

* **Eeshaan Sourabh Reddy**
* **Jaime Meyer Beilis Michel**

**Introduction**

This report explores the integration of Unity and Touchdesigner to create interactive and immersive environments for virtual reality (VR) and physical spaces. By utilizing tools such as KlakSpout for video streaming and Augmenta for simulation, we aim to develop engaging experiences in various planetary environments.

**Given Timeline:**

1. **August 15**: Submit first draft of the show.
2. **October 1**: Complete all preparations and finalize the project.
3. **November**: Launch exhibition and physical opening.

**Note**: All work must be finished 40 days before the exhibition opening.

**Expected timeline:**

1. August 15: Submission of a storyline, some assets utilized on the game
2. October 1: Completion of the first draft of the project, interactivity
3. Dependent on the in physical opening or VR game distribution, the November

deadline would change the way that model mappings would be implemented

1. December: Full implementation of post processing, shaders, and musical scoring technologies.

**Planets to Explore (Example/Can add or modify)**

1. **Ice World**
   * **Description**: A cold, icy landscape with glacial structures and snow-covered terrain.
   * **Features**: Reflective surfaces, dynamic snow effects, and icy caverns.
2. **Hot World**
   * **Description**: A fiery, volcanic environment characterized by lava flows and high temperatures.
   * **Features**: Lava animations, heat distortion effects, and active volcanoes.
3. **Water World**
   * **Description**: An aquatic planet with extensive oceans, coral reefs, and underwater landscapes.
   * **Features**: Water simulations, underwater effects, and marine life.

**Integration of Unity and Touchdesigner**

**Unity**: Chosen for its robust 3D capabilities and support for VR. Unity allows for the creation of detailed, interactive environments with depth and interactivity.

**Touchdesigner**: A visual programming environment ideal for creating real-time interactive visuals and effects.

**KlakSpout**: A Unity plugin that facilitates the sending and receiving of video streams between Unity and Touchdesigner. It allows real-time interaction and data exchange between the two platforms.

* **Link to KlakSpout**: https://github.com/keijiro/KlakSpout

**A screenshot of a computer

Description automatically generated**

**Objectives**

1. **Spot to Render Material on Object in Unity**
   * **Goal**: To test and visualize how materials can be dynamically applied to objects within Unity, leveraging live video streams from Touchdesigner.
2. **Receive Live Camera Feed from Unity to Touchdesigner**
   * **Goal**: To use live video data from Unity to influence and modify interactive visuals and effects in Touchdesigner.

**Optional Method for Particle System:**

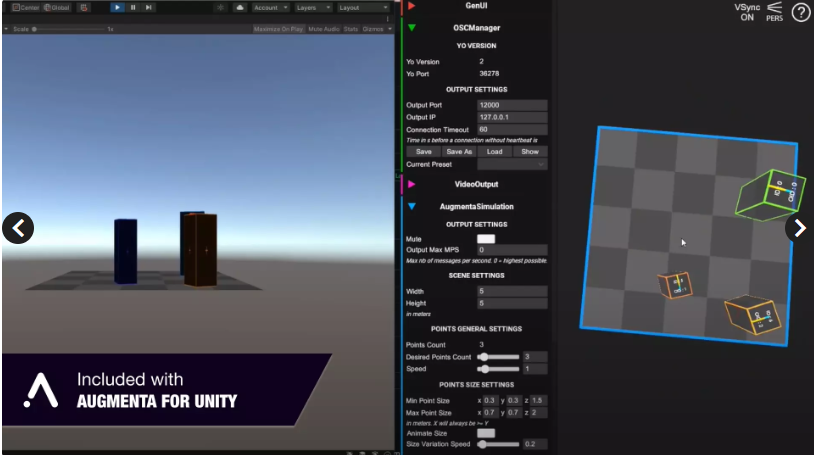
**Touchdesigner**: Export particle data from Touchdesigner and import it into Unity to recreate the particle system.

1. **Export Data**: In Touchdesigner, export the particle positions, velocities, and other relevant data to a suitable format, such as CSV, JSON, or custom binary format.
2. **Import in Unity**: Write a script in Unity to read this data and create a particle system using Unity's Particle System component.
3. **Recreate the Effect**: Use the imported data to position particles and set their properties like size, Color, and velocity.

**Augmenta Integration**

**Augmenta** is a spatial tracking and simulation tool that provides real-time tracking and interaction capabilities within physical and virtual environments. It enhances the interactive experience by mapping and understanding physical movements and translating them into the virtual space.

1. **Simulation Tools**:
   * **Functionality**: Augmenta offers tools for simulating various physical interactions and environmental effects within a virtual space.
   * **Application**: These tools can be used to model and simulate realistic interactions, such as object manipulation, environmental changes, and user behavior, within Unity.

