Manual

VR Equipment and Laptop Setup

The following outlines the setup of the specific use-case of running the project on the University-provided laptop; the project can run on any VR-capable PC, but the physical setup will vary by platform.

HP Reverb G2 VR Headset Setup:

- Connect the power cable to a wall outlet, after ensuring the power output cable from the hub is connected to the power adapter.
- Connect the input cable (*DisplayPort*) to the port on the headset behind the mounted magnetic gasket (which must be removed to connect, then reattached afterward).
- Connect the display cable output (*DisplayPort*) to the *DisplayPort-to-USBC* adapter, then connect this to the laptop's *USBC* port.
- Connect the data cable (*USBC*) to the *USBC-to-USBA* adapter, then connect this to the laptop's *USBA* port.
- Press the button on the square hub into which all cables are routed to power on the headset.
- Ensure the controllers are charged (*AA* batteries can be accessed through sliding down on the panel near the thumb grip area). Turn them on through holding down the **Windows** button underneath the analogue sticks.
- Launch the **Windows Mixed Reality** application on the computer; you are ready when the application's default scenery renders to the headset, and you can see both controllers tracked to where you physically hold them.

System Requirements

ML Pipeline Requirements

- You must have approximately **200GB** of free storage
- You must have an active internet connection to run Monocular Depth Estimation
- You must have a system with an `NVIDIA` graphics card due to the *CUDA* dependency
- All the individual components of the pipeline require different *Anaconda* environments to run, except for *360MonoDepth* which uses a Docker container. The environment requirements for the different modules can be found and set up using the instructions provided on their repositories:
    - *360MonoDepth* - https://github.com/manurare/360monodepth

    - *DBAT* - https://github.com/heng-yuwen/Dynamic-Backward-Attention-Transformer/tree/main
    - *EdgeNet 360* - https://github.com/SimonLisowski/Mesh-Generation
      Note: The Docker container should be set up using the modified 360MonoDepth code on the main project repository of this due to the modifications made to make it compatible with the pipeline.
- While not a requirement, it is recommended to run this on a newer machine with an SSD

Unity Requirements

- You must run the program using the *Unity Hub* (into which the *Unity* project folder: `Project/AV-VR/Unity/Project/AVVR` is imported as a "project on disk"), which must have installed the *Unity Editor* version **2022.3.11f1**
- An NVIDIA graphics card is required to enable *Steam Audio* functionality
- A VR headset is required to output the scene's render: the *HP Reverb G2* is recommended, but the project is designed to run platform-independently
- The VR environment of the headset must be running (i.e. *Windows Mixed Reality*, *SteamVR*, etc.) before pressing play when the scene is loaded

Generating a scene

Generating a scene is an effortless task. This section provides guidance for running the pipeline for 3D scene reconstruction for VR from a single omnidirectional input image. The whole pipeline includes namely depth estimation using *360MonoDepth*, Material Recognition using *Dynamic Backward Attention Transformer*, Semantic scene completion and Material mapping using *EdgeNet360*, with a separate final import of the scene in *Unity* which is outlined in the next section of this manual.

The pipeline provides three distinct ways to run the pipeline:

1. *Using any RGB image with our* Monocular Depth Estimation *module*
2. *Using any RGB image and its ground truth depth maps from the* Stanford2D3D Dataset Area 3
3. *Using any RGB image and its corresponding depth map (already produced using stereo matching)*

To run the pipeline, the `GUI.py` script provided in the scripts folder should be run. The User Interface provides the user with the choice to run the pipeline using the options outlined above. This is demonstrated in *Figure 1*.

