Data Visualization workflow

The purpose of this guide is to develop a tool to convert numerical simulation data from any kind of CFD model (e.g., from ANSYS or OpenFOAM) into a 3D visualization build inside a virtual reality environment using the Unity3D engine. Software like ANSYS can provide certain levels of post-processing and visualization (e.g., CFD-post), while simulation tools such as OpenFOAM require additional software like ParaView to handle the post-processing and visualization parts.

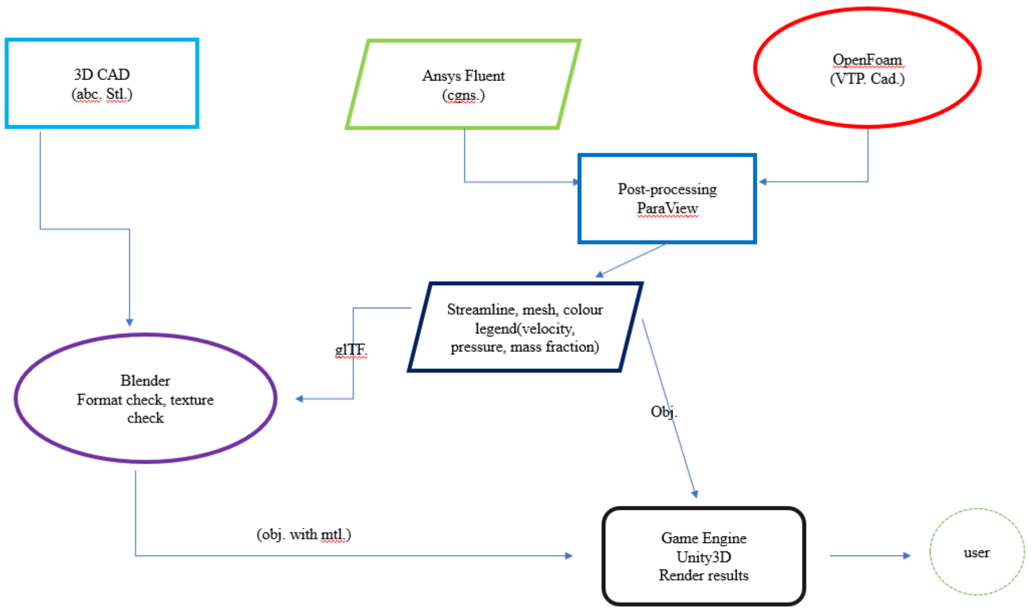
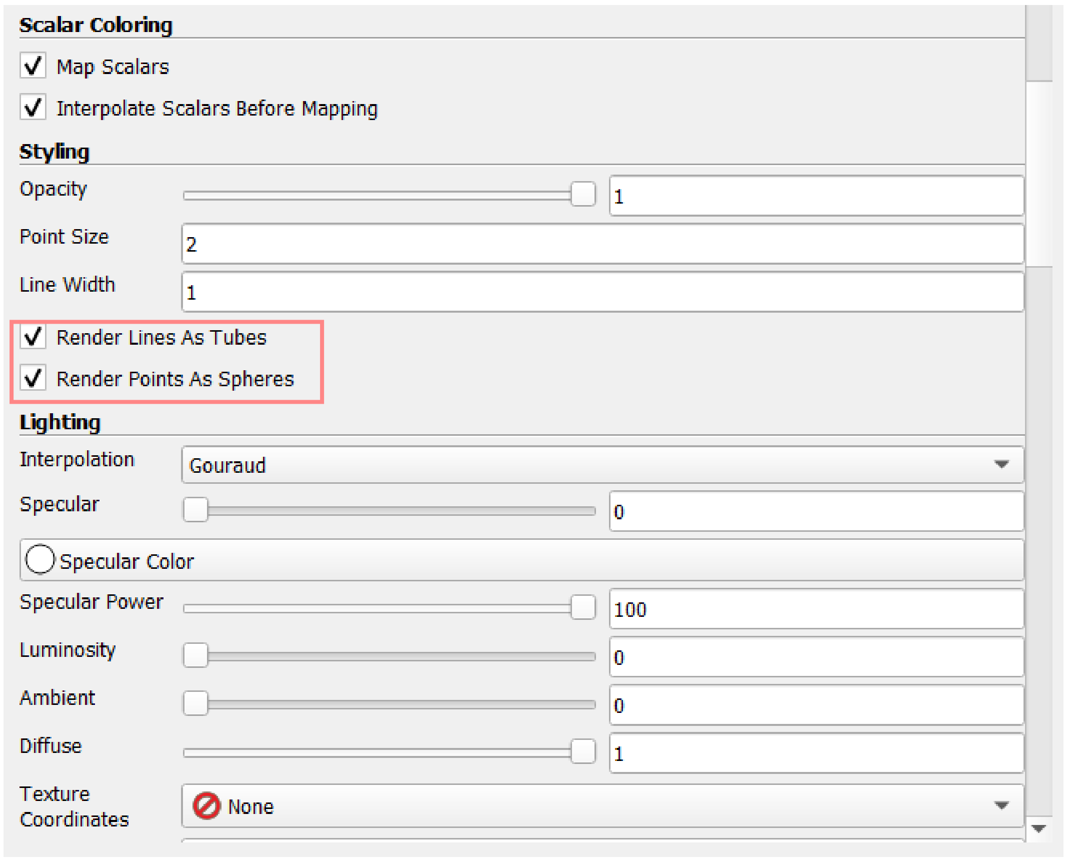


Figure 1. Human-centric data visualization workflow chart.

Post-processing using ParaView:

ANSYS Fluent produces geometry data including meshes and nodes inside the simulation that can be directly storage as Position files, and the mechanical properties are also stored along with position. However, it is not possible to edit this Position file. The only format that this mesh data with geometry data can be export to is a text-like file called .cgns. As for OpenFOAM, the geometry data is stored separately as a CAD file using .stl or .obj format. The simulation results (velocity, pressure, etc.) with mesh data can be provided as Virtual ToolKit surface models (polydata) in .vtp format. All these files can be further examined through ParaView to generate streamlines, contours, iso-surfaces, triangulate, and add filters for customizing the visualization before exporting. Almost all Virtual Reality visualization methods will not clearly display default lines and points including Unity3D and the ParaView OpenXR plug-in. Therefore, two filters call “Tube” and “convert into point cloud” are commonly used to render lines into tubes and render points into spheres or a point cloud. Also, Unity3D is optimized for rendering elements with no more than 4 faces. So, for best results, another filter called “triangulate” will reduce the mesh elements to less than 4 faces. The final step is to generate “material” files by adding color legends to represent the different properties (e.g., velocity of the flow). Each property (velocity, pressure, mass fraction, etc.) is given its own color legend and will be exported as one material file. For maximum performance and computational efficiency, only store the material file required.

Useful settings within ParaView are listed below:



Scalar Coloring
- ✓ Map Scalars
- ✓ Interpolate Scalars Before Mapping

Styling

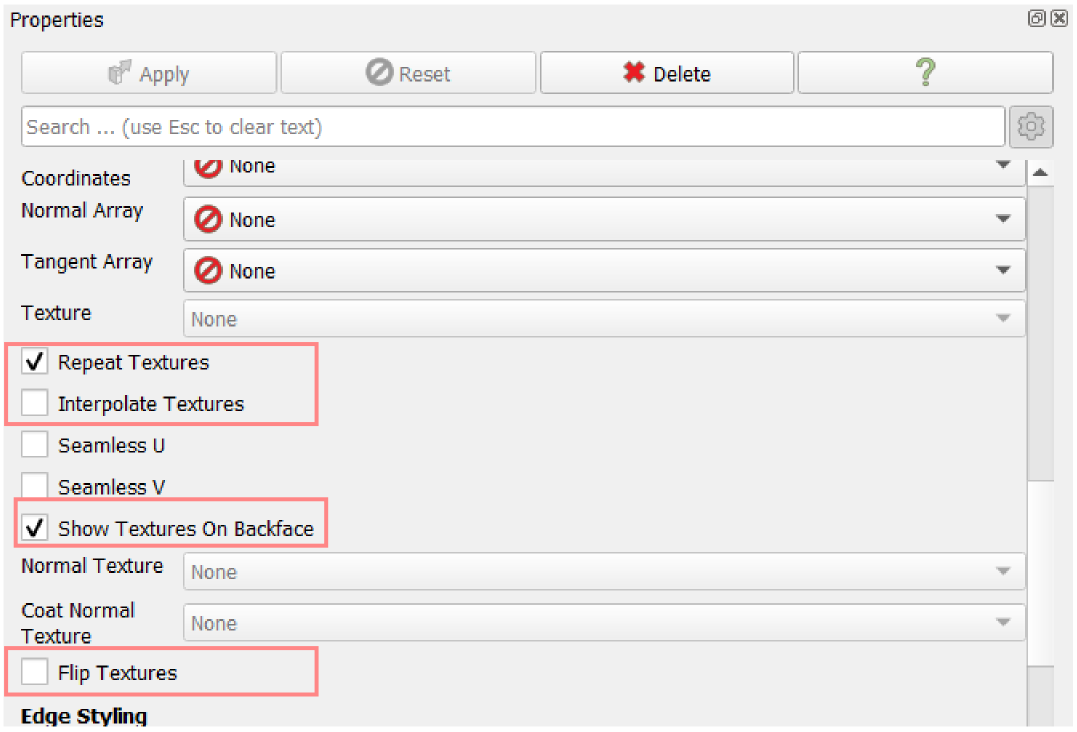
Opacity	1
Point Size	2
Line Width	1

- ✓ Render Lines As Tubes
- ✓ Render Points As Spheres

Lighting

Interpolation	Gouraud
Specular	0
Specular Color	
Specular Power	100
Luminosity	0
Ambient	0
Diffuse	1
Texture Coordinates	None

This setting will help you to quickly set up 3D rendering, but this only works with simple geometries. For a complex model, go to the filter tab and manually add "Tube" filter and "convert to point cloud" filters.



Coordinates	None
Normal Array	None
Tangent Array	None
Texture	None

- ✓ Repeat Textures
- ☐ Interpolate Textures
- ☐ Seamless U
- ☐ Seamless V
- ✓ Show Textures On Backface

Normal Texture	None
Coat Normal Texture	None

- ☐ Flip Textures

Edge Styling

Selecting the “repeat textures” option can help save time and space when exporting .obj files when all the materials/textures are the same.

Selecting the “Show Textures on Backface” option can help with processing Normals. Sometimes users can face an 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 has one normal and the Backface is not baked with textures. This will help fix the problem. However, when the model is very complex, users may need to manually adjust the normal in another software such as Blender.

Finally, selecting the “flip texture” option can directly flip the normal of the texture which may be required for cases.

Data export in ParaView:

ParaView can export several file types, one of these being .obj files. Wavefront .obj format is an encoded surface geometry file that can be generated by ParaView’s “save data” function. This is normally used for a large number of time steps (e.g., > 200 time steps) or large number of simulation result files (e.g., > 200 .cgns, .vtk or .vtp files). The disadvantage of directly exporting .obj format files from ParaView is that ParaView will not pack the corresponding material file with the .obj file. Therefore, users will need to export again for just obtaining the material file, use another post-processing software, or make their own material file inside Unity3D. This procedure is also required for cases where all .obj files share the same material file without the need of extracting individual material for each obj. file. The second kind of file that can be exported by ParaView is .glTF format. The .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 all the materials associated with the meshes. The disadvantage of this method is that user needs to do an additional step to convert this .glTF file into the 3D model .obj format files with all material .mtl format files.

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

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

GLTF files are for users who not only want to export the geometry, but also the texture of the model. By converting the GLTF model into .obj files, a STL file will also be generated which contains the texture properties.

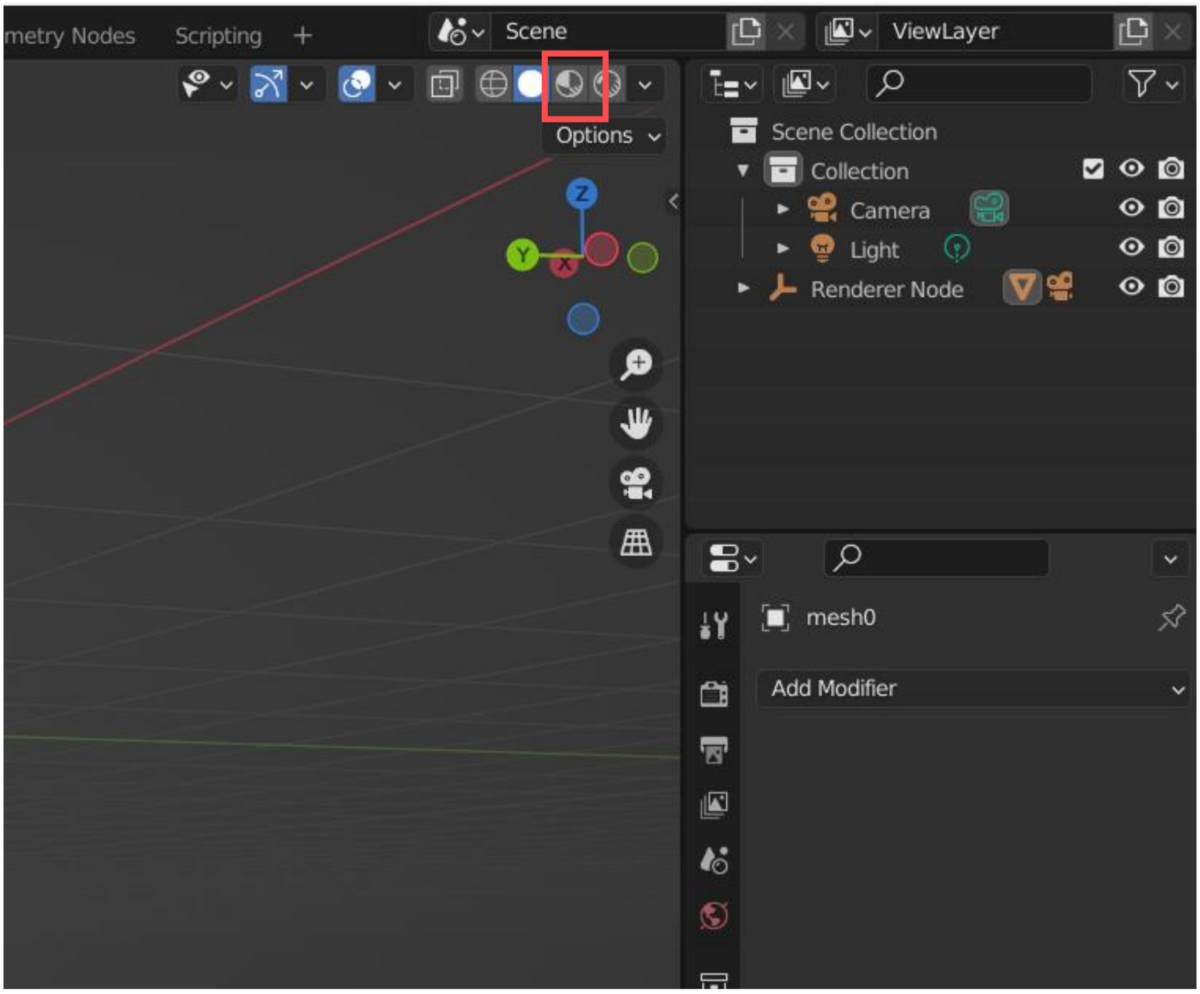
If the post-processing step or modeling step uses Autodesk, then it can directly export .obj files through the export functions. There is no need to first convert everything into X3D or GLTF files.

