SmashIT VR

Final Project Documentation

Kevin Li (kl3285),

David Rios (dar2209),

Tina Tsai (tyt2111),

Kei Asano (ka2836)

COMS 4172

Professor Steven Feiner

May 7, 2024

Table of Contents

Introduction **2**
Application Design and Architecture **2**
Scene Design 5
Level 0: Tutorial 5
Level 1: Basic Room 5
Level 2: Countdown 6
Level 3: Moon 7
Level 4: SMASH 8
3D UI Interactions **9**
Selection 9
Manipulation 10
Travel 11
Wayfinding 12
UI Interaction 13
Breaking Mechanism **14**
Type 1: Replacement (Levels 0, 1, 2, 3) 14
Type 2: Detachment (Level 3) 15
Type 3: New Mesh (Level 4) 16

Introduction

SmashIT VR is a VR-based application that provides a safe and immersive alternative to rage rooms, which can be costly, dangerous, and have restricted access to specific facilities and locations. By simply putting on a VR headset, users are able to immerse themselves in a virtual environment where they can engage in a similar adrenaline-filled experience to relieve stress but without the hazards. By using their controllers and moving their physical body to interact with the virtual environment, users can pick up and destroy virtual breakable objects. Additionally, various tools in the virtual environment can aid users in destroying breakable objects in a multitude of ways. This, coupled with the sound and haptic feedback, provide users with a fun and engaging experience.

Application Design and Architecture

SmashIT VR is composed of various Unity scenes. Upon starting the game, the user is directed to the Level Selector scene, where they can choose 1 of the five levels to play. Users interact with the floating user interface (UI) with their controllers. Raycast and buttons are used in this scene for the user to make a selection.

Figure 1: Level Selector scene with selection UI

The 5 levels and their corresponding objects that make up our application are as follows:

Level	**Objective**
Level 0: Tutorial	Teaches the player to interact with breakable objects in the environment and how to break them through throwing or using other tools. Players have to follow the tutorial properly to complete the level.
Level 1: Basic Room	Destroy all vases in the sandbox environment.
Level 2: Countdown*	Destroy all vases within the time limit.
Level 3: Moon*	Find and hit the gold Wall-E (inspired "Where's Wally?"). The level uses 1/6th of Earth's gravity parameter to simulate moon motion.
Level 4: SMASH	No objective. This level is designed to be an experimental level where players have a different hammer tool. The 'S,' 'm,' 'a,' 's', 'h,' 'l,' 'T,' letters only chip off when hit by greater forces, requiring faster and more forceful swings.

**These levels contain more complicated scenes and are best run using Quest Link or on a VR headset with more processing power.*

Once a user has selected a level, the corresponding Unity scene is loaded. The user can start the game and complete the objective, which is always displayed as a view-anchored UI element. Once the objective has been met (for levels 0-4) another view-anchored UI window appears, prompting the user to either move to the next level, retry the level, or return to the main menu (i.e., the level selector). For level 2, if the countdown reaches 0 and not all objects have been destroyed, a similar window will appear but only offer the user the option to retry or return to the main menu. During gameplay, the user will also be able to pause the game using a single button press. A pause menu provides the option to resume, return to the main menu, or quit the application entirely.

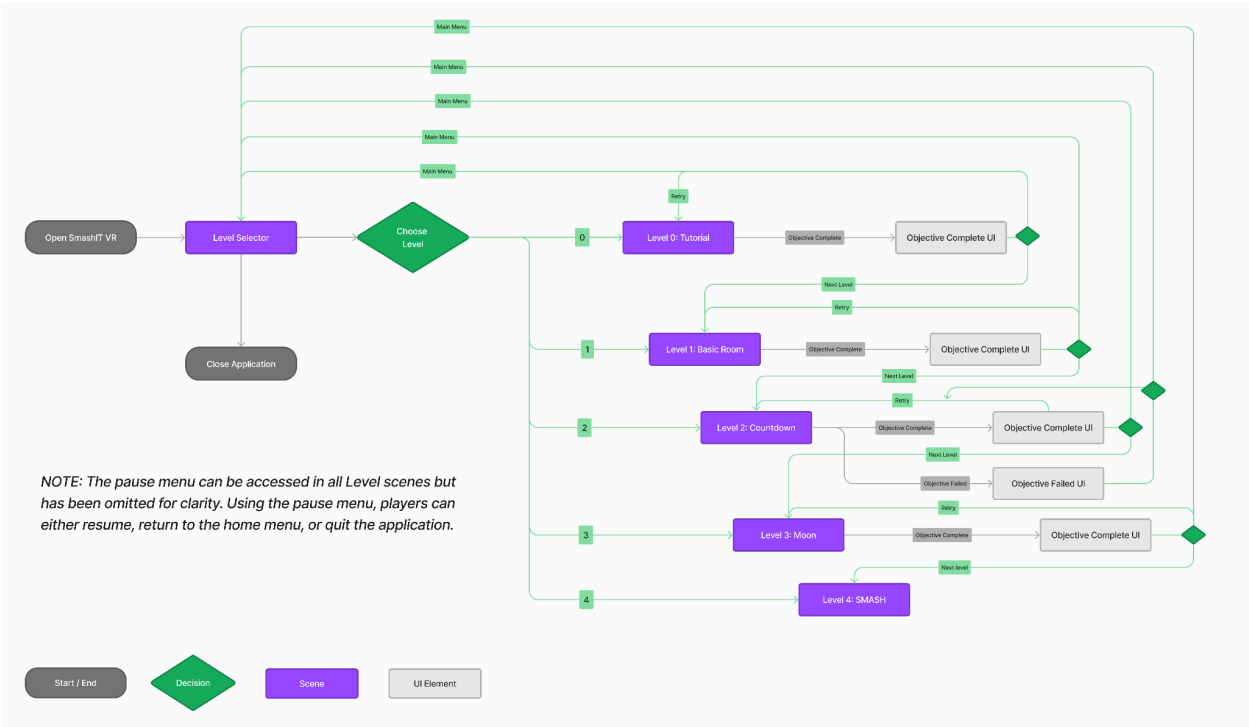

Figure 2: SmashIT VR User Flow Diagram

Scene Design

Level 0: Tutorial

The tutorial was designed to teach users how to 1) pick up and throw a breakable object, and 2) use another object to break another object. This level thus has two parts, one for each of the above. The red-brick room has a single scroll gameobject placed on a wall. This environmental object serves as a wayfinding method, allowing the user to orient themselves in space and identify a ‘north’ direction. Additionally, the scroll responds to interactions. Before the user picks up the bat in the first part of the level, the scroll prompts the user to pick it up. After the bat has been picked up, the scroll’s text changes and it prompts the user to smash the vase. The scroll, in addition to the world UI elements with clear instructions, helps the user understand how to proceed the level. This adheres to Nielsen Jakob’s first Usability Heuristic: Visibility of System status. The environment responds to user actions and helps the user identify the next intended action.

Additional game logic has been implemented to ensure that the player has to pick up the bat and use it to destroy the vase in the second part of the tutorial. This is achieved by ensuring that the second vase only appears when the player is holding the bat.

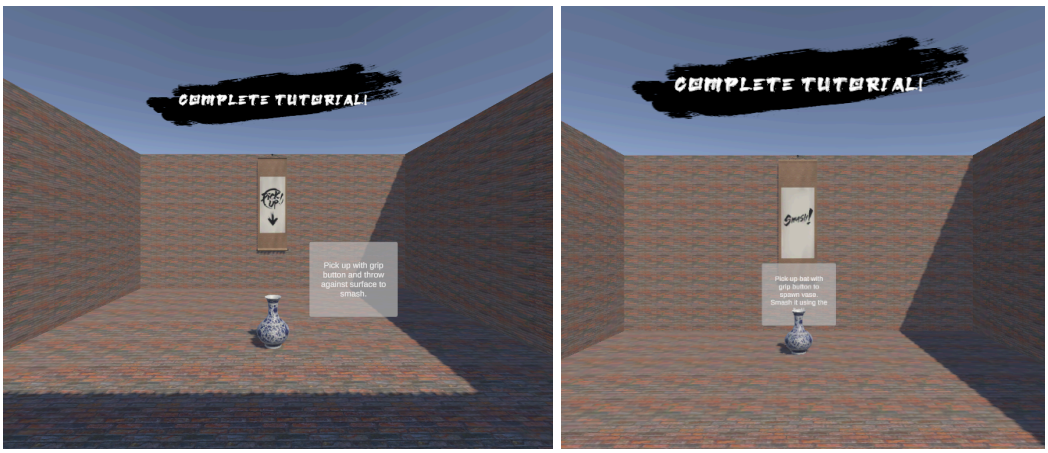

Figure 3: Tutorial scene images demonstrating the responding environmental objects and overlaying UI instructions; the scroll prompts the user to pick up the vase in part 1 (left) and smash a vase in part 2 (right).

Level 1: Basic Room

This room contains more vases for the user to familiarize themselves and practice interaction techniques learned in Level 0. The scroll is also present in this scene and allows the user to orient themselves in space (by providing a constant ‘North’ direction).

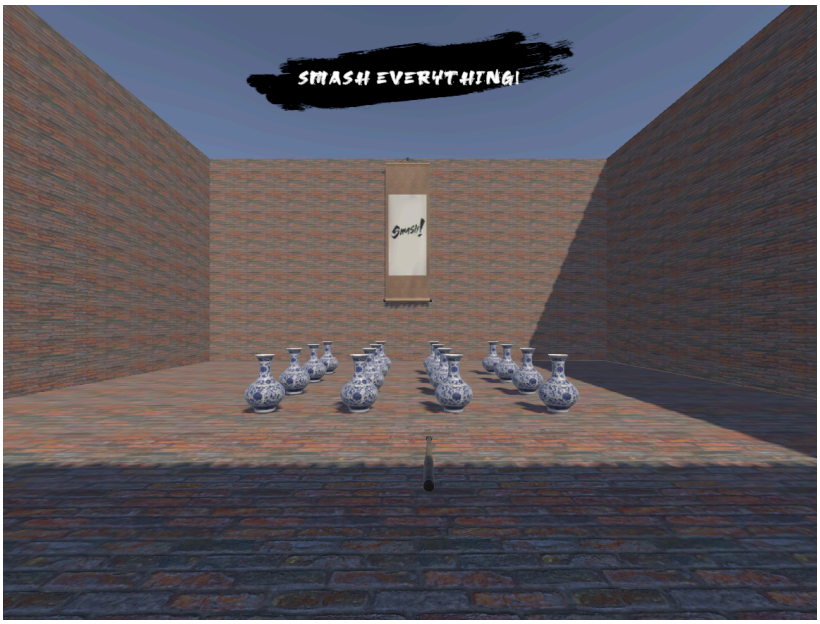

Figure 4: Basic room setup with vases lined up. A bat is also available for the player to use.

Level 2: Countdown

This room extends on Level 1. More vases are placed in the virtual environment. The fixed-scroll now serves as a countdown clock. The user, while quickly smashing objects, will be able to refer to the scroll easily to track remaining time.

Figure 5: Scroll in the background indicates the remaining time to the player.

Level 3: Moon

This level departs from all previous levels both in terms of environment and physics. The player needs to find the gold Wall-E and hit it to complete the level. Initially, a ghost gold wall-E appears, clearly communicating the objective and building the initial atmosphere. To support the user in finding the real gold Wall-E successfully, we have introduced a wayfinding tool: A minimap. The map is fixed to the right controller. The user can refer to the map to identify their location in space, as well as previously traversed parts of the terrain.

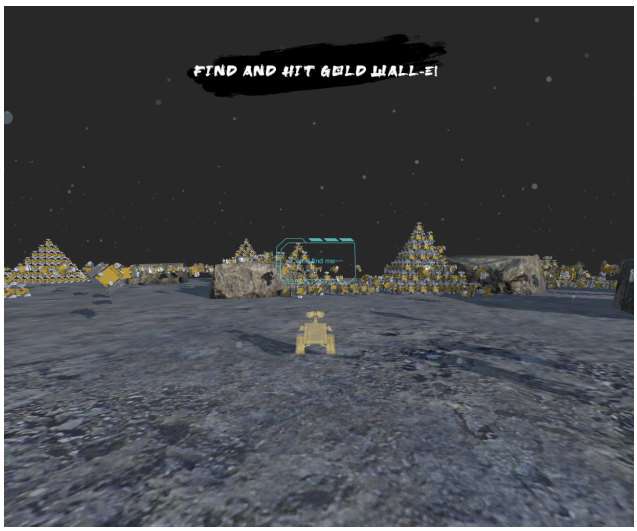

Figure 6: Ghost gold wall-E will prompt the user to follow it before disappearing.The player then starts their adventure of finding the real gold Wall-E by smashing normal ones around.

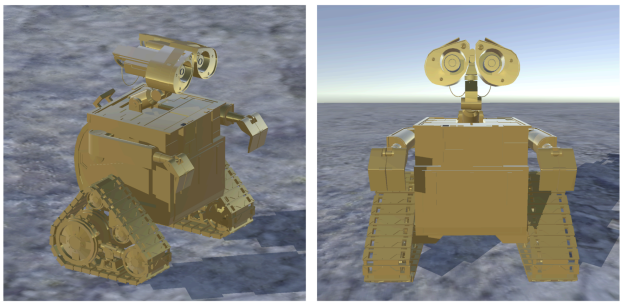

Figure 6.1: The gold Wall-E.

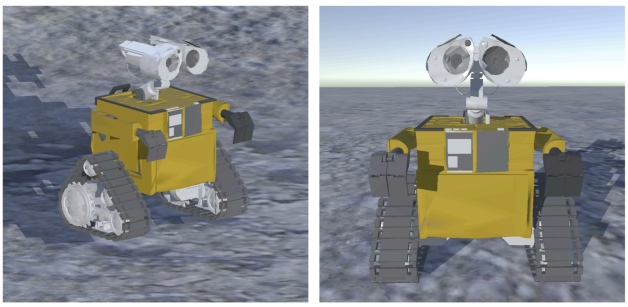

Figure 6.2: Normal Wall-E.

Level 4: SMASH

The final level does not have an objective and serves as a room where users can continuously smash the letter objects to their liking. It is the ultimate stress relieving level. What is special, however, is that the letters only chip away and respond to greater forces, requiring the user to swing their controller harder. This level utilizes a different kind of logic for the breaking mechanism, more of which is detailed later in this paper.

Figure 7: Dark environment where there is no game objective. Players can smash for as long as they desire.

3D UI Interactions

Both of the controllers can be used to navigate the application and interact with scene objects.

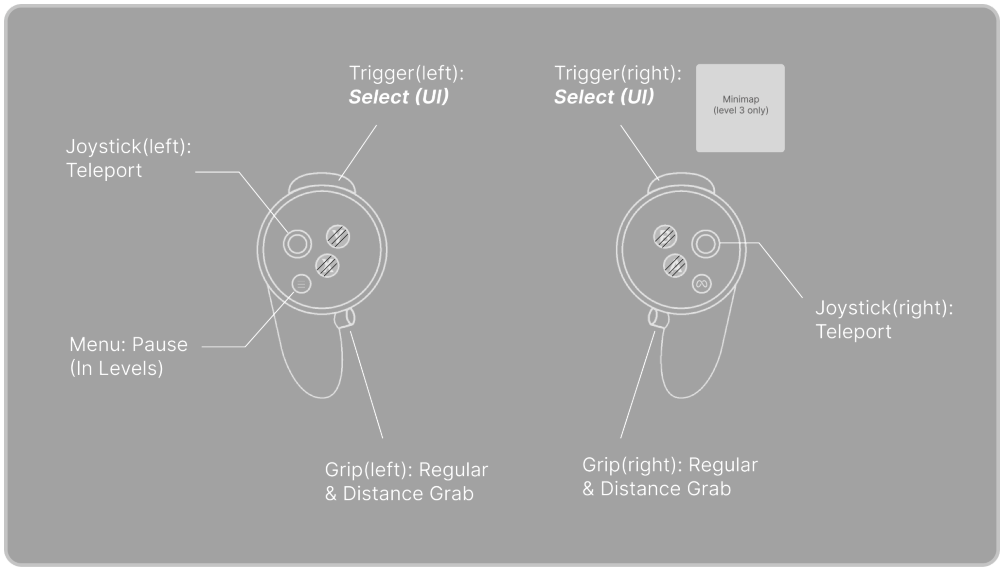

Figure 8: Controller button bindings

Selection

Figure 9: Object Selection (Left: before selection, Right: after selection)

Users can grab an object by pointing at the object using the controller and subsequently pressing the grip button for a distance grab operation (Figure 9). Users can also normally grab an object by holding the controller next to the virtual object and pulling the grip button.

Manipulation

Users can break an object in several ways: Breaking with a bat (Figure 10), breaking with another breakable object (Figure 10), and by throwing (Figure 11).

Figure 10: Breaking a vase using a bat (Left: before breaking, Right: after breaking)

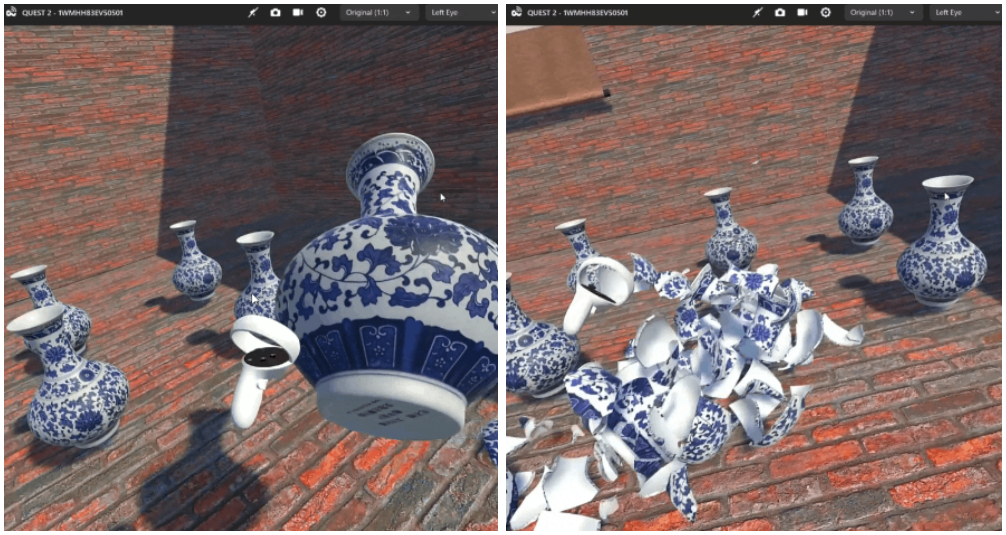

Figure 11: Breaking a vase with another vase (Left: before breaking, Right:after breaking)

Figure 12: Breaking a vase by throwing (Left: before breaking, Right:after breaking)

Travel

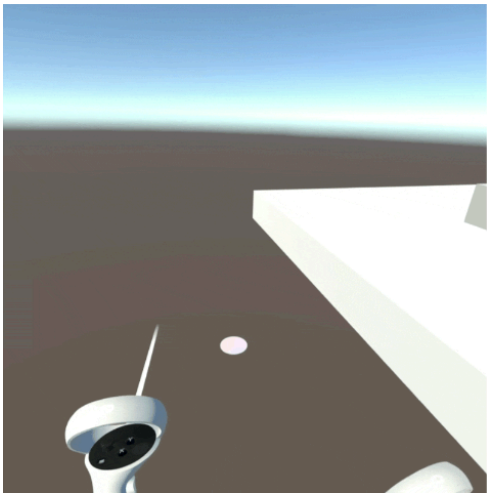

Figure 13: User teleporting into a different position

To change positions, users can either 1. Physically move their body or 2. Teleport to a specified location using the controller while pressing the joystick forward.

Wayfinding

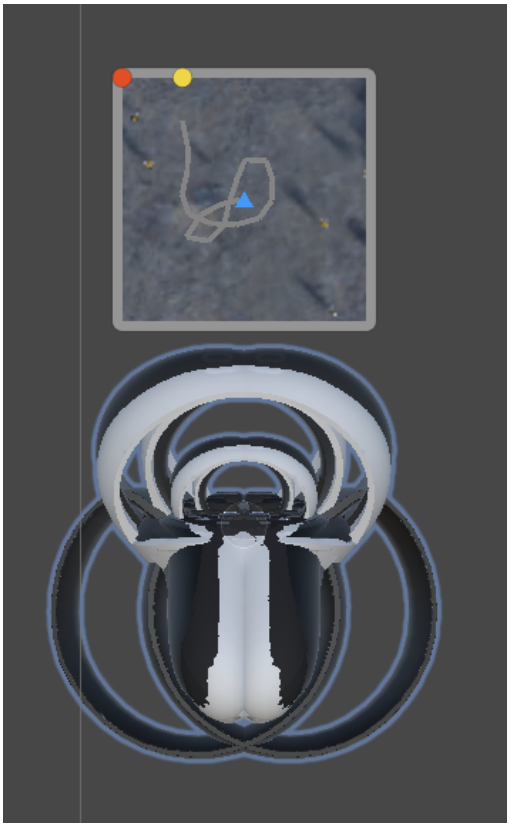

Figure 14: Minimap on right controller; blue triangle indicates player position, red circle indicates next objective, and yellow circle indicates the position of the ghost Wall-E.

The minimap is intended to guide the user through the objectives in the moon level (Level 3), helping them navigate to the next objective (potential Wall-E location) while keeping track of their elapsed path to help them maintain their orientation in the virtual world. The minimap is anchored to the user's right controller, inspired by the accessibility of accessories like wristwatches. The displayed map rotates with the user, while the icon denoting the user is triangular, always pointing North regardless of the user's orientation.

There are three different types of icons displayed on the minimap: a blue triangle (player), a red circle (next objective), and a yellow circle (ghost Wall-E). The blue triangle denoting the player is always centered on the minimap. The red circle shows the user the location of the next closest pyramid of Wall-Es, which often is the next physical object they should search in hopes of finding the golden Wall-E and winning the level. The yellow circle is meant to give a temporary visual indicator of the ghost Wall-E, and disappears along with it.

As the player navigates through the world, a gray path is drawn on the minimap, showing the user the path they have taken to help them orient themselves in the world, aiding them to navigate to the next objective without retreading their steps.

UI Interaction

During gameplay, the pause menu can be toggled by pressing the ‘Start’ (menu) button on the left controller. Users can use the pointer emitted from the controller and press the trigger button to activate a button. The user interface panel moves according to the head orientation/position, ensuring that the panel always appears in front of the user. Players can also interact with the win and loss screens in the same way.

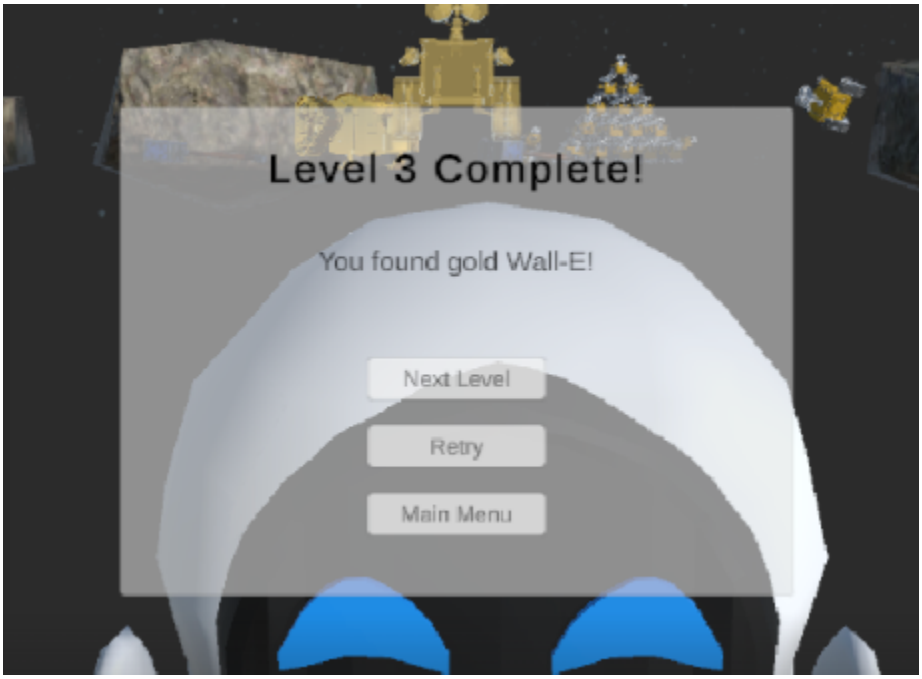

Figure 15: User Interface after the user completes the level

Breaking Mechanism

In this game, 3 types of breaking mechanisms have been incorporated into the 5 levels. The mechanics are used hybridly to achieve more variation and optimized runtime for the gameplay.

Type 1: Replacement (Levels 0, 1, 2, 3)

This is the mechanism used in most scenes, where the intact model is swapped with a pre-broken model directly when an on-hit event is triggered. The pre-broken model can be customized to its broken pieces in other modeling software like Blender and Rhino3D to achieve different piece counts and shapes (Figure 15).

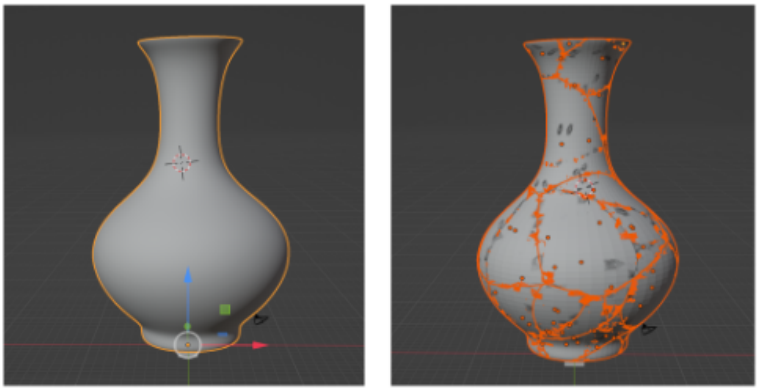

Figure 16: Vase model is "cell fractured" in blender.

The replacement is also used with the Detachment type shown below. By creating a lower mesh unbreakable (non-interactable) version from a complex model reversely, the lower mesh version is initially used in the scene and swapped to the complex one when hit. This can be used with objects that need to be broken with stronger hitting force, such as machines with metal material. Thus, when the player first hits it, it isn't too odd that the object didn't break. When the player hits it the second time, it starts to be "breakable" since it has changed to a complex interactable one. (Figure 16)

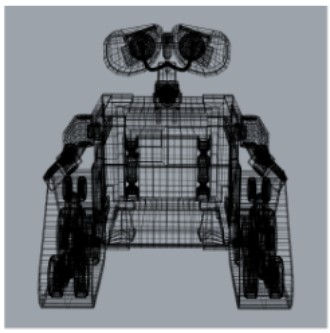

Figure 17: Complex meshes are joined in external modeling and could also be joined by using Unity ProBuilder or Mesh Reduce API.

Type 2: Detachment (Level 3)

The detachment type used in the model is handled by following its physical structure (Figure 17). The model consists of small parts and components as a whole, while the detachment happens depending on how and which part of the model the player hits.

