Geometry Simplification Methods for Virtual Reality Applications

Alexander Sharmazanashvili [1*], *Roger* Jones[2], *Alexander* Alikhanov[1], *Giorgi* Mirziashvili[1], *Kote* Tsutskiridze[1], *Ela* Abramovi[1], *Avtandil* Khelashvili[1]

[1]Georgian Technical University, Nuclear Engineering Center, 0160 Tbilisi Kostava Str. 11, Georgia
[2]Lancaster University, Department of Physics, LA1 4YB, UK

On behalf of the ATLAS Collaboration

Abstract. Virtual Reality (VR) applications play an important role in HEP Outreach & Education. They enable the organization of virtual tours of the experimental infrastructure by allowing users to interact virtually with detector facilities and describe their purpose and functionalities. However, nowadays, VR applications require expensive hardware, like the Oculus headset or the MS HoloLens, and powerful computers. As a result, this reduces the reach of VR application implementation and makes their benefits questionable. An important improvement to VR development is thus to facilitate the usage of inexpensive hardware, like Google cardboard and phones with average computational power.

Requirements to use inexpensive hardware and achieve quality and performance close to the advanced hardware bring challenges to VR application developers. One of these challenges concerns the geometry of the 3D VR scenes. Geometry defines the quality of the 3D scenes and at the same time, causes big loads on the GPU. Therefore, development methods of the geometry make it possible to find a good balance between the quality and performance of the VR applications.

This paper describes methods for simplifying the "as-built" geometry of the ATLAS detector, ways to reduce the number of facets to meet GPU performance limitations, and ensure smooth movement in VR scenes.

1 Overview

Visualization is one of the key factors for the Outreach & Education activity of the High Energy Physics (HEP) experiments. 3D visualization with advanced VR (Virtual Reality), AR (Augmented Reality), and MXR (MiXed Reality) extensions make it possible to visualize detectors' facilities, explaining their purpose, functionalities, development histories and collaboration.

The HEP visualization applications for Outreach & Education serve a large audience. Therefore, they should be extensive, easily accessible, compatible with most hardware and operating systems, simple to use, and with a well-developed user framework and open source. The browser-based software applications built on the gaming engines, bring a best fit to these requirements. As a result, there are no binary codes for the application. They will be downloaded from the server and run inside the browsers without any installations. Because browsers are compatible with the majority of hardware and software platforms, the applications will derive the same benefit. Applications will also be easily reachable. Typing the URL address in the browser will be enough to run the application.

Almost all browsers use the WebGL API [1] for the rendering of interactive 2D and 3D graphic scenes [2]. WebGL is a JavaScript API, based on OpenGL. There are several WebGL JavaScript libraries for 3D visualization in browsers - Babylon, three.js, Phaser.js, etc. They bring huge benefits because they are open-source and fast-growing platforms. They deliver powerful features of WebGL-based and physical-based 3D rendering, directional point-spot lighting, static and skinned meshes, texturing, importing of geometries, etc.

Gaming engines also bring the possibility to build a VR experience on cheap cardboard headsets by splitting the mobile device screen into two symmetrical parts and synchronizing screen updates according to the movement of the cardboard. The cardboard orientation in the space ensures the scenes' control through the portable devices' gyroscope.

At the same time, browser-based applications cause limitations in performance compared to binary applications, while the text code is interpreted during the execution by the browsers. Also, gaming engines cannot render the complex geometries of the HEP facilities or slow down the process so that the browser kills it.

Thus, important assets for browser-based visualization applications are geometries. They play a key role in the definition of the quality of the 3D scenes

* Al. Sharmazanashvili: lasha.sharmazanashvili@cern.ch

and the performance of the applications. Quality and performance are two antagonistic measurements of the application. The high quality of the 3D scenes causes low performance, and vice versa. Geometry is a factor ensuring a good balance between quality and performance.

The best immersive VR experience brings so-called as-built geometries, which express the existing real-life hardware of the detector and, therefore, have the highest level of detail. However, as-built geometries are extremely complex, which makes it impossible for gaming engines to visualize them. For instance, the as-built geometry of the ATLAS detector consists of about 45 million solid geometry primitives [3], which translates into tens of billions of triangles for the mesh geometries. Therefore, it brings the necessity for geometry simplification, ensuring a balance between the quality of the 3D scenes and the performance of the application.

The development of effective methods and tools for geometry simplification is especially important for VR applications. VR screen foresees the existence of the two frames with the two symmetrical 3D contents. It doubles the overall scene weight and issues more strict requirements for geometry simplification to find the quality-vs-performance balance.

2 Geometry Development Life Cycle

As-built geometries of HEP facilities are represented by Computer-Aided Design (CAD) 3D models in the engineering databases. They are solids geometry with the highest level of details, are editable, and are placed in widely used repositories and formats. However, CAD geometries can not be implemented by the engines directly, and therefore, geometry transformations are required. Gaming engines have internal methods for the description of the solid-based geometries. However, the way to transform CAD solid-based geometry into engine solid is unlikely to be possible because of:

1. Solid-to-solid transformations concerned with geometry structural tree translation from CAD to the engines, which always cause geometry faults.
2. Engines have limited possibilities to represent the solid geometries and are not suitable for the complex geometries of the HEP facilities.
3. Engines have very poor Boolean processors making it impossible to do cuts on complex geometries.

Thus, the only way to bring the as-built CAD geometries into the visualisation engines is to use the mesh-based geometries representing just the surfaces of the objects.

CAD platforms have internal methods for the export of the meshes from the solids. In the majority of cases, it is possible through the WRL or CGR files. However, CAD meshes can not be implemented by the engines because they don't contain information about the texture and lighting necessary for rendering. Therefore, an additional transient graphical modelling platform is needed to deliver CAD meshes to visualization engines. Good results can be delivered by the Blender software application. However, it raises another problem of importing the CAD meshes in Blender. As mentioned earlier, HEP facilities have complex geometries, and meshes can contain tens of millions of triangles and hundreds of thousands of objects. Blender fails to read such kinds of meshes - in most cases, crashing or spending tens of hours reading the geometries.

Below is the case study of the *Scene-2* geometry of the VR application, which represents the Muon Barrel region of the ATLAS detector in the Cavern (Fig.1).

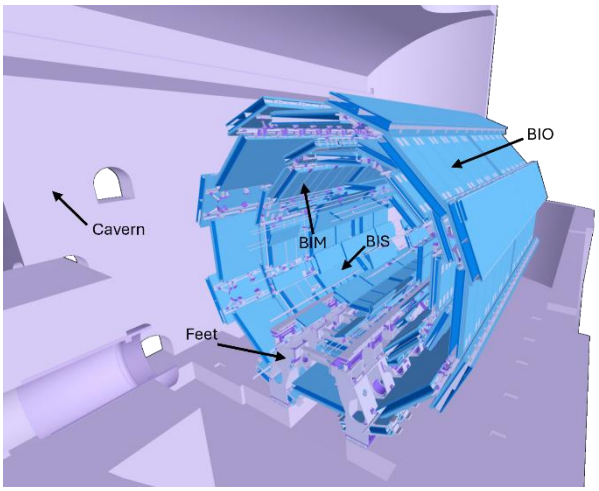

Fig. 1. *Scene-2* of the Muon Barrel region in VR

It consists of the objects of BIO Outer Chambers, BIM Middle Chamber, BIS Small Chambers, Feet, and Cavern [6]. The overall characteristics of the CAD mesh are given in Table 1.

Table 1. Overall parameters of *Scene-2*

	Triangles	Objects
BIS Small Chambers	149'600	6'544
BIM Middle Chambers	825'344	17'400
BIO Outer Chambers	45'962'224	163'155
Feet	9'760'647	6'175
Cavern	140'000	529
Total:	**56'837'815**	**193'803**

For this amount of triangles and objects, Blender crashes when reading the mesh. Just for one component - BIO Chambers with the amount of triangles 45'962'224 and 163'155 objects Blender successfully read the mesh geometry. This task took 14 hours of the Intel core I7 4770/RAM 8Gb/Win11 machine and used 50% CPU load with 8Gb RAM.

Thus, an additional graphical platform is needed between the CAD and Blender. Good results are delivered by Keyshot application[7].

The final life cycle of geometry development used for the browser-based VR application is presented in Fig. 2. CATIA, as a CAD platform, is responsible for meshing - the creation of the vertex matrix with *xyz*

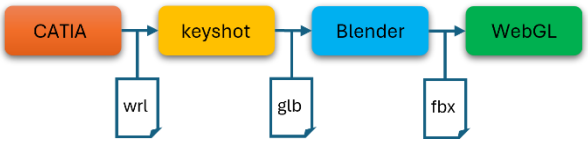

Fig. 2. Geometry development life cycle

meshing coordinates from the solids and all necessary geometry transactions, and generation of the output in WRL.

- Keyshot is responsible for triangulation, normals generation, and tesselation. Output generated in GLB.
- Blender is responsible for UV mapping, texturing, and lighting. Output generated in FBX.
- WebGL/three.js is responsible for rendering and representation of geometry in 3D scenes.

However, geometry transformations in the chain bring additional problems related not to geometry faults but to mesh structure. It causes an increasing number of triangles in the mesh and as a result, makes geometries heavier. During the tesselation and handling of the imported geometry data, many vertices are duplicated or added alongside the edges. Therefore, additional steps for instant cleaning should be done on the imported meshes. These steps include:

- Removing the duplicated vertexes
- Deleting the unused materials
- Simplifying the mesh geometry
- Fixing the normals
- Removing the loose geometry
- Cleaning up the mesh topology.

These steps reliably bring CAD meshes into the WebGL/three.js graphical engine for visualization. However, it is not enough because CAD meshes contain as-built geometry with a lot of detail. They are much heavier than engine limitations. The above-described *Scene-2* has about 57 million triangles (Table 1). VR application duplicates screens. Thus, the presented geometry of *Scene-2* exceeds by 30 times the limitations of the WebGL/three.js engine. Therefore, after successful import in Blender, the geometry should be simplified to find a good balance between the quality and performance of the VR applications.

3 Geometry Simplification Methods

There are several ways to reduce the number of triangles in the description. One way is to increase the approximation and reduce the quality of visualisation. Then, for each screen resolution, the optimal approximation value has to be chosen. Also, an essential parameter for selecting the approximation is the zooming rate, which describes the maximal zooming coefficient of the scene. Therefore, the approximation should be slightly visible for the maximal zooming rate of the scene. The good approximation value should express the excellent balance between the number of triangles in the scene and the quality of visualization of the scene.

Another way to reduce the number of triangles strongly is to remove the holes and circular parts and replace rounded profiles with linear ones. So, most surfaces must be described by the quads (2 triangles). The optimal approximation value should be found for the surfaces where such modification is impossible.

The third way of reducing the number of triangles is to rate the components of the detector according to their visibility in the visualization scene. Dimensions of detector components vary from tens of centimetres up to the tens of meters. For instance, the Pixel detector in the ATLAS detector [4] has dimensions (L=1.8m/D=0.245m) in comparison to the Tile Calorimeter [5], which is the middle part (L=11.4m/D=8.6m) and Muon detectors [6] (L=30m/D=24m). The imported geometry of the pixel detector contains 356’000 triangles. Thus, there is no reason to import detailed geometry descriptions of the Pixel detector because it will be almost invisible because of its comparatively small size. At the same time, it dramatically increases the overall number of triangles in the scene.

Therefore, for the given maximal zooming rate of the scene, some components will stay in the scene less visible because of their small dimensions (D_I/L_I in Fig.3). Therefore, there is no need for details, and primitive geometries can describe them. Components with a medium visibility rate (D_{II}/L_{II} in Fig.3) will have some details, and it is better to describe them by the envelope geometry. The last category of components with large dimensions (D_{III}/L_{III} in Fig.3) will have the highest visibility rate; most likely, it is better to implement detailed geometry descriptions for their import. Finally, all detector components can be grouped into three categories according to their visibility rate. The recommended type of geometry description for the import can be assigned to each detector component (Fig.3).

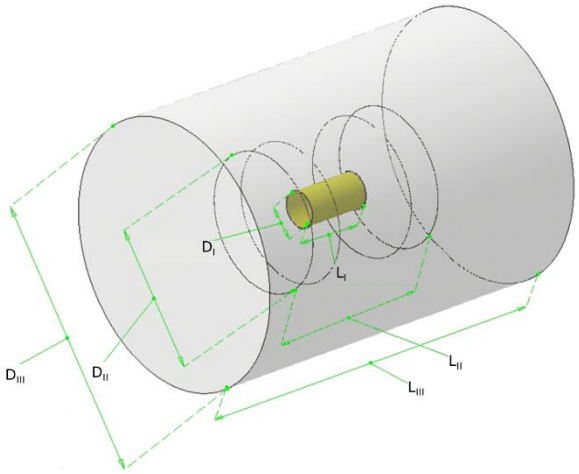

Fig. 3. Visibility categories of detector components

Geometry simplification methods are described in the book Ref. [3]. The method should include simplification steps, considering all the factors mentioned above, playing an essential role in reducing

the number of triangles. The geometry description simplification method with seven consecutive steps is described below.

STEP#01: PARAMETERS DEFINITION

The first step for simplification is the definition of values of general parameters of the visualization scene, which will impact the further steps of simplification. These parameters are:

1. R_{sc} - the screen resolution. Preliminary, should be decided on the purpose and domain of the application. The browser-based VR applications are run on portable devices. Therefore, the R_{sc} parameter will correspond to the screen resolution of the mobile phones - 360 x 640 pixels.
2. Minimal dimensions of the components, that ensure their visibility in the scene. Two parameters can be considered - the minimal diameter D_{min} because most of the components of the detector have cylindrical shapes and the minimum length of the components is L_{min}. Dimensions of the detector components differ from tens of meters down to centimetres. Therefore, the application developers and user community decide what components are not essential to visualize because of their sizes. As a result, all the components with dimensions less than D_{min}/L_{min} will be deleted from the geometry.
3. Z_{max} - maximal zooming rate coefficient of the scene describes the maximal scale of the visualization of the components. The Z_{max} value should be defined according to R_{sc} and D_{min}/L_{min} values.

STEP#02: REMOVAL OF THE COMPONENTS

Removal of the components with dimensions less than D_{min}/L_{min}. Because D_{min}/L_{min} sets the minimal dimensions of the components to ensure their visibility in the scene, all components less than those values should be deleted. For this procedure, editable 'as-built' geometry descriptions are needed. They will be modified by erasing the parts with dimensions $<D_{min}/L_{min}$.

STEP#03: REMOVAL OF THE PARTS

Removal of inessential parts from the geometry description. 'As-built' geometry descriptions of the detector contain the whole spectrum of parts and sub-assemblies of the different purposes - main components of facilities; services - cooling pipes, electrical boxes, cable trays, pumps; mechanical, support and access structures and many others. Depending on the VR scenes, some categories must be deleted. For instance, if the main purpose of the VR scene is a visualization of the main components of the detector, then all other types like services, mechanical structures, etc., must be deleted. If the purpose is a visualisation of mechanical engineering facilities, then components of the detector can be presented in a very generic way through the primitives.

STEP#04: REMOVAL OF THE HOLES

Holes in the geometry descriptions dramatically increase the number of triangles. This happens because each hole is a set of lines with a number of vertices. In facet-based geometry descriptions, all vertices are considered equally. There is no difference between the vertices of the hole or edge, etc. Triangles are created for all the vertices. As many vertices are on the surface, so are many triangles describing the surface. Fig.4 shows surface representations with and without the hole. Surfaces A and C have nine vertices. Eight triangles represent each surface. Surface B has eight vertices on the edge and 56 vertices belonging to the hole. As a result, surface B is described by 64 triangles. So, the hole's presence increases the number of triangles by a factor of 8. Because of this, it is necessary to remove holes from the geometry description as much as possible. Only exceptional purpose holes describing essential functionality or facility structure features should remain. Usually, detector facilities contain many holes, and their removal is a critical step in simplifying the geometry.

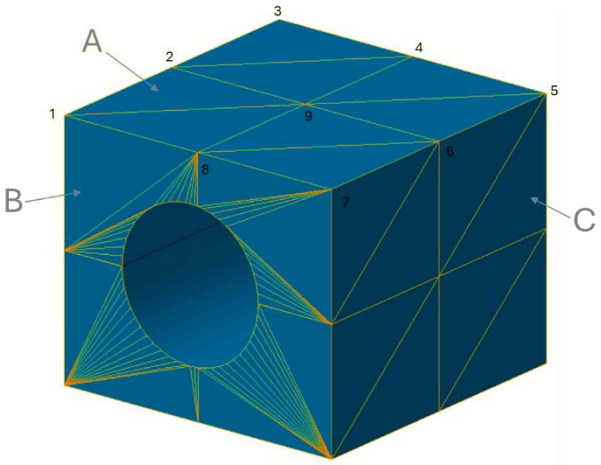

Fig. 4. Facet representation of the surface with the hole

STEP#05: TRANSFORMATION OF THE ROUNDED PROFILES

Transformation of rounded connections to straight-line connections. The 'As-built' geometry of the detector has many construction elements for mechanical engineering purposes. The mechanical engineers add them for the construction, design, manufacturing, and installation. One of the categories of those constructive elements is the typical joining of surfaces. The galettes and chamfers are usually used as a standard constructive joining element. The existence of such features in the geometry description is the source of the considerable increase in the number of triangles in the scene. Fig.5

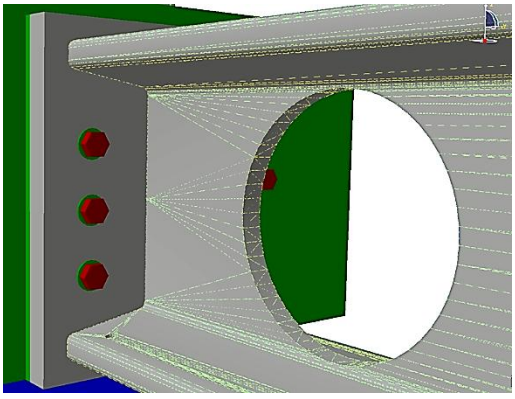

Fig. 5. Triangles on surface with rounded connections

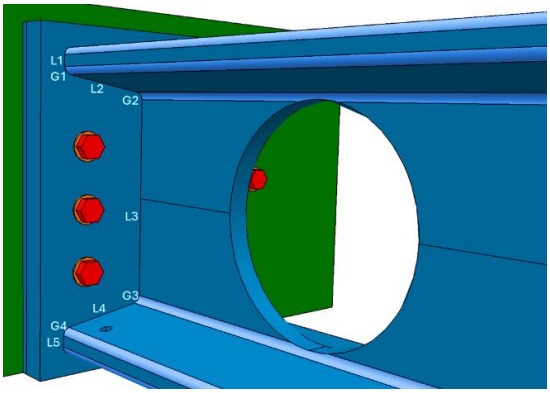

Fig. 6. Galettes on the typical joining

shows triangles on the surfaces with rounded connections of the Feet construction of the ATLAS detector. They have to be replaced by straight-line connections. Fig.6 illustrates the number of galettes used for joining the straight-line profiles on the joining of the surfaces - G1 on the union of the L1-L2; G2 - for the L2-L3; G3 - for the L3-L4 and G4 - for the L4-L5.

Each galette presented in the geometry is not an independent solid, but they are part of the profile sketch inside the one body (Fig.7). Therefore, their modifications are possible through the editing of the sketch geometry. It is more complex work than the removal of the holes considered above. Sketch geometry, as usual, in addition to primitives, also has several geometry constraints. Therefore, galette modification or removal requires first their isolation and then editing.

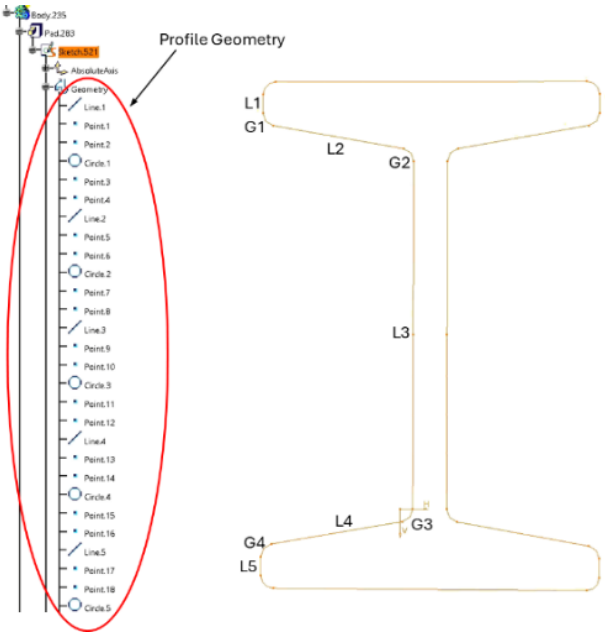

Fig. 7. Sketch profile with galettes in CATIA

STEP#06: TRANSFORMATION OF THE COMPLEX SURFACES

HEP detector facilities are characterized by the massive repetitiveness of the components. There are complex components built by identical subcomponents. Depending on the visualisation purpose, those subcomponents can be replaced by one generic description. For instance, the TRT EndCAP disks of the ATLAS detector have assemblies with eight disks inside of each. Therefore, it is possible to replace eight disks with the description of 1 generic disk. As a result, each assembly will have one disk instead of 8, and the number of triangles for the assembly in the facet-based representation will be cut down by a factor of 8.

STEP#07: SELECTION OF THE APPROXIMATION VALUE

The approximation value dramatically impacts the number of triangles in the scene and the quality of the visualization scene. As significant is the approximation value as triangles in the scene and vