Visualization workflow

The purpose of this job to develop tool to convert any kind of CFD simulation (not just limited to road transportation, but also appliable for water transports and air transports) from Ansys and Openfoam into a 3D visualization build inside a virtual reality environment using Unity3D engine. Software like Ansys can provide certain levels of post-processing and visualization, and simulation tools like Openfoam need additional software like Paraview to handle the post-processing and visualization parts.

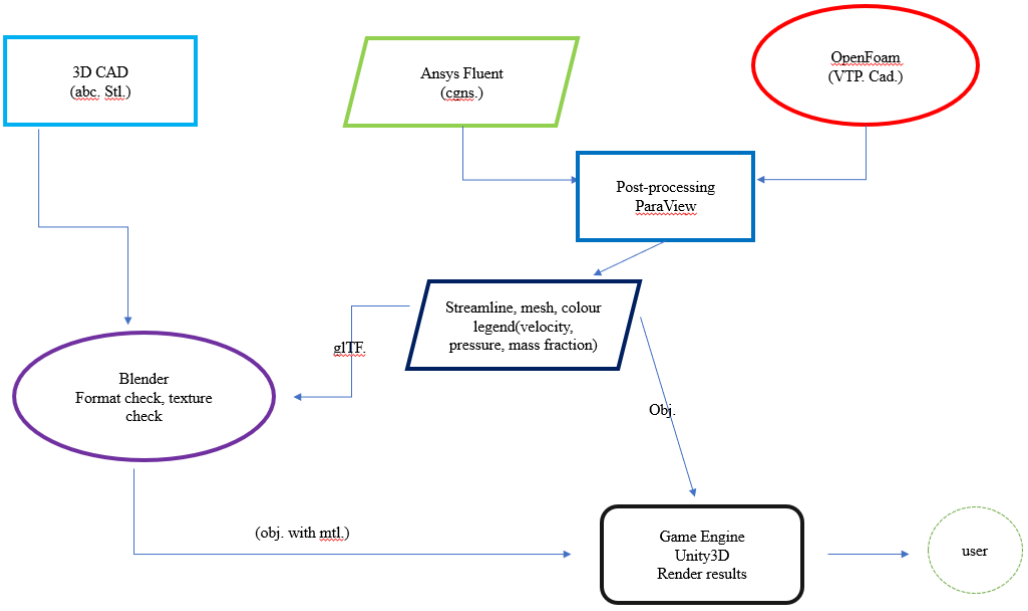
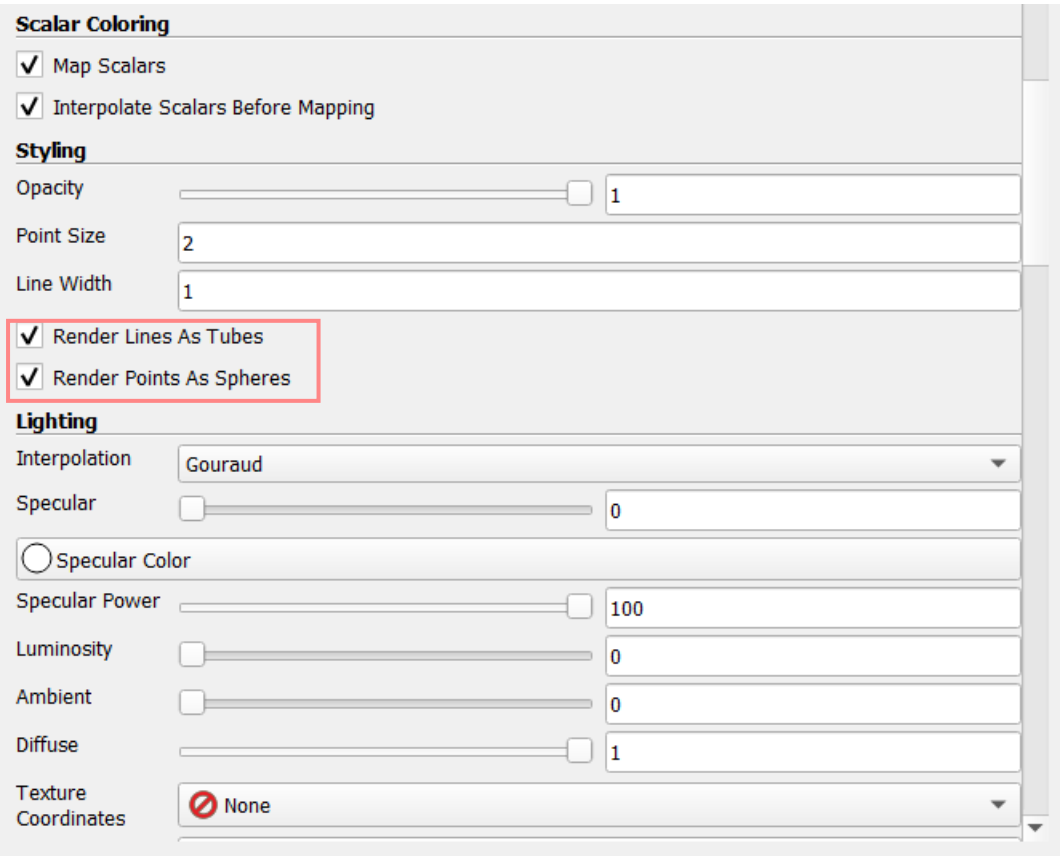


Figure 1. visualization workflow chart.

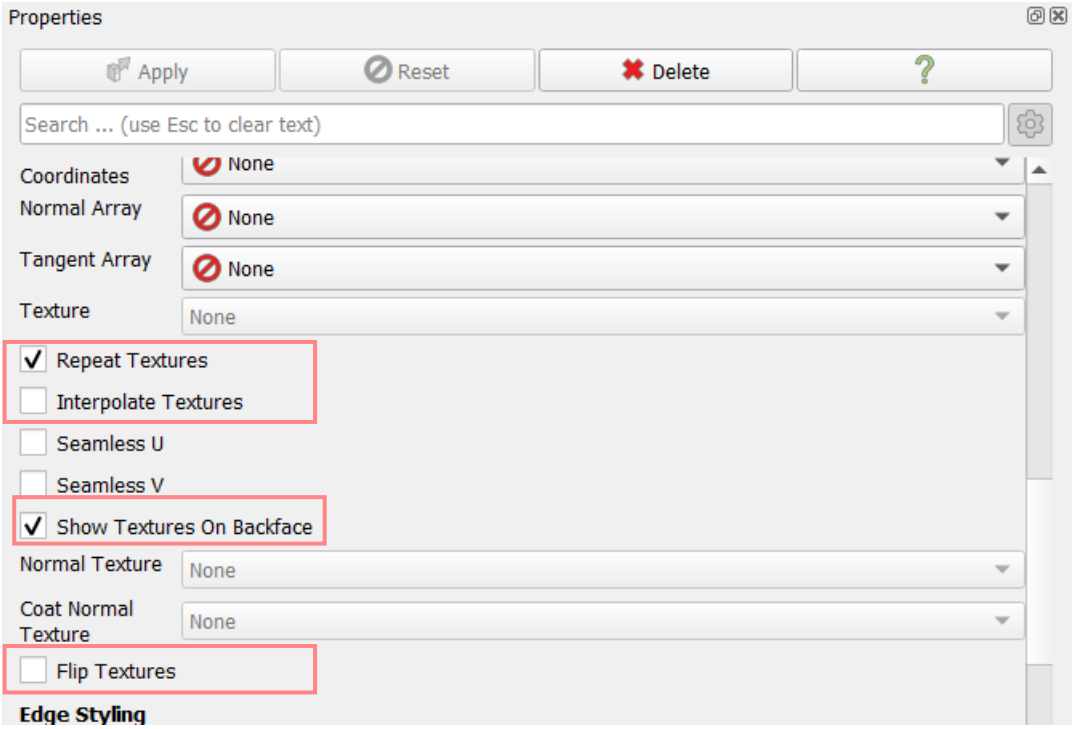
post-processing:

Ansys Fluent produces geometry data including meshes and nodes inside the simulation that can be directly storage as Position files, and the mechanical properties is also storage along with position. But this Position file is unable to edit. The only format that this mesh data with Geometry data can be export is a txt. like file called cgns. As for OpenFoam, the geometry data is stored separately as a CAD file with STL. Format. The simulation results (velocity, pressure, mass fraction and critical Q value) with mesh data are provided as a Virtual ToolKit surface models (polydata) in VTP. Format. All those files will be further examinate through ParaView to generate streamline, contour, iso-surface, triangulate, and adding filters for easier converting process. Almost all Virtual Reality visualization method will not clearly display lines and points including Unity3D and ParaView OpenXR plug-in. Therefore, two filters call “Tube” and “convert into point cloud” are commonly used to render lines into tube and render point into sphere or point cloud. Also, Unity3D is optimized for rendering elements with no more than 4 faces. So, for best results, another filter called “trangulate” will reduce the mesh elements to less than 4 faces. The final step is to generate “material” files by adding colour legend for different properties. Each property (velocity, pressure, mass fraction…) will create its own colour legend and will be exported as one material file. For maximize the performance, only store the material file needed.

Other settings that may be worth to try are listed below:



This setting will help you to quickly set up 3D rendering, but this only works with simply geometries. For complex model, still needs to go to the filter tab and manually add “tube” filter and “convert to point cloud” filters.



By selecting the “repeat textures” option can help same time and space exporting Obj files when all the materials/textures are the same.

By selecting the “Show Textures on Backface” option can help with processing Normals. Sometimes user will be facing a issue that the model/texture can only be seen from certain angles, and it will become invisible when rotating the view. This is because the CAD model only have one normal and the Backface is not baked with textures. This will help fix the problem. But sometimes when the model is too complex, user will need to manually adjust the normal in other software such as Blender.

By selecting the “flip texture” option can directly flip the normal of the texture.

Data export:

There are two kinds of file can be exported by ParaView. The first one is directly using obj. files. Wavefront obj. format is an encoded surface geometry files that can be massively generated by ParaView’s “save data” function. This is normally used for large number of time steps (greater than 200 time steps) or large number of simulation result files (more than 200 cgns. Files or VTP files). The disadvantage of direct exporting obj. format file from ParaView is that ParaView will not pack the corresponding material file with the obj. file. Therefore, user will need to export again for just obtaining the material file or using other Post-processing software or making their own material file inside Unity3D. This is procedure is also used for the case that all the obj. file can share the same material file without the need of abstract the individual material for each obj. file. The second kind of file that can be exported by ParaView is glTF. Formal. glTF format is a 3d file format that stores 3d model information in JSON format. This format is optimized for packing both 3D assets size with the runtime processing needed. This format can automatically generate file sequences based on time step sequences and store the all the materials associated with the meshes. The disadvantage of this method is that user need to do an additional step to convert this glTF. Format file into 3D model obj. format files with all material mtl. Format files.

To export obj. files, user can go to “File” >> “Export scene” then select file type to “GLTF” or “X3D”.

X3D file is for user who only wants to export the geometry of the model or only wants to use the mesh. This way user can customize the material, texture, lighting and other options later using Blender.

GLTF files is for user who not only wants to export the geometry, but also the texture of the model. By converting GLTF model into obj, files, it will also generate a STL file containing the texture properties.

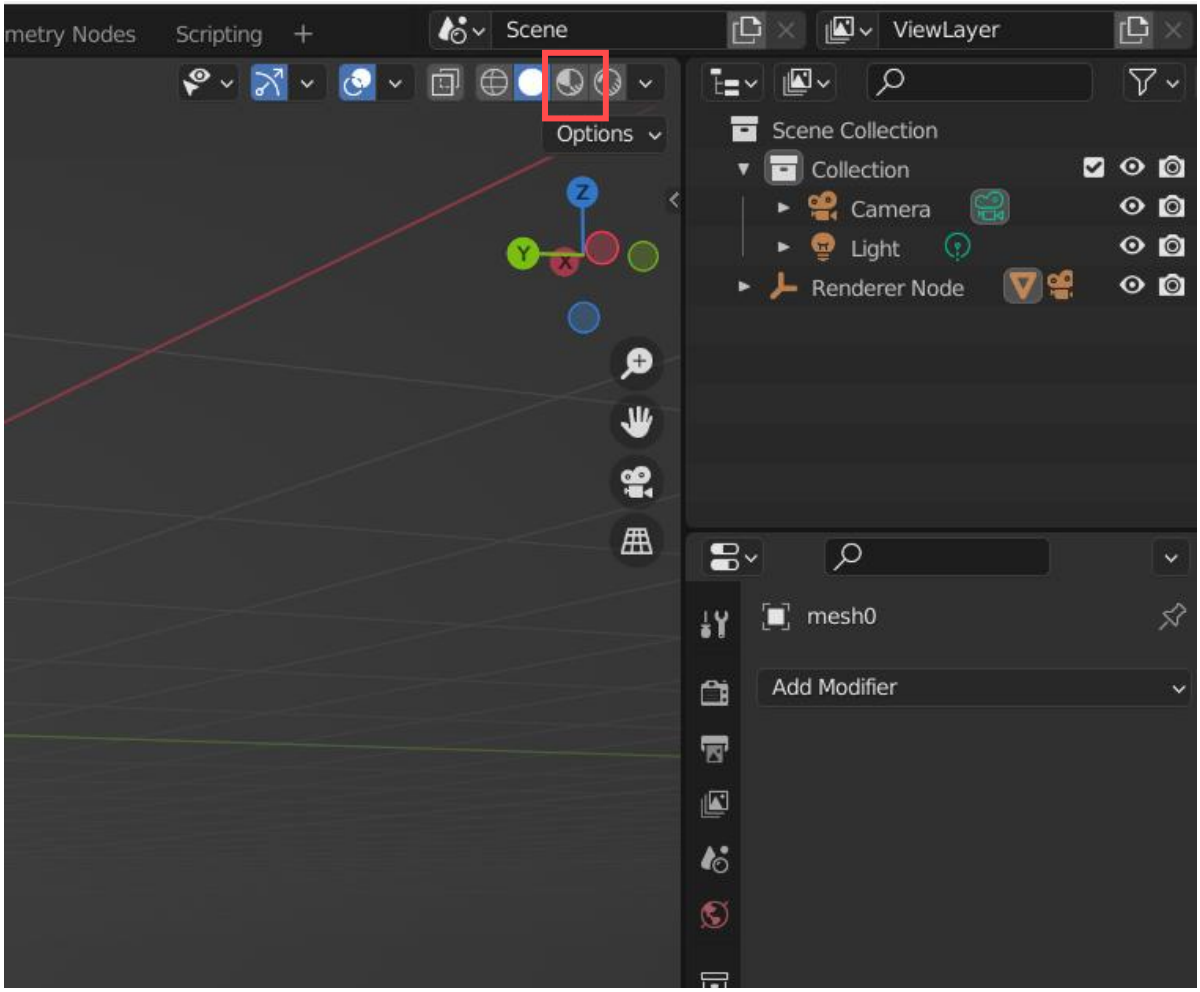
If post-processing step or modelling step is using Autodesk, then it can directly export obj. files through the export functions. There is no need to first convert everything into X3D or GLTF files.