Blender for file type conversion (optional)

If using the second kind of file exporting method (.glTF), all the geometry files (stl.) and mesh files (.vtp, .cgns) will be converted into only one type of .glTF file. 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. All time-steps can be imported at once 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 has correctly 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.

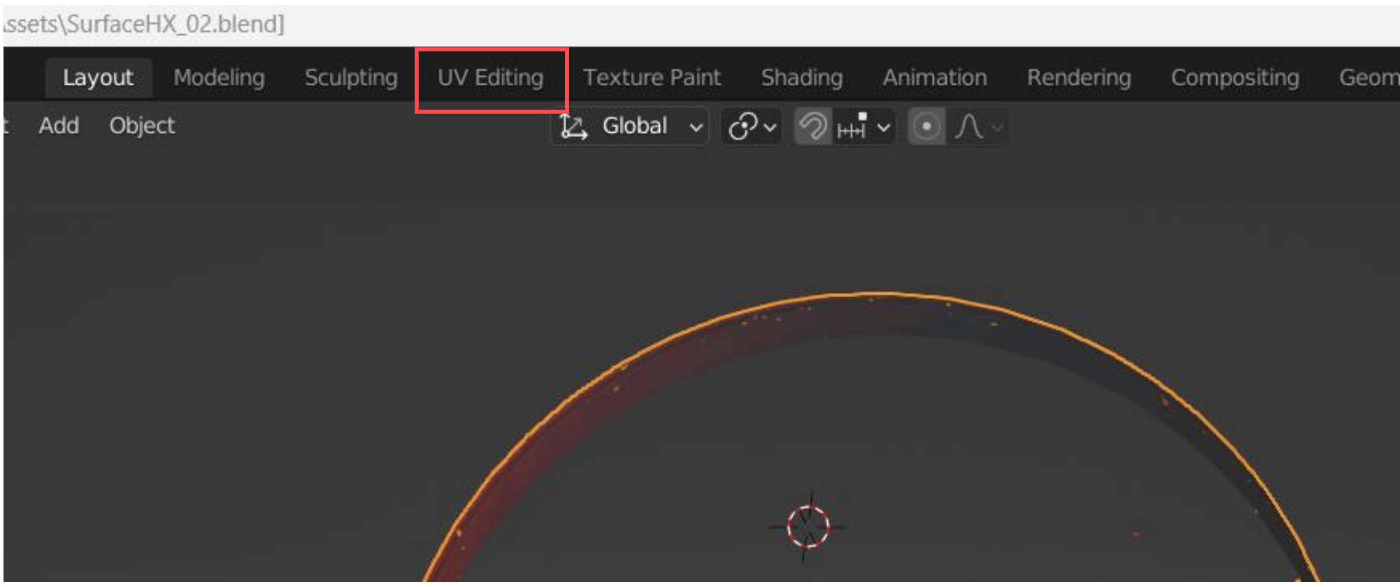
Blender UV editing and unwrap:

After loading the model (.glTF file) into Blender, first 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, users can find the following button on the right up corner.



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

The next step is to check the normal. By rotating the model, users can see the normal direction. After determining 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 many video guides on YouTube. Below are some useful links by other content developers:

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

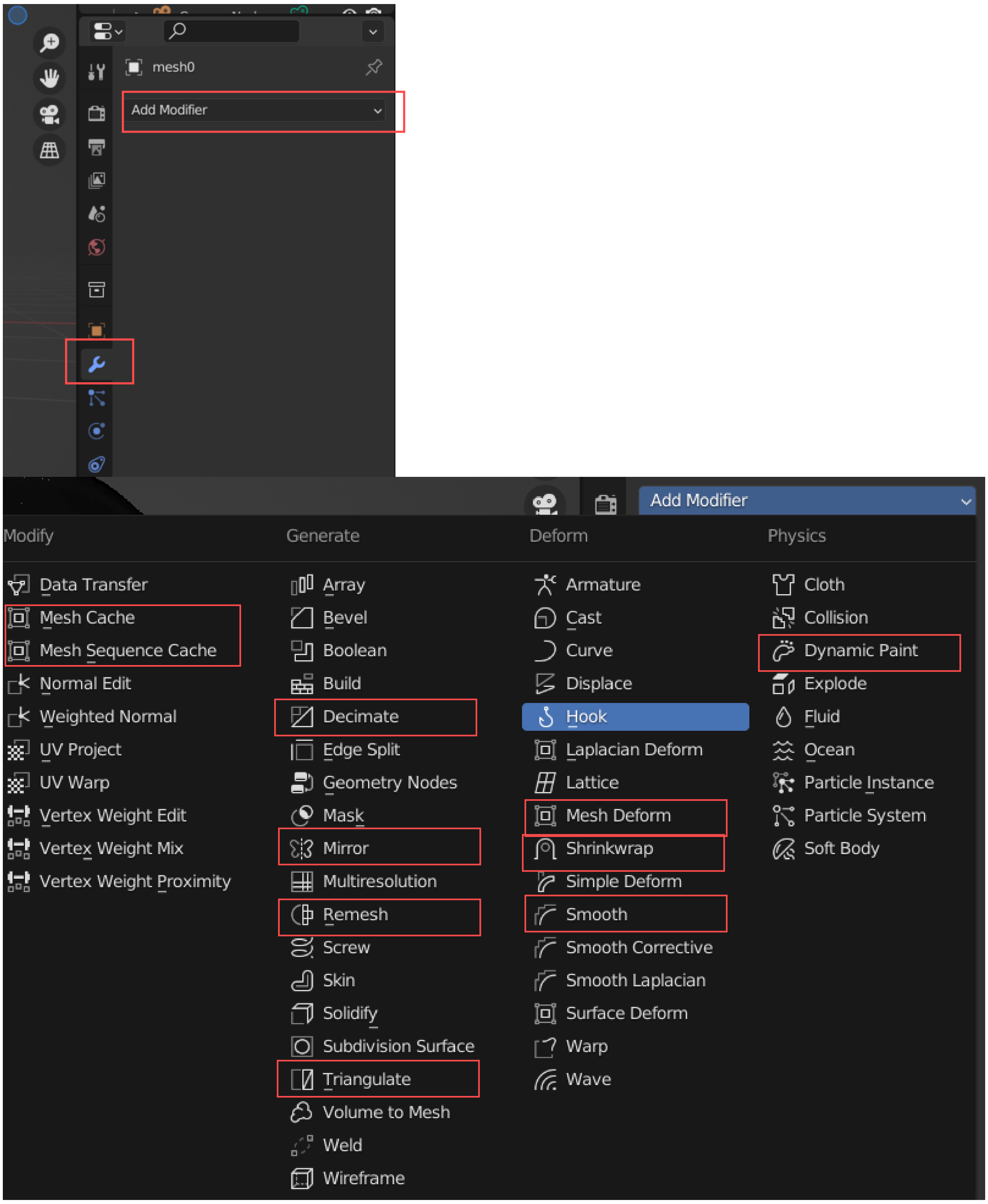
https://www.youtube.com/watch?v=scPSP_U858k&t=681s

For complex models with multiple textures/materials, below is another video guide on YouTube

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

Note that Blender add-ons and other installed features may not work on different versions. Users will need to check the version compatibility.

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

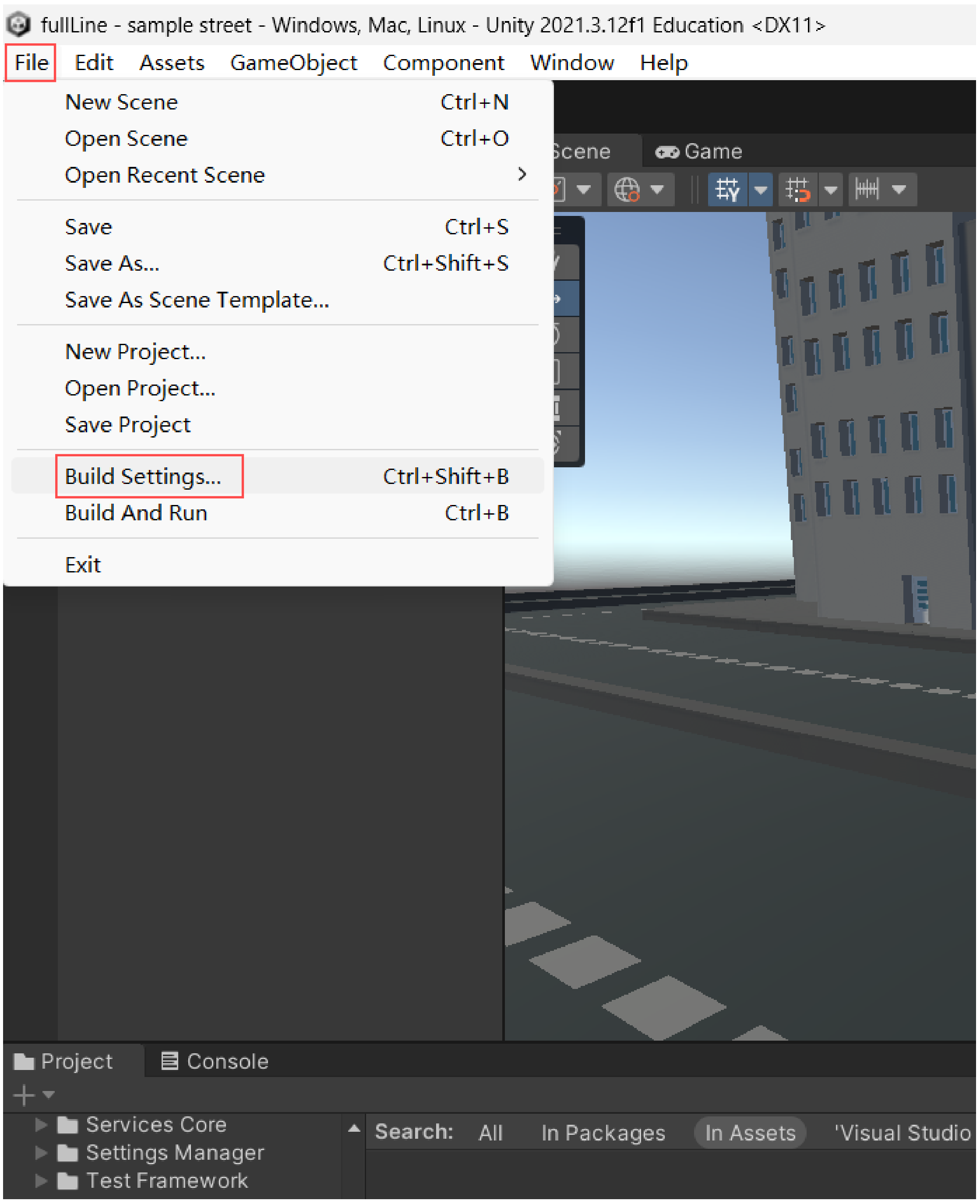


For example, "Decimate" can help reduce mesh number, this can reduce computational demand for the unwrap process. "Smooth" can help after using decimate to smooth the model. "Dynamic paint" can help customize the texture.

Creating frames using .obj sequence in Unity3D Virtual Reality environment

Pre-build settings:

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

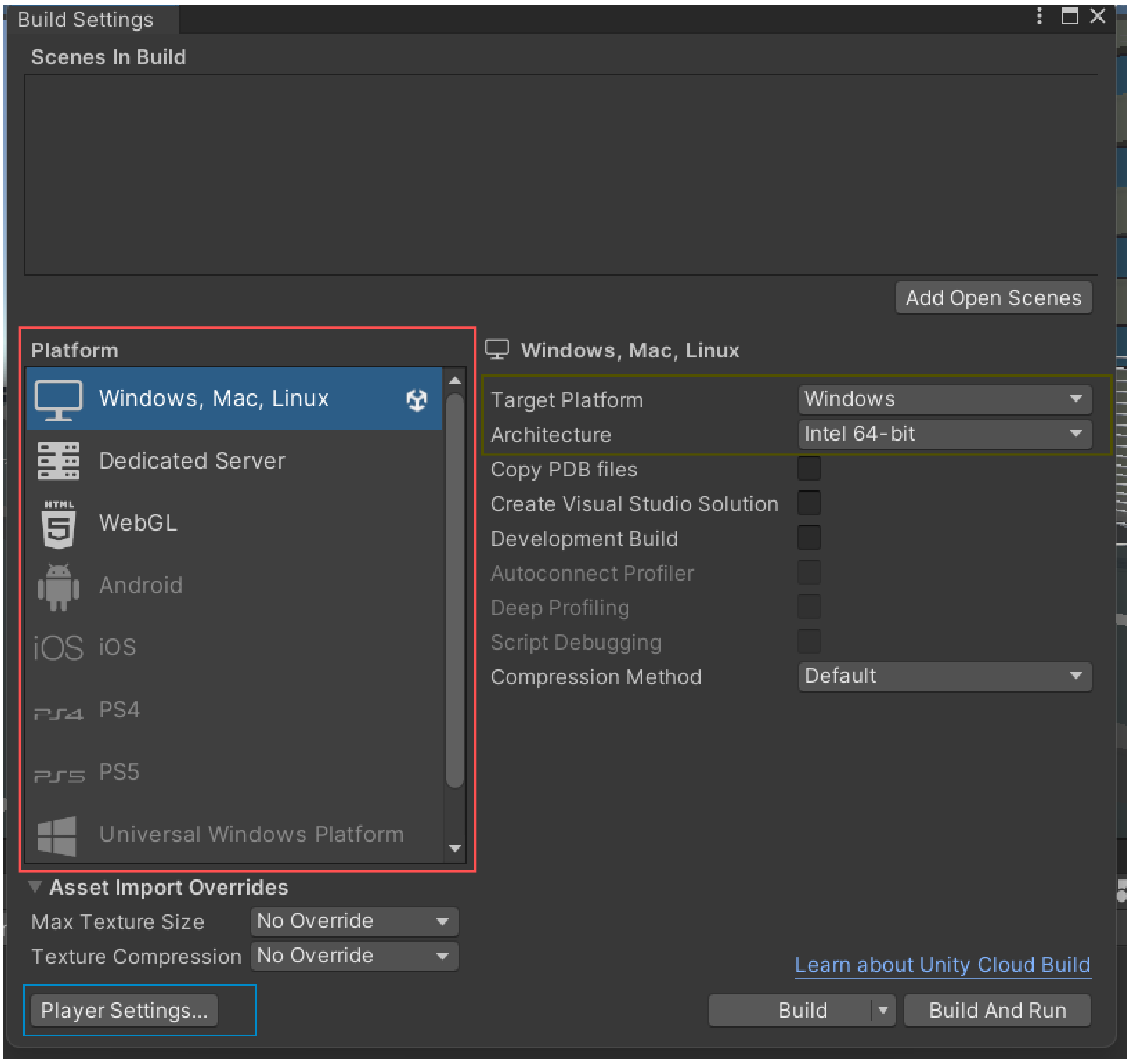


Inside the build settings, there will be the "Scenes In Build" panel, which includes all the scenes that have already been built in this project. Users need to manually add all scenes required for this setting.

Also within build settings, users can change "Platform", which can be seen in the red box.

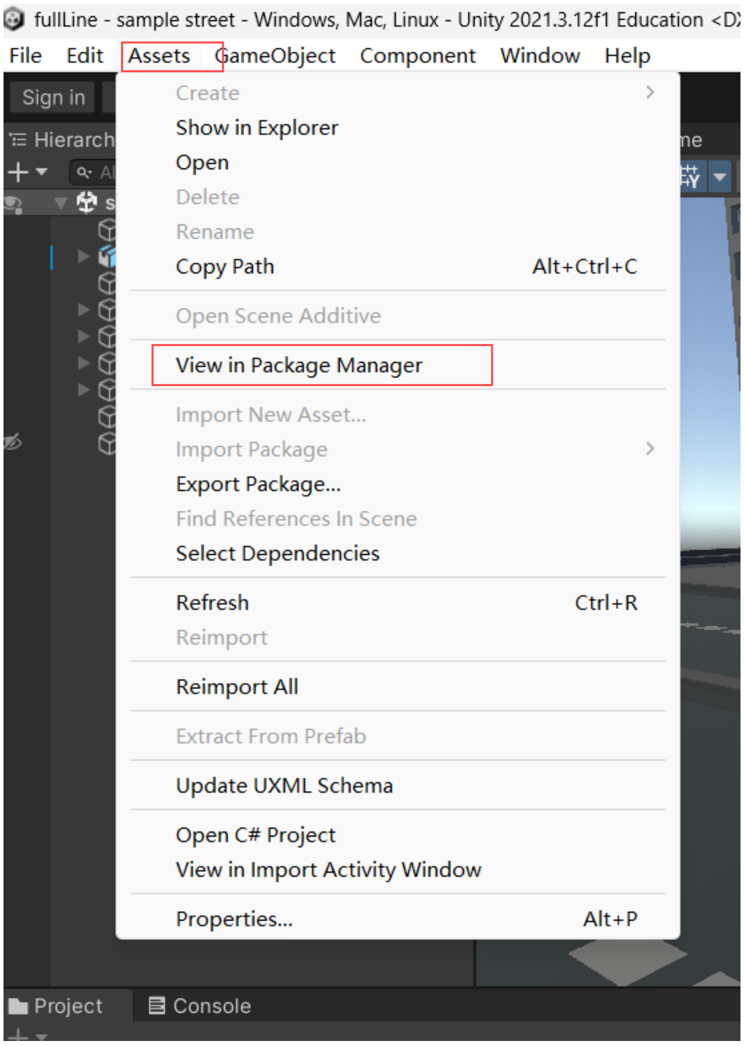
Users can also change the "Target Platform" between Windows, Linux and Mac by changing the setting in the yellow box.

If users want to explore more player settings, they can click on the "Player Settings" button in the blue box.

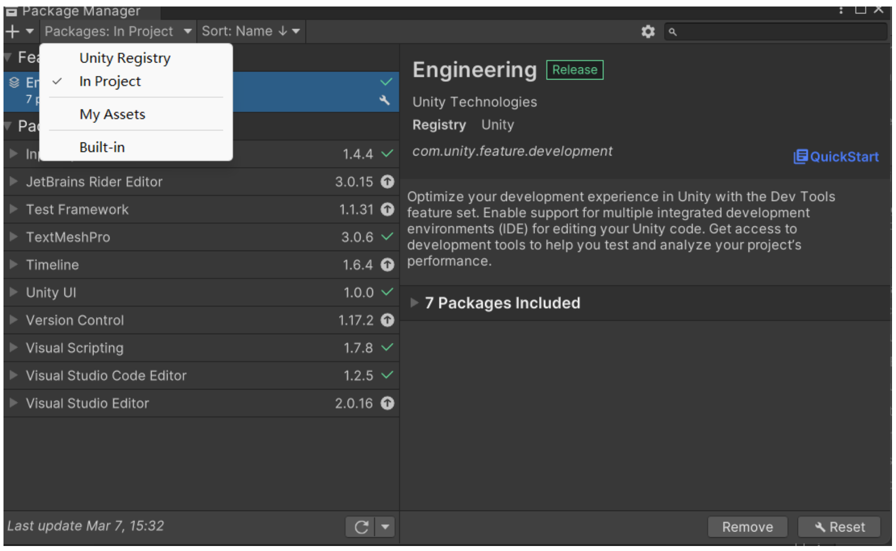


Target Platform: Windows
Architecture: Intel 64-bit
Copy PDB files
Create Visual Studio Solution
Development Build
Autoconnect Profiler
Deep Profiling
Script Debugging
Compression Method: Default
Max Texture Size: No Override
Texture Compression: No Override

Besides the platform settings and player settings, there are some plugins and packages that 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 includes all the packages that are commonly used in Unity3D and any package the user has selected to install during the installation process. On the top menu, next to the plus sign, there is a drop down menu displaying the current packages used in this project. Users can also install any package from the installation process by switching to Unity Registry. If the target plugin/package cannot be found inside the Unity Registry, users can also search for the package from the Unity Asset Store and add to “My Assets” and install from there.



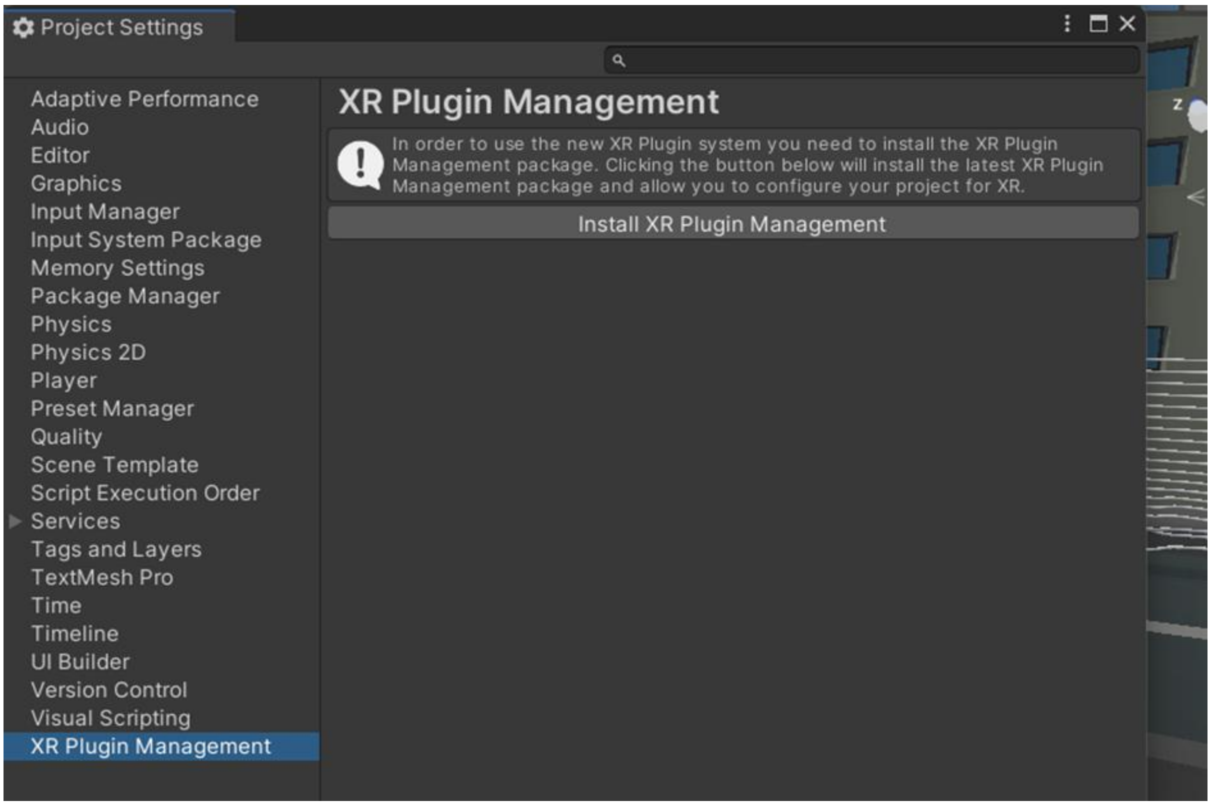
In	1.4.4 ✓
JetBrains Rider Editor	3.0.15
Test Framework	1.1.31
TextMeshPro	3.0.6 ✓
Timeline	1.6.4
Unity UI	1.0.0 ✓
Version Control	1.17.2
Visual Scripting	1.7.8 ✓
Visual Studio Code Editor	1.2.5 ✓
Visual Studio Editor	2.0.16

Unity Technologies
Registry Unity
com.unity.feature.development

Optimize your development experience in Unity with the Dev Tools feature set. Enable support for multiple integrated development environments (IDE) for editing your Unity code. Get access to development tools to help you test and analyze your project's performance.

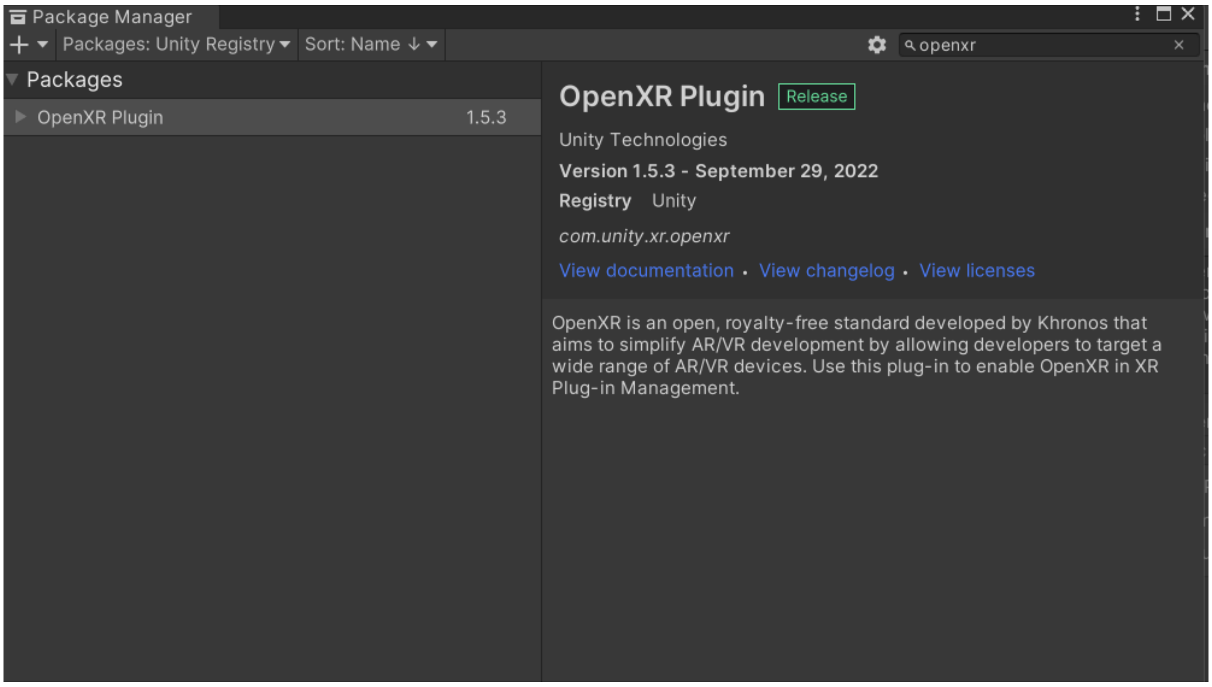
Last update Mar 7, 15:32

There are a few packages that need to be installed before users can work on a VR project. The first one is XR Plugin Management and OpenXR plugin. Just search those in the search bar and install. Some versions of Unity may need to install TextMesh Pro if you'd like to add text boxes in the scene.



XR Plugin Management

In order to use the new XR Plugin system you need to install the XR Plugin Management package. Clicking the button below will install the latest XR Plugin Management package and allow you to configure your project for XR.



OpenXR Plugin

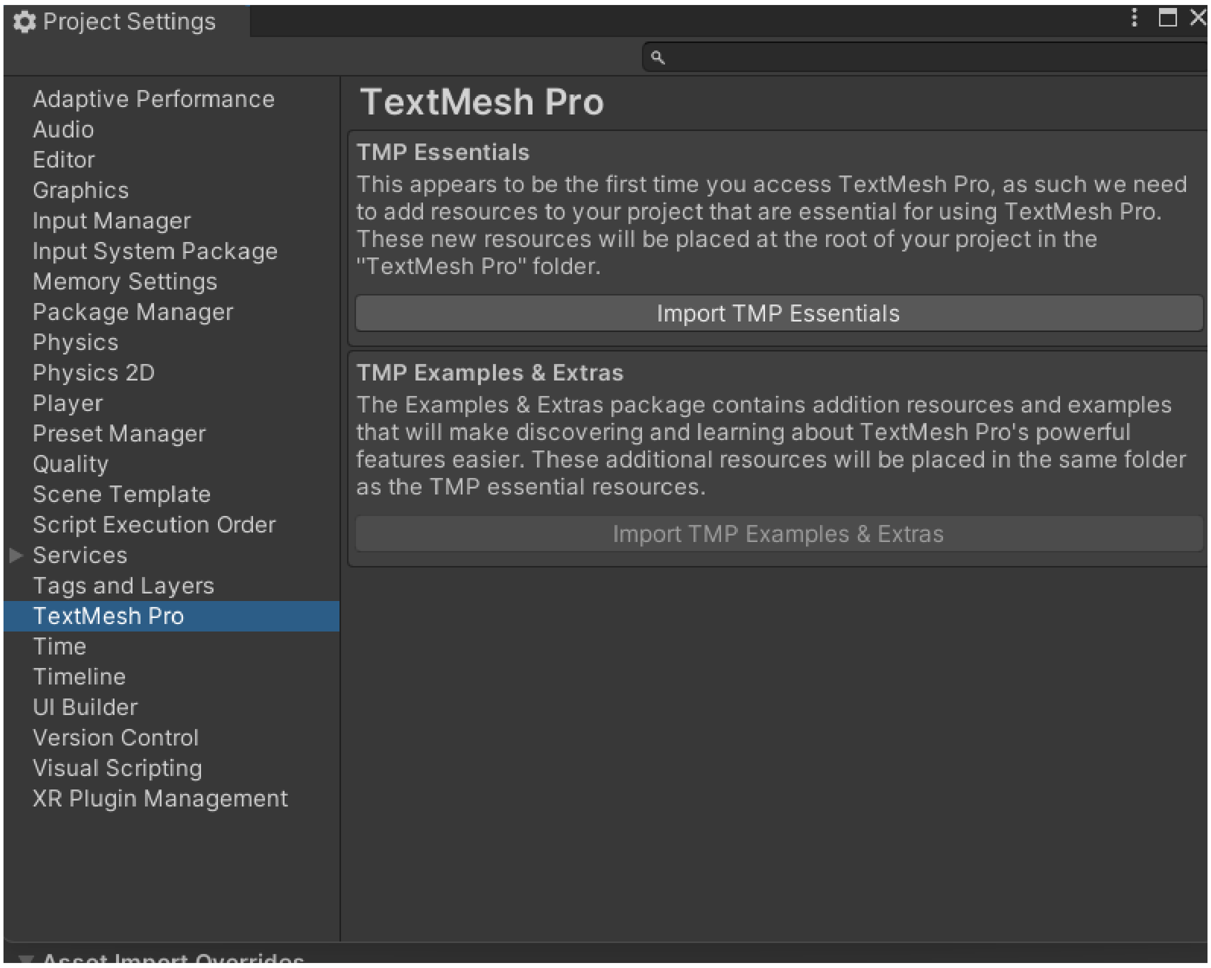
Unity Technologies

Version 1.5.3 - September 29, 2022

Registry Unity

com.unity.xr.openxr

OpenXR is an open, royalty-free standard developed by Khronos that aims to simplify AR/VR development by allowing developers to target a wide range of AR/VR devices. Use this plug-in to enable OpenXR in XR Plug-in Management.



TextMesh Pro

TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

TMP Examples & Extras

The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

If users want to add proper interaction features, one plugin that can be used is called “com.unity.xr.interaction.toolkit”. Type this into the search bar and install this toolkit. It helps users add interaction features such as grabbing and throwing. There are many ways to achieve those features and there are lots of functionalities that can be added. For other ways to make VR applications, there are a few video guides below that can also help:

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

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

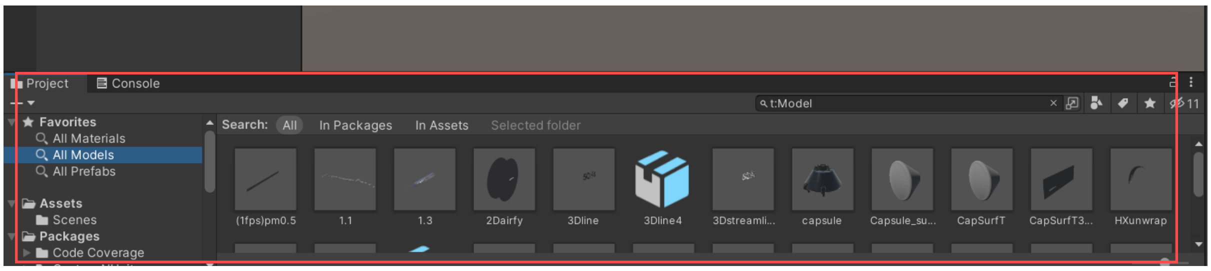
Below is a VR course from LinkedIn Learning which requires a certain subscription or paid services. But it will offer more detailed and structured learning experiences.

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 models and building models for free from the asset store with excellent quality. Credits to some of the developers who have put their work here free of charge. Final Form Studio offers lots of detailed car models from high-poly to low-poly models. PoèMe offers a few city build assets containing lots of different models that can help users 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 your own assets from other source, Unity Asset Store is the most safe and reliable source.

To create your own assets, one common way is to import a 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 shown below.



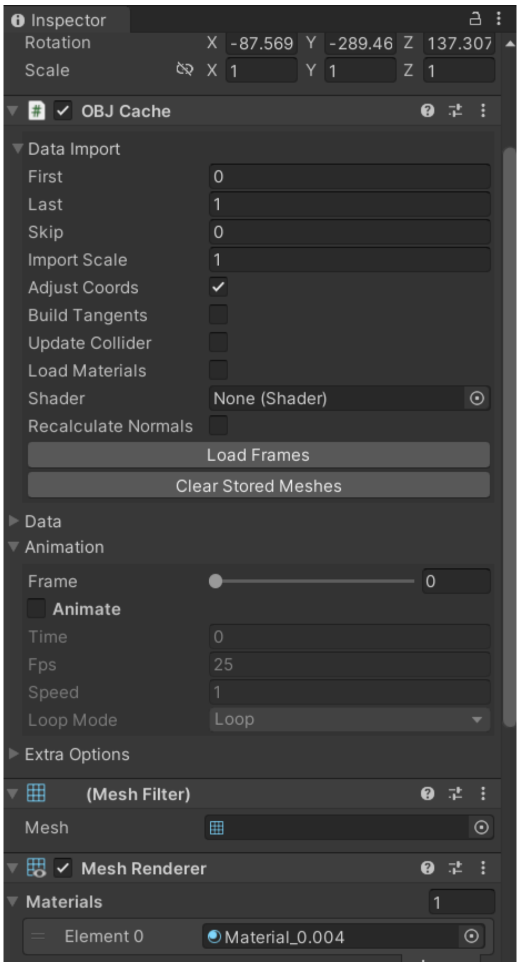
The texture file from Blender is usually saved as a picture file, and Unity cannot directly read the texture file. Users need to create a new material and add the texture file into the new material, applying the material to the asset manually.

Another way is to import an FBX file as a new asset. This format is more acceptable for other 3D modeling software. FBX files are usually smaller than .blender files and can be commonly found in the asset store. The same texture problem remains with FBX files. Users need to manually create a material file and add the texture into the material file. Then apply it to the asset.

There are multiple ways to import .obj mesh sequences. 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 accurately. The downside of this is it introduces a large work load and requires a powerful computer to run it. One tool from the asset store that is available is called Megacache and offers a basic .obj sequence reading and loading functionality. Using the scripts provided by Megacache, a window is opened 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 where you can review the asset.

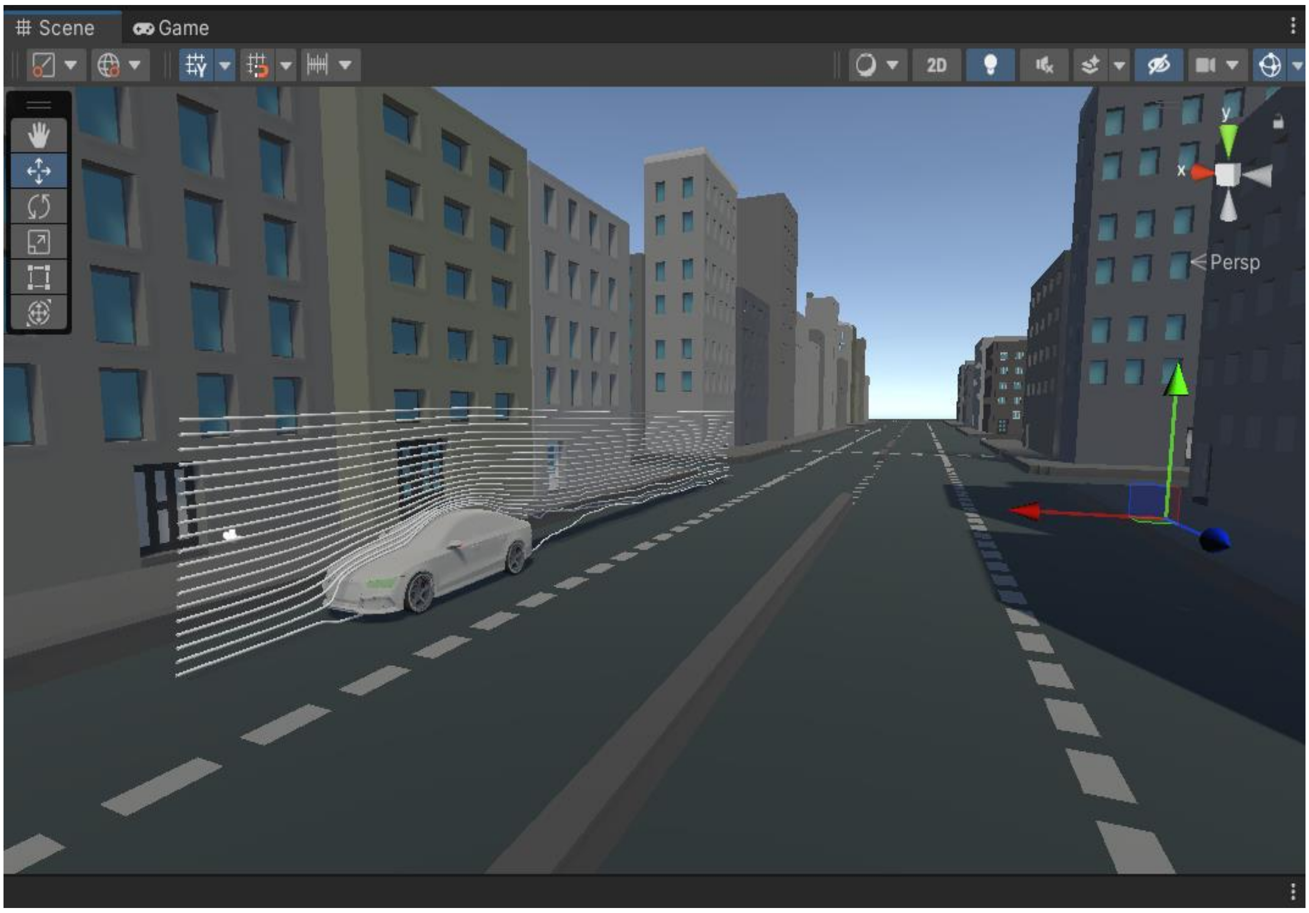
https://assetstore.unity.com/packages/tools/modeling/mega-cache-26522


```
Inspector
Rotation             X -87.569  Y -289.46  Z 137.307
Scale                X 1        Y 1        Z 1
OBJ Cache
Data Import
First                0
Last                 1
Skip                 0
Import Scale         1
Adjust Coords        ✓
Build Tangents
Update Collider
Load Materials
Shader               None (Shader)
Recalculate Normals
Load Frames
Clear Stored Meshes
Data
Animation
Frame                0
Animate
Time                 0
Fps                  25
Speed                1
Loop Mode            Loop
Extra Options
(Mesh Filter)
Mesh
Mesh Renderer
Materials            1
Element 0            Material_0.004
```

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

Project Result Demonstration

By using the workflow and tools introduced above, a CFD simulation modeling non-exhaust pollution flows is made into a 3D Virtual Reality environment with accuracy. The snapshot below shows streamlines from an animation using 200 frames with 25 frames per second. A 3D iso-surface of PM2.5 at 5% mass fraction representing the road dust resuspension, brake and tire wear is generated also as animation around the vehicle. Other models withing the VR environment contained constant velocity (cruising) and deceleration cases with 400 frames with 50 frames per second. All models including vehicle, building and road are set as real-life size and the main camera will follow VR headset movements. The whole city's floor allows user to teleport using the hand controllers. Users can also use the controller to snap turn 45-degree angles and exit the game. Users can also walk/teleport into the simulation and observe how high and how far the emission is distributed around passing and stopping vehicles.



Other visual representations can also be added into the scene. These include objects like walking pedestrians, smoke particles, audio feedback and instruction. A following camera with moving vehicle as reference can also be added to provide users with 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 need 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 fraction. Therefore, each property has its own material file, and one single time step (0.01 sec) file could have up to five material files associated to it. The .obj file sequence is stored in the computer RAM space for faster loading time; therefore, it is limited by the RAM space. Any .obj sequence larger than 4 GB will cause problems with loading time and potentially crash the software. Another kind of 3D visualization is the common Unity3D asset file called prefab. This is a packed file containing 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 sequences, material, adding building, road, and pavement, adding wheel rotation, jumping between scenes and so on… All of these need to be written into corresponding scripts. Learning C# language, debugging the code, and finding the right tutorial can be challenging for newcomers.