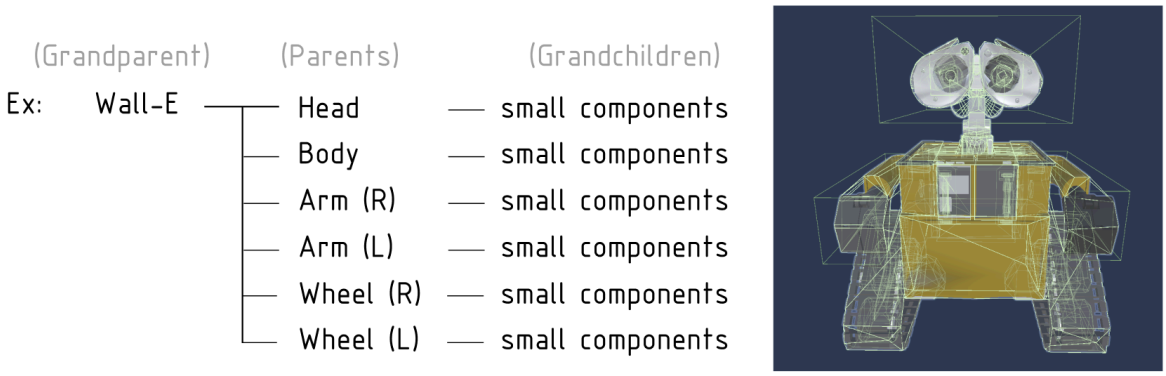

Figure 18: Wall-E is structurally organized into 3 hierarchy.

The parts and components outside of the range would not be affected unless the "Body" (tagged) was being hit; then, these unaffected parts would be detached from the grandparent (main) as a group separately (Figure 18).

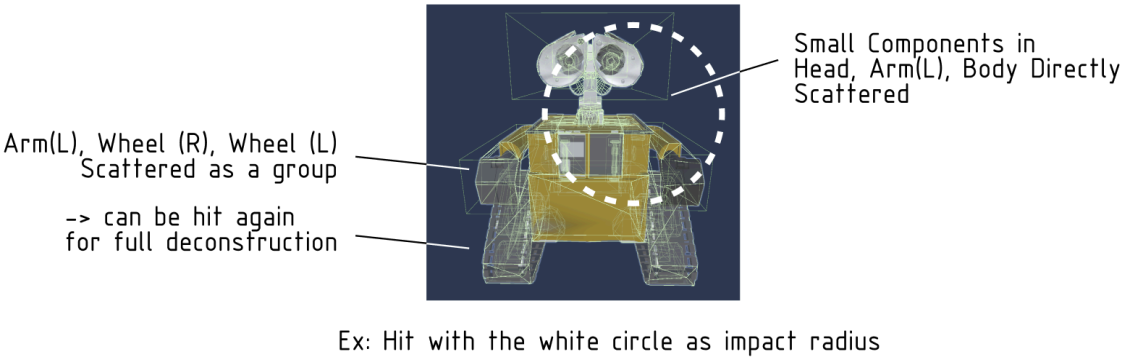

Figure 19: Wall-E's detachment example with the impact in the white circle.

This approach requires more careful settings on Rigidbody and Collider settings in Unity, especially when integrating the interaction methods with VR controllers with the Kinematics, Provide Contact, and Is Trigger settings. The impact radius and threshold force minimum also affect the success of the ideal detachment as well. In some tests, if the parameters didn't match, the unaffected parts would not be detached where they stay at the same location, which could leave an odd bug in the scene.

Type 3: New Mesh (Level 4)

The third mechanism is applied in Level 4. The mechanism involves creating new meshes of the broken objects during runtime. The objects break apart into smaller pieces in a dynamic and visually interesting way. By dividing the mesh and randomly generating a plane that passes through it, newly created sub-meshes represent one of the halves of the original mesh as it is cut by the plane. Once the mesh is fully divided, each piece is turned into a separate game object with its own physics properties. (Figure 19) These new objects are also pushed away from the center of the original object, creating an explosion effect. The logic with more geometric mathematics to handle the meshes can be found in the reference script.[1]

Figure 20: The object gets hit and divided into multiple pieces, with new meshes created.

To limit the number of new meshes, the threshold force minimum is set to be significantly larger than other levels so that the player can still keep smashing but decrease the possibility of heavy runtime loading.

[1] https://gist.github.com/ditzel/73f4d1c9028cc3477bb921974f84ed56