On the road to build an open-source VR workflow in a Linux restricted network environment.

Adrien Gomez* CEA, France

Fabien Vivodtzev† CEA, France

ABSTRACT

In this ongoing VR project leads at The French Atomic Energy Commission (CEA) , we are successfully extending Virtual Reality (VR) experiences for the analysis of Computer-Aided Design (CAD) and simulation data as close as possible to the scientist's workstation. Due to security requirements, the deployment of such a VR environment on a limited network can be cumbersome. Therefore, in this project we explore open source solutions that can be packaged to enable VR exploration of scientific data in a Head Mounted (HMD) from a restricted network without an a priori VR ready environment.

Keywords: VR environment, CAD, open-source, Unity, Monado, Godot

1 CONTEXT

Virtual Reality (VR) technologies can be used in various fields. In this work we are interested in creating a VR environment for data exploration in the field of CAD and numerical simulation. There are many frameworks like Unity [1], Ansys EnSight [2], or Unreal Engine [3]. Most of these solutions work well with different HMDs connected to middleware and drivers, mostly in a Windows operating system environment. The goal of this project is to develop a VR workflow that runs on a restricted network isolated from the Internet and running mainly on Linux. The CAD to explore and the simulation results to analyze are generated on supercomputers connected to the same restricted network. In order to offer the users, VR capabilities on their data, we are investigating solutions to set up an environment on Linux that minimizes the transfer between networks. We are also interested in open source solutions to meet the security requirements of such a restricted network. In this project, we are using a workstation with an nvidia GeForce GTX 1080 to which an HTC Vive Pro [4] is connected. Our first demonstrations with this HMD under Windows using the SteamVR software [5]. With this configuration, we successfully set up several proofs of concept, which will be discussed later. Based on these use cases, we will discuss the roadmap for deploying such VR environments in a Linux isolated network environment with various open source software such as Monado [6] and the Godot [7] engine.

2 USE CASES

2.1 CAD exploration

Our first use case corresponds to a CAD visualization of an assembly line. The 3D model contains 7000 objects and 800 000 polygons. The goal was to create a VR environment directly from the CAD file exported from the engineering office via the CATIA [8] software, without spending too much time processing it or creating a hierarchy within the VR software. Many software can be used to explore this type of CAD model, such as TechViz XL [9], 3DX [10], CAD-to-VR [11].

*e-mail: adrien.gomez@cea.fr

†e-mail: fabien.vivodtzev@cea.fr

For this use case, we used Unity [1] and various assets to load and manipulate the 3D model. First the plugin PiXYZ [12] plugin allows us to load the hierarchy of the CAD file and and create a Unity object with the right meshes and colliders to prepare the VR manipulation. Then we use the Unity plugin Interact [13] to create the scene with interactions and some default material properties for the objects.

2.2 Identifying Mesh Failures

Our second use case is to visualize a mesh and try to provide a tool to explore singularities or errors introduced into the mesh during the generation. In this project we decided to implement a specific interaction and render adapted to meshes. Thus, we implemented a mesh reader in Unity in C# and added VR functionalities using the VRTK [14] Unity asset. The implementation allows us to define material properties, rigid bodies for physics, highlighting of objects and volume definition for the colliders.

In this work, we focus on the design of simple 3D interactions for non-expert users in VR to allow them to easily get familiar with the visualization of a mesh in the HMD. To facilitate the navigation, the workstation shows the user's view and some controls to change the size and the view in case the user gets lost. We also add a simple collaboration tool between the user and the controller to allow the user to point with a laser-style ray some artifacts into the mesh. Non-expert VR users have had a very positive experience with this immersive mesh exploration. They felt comfortable in the VR scene and appreciated the high visual impact of this visualization, especially around mesh inconsistencies.

2.3 Visualization of Simulation Data

Our third use case is the exploration of simulated data with a scientific visualization pipeline to extract isosurfaces or perform volume rendering. To take advantage of existing solutions, we have used the the VR plugin of ParaView [19] named *OpenVR* (recently renamed to *XRInterface* in ParaView 5.11). The data is loaded from ParaView GUI on the workstation and the visualization pipeline is set up. The rendering view is then sent to the HMD display with several interactions adapted to scientific visualization, such as translation, rotation and scaling but also cropping and picking.

3 VR OPEN-SOURCE SETUP

The experiences described above show that VR could help scientists to understand different types of data. Even new VR users can benefit from this technology with visualizations adapted to the application domain (CAD, meshing, simulation). The various experiments also show that VR workstations should be as close as possible to the working environment in order to be easily used by the expert. A dedicated room for VR far from the user's workstation or an additional transfer between networks should be avoided as much as possible. Therefore, in this project, we are trying to reproduce such VR experiences, but on the network of the supercomputer where the data are explored.

