bachelor's thesis title in Czech:

Nové techniky výběru, manipulace a umístění objektů pro editaci světa ve VR

Guidelines:

Review state-of-the-art and industry standards for VR and non-VR world creation. Focus separately on manual creation and procedural generation in the game and art industry. Explain the differences between world creation actions such as searching, selecting, manipulating, and placing an item. Look at available technology (AI, gesture recognition, voice recognition, etc.) and try to suggest new approaches to these actions for world creation inside VR. Design a list of scenes showcasing different types of editing (e.g. furnishing a room, panning out a landscape, placing particle effects), and for each scene, propose a technical execution of editing actions incorporating some of the new approaches. After consulting your mentor, choose three scenes to implement and later use in the testing scenarios. Conduct user testing of the implemented actions with five participants with previous game-making and VR experience.

Bibliography / sources:

[1] Difeng Yu, Xueshi Lu, Rongkai Shi, Hai-Ning Liang, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves. 2021. Gaze-Supported 3D Object Manipulation in Virtual Reality. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21). Association for Computing Machinery, New York, NY, USA, Article 734, 1–13. https://doi.org/10.1145/3411764.3445343

[2] Joanna Bergström, Tor-Salve Dalsgaard, Jason Alexander, and Kasper Hornbæk. 2021. How to Evaluate Object Selection and Manipulation in VR? Guidelines from 20 Years of Studies. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21). Association for Computing Machinery, New York, NY, USA, Article 533, 1–20. https://doi.org/10.1145/3411764.3445193

Name and workplace of bachelor's thesis supervisor:

Ing. Uršula Žákovská Department of Computer Graphics and Interaction FEE

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **17.02.2023** Deadline for bachelor thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

Ing. Uršula Žákovská Supervisor's signature	Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

. Date of assignment receipt	Student's signature

Acknowledgements

Thanks to my family and friends for their help and support. Special thanks to my supervisor, Ing. Uršuľa Žákovská, and and Ing. David Sedláček, Ph.D.

Declaration

I hereby declare that I have completed this thesis independently and that I have used only the sources (literature, software, etc.) listed in the enclosed bibliography.

In Prague on 24. May 2023

Abstract

As the popularity of Virtual Reality (referred to as VR in later parts of the document) grows, more and more developers and designers are working with this type of application. However, when it comes to creating scenes in VR, challenges arise. While scenes can be created on a regular computer monitor, it becomes difficult to perceive them from the player's perspective, requiring developers to test them using a VR headset. In my project, I aim to address this problem by exploring the concept of pre-creating scenes directly in VR.

Keywords: VR, Unity, object selection, object manipulation

Supervisor: Ing. Uršuľa Žákovská

Abstrakt

S rostoucí popularitou virtuální reality s tímto typem aplikací pracuje stále více vývojářů a designérů. Při vytváření scén ve VR však dochází k problémům. Zatímco scény lze vytvářet na běžném monitoru počítače, jejich vnímání z pohledu hráče je obtížné, což vyžaduje, aby je vývojáři testovali pomocí náhlavní soupravy VR. Ve svém projektu se snažím tento problém řešit zkoumáním konceptu předtváření scén přímo ve VR.

Klíčová slova: VR, Unity, výběr objektu, manipulace s objekty

Překlad názvu: Nové techniky výběru, manipulace a umístění objektů pro editaci světa ve VR

Contents

Figures

Tables

Chapter 1

Introduction

As the popularity of Virtual Reality grows, more and more developers and designers are working with this type of application. However, when it comes to creating scenes in VR, challenges arise. While scenes can be created on a regular computer monitor, it becomes difficult to perceive them from the player's perspective, requiring developers to test them using a VR headset. In my project, I aim to address this problem by exploring the concept of pre-creating scenes directly in VR.

In this work, my focus will be on observing the current methods used for object selection, manipulation, and placement during editing in VR. I will examine the existing techniques and their limitations. Following that, I will propose and implement my own solutions to address the challenges associated with object selection in VR. Those solutions will then be tested to evaluate their effectiveness and usability.

Chapter 2

Current object manipulation, selection and placement methods in VR

2.1 Introduction

The challenge of effectively manipulating 3D objects and creating virtual scenes has been a long-standing problem. One example of an attempt to address this issue is the paper "3D Palette: A Virtual Reality Content Creation Tool"[BBMP97] from 1997. In this paper, the authors proposed an interface solution that combined a graphics tablet, voice recognition technology, and various tracking devices. The system utilized a Wacom tablet and digitizing pen, which were both tracked in space using Polhemus Fastrak electromagnetic sensors. Additionally, a head-mounted microphone and CrystalEyes stereoscopic shutter glasses were employed. This setup allowed users to hold the tablet in one hand while interacting with the scene using a combination of 2D and 3D pen gestures, as well as voice commands.

I have found several noteworthy examples of real-world applications that involve creating scenes in virtual reality (VR). One such example is ShapesXR[sha], a powerful tool that enables users to design 3D models, incorporate sounds, and create immersive scenes within the VR environment. The created content can then be exported to the Unity game engine, providing a seamless integration for further development and utilization. What sets ShapesXR apart is its collaborative functionality, allowing teams to work together on a single project. This collaborative aspect enhances productivity and accelerates content production.

Another notable tool in this realm is VR Sketch[vrs], which allows designers to sketch and conceptualize their ideas directly within the VR space. By immersing themselves in the virtual environment, designers can intuitively manipulate and refine their designs, promoting a more organic and immersive creative process.

Additionally, the realm of augmented reality (AR) also offers compelling options for scene management and design. For instance, IKEA Place[ike] and PLNAR[pln] are noteworthy AR home design apps that empower users to envision and place virtual objects within their own living spaces.

2.2 3D objects from 2D photos

Real-world objects are commonly used as references for creating accurate and detailed 3D objects. However, advancements in technology have made it possible to automate the conversion of 2D photos into 3D objects. One approach is through the use of a 3D scene generator, which decomposes an image into separate 3D features for each object and then combines them to form a unified scene representation. This technique allows for the creation of scenes composed of multiple 3D objects[NPRM+20].

One key advantage of this approach is its ability to learn from unlabeled images. By leveraging the features extracted from the images, the generator can create realistic 3D objects that accurately reflect the visual content of the input photos.

Furthermore, this method enables various object manipulations within the scene, including rotation, movement, and scaling. Users can interact with the objects and modify their properties to suit their creative needs or design requirements.

2.3 Scene generating using speech

Additionally, human speech is an excellent source of inspiration. There is a paper discussing an approach that assists non-professionals in generating 3D scenes using natural language recognition - "3d scene creation using story-based descriptions". In my opinion, this project can also greatly benefit individuals with visual impairments, enabling them to actively participate in the creation of 3D worlds and express themselves more fully. Tests have demonstrated the successful functionality of the methods employed, particularly with non-professionals. [ZMG05].

2.4 Realistic grasping system

VR provides immersive visual and auditory experiences that closely resemble the real world. However, certain techniques within VR still have room for improvement. In the paper titled "A visually realistic grasping system for object manipulation and interaction in virtual reality environments", an algorithm is proposed to address the limitation of relying on pre-made animations for visual grasping.[OMGGG+19]

The proposed system allowing users to freely move and interact with virtual objects using handheld controllers within the simulated world. The grasping system described in the paper enables interaction and manipulation of objects regardless of their geometry. This is achieved by automatically adapting the virtual hand to the shape of the object, eliminating the need for specific grip animations for each object. By starting with a user-predefined 6D hand pose, the virtual hand can be adapted to fit multiple object shapes, simplifying the process of grasp synthesis.

2.5 Visual programming

Visual programming is a technique where traditional programming code is replaced with visual nodes. One of the most popular realizations of this concept is Scratch, which is designed for beginner programmers who want to understand programming principles. Another example is Unreal Engine Blueprints Visual Scripting, which is focused more on game designers with little programming experience and is used for game development.

Additionally, there is a paper titled "EntangleVR: A Visual Programming Interface for Virtual Reality Interactive Scene Generation" that discusses a visual programming system based on quantum computing.The paper explores the application of quantum computing concepts to creative computing, such as interactive scene creation for virtual reality and game design. The phenomenon of entanglement, which lies at the core of the disparity between classical and quantum physics, is described as a correlated relationship between measurements of spatially separated particles. In the context of quantum computing, entanglement enables the transformation of individual object states into non-separable shared states. This means that a single interaction with one object's state can have an impact on all the entangled object states as a whole.

Furthermore, the paper highlights the future expansion of the system by incorporating entanglement at different scales of scene creation. This could involve procedural scene generation and the entanglement of multiple scenes, amongst other possibilities. [CPS21]

2.6 Manipulation techniques that are not using hands

If there are reasons for not using hand tracking and manipulation in VR, or if an additional input method is desired, gaze-supported manipulations can be employed. While hand tracking is a well-known technology in VR, gaze-supported manipulation is not as widely adopted. However, there have been several research studies and implementations exploring this area.

For instance, a paper titled "Gaze-Supported 3D Object Manipulation in Virtual Reality" [YLS+21] discusses the challenges associated with gaze-based interaction, such as the lack of precision and the difficulty of confirming a selection. To address these challenges, many techniques combine gaze with an additional modality, following the principle of "gaze select, hands manipulate." One example is the Gaze-touch proposed approach, which allows users to indirectly control gaze-selected targets using multi-touch gestures on interactive surfaces. Another method, proposed by Turner et al., maps the object that the user is looking at to the touch or cursor position, enabling further manipulation.

New Meta's headset - Meta Quest Pro also has abilities to track gaze direction, so this technology might be growing in popularity with game developers and users.

2.7 3D sketching in VR

Virtual reality offers a means to expand our real-life experiences and provides plenty space for creativity. In the paper "VRSketchIn: Exploring the Design Space of Pen and Tablet Interaction for 3D Sketching in Virtual Reality," the authors present a method of creating art in VR using a pen and a tablet. They highlight that immersive environments can offer a delightful full-body painting experience. While previous works also utilize a pen, tablet, and VR, they do not specifically focus on sketching or defining a design space. It's important to note that older works were implemented with limited hardware.

2.8 Procedural generation

Procedural generation is a highly beneficial technique that allows for the creation of numerous models, expansive terrains, and more, using minimal input data. While implementing algorithms for procedural generation can be somewhat challenging, working with and understanding existing solutions is relatively straightforward. In fact, there have even been proposals for applying procedural generation to the creation of stories. According to research - "A survey on the procedural generation of virtual worlds, Multimodal Technolo- gies and Interaction 1 (2017)"[FE17], there are several advantages to this approach. Firstly, traditional story creation can be a resource-intensive process involving various professionals such as artists, content designers, programmers, and audio engineers. Employing procedural generation techniques for storytelling can significantly reduce the required effort. Also, generating new variations of stories dynamically at runtime can enhance player engagement and motivation when replaying a game. Finally, the use of procedural generation enables the creation of stories that automatically adapt to the player.[FE17]

2.9 World Creation

2.9.1 Manipulation and placement

Object manipulation involves altering an object's position, rotation, and scale. For example, in the Unity editor, this can be accomplished by using the mouse to interact with objects or by directly adjusting the values in the object inspector.

In virtual reality, object manipulation can be achieved through manual adjustments of object values or by utilizing a controller. For instance, with the Oculus Quest, you can select an object by pointing at it and then grasp it using the Grip button located on the side of the controller. Once grasped, the object will respond to your hand movements, allowing you to change its position and rotation as if you were holding it in your hand.

2.9.2 Searching

The search process can vary depending on the needs and preferences of the user or application developer. Both Unity and VR scene creation methods provide users with the ability to browse through all the available assets and find the desired ones. Additionally, they may(Unity has this feature) incorporate keyword-based searching, allowing users to input specific keywords or tags to narrow down their search and find assets more efficiently. These search functionalities aim to enhance the user experience and make it easier to find the desired assets within the vast collection of available folders.

2.9.3 Selection

Object selection and object searching are closely interconnected methods. In most cases, when users engage in object searching, they subsequently proceed with object selection. Advanced searching techniques, such as keyword search, can greatly streamline the object selection process. The common object selection approach involves pointing a controller or mouse pointer at the desired object and confirming the selection. Once selected, the object is then spawned into the scene. Additionally, in Unity, users have the option to conveniently drag and drop items directly from the assets folders into the scenes, offering an alternative approach to object selection and placement.

Chapter 3

Proposal

Using AI(Artificial Intelligence) to assist in world editing can indeed be highly beneficial. Voice recognition technology can be utilized to enable actions such as spawning objects, moving them, or even rotating them based on specific parameters specified by the user. One notable advantage of this approach is that it allows for hands-free interaction, making it particularly useful when the user's hands are occupied or when assisting individuals with limited hand mobility.

My goal is to choose object selection techniques that are more interesting, so, in my current work, I will be focused more on the selection. For implementation, I will also implement object placement techniques. As I observed, there are ways of making a usable and comfortable environment for creating scenes in VR such as these:

- pattern recognition, choosing objects by drawing shapes
- voice recognition, choosing an object by saying its name
- selecting an object from the list
- their combinations

My work will be focused on object selection using pattern recognition technology, voice recognition technology and selection from list. In the first method the user will be drawing shapes with their controller motion capture, which then will be translated to text using AI recognition algorithms. That text will represent the object name, with which we can then add the object to our scene. Second method will capture the voice of the user, commands and will do whatever the user told it to do (selecting object, possibly movement, rotation and scale of objects). In the third method the list with objects will be used.

It's a great idea to add the features of ShapesXR, such as object transforming, modeling, maybe audio editing, and exporting to Unity. Additionally, the ability to collaborate with colleagues for scene and shape creation would be a remarkable addition.

For now, I will be implementing all three approaches, but in use with object selection. For the pattern recognition I was inspired by an article on medium.com - "Neural Nets + VR = Magic!" [neu].

Chapter 4

Implementation

4.1 Introduction

Source code and scenes can be found using the link provided in [git]. It is recommended to run it directly in Unity to select a scene for testing.

Movement and object manipulations, such as moving and rotating, are made by the XR Interaction Toolkit, an integrated package within the Unity Game Engine. In order to evaluate methods, the test scenes were constructed using the POLYGON - Fantasy Kingdom Pack from the Synty Store[Fan].

The application was specifically designed and optimized for use with the Oculus Quest 2, the second model of the Oculus Quest headset. This particular headset was utilized for both deployment and testing purposes throughout the development process. The final project functions as a versatile application that can be utilized as both a standalone and connected to a Personal Computer application for the Oculus Quest and Oculus Quest 2. Additionally, the application has the functionality to save placed game objects into a separate save file.

For my implementation, I focused on three methods:

- Gesture Recognition

- Voice Recognition
- Selection from the list

As for the testing scenes, the following use cases were created:

- Interior design
- Landscape design
- Particle effects placement

4.2 Object selection implementation

4.2.1 Gesture recognition

I began my work by developing a pattern recognition algorithm, which offers an intriguing solution to the object selection problem. This algorithm involves hand movements, providing a valuable approach for creating multiple identical objects, especially when using small gestures that are fast to reproduce.

In the Unity Asset Store, there is an extension called "PDollar Point-Cloud Gesture Recognizer"[pdoa] that captures points from input gestures. With the help of cloud technology, it provides a guessed gesture name (if recognized) along with a score (zero if not recognized). This gesture recognition system defines a gesture as a single stroke, without any time limitations. However, each point that composes the gesture will disappear after 5 seconds.

To understand and utilize the PDollar library effectively, I referred to a two-part video tutorial series by Valem's channel titled "How to Detect a Movement in VR - Unity VR Tutorial"[pdob]. These tutorials served as a helpful reference for incorporating the PDollar library into my project.

To use gesture recognition, the user must press the trigger button on the right controller and draw the gesture until releasing the button. To train a gesture, the user must write the gesture name in the ObjectManipulation object properties, press the secondary button on the left controller and then draw the gesture as usual.

It is worth mentioning that composing gestures for particle effects was a difficult task.

Here are some examples of gesture sketches and their corresponding representations within the application:

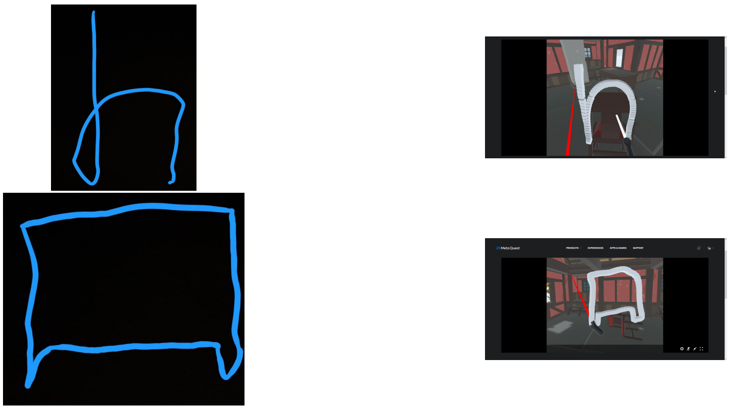

Figure 4.4: Cabinet Gesture

4.2.2 Voice Recognition

Using voice recognition was made easier with the help of wit.ai[wit], a powerful voice recognition system developed by Meta company. The wit.ai platform offers an intuitive and user-friendly interface, allowing users to define their intents (functions) and entities (object parameters in this case) easily. I was inspired by the "How to Create Custom VR Voice Commands in Unity"[vrV] video on the YouTube platform.

To incorporate voice recognition into my project, a script was implemented that activated the voice recognition functionality and processed the gathered information. This enabled the system to determine the appropriate function to use (specifically, the spawn object functionality) and the entities (in this case, the objects to be spawned).

4.2.3 Selection from list

Selection from a list was chosen as the object selection technique, as it is commonly used not only in VR but also in traditional scene creation (i.e., scene creation on a regular monitor).

For the selection from the list, default Unity User Interface components were utilized. The process began by creating a preset for the button, which consisted of the button itself, a placeholder for text, and a placeholder for an image. Subsequently, a Canvas with a Scroll View and a Panel within it were created. To populate the Panel with buttons from the assets folder (assets dedicated to the specific scene), a script was developed.

Additionally, the script utilized the Runtime Preview Generator[pre] Unity asset to enable the generation of preview images from object presets, thereby improving the visual representation of the buttons. Furthermore, the script implemented object creation functionality for each button, allowing users to spawn objects associated with buttons.

It is important to highlight that the Runtime Preview Generator did not support particle effects, as their previews were not displayed even in the Unity Editor's File Manager. However, users were able to navigate through the available options using text descriptions. An attempt was made to create a script that would add images from a folder to the buttons, but the images were not displayed. This issue could potentially be caused by incorrect image sizes or other factors.

To open the list, the user can click on the left stick. When the list is displayed, gesture recognition and the pointer ray from the left controller are disabled. This allows the user to select items from the list using the trigger on the right controller.

4.3 Scenes

4.3.1 Interior design

The interior design involved creating the layout and arrangement of objects within a small one-room house. Testers were provided with a variety of objects such as tables, chairs, wardrobes, and more to be placed in the interior. Initially, the prototypes included pre-made house designs for testers to work with. However, it was later decided to discard this idea in order to allow testers to express their creativity and not restrict them to predefined designs.

4.3.2 Landscape design

For landscape design, an improvisational location was created, featuring elements such as an ocean, rocks, mountains, and a road. Testers were given the opportunity to place various objects such as houses, trees, bushes, and more within the landscape. As the objects in the landscape were larger in size compared to interior design objects, additional work was required to ensure proper scaling of the in-application avatar to match the scene and provide an accurate representation of the user's interaction with the objects.

All objects in the scene were set to be kinematic, which means they have no gravity and do not fall or respond to external forces. This allows the user to freely place and manipulate the objects in the desired positions without any interference from physics-based movements.

4.3.3 Particle effects placement

The particle effects scene consisted of a small village situated in a valley, with four buildings placed in a circular arrangement and surrounded by mountains. At the center of the village, there was a campfire site. Each of the houses in the village was furnished with various objects. Testers had the option to select from a variety of particle effects, including butterflies, flames, steam, smoke, water, and more.

Since particle effects do not have collision properties, a decision was made to add a platform at the bottom of each spawned particle effect. The platform was designed as a cube, with the Y-axis scaled down. The object had white dot in the center and the edges of the platform were transparent, allowing the user to visually understand where they could grab the object and manipulate it within the scene. This provided a clear reference point for interacting with the particle effects and enhanced the user's control and understanding of their placement and behavior. In this scene, all objects were also set to be kinematic.

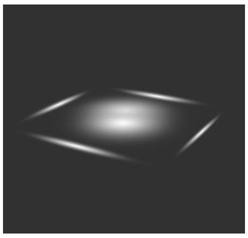

Figure 4.5: Particle platform

All three scenes were created using Polygon Fantasy Kingdom assets [Fan].

4.4 Controls

- Right Controller:
  - Trigger - draw gesture
  - Grip button - grab an object
  - Controll Stick - left
right - camera turn, forward
backward - adjust object distance.
  - Secondary button - Activate Voice recognition
- Left Controller:
  - Primary button - save created objects
  - Secondary button - change mode(Training Gesture/ Gesture Recognition)
  - Grip button - delete object
  - Control Stick - all directions - move the camera
  - Control Stick Click - open items list

4.5 Errors

1. When an object is grabbed, it is moved slightly in the player's direction, and its rotation is also adjusted. Despite attempting to resolve this issue by following a video tutorial[off], the problem persisted, and none of the other available options seemed to have any effect on the issue.

2. When an object is grabbed and brought very close to the camera, it can cause a slight change in the player's position. This issue can potentially be resolved by adjusting the camera and object layers in Unity.

3. Camera turns are mapped on the control stick in the right controller. Also, the selected object rotation is mapped on the same stick. That causes camera movement to the left and right when trying to rotate the selected object in these directions. However, sometimes when you move the stick a little to the front and then to the left or right, the camera won't move.

4. The voice recognition algorithm performed poorly with objects whose names consisted of more than one word or difficult-to-pronounce words. For example, the word "wardrobe" was often interpreted as "ward", and "tree cluster" was recognized as "tree".

5. In the second scene, it was decided to increase the spawn distance to fit the larger size of some objects. However, this led to a problem where some objects would spawn underground.

Chapter 5

Testing

5.1 Introduction

The test application was designed for both standalone and connected use cases, allowing it to be used with or without a computer. Tests were conducted using an Oculus Quest 2 headset, which was connected to a PC (laptop) equipped with an NVidia GTX 1060 graphics card and an Intel i7 8th generation processor, using a USB-C cable.

For the testing, three sample scenes were created using the Polygon Fantasy Kingdom assets [Fan]. The first scene consisted of houses without interiors, where testers were required to create the interior themselves. The second scene included fields, mountains, and an ocean, and testers were tasked with creating the landscape. In the third scene, testers were placed in a small village and were asked to add particle effects.

Testers were instructed to initially compose the scenes on a regular monitor and then recreate the scenes in the VR application using all three methods: gesture recognition, voice recognition, and selection from a list. After completing all three scenes, testers were asked to fill out forms that contained questions about which method of object selection worked best for each use case (interior design, landscape design, and particle effects placement).

While it deviates slightly from the original assignment, the decision was made to let testers determine the preferred method of object selection. Testers were also asked to choose the more reliable object selection method and evaluate whether VR scene creation was better than on a regular monitor. They were also provided with a field to write any additional comments they wished to share. Each tester conducted the tests separately and was unaware of the specific scene to be composed until the test began. Testers' feedback during the tests was also taken into account.

5.2 Test group

I have tested my implementation on 6 people mostly of age from 20 to 26. All of the testers had previous game development experience, but, unfortunately, one of them haven't had previous VR-using experience.

5.3 Results

After analyzing feedback from the testers, the methods used for testing can be evaluated. Out of the participants, 66.7 percent (4 people) preferred scene creation on the monitor. Surprisingly, voice recognition was found to be the most effective selection technique for all three scenes. However, for a more reliable and consistent method, selection from a list was the preferred choice.

Some of the testers suggested adding a feature to disable collision temporarily during the object placement phase and incorporating object attraction functionality, such as automatically aligning one of the axis coordinates of the placed object with neighboring objects. Additionally, they recommended expanding the object rotation capability to include 360-degree rotation along all three axes using the Control Stick on the controller, as the current implementation only allows rotation along a single axis.

Testers provided feedback regarding the reliability of gesture interpretation. They observed that some gestures were interpreted inconsistently, which could be attributed to various factors. One potential reason could be the limited quantity of training data provided to the PDollar asset, resulting in less accurate recognition. Another factor could be errors or limitations within the PDollar asset itself. Additionally, since the model was trained on my own handwriting, it may not generalize well to other individuals' handwriting styles. Testers found that when they attempted to mimic my handwriting, the gesture recognition worked more reliably. This highlights the importance of training the gesture recognition model the game designers themselves.

Also, testers provided valuable suggestions to improve the gesture recognition method. One of the suggestions was to utilize gestures based on letters. This approach would involve users drawing gestures resembling specific letters, and upon recognition of a letter, a list of items beginning with that letter would be spawned. By incorporating letter-based gestures, users could potentially find desired objects more quickly and accurately. Additionally, the situation can be improved by adding hints for gestures. "Hints on what gestures exist would make their usage much easier" - one of the tester's words.

One of the testers wanted to add an item highlight and navigation with a stick to the list.

Some suggestions were about voice recognition. One of the testers suggested having the option to rotate and move objects using voice recognition. Furthermore, it was suggested to include a message indicating which item was recognized, similar to how it is displayed in gesture recognition.

It is worth mentioning that testers who were not informed about the bugs during the tests did not notice all of them.

Error feedback and possible ways to solve it are in the Implementation section.

Chapter 6

Conclusions

The main objective of this study was to evaluate and identify the optimal method for object selection in VR. This research serves as an expanded version of the Unassisted project, building upon its findings and further investigating the topic of object selection in VR environments using different methods.

The preferred method, chosen by users for all three use cases, is voice recognition. This is quite surprising, as the most reliable method chosen is object selection from a list. Additionally, as mentioned by one of the testers, selecting from a list is faster when you need a large quantity of a single object. But, if voice recognition quantity choosing functionality will be added, this disadvantage would not be present.

While implementing, I have observed certain disadvantages of the gesture recognition:

- 3D artists must do additional work, for example, write gestures for models, train AI to recognize gestures, and add components to move objects in VR
- It is difficult to invent new gestures for objects, the more objects are, the less possibilities for new object gesture you have. Also, all the gestures are drawn in 2D space and with one stroke(in my representation)
- It is not reliable

In my opinion gesture recognition would be preferable in gameplay, for entertainment, rather than in scene creation. As for VR world creation, development in VR gives more perspectives. Errors are found faster, their fixes are immediate, and you don‘t have to take off your VR headset.

Worth mentioning that the current recognition methods are not very reliable at the moment. It would be great to obtain more reliable solutions, or, even train my own models for gestures and voice. Unfortunately, those are massive projects that require a huge amount of machine learning knowledge and training data, so they are not part of my Bachelor's thesis, but, maybe as a future extension to that project, it will appear someday. Also, it would be great to add the ability to select the quantity of objects.

It would be great to combine all of these methods (list, gesture recognition, procedural generation, etc.) for game developers to be able to choose methods that are more suitable for them and their team's purposes. As mentioned in the Testing chapter, it would also be a great idea to combine letter recognition using Gesture recognition with selection from list.

Using Virtual Reality to create VR worlds is very immersive, and needs to be implemented more. It is not suitable for everyone, due to personal preferences or health conditions, but I hope that it will soon be more popular among developers.

Appendix A

Bibliography

[BBMP97] Mark Billinghurst, Sisinio Baldis, Lydia Matheson, and Mark Phillips, *3d palette: A virtual reality content creation tool*, ACM Symposium on Virtual Reality Software and Technology, Proceedings, VRST (1997), 155–156.

[CPS21] Mengyu Chen, Marko Peljhan, and Misha Sra, *Entanglevr: A visual programming interface for virtual reality interactive scene generation*, Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology (New York, NY, USA), VRST '21, Association for Computing Machinery, 2021.

[Fan] *POLYGON Fantasy Kingdom - Low Poly 3D Art by Synty*, `https://assetstore.unity.com/packages/3d/environments/fantasy/polygon-fantasy-kingdom-low-poly-3d-art-by-synty-164532`, [Online; accessed 24-May-2023].

[FE17] Jonas Freiknecht and Wolfgang Effelsberg, *A survey on the procedural generation of virtual worlds*, Multimodal Technologies and Interaction **1** (2017), no. 4.

[git] *Repository with Bachelor's project*, `https://github.com/dmytrokadar/SemestralVR`, [Online; accessed 25-May-2023].

[ike] *IKEA Place*, `https://apps.apple.com/us/app/ikea-place/id1279244498`, [Online; accessed 24-May-2023].

[neu] *Neural Nets + VR = Magic!*, https://medium.com/hackernoon/neural-nets-vr-magic-3b066538aa5d, [Online; accessed 24-May-2023].

[NPRM+20] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra, *Blockgan: Learning 3d object-aware scene representations from unlabelled images*, vol. 33, Curran Associates, Inc., 2020.

[off] *Simple Offset Interactable for XR Toolkit (Update) - Andrew*, https://www.youtube.com/watch?v=XRaNYOXU0hw, [Online; accessed 24-May-2023].

[OMGGG+19] Sergiu Oprea, Pablo Martinez-Gonzalez, Alberto Garcia-Garcia, John A. Castro-Vargas, Sergio Orts-Escolano, and Jose Garcia-Rodriguez, *A visually realistic grasping system for object manipulation and interaction in virtual reality environments*, Computers Graphics **83** (2019), 77–86.

[pdoa] *PDollar Point-Cloud Gesture Recognizer*, https://assetstore.unity.com/packages/tools/input-management/pdollar-point-cloud-gesture-recognizer-21660, [Online; accessed 24-May-2023].

[pdob] *video with PDollar usage*, https://www.youtube.com/watch?v=GRSOrkmasMM, [Online; accessed 24-May-2023].

[pln] *Plnar*, https://plnar.ai, [Online; accessed 24-May-2023].

[pre] *Runtime Preview Generator*, https://assetstore.unity.com/packages/tools/camera/runtime-preview-generator-112860, [Online; accessed 24-May-2023].

[sha] *ShapesXR - VR Creation and Collaboration Platform for Remote Teams*, https://www.shapesxr.com/, [Online; accessed 24-May-2023].

[vrs] *VRSketch - Design inside virtual reality*, https://vrsketch.eu, [Online; accessed 24-May-2023].

[vrV] *How to Create Custom VR Voice Commands in Unity - Linkira Studios*, https://www.youtube.com/watch?v=SJ96P-ZhBoc, [Online; accessed 24-May-2023].

[wit] *wit.ai - Build Natural Language Experiences*, https://wit.ai, [Online; accessed 24-May-2023].

[YLS+21] Difeng Yu, Xueshi Lu, Rongkai Shi, Hai-Ning Liang, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves, *Gaze-supported 3d object manipulation in virtual reality*, Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (New York, NY, USA), CHI '21, Association for Computing Machinery, 2021.

[ZMG05] Xin Zeng, Qasim Mehdi, and Norman Gough, *3d scene creation using story-based descriptions*, University of Wolverhampton, School of Computing and Information Technology, 2005, Developments in the creation of 3D virtual environments have made this a popular approach in a variety of applications. However, creating 3D scenes still requires professionals to spend an enormous amount of effort and time on it. In this paper, we present a fully implemented prototype system called 3DSV (3D Story Visualiser) that generates a virtual scene by using simplified story-based descriptions. This prototype system is based on the integration of advances in computer graphics and text-tovisualization technology to generate 3D virtual environment. This makes it easier and faster for non-professionals to create 3D environments compared to using traditional 3D design packages. Online; accessed 24-May-2023.