Blender for file type conversion (optional)

If using the second kind of file exporting method, all the geometry files (stl.) and mesh files (VTP. Cgns.) will be converted into only one type of glTF. Files. The number of files will be equal to the number of time-steps available inside the simulation results. Then import all glTF. Files into Blender software which all time-step can be imported all at one as a packed file. It is optional to add a smoothing filter to smooth out the mesh. Double check the material file by going into the UV editing window to see whether the material file is properly imported. It is also optional to select specific shading settings for user's preferences. The final step is to export the 3D models as time stepped obj. file sequence. By using Blender software, the associated material file in mlt. Format will automatically generate when exporting into wavefront obj. format.

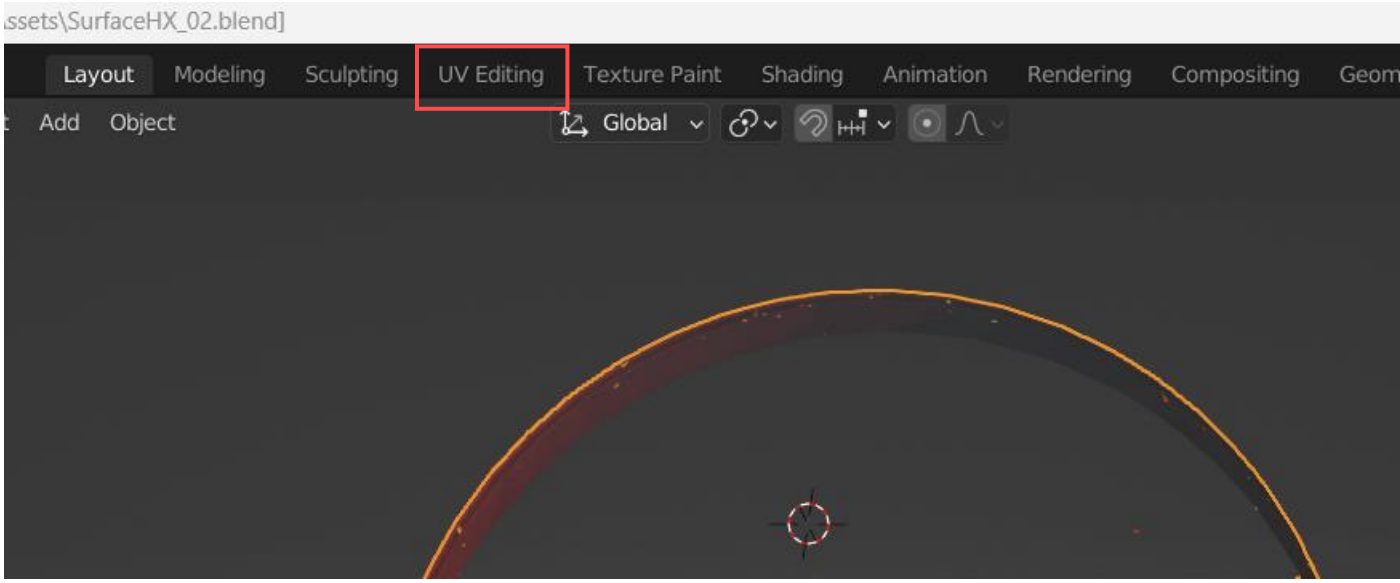
Blender UV editing:

After loading the model (GLTF. file) into Blender, first should check whether the texture is baked with the model. In the layout window, the model should look like a grey solid body. To check the texture, can find the following button on the right up corner.



By selecting this option, it will show the currently texture with the lighting condition. If the texture is loaded properly, user should be able to see the texture.

The next step is to check the normal. By rotating the model, user can see the normal direction. After find out the normal direction, go to the UV editing window by selecting “UV editing” on top.



For more information about how to UV unwrap and edit texture, there are lots of video guide on Youtube. There below is some useful link that I used to help me with my project.

https://www.youtube.com/watch?v=7JUNlj6mR0U&t=88s

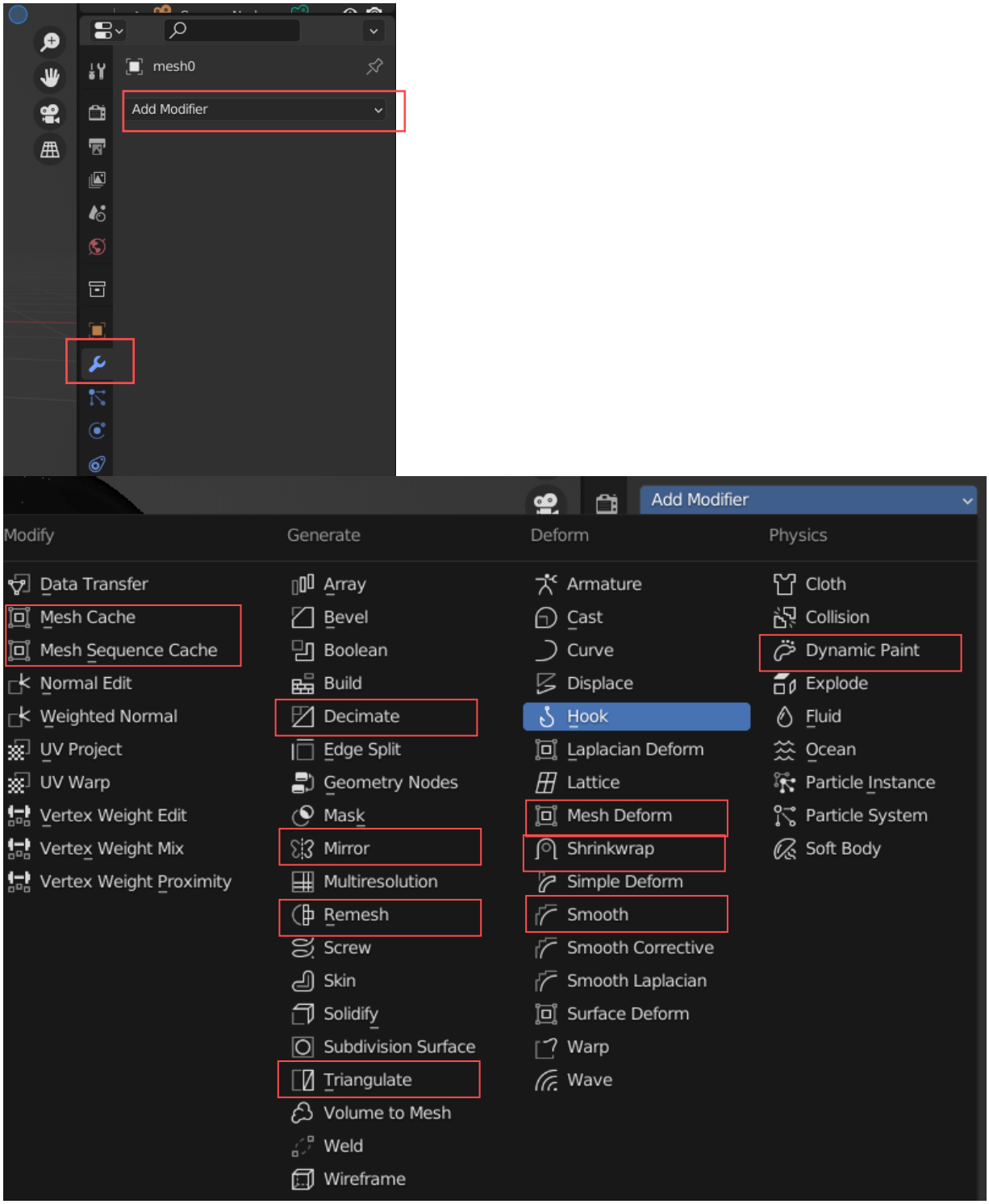
https://www.youtube.com/watch?v=scPSP_U858k&t=681s

for complex models with multiple textures/materials, there below is a video guide on Youtube

https://www.youtube.com/watch?v=wG6ON8wZYLc&t=384s

Be aware that Blender add-ons and other installed features may not work on different versions. Need to check the version compatibility.

Some of the useful modifiers to use can be find from the picture below.

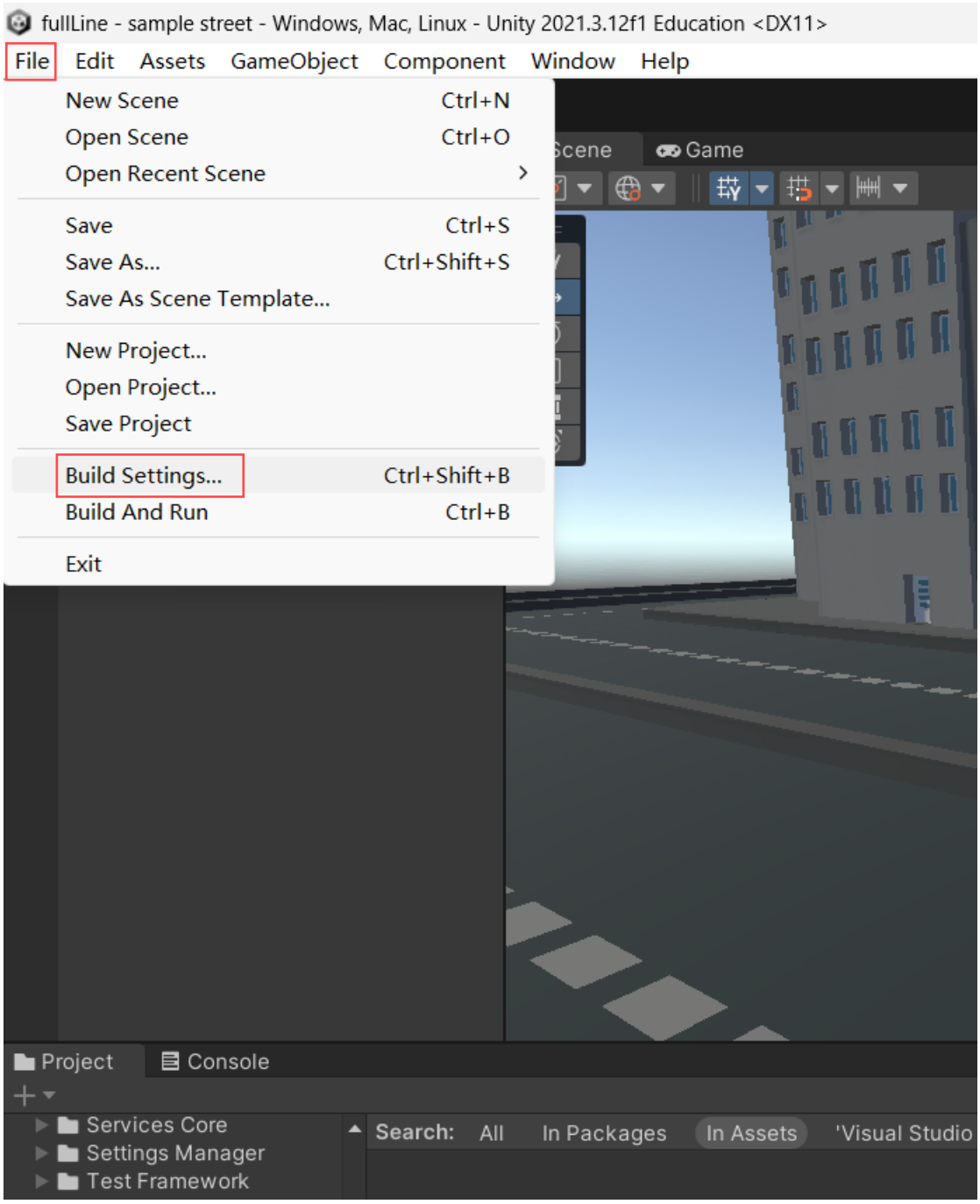


For example, decimate can help reduce mesh number, this can reduce the work required for unwrap process. Smooth can help after using decimate to smooth the model. Dynamic paint can help customize the texture.

Create frames using obj. sequence in Unity3D Virtual Reality environment.

Pre-build setting

After installed Unity3D, there are some settings need to be done before it can be used for VR production. When start a new project, there is an option to start a VR build using core rendering. There is no need to only using this build. Any 3D build can work with VR and those settings can be modified later inside the newly build project. When open a new project, user can click on the file button on top and open the drop menu below, then select build settings.

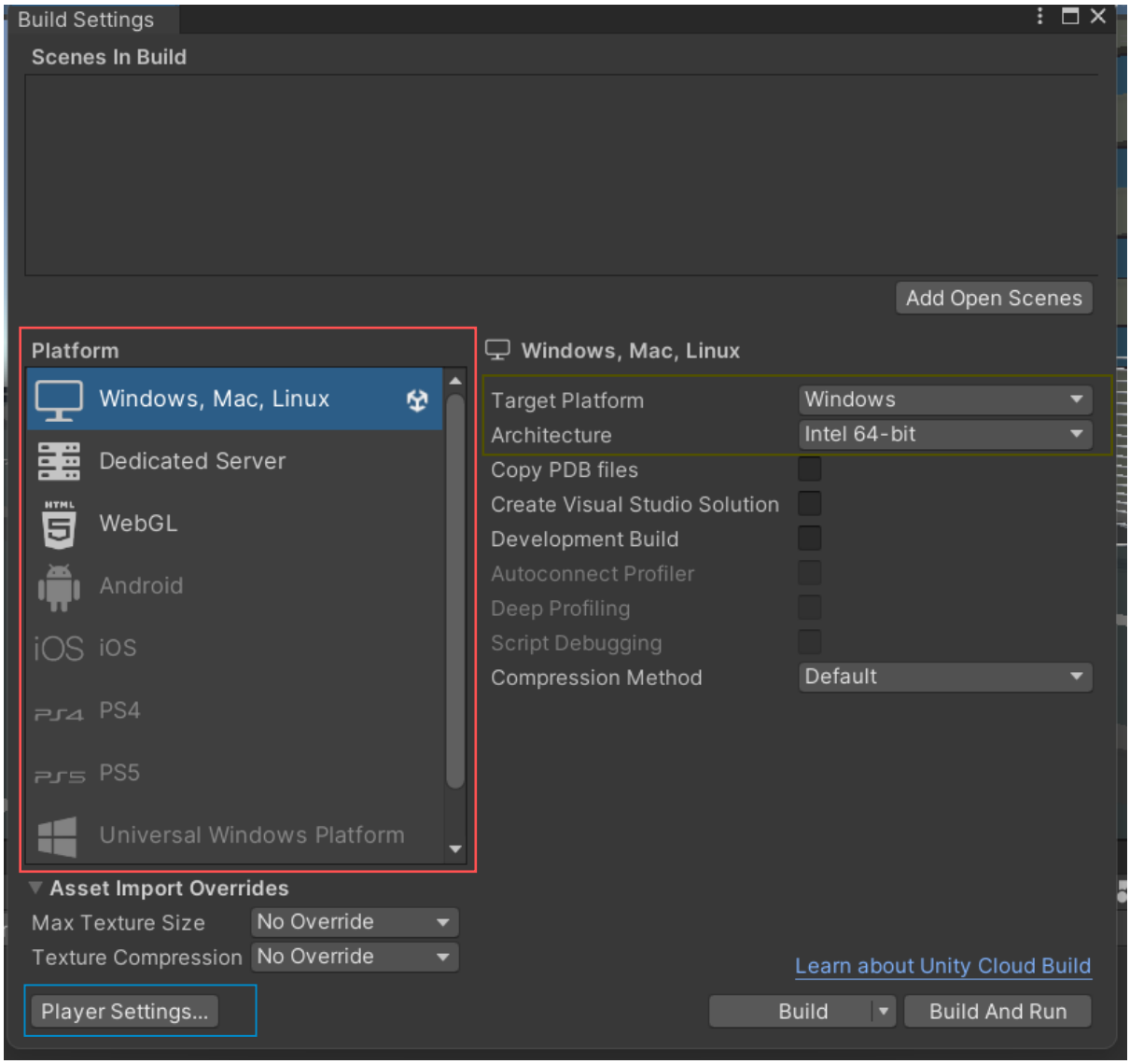


Inside the build setting, there will be the scenes in build, which including all the scenes that already been build in this project. User needs to manually add all scenes wanted for this setting.

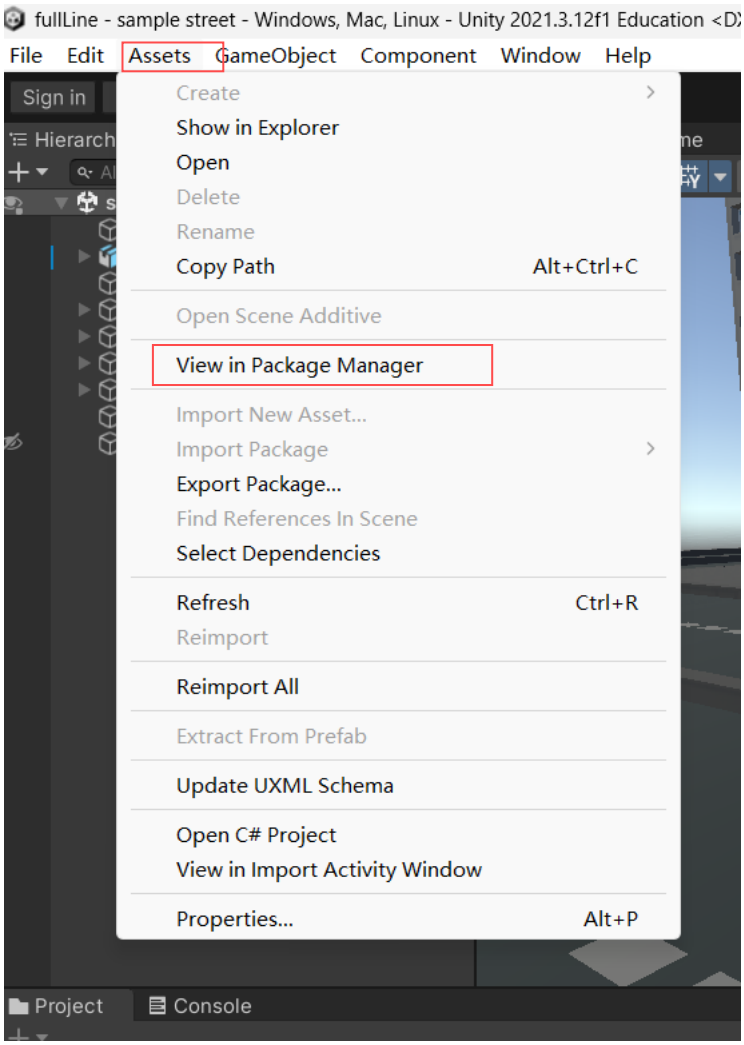
Over here at the build setting, user can change platform, which can be seen in the red box.

User can also change the target platform between windows, Linux and Mac by changing the setting in the yellow box.

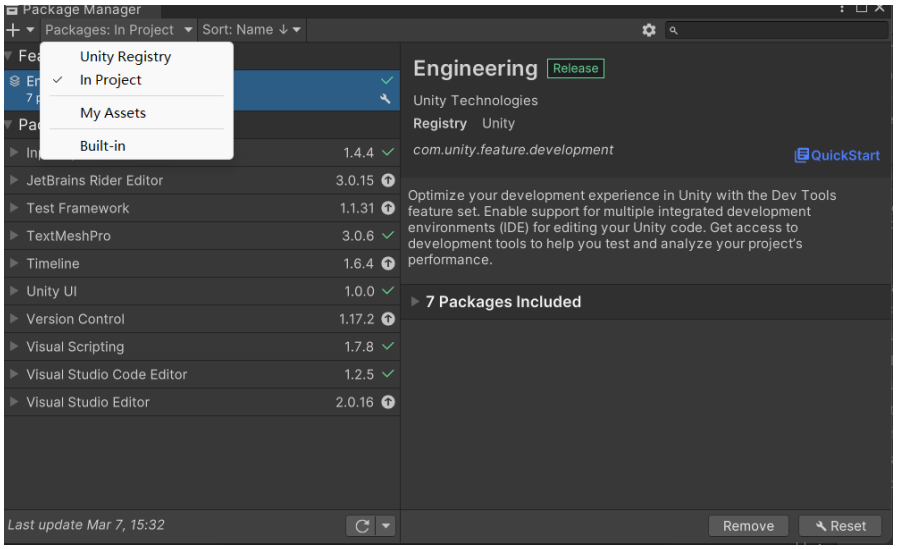
If user wants to explore more player settings, can click on the player settings button in blue box.



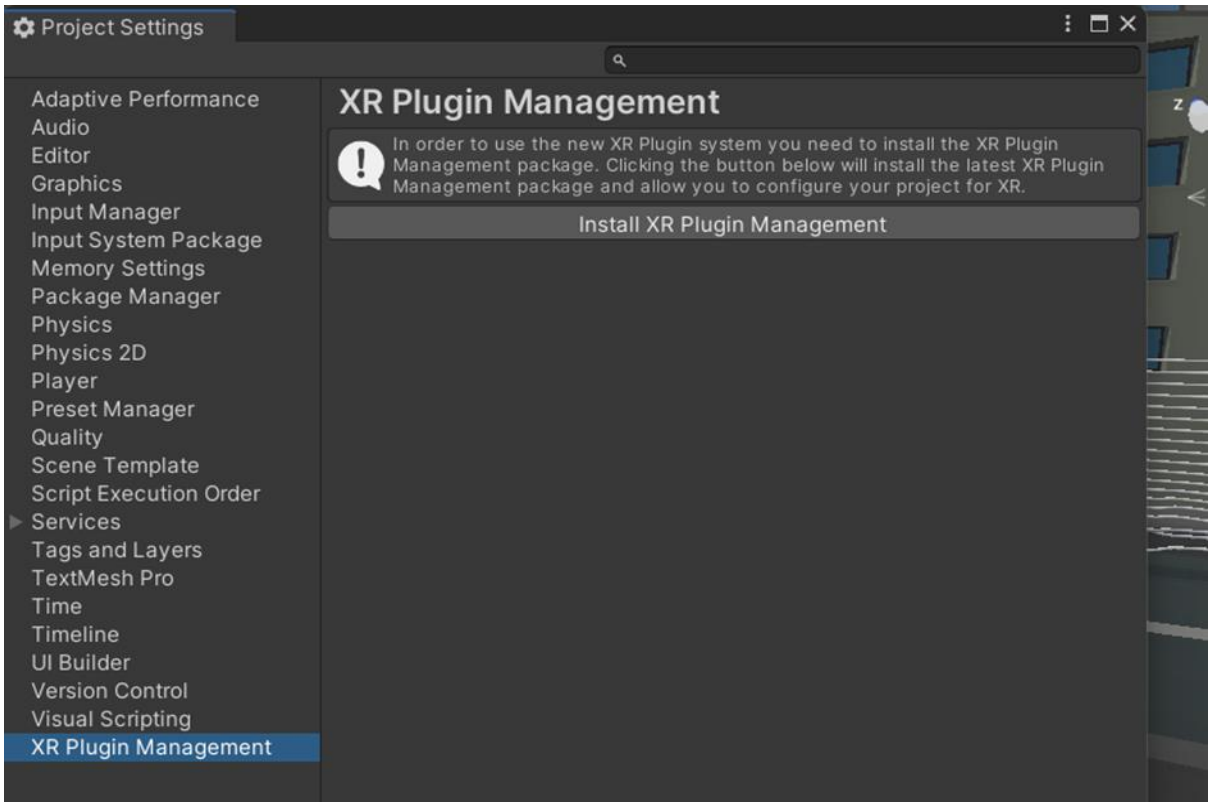
Besides the platform settings and player settings, there are some plugins and packages need to be installed before it can run VR properly. To open the package manager, first select the Assets menu on top, and on the drop menu, select “View in Package Manager”

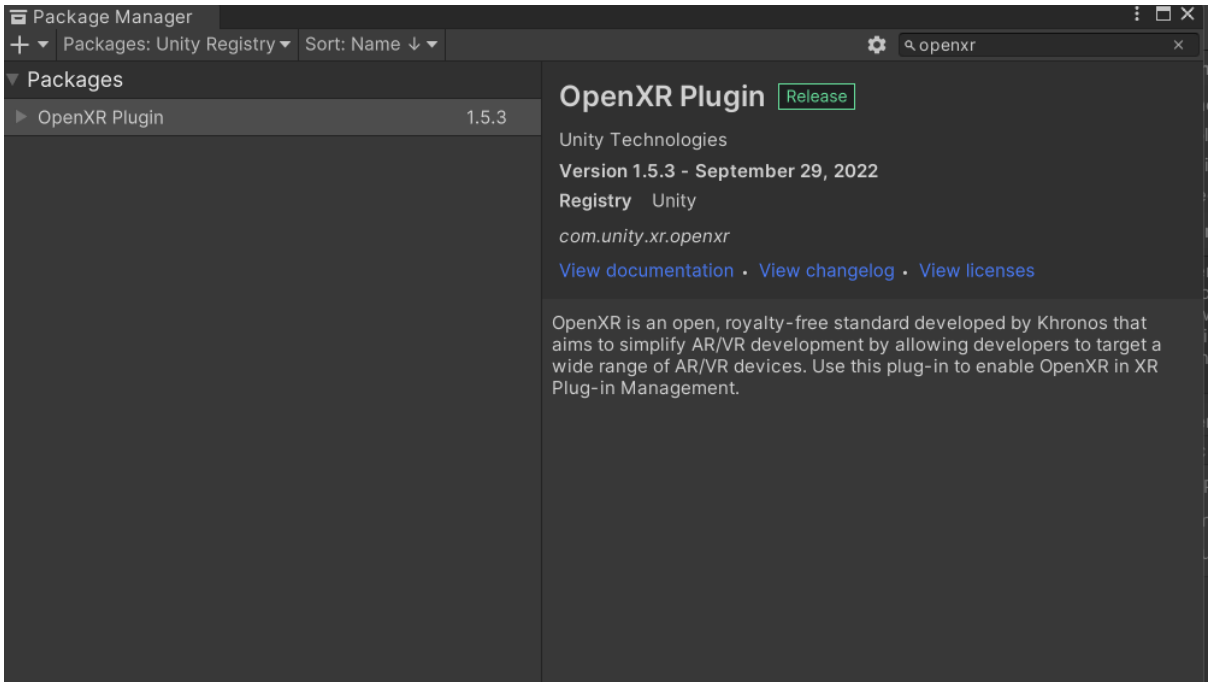


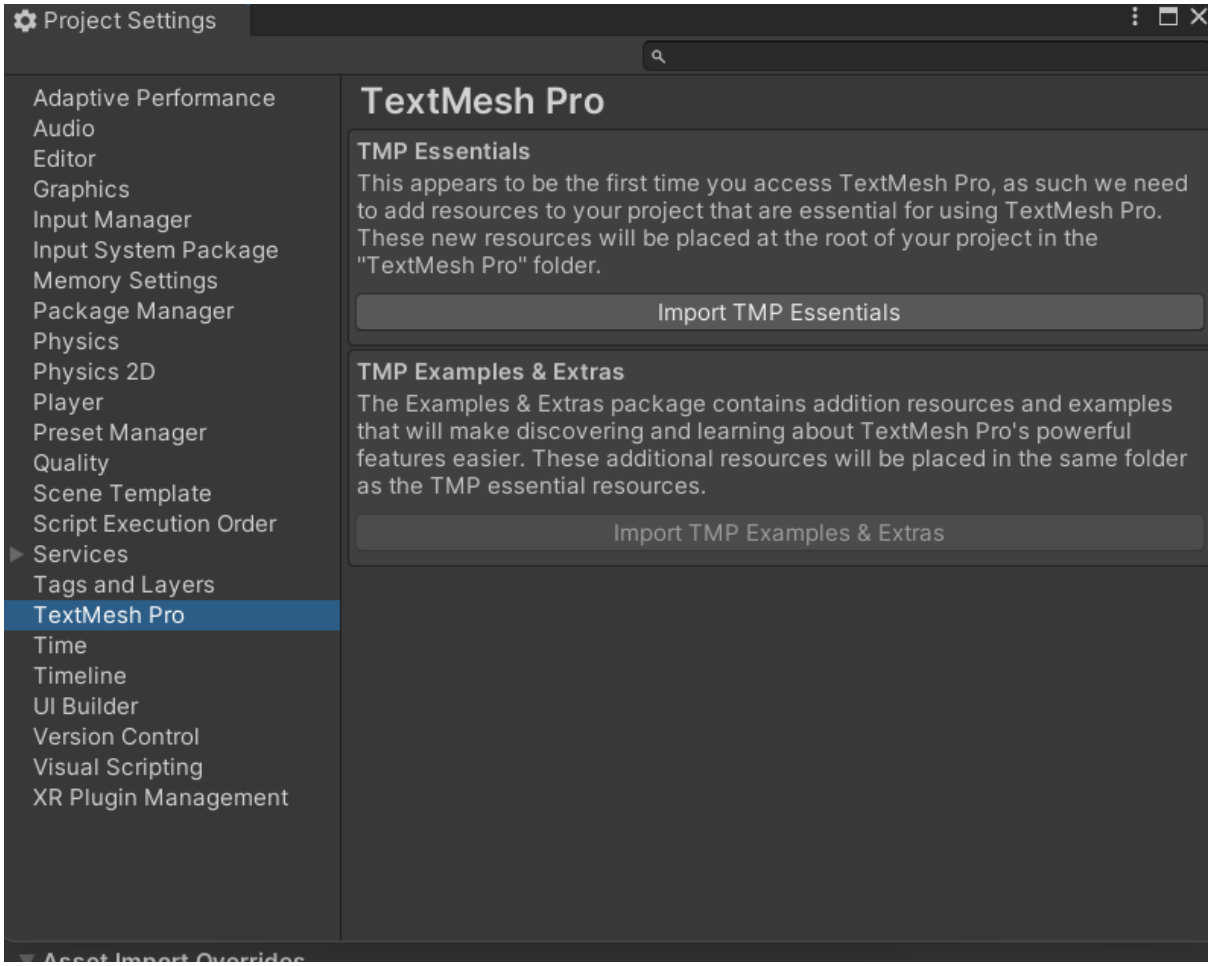
Package manager including all the package that is commonly used in the Unity3D and any package user selected to installed during the installation process. On top next to the plus sign, there is a drop menu display the current packages used in this project. User can also install any package from the installation process by switching to Unity Registry. If the target plugin/package can not be found inside the Unity Registry, user can also searching for the package from the Unity asset story and add to “My Assets” and install from there.



There are a few packages need to be installed before user can work with VR project. The first one is XR Plugin Management and OpenXR plugin. Just search those in the search bar and install. Some version of unity may need to install TextMesh Pro to add textbox in the scene.







If user want to add proper interaction features, one of the plugin can be use is call “com.unity.xr.interaction.toolkit”. Type this into the search bar and install this toolkit can help user add interaction features such as grabbing and throwing. There are lots of more ways to achieve those features and there are lots of functionalities can be added. For more way to make VR applications, there are a few video guide below that can help.

https://www.youtube.com/watch?v=fM0k2n7u8sc&list=PLpEoiloH-4eP-OKItF8XNJ8y8e1asOJud&index=1

https://www.youtube.com/watch?v=RpHAZ0N5W1s&list=PLwz27aQG0IIK88An7Gd16An9RrdCXKBAB

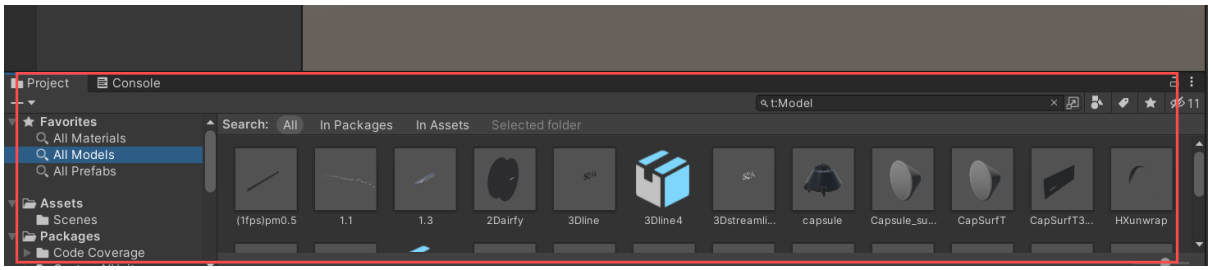
This is a VR course from Linkedin Learning which requires a certain subscription or paid services. But it will offer more detailed learning experiences.

https://www.linkedin.com/learning/unity-building-vr-user-interfaces/a-better-occlusion-strategy?u=54776729

Loading assets, FBX files, Blender files and Obj. mesh sequences

Assets can be purchased from Unity asset store directly. There are lots of free assets available to use. For example, I have personally downloaded lots of car model and building models for free from the asset store with real good quality. Credits to some of the developers who has put their work free of charge. Final Form Studio offers lots of detailed car models from high-poly to low-poly models. PoèMe offers a few cities build assets contains lots of different models that can help user build a whole city in hours. AndaSoft has a free version of Easy road building asset that will satisfy most user's need. Although it is possible to install own assets from other source, Unity Asset story is the most safe and reliable source.

To creating own assets, one common way is to import model from Blender. Blender files can be directly read in Unity. Simply by clicking "Assets" >> "Import new assets" and select the .blender file or drag and drop the .blender file into the project area. Like show below.



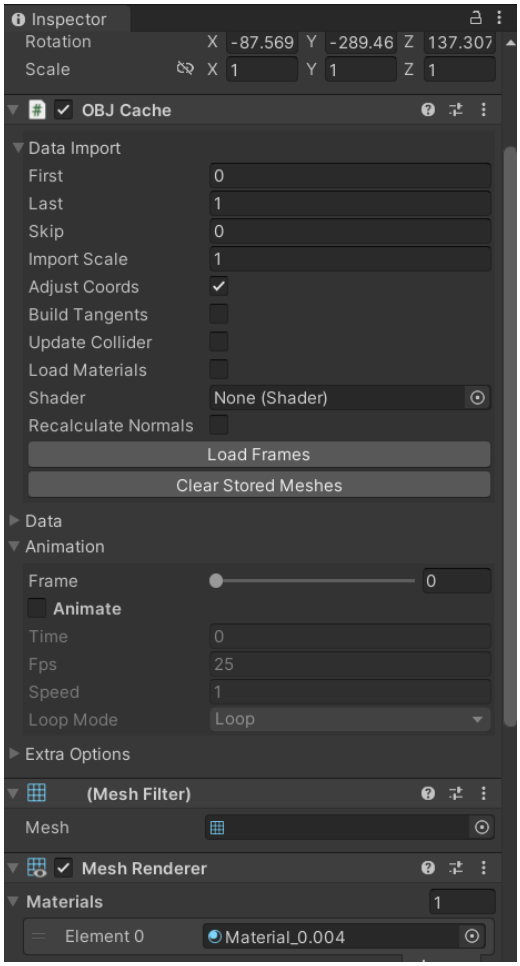
The texture file from Blender is usually saved as a picture file, and Unity can not directly read the texture file. User needs to create a new material and add the texture file into the new material and apply the material to the asset manually.

Another way is to import FBX file as a new asset. This format is more acceptable for other 3D modelling software. FBX file are usually smaller than .blender file and can commonly found in the asset story. The same texture problem remains with FBX file. User needs to manually create the material file and add the texture into the material file. Then apply to the asset.

There are multiple ways to import obj. mesh sequence. One way is to use the animation toolbox build-in Blender and Unity. The method used in this project is to load all obj. files in sequences with hundreds of obj. files. This is the most accurate way to re-create the simulation results. The result is nearly pixel perfect. But the downside of this is a huge work load and requires a powerful machine to run it. One tool from the asset store that is available call Megacache offers a basic obj. sequence reading and loading functionality. Using the scripts provide by Megacache can open a window inside Unity inspector to load frames from obj. sequence as obj cache.

Credit to Chris West for coding the scripts. There is a link below that can direct to his page and review the asset.

https://assetstore.unity.com/packages/tools/modeling/mega-cache-26522



As shown above, the material files can also be imported into the mesh renderer inspector window for each frame. Each individual frame is created by one obj, files and one element that can be used to upload material file. Animation can also be generated by check the animate box and parameters like time, FPs, speed, and loop mode and be adjusted mode the inspector window. Besides the time-stepped simulation results, Unity3D is also very powerful on creating scenes and adding interactive features. In the NEE study case, an urban environment is created as background and basic interaction like grab, snap turn and teleporting is added using Unity3D SteamVR input toolkit.

Project Result Demonstration

By using the tools introducing above. A same CFD simulation results animation for the NEE study case is made into a 3D Virtual Reality environment with almost pixel perfect accuracy. The animation shows the streamline with 200 frames with 25 frames per second. A 3D iso-surface of PM 2.5 at 10% mass fraction representing the road dust resuspension and wheel wears is generated also as animation around the vehicle shows the constant velocity and deceleration case with 400 frames with 50 frames per second. All models including vehicle, building and road are set as live size and the main camera will follow VR headset movements and offset as 1.5 meter above the ground level. The whole city's floor allows user to teleport using the controller. Users can also use controller to snap turn for 45-degree angle and exit the game. Users can also walk/teleport into the simulation and observe how high and how far the emission is distributed.

Other visual representation can also be added into the scene. Things like walking pedestrians, smoke particles, audio feedback and instruction. A following camera with moving vehicle as reference can also be added to provide user a first-person point of view from the moving vehicle.

Limitation:

The difficult part of this project is the combination of two different kinds of 3D visualizations. One is the time-stepped mesh files that needs to be played in time order to create a movie-liked animation inside the virtual reality environment. Each time step (0.01 sec) is its own mesh with different number of elements, nodes, geometry, and material. The material files represent the mechanical properties data like the colour map of pressure, velocity, and mass fractions. Therefore, each property has its own material file, and one single time step (0.01 sec) files could have up to five material files associated to it. Another kind of 3D visualization is the common Unity3D asset file called prefab. This is a packed file contains meshes, renders, shading, UV setting, location transfer data and scripts. Scripting using C# programming language is the main method to adding functionalities into the Unity3D platform. Rendering obj. sequence, edition material, adding building, road, and pavement, adding wheel rotation, jumping between scenes and so on… all of those need to write the corresponding scripts. Learning C# language, debugging the code, and finding the right tutorial is difficult. The obj. file sequence is stored in the computer RAM space for faster loading time; therefore, it is limited to the RAM space. Any obj. sequence larger than 4 GB will cause problem with loading time and potentially crash the software.