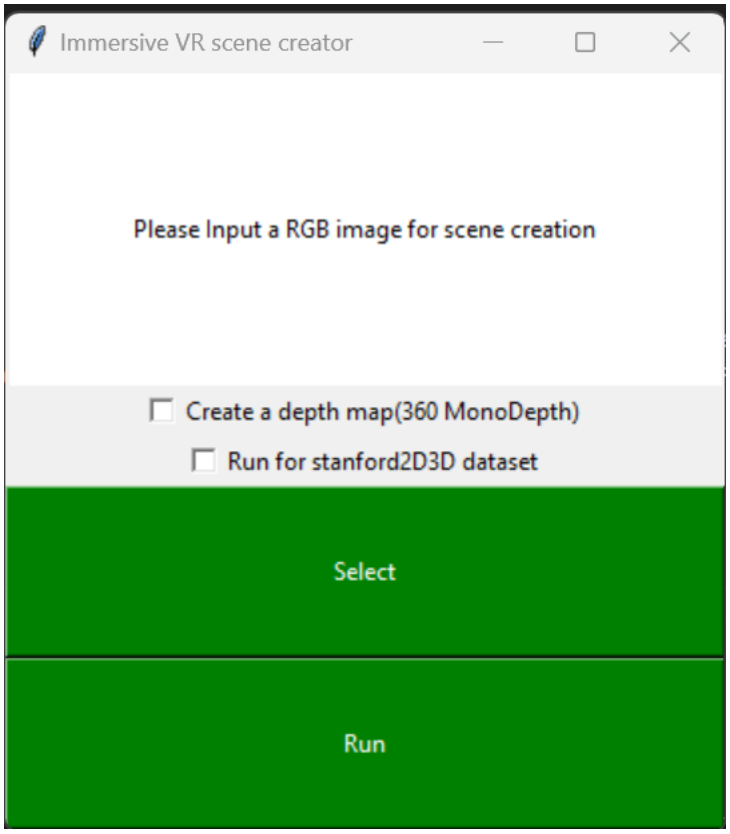


Figure 1 GUI providing the options to run the 3d reconstruction pipeline for VR

The scripts folder contains all the necessary scripts and python files used for automating the pipeline.

Running with the Monocular Depth Estimation Module

Running this component is simple. As a pre-requisite, when 360MonoDepth has been set up and built with Docker initially, you must manually edit all the .ps1 scripts except for masterscript.ps1 and ensure that the $imageNameOrId variable is set to the image ID (and not container ID) of the associated 360MonoDepth Docker image (can be retrieved using the Docker Desktop application). This step is only required at the very initial set up stage.

To run the pipeline with our Monocular Depth Estimation module, run `GUI.py` and make sure the **"create a depth map"** option is checked.

Clicking the green **Select** button will then provide the user with a dialogue prompt to navigate to and select the RGB image on which they wish to apply Monocular Depth Estimation.

Clicking the **Run** button next will run the Machine Learning Pipeline without the need for the user to manually intervene. The main part of the User Interface displays the status of the pipeline, whereas, the command line displays exactly what is being executed.

The output depth map is saved in the `Data/Input` folder within the *EdgeNet360* directory ready to be passed into that module. The *`splt_img.py`* file runs and splits the RGB image

and saves the cube face images in the `material_recognition/Dynamic-Backward-Attention-Transformer/split_outputs` folder.

The output material recognised cube face images from *DBAT* are saved in the `material_recognition/Dynamic-Backward-Attention Transformer/output/split_output"` folder. The omnidirectional material map image is then saved in the `Input` folder of `Edgenet360`.

Check the command prompt whilst the scene is creating. The command prompt displays the material assigned to each objects for verification, as seen in *Figure 2*. Once the command prompt displays the line seen in *Figure 3*, the pipeline execution is finished, press any key. *EdgeNet360* creates the necessary output `.obj` and `.mtl` files for the voxel grid and the mesh saving them in the `edgenet360/Output` folder.

Restart the widget to create another scene, to rerun the pipeline.

Once the mesh has been created, please refer to the next section for the procedure of generating and rendering an acoustically-modelled VR scene from it.

Running with Stanford2D-3D

To run the pipeline using the images from the *Stanford2D3D* dataset and their corresponding ground truth depth maps, firstly download the *Stanford2D3D* dataset and run the *Ground Truth Extraction* on the images in the **Stanford** folder as instructed in the *EdgeNet360* repository.

Then all you need to do is run the `GUI.py` python script and check the option: Run for *`stanford2D3D`* dataset and provide the input RGB image using the select image option, insert which area the image is in, as seen in *Figure 2* and select **Run**. The output files i.e. the material recognised images, cube face images and the `.obj` & `.mtl` files for the voxel grid and mesh will be saved in the same directories as explained before. The material image is saved in the appropriate folder by the scripting process so that it can be provided to *EdgeNet360* along with the depth map. The final mesh scene is saved in `edgenet360\output`.

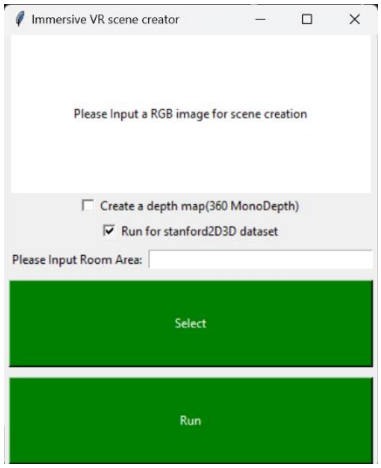
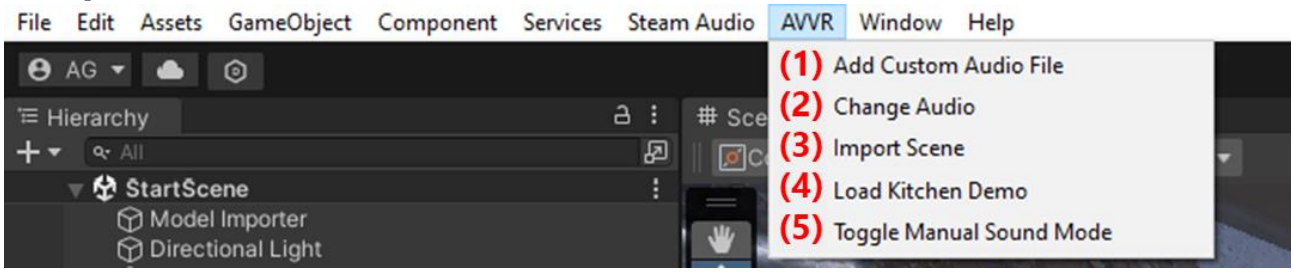


Figure 2: The GUI input for running the Stanford2D3D dataset. Insert Image and area of the Stanford dataset, e.g. area_3

Running a scene

Once a mesh has been created by the ML pipeline process outlined previously, it is time to render a Virtual Reality scene to experience the acoustic modelling that can be generated with it.

Setup



*The **AVVR** custom menu and its items.*

1. Run the **AVVR** project from the *Unity Hub* application to launch the Unity environment, this should launch into *"Edit"* mode.
2. Select the **AVVR > Import Scene (3)** context menu in the top bar of the Unity window to import the *.obj* file generated in the previous section. A file selector will open, starting in the default output folder of the ML pipeline; any .obj file on the PC can be selected, however. Once selected, the scene presented will dynamically change to allocate the new mesh.
3. By default, the audio that will play in the VR scene is a clip of some music played on a clarinet.
    - If desired, this can be changed by selecting the audio source in the project hierarchy (**AVVR > Change Audio (2)**, on the top context menu) and selecting the desired audio under the *"AudioClip"* field of the popup window. Custom audio files can be added to this list by using the **AVVR > Add Custom Audio File (1)** menu item.

    - The default behaviour of the sound can be changed; with the **AVVR > Toggle Manual Sound Mode (5)** menu item, this can be switched between playing automatically or manually with a controller button press (mapping listed in the following *Play Mode* section of this manual)
4. Ensure the Windows Mixed Reality application (or the corresponding environment for the chosen VR headset, such as the SteamVR application) is running, then press the play button at the top of the window; this will render the scene to the PC's connected headset.

Play Mode

Upon pressing play, the spatialised audio (with reverberations for the room applied to the mesh generated by the ML pipeline) will be playing throughout the scene, which you can navigate throughout to experience.

VR Controls

- **Left or Right Controller Trigger (Back Button):**
    - ***Teleport:*** A selection sphere will appear on the ground; release the trigger to instantly teleport your position to its location.
    - ***Move Sound Source:*** Point a controller at the Audio Source (a red sphere at the middle of the room), and press the trigger to pick it up. The source will stay in attached to your hand until you let go of the trigger, allowing you to move it to any position within the room.
- **Left Controller Analogue Stick:**
    - ***Smooth Movement:*** Moving the stick in any direction will allow yourself to move smoothly along the ground within the scene (relative to where you are facing).
- **Right Controller Analogue Stick:**
    - ***Quick (i.e. "Snap") Turn:*** As an alternative to physically orienting yourself in the direction you wish to face, you may also position the stick left or right to quickly turn yourself (facing 45 degrees to the left or right from your previous gaze, respectively).
- **Left or Right Controller Buttons (X/Y):**
    - ***Play Sound:*** With **AVVR > Toggle Manual Sound Mode (5)** switched on, this will execute the playback of the sound source.

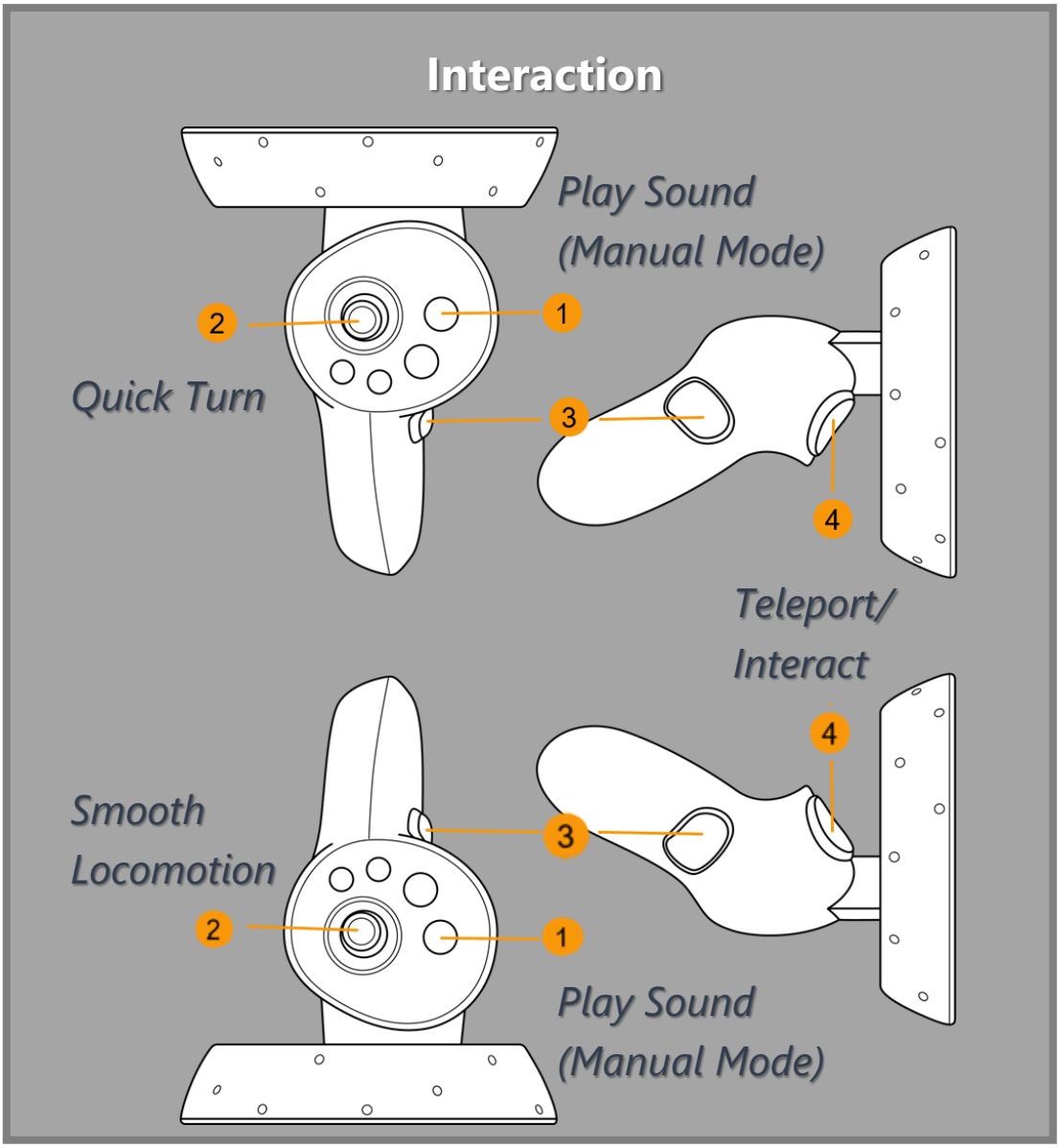
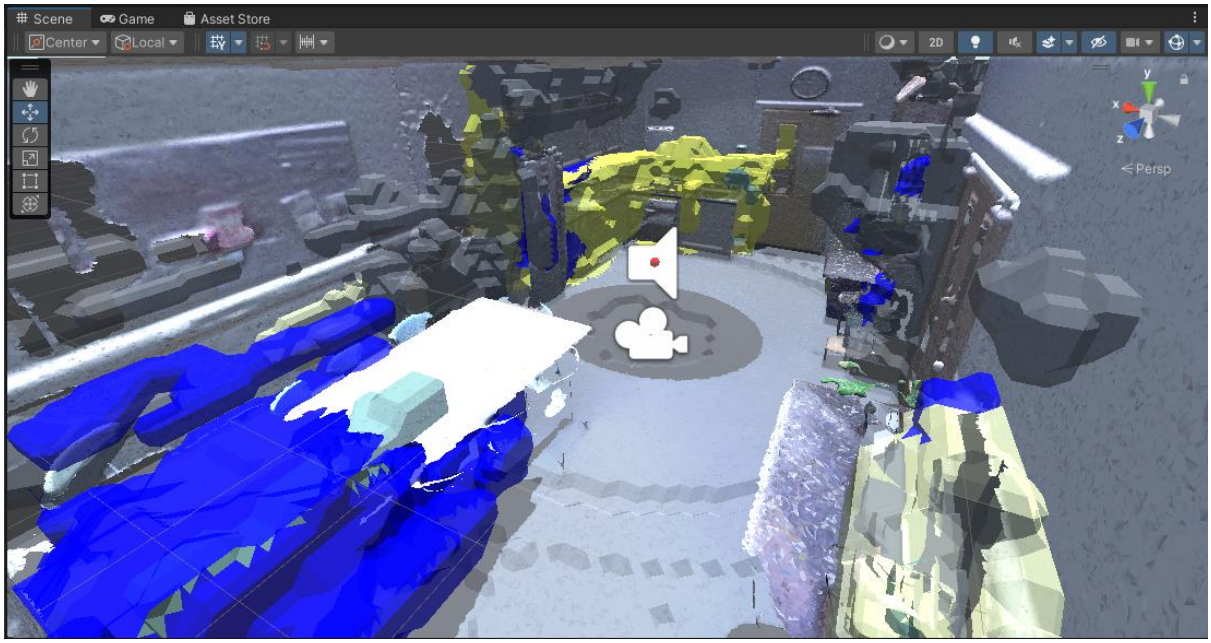


Diagram of controller inputs for the HP Reverb G2.

Demo Scene

Additional functionality exists under the **AVVR > Load Kitchen Demo (4)** menu item; this will load a pre-prepared scene generated of a kitchen scene, with the automatic overlay of a LIDAR scan of said kitchen with the pipeline output set to transparent. This scene is useful for helping demonstrate the real-life scene corresponding to the mesh our pipeline generates, and is more visually appealing than the scenes obtained from the default pipeline mesh import.



Unity Edit Mode view of the Demo Scene.

NB: A video demonstrating this part of the pipeline is present in the Project Archive, named "VR Demo.mp4".