****

**Role in the Project:**

* **Enhancing Interaction**
* **Simulate Virtual Movement**
* **Testing and Optimization**

**Tests and Evaluation**

**Test 1: Spout from Touchdesigner to Unity 3D (VR/Physical Space)**

* + **Objective**: To establish a connection where Touchdesigner’s visual outputs are integrated into Unity’s 3D environment, allowing for real-time visual feedback in both VR and physical spaces.

**Test 2: Using Augmenta (Physical Space)**

* + **Objective**: To incorporate Augmenta's spatial tracking capabilities into a physical space setup, exploring how physical movements and interactions can be mapped and used within the Unity environment.

**Test 3: Simulate Movement (VR)**

* + **Objective**: To simulate and test user movement within a VR environment, analyzing how well the Unity and Touchdesigner integration supports immersive interactions.

**Test 4: Test Augmenta with Unity Environment (VR)**

* + **Objective**: To integrate Augmenta's simulation tools with Unity, evaluating how effectively Augmenta’s data can be used to enhance virtual environments and interactions.

**Test 5: Use the Interactive Environment to Touchdesigner to Add Effects (VR/Physical Space)**

* + **Objective**: To leverage Unity’s interactive environments to send data back to TouchDesigner, enabling the addition of real-time effects based on user interactions within Unity.

**Test 6:** Model instantiation and modern rendering techniques:

* + Objective: Exploring resource management options in the Unity Game engine to bring together a minimal use of draw calls / stored meshes.

As an important step in optimization, we have discussed the utilization of the least possible meshes with the most possible uses of instantiation, leading us to limit to the least draw calls needed to create a good in-game environment.

**Test 7:** Using spatial information from the user to generate parameters implemented on synthesizers – audio technologies

* **Objective:** Making use of streams and buffer data types in C to deliver information to VST. A good contender for this would be Surge XT VST, which allows modification of its open source. This is still undetermined as the array of possible technologies is also dependent on the creative outlook for the scoring of the experience.

**Test 8:** Utilizing API calls on TouchDesigner to generate parameters implemented on synthesizers – audio technologies

* + Objective: TouchDesigner offers extensive mathematics, and sound library options that could perform easy computations and arrangement of both visual and numerical inputs into sound parameters.

For this proposition, there is the possibility to generate sounds from analog synthesizers by using TouchDesigner as a trigger, but this is dependent on either having a physical opening or a distribution of the software; while, for a fully software based solution, API calls can be stored and read from a hash table that matches the hashing of a customized LFO range, or read linearly, making calls from the API on specific time frames based on the progression of the synthesizers release, or the tempo of the music.

The following is a basic graph showing how this can be achieved.

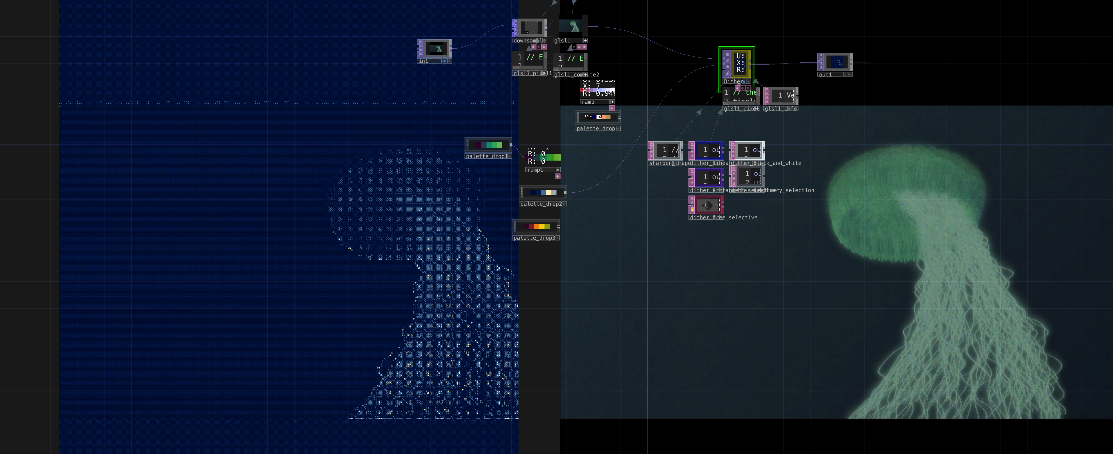


**Test 9:** Utilization of TouchDesigner as a Post-Processing and shaders tool, through Spouting

* Objective: Due to the possible implementation of AI technologies, and the many stemming computations that refer to spatial tracking done in VR, making a game with stylization would reduce the necessity for complex meshes and light simulations that a realistic rendering would otherwise require. For this purpose, the spouting to touchdesigner could be used for innumerable types of stylizations such as dithering, colour quantization to give it a retro look, or adding filters, blooming, colour correction, normal mapping, all of which, would give a solid style to the game and reduce the necessity for expensive computations.

This is specifically important if the software is meant to be distributed, but it also offers advantages in making a creative and unique look for the game.

The following is an example of stylization in touchdesigner, where a jellyfish animation is being passed to a sharpening, dithering and colour quantization filter live in touchdesigner with a GPU cook time close to 0 milliseconds, while this is not necessarily the style we need to strive for, this is an example of the possible advantages of Spouting.



**Test 10:** Utilization of TouchDesigner TOPs for computing terrain / spatial information.

* + Objective: Samples from the Noise TOP, and the capabilities of transformations and translations of it in TouchDesigner could be implemented to form terrain and user position information with simple operations. And a vast control of simulation basics such as collision.

**Unity 3D Env:**

* + - 1. Added 3d Environment
      2. A screenshot of a computer generated image of a snowy mountain

         Description automatically generatedSnow covered hills and blue sky

         Description automatically generatedAdded Background particle system

IceWorld\_ENV

A landscape of a mountain

Description automatically generated with medium confidenceA mountain with lava on it

Description automatically generatedFireWorld\_ENV

**Assets usage Licence:**

Standard Unity Asset Store EULA (End User License Agreement).

* **Permitted use**: You can use the asset in any project, including public projects.
* **License scope**: The license is granted per-seat, meaning it's tied to the Unity account used to purchase the asset.
* **Restrictions**: You cannot redistribute, resell, or sublicense the asset.
* **Ownership**: The asset creator retains ownership and intellectual property rights.

**Leap2 Hand Tracking:**

* 1. Using Hands into the environment
  2. Able to interact with objects
  3. The software contains scripts for hand controls

A white and orange objects on a black surface

Description automatically generatedA screenshot of a video game

Description automatically generated

**Transition Screen/Main Menu**

1. To select planet
2. Start Screen
3. Currently 2d, will be made 3d

A screenshot of a computer screen

Description automatically generated

**Optimization Tests:**

A screenshot of a computer

Description automatically generated

Optimizing for VR sets:

1. **Single-Pass Stereo Rendering**: ender both eyes simultaneously, reducing overhead.
2. **Occlusion Culling**: Use occlusion culling to hide objects outside the user's view.

**Final Environment:**

After carefully picking the assets needed for the project, I have built 4 scenes, MainMenu, FireWorld, WaterWorld, IceWorld. The player moves through these scenes by selecting the planets in the MainMenu (Start screen).

The development process involved the following key steps:

1. **Asset Selection and Scene Creation**:
   * Unity tools such as **ProBuilder** and **Polybrush** were utilized for terrain modeling and detailing.
2. **Ray Projection for Interaction**:
   * Using the Leap Motion SDK, we captured hand motion data and mapped it to a ray projection system.
   * The system was tested extensively to ensure accurate and intuitive selection of objects.
3. **Music Synchronization via OSC**:
   * OSC communication was configured between Unity and TouchDesigner to ensure that the correct music played immediately upon entering a scene.
   * Multiple test cases were run to confirm the reliability of OSC signals and smooth transitions.
4. **Integration Testing**:
   * We tested the entire system in VR to confirm that hand tracking, scene navigation, and music synchronization worked seamlessly.

**Music Integration with TouchDesigner**

One of the key features of the project is its dynamic music system, which is closely tied to scene transitions. Each scene has its own unique music that matches its theme:

* **FireWorld**: Intense and energetic music to reflect its fiery atmosphere.
* **WaterWorld**: Calming and peaceful music to suit the underwater vibe.
* **IceWorld**: Cool, crisp tones to complement the icy environment.

To make this work, we used **Open Sound Control (OSC)**, a protocol that lets Unity and TouchDesigner communicate with each other in real time. Here’s how it works:

1. When the player selects a scene, Unity sends a message to TouchDesigner through OSC.
2. TouchDesigner receives this message and starts playing the music track for the selected scene.
3. The music continues playing as the player explores the scene, enhancing the overall experience.

This system ensures smooth transitions between visuals and music, creating a more immersive and engaging environment for the player.

**The below link has all the video demos and final product.**

Box Link: https://arizona.app.box.com/s/3l2vt3l11qaok47o1bdyps3a6bw6vafb

**Failed Attempts**

There were few systems we could not build:

**Particle System:** Implementing particle system in unity using Touchdesigner was not possible due to not finding the right program to do it. We tried many ways to apply particles system by sending the image into unity, but it looked more 2D and did not match the environment. Designing a particle system manually would take a lot of time to design.

Moving forward, several enhancements can be considered to refine and expand the project:

1. **Adaptive Soundscapes**: Introduce real-time music generation based on the player's actions or position within the scene.
2. **Physics-Based Interaction**: Add more interactive elements within each scene, such as objects that react to hand gestures.
3. **Expanded Scene Library**: Build additional themed worlds, each with unique environmental and auditory features.
4. **Performance Optimization**: Optimize hand tracking and OSC communication for smoother performance in more complex scenes.

**Ishowcase**

We presented our project in iShowcse event on 11th December, the project received many praises and everyone loved the designs and music.

A person sitting on a table with a computer

Description automatically generated