This restricted network is under Linux, with high software security requirements, and without internet access. The use of SteamVR and Unity cannot be deployed for every user due to the security requirements. That's why we are investigating the use of open source

software running on Linux to control the HMD. We also define a pipeline to prepare the data models for the VR scene tools.

3.1 Software and Data Preparation for VR

First, a CAD file is exported from CATIA and converted with the open source FreeCAD [15] software to a glTF [16] file. The hierarchy of the complex CAD model is preserved and ready to be imported into VR software.

3.2 Packaging VR software

In order to simplify the deployment of all required VR software on the restricted network, we prepare a container using the Flatpack [17] tool. From an open network, we first configure the proxy to download the required packages. Then the home directory and download repository are set up. After the configuration is done into this container and an archive is created for transfer and deployment on the restricted network. The use of such installed software is made by the run command through the flatpack tool.

3.3 Build VR scene

As mentioned earlier, there is a lot of software available to build a 3D scene ready for VR. In this project, we focus on lightweight open source solutions. Therefore, we will use the Godot 4 engine, which is a free open source software to build desktop and mobile applications in 2D and 3D. We package the engine as described in the previous section by adding the adding the VR capabilities. This is done using the Godot XR tool [18] plugin and special shaders to enable the stereoscopic views in the engine, as explained at the end of this section.

The first step is to configure the 3D by creating a Node3D named Main, an XROrigin3D node as child of Main, an XRCamera3D node as child of XROrigin3D, and two XRController3D nodes as children of XROrigin3D for the left and right hand. In the Inspector panel, configure the Tracker parameter for the left and right hand. Then add a mesh as a child of the left and right hand to visualize them and different 3D objects like a plane mesh for the ground. Add some light and a world environment with a sky and a PhysicalSkyMaterial. Finally, add a script to ensure that the OpenXR interface is loaded successfully with the function XRServer.find interface(" OpenXR").

To activate the OpenXR tools for the scene, create a new folder in the project folder, create a new folder named addons and extract the godot-xr-tools folder into it (or download it if you have an internet connection from the AssetLib tab). Then add a VRCommonShaderCache node to the XRCamera3D. This will tell Godot to use certain shaders from the XR Tools library. With all the previous steps we successfully created a simple VR scene including the CAD model with appropriate material properties, object hierarchy and simple interaction tools.

3.4 Easy XR Driver Deployment

In order for the HMD to work with the VR workstation drivers, a middleware needs to be configured. Our first experiment was made on the open network, with a support of the HTC Vive Pro in Godot using SteamVR. Since we want to simplify software installation on the secure side we are now investigating how to use an open source alternative like *Monado* [6]. This open source XR runtime brings immersive XR experiences to workstations, mobile phones, and other devices. Monado aims to be a complete implementation of the OpenXR API created by Khronos and runs on Linux.

4 Conclusion and Future Work

Based on several VR experiments with different software and data, we successfully identify real-world applications of VR for CAD and scientific data exploration used by scientists in the laboratory. The experience shows that to enhance VR capabilities, users need to have a running VR environment as close as possible to their data. Therefore, in this project, we show how to set up a lightweight and portable VR environment on a restricted network. This ongoing work needs to continue on the driver side by replacing SteamVR with Monado to finally have a VR-ready open-source environment that can be easily deployed on a restricted network with an automated pipeline to explore complex CAD models in VR.

References

[1] Unity, Unity Technologies, https://unity.com/.

[2] Ansys EnSight, Ansys, https://www.ansys.com/products/fluids/ansys-ensight.

[3] Unreal Engine, Epic Games, https://www.unrealengine.com/.

[4] HTC Vive Pro, HTC, https://www.vive.com/eu/product/vive-pro/.

[5] Steam VR, Valve, https://store.steampowered.com/app/250820/SteamVR/.

[6] Monado, https://monado.dev/.

[7] Godot, https://godotengine.org/.

[8] Catia, Dassault Systèmes https://www.3ds.com/fr/produits-et-services/catia/.

[9] Techviz XL, Techviz https://www.techviz.net/en/.

[10] 3DX, Dassault Systèmes https://www.3ds.com/3dexperience.

[11] CAD-to-VR, Autodesk, Dassault Systèmes https://www.autodesk.com/developer-network/certified-apps/cad-to-vr.

[12] Pixyz, xxxx https://www.pixyz-software.com/.

[13] Interact, LS Group xxxx https://www.ls-group.fr/fr/interact.

[14] VRTK, https://www.vrtk.io/.

[15] FreeCAD, https://www.freecad.org/.

[16] glTF, https://www.khronos.org/gltf/.

[17] Flatpak, https://flatpak.org/.

[18] Godot XR tools, https://github.com/GodotVR/godot-xr-tools/releases.

[19] J. Ahrens, B. Geveci, and C. Law. ParaView: An end-user tool for large data visualization. In *Visualization Handbook*. Elesvier, 2005. ISBN 978-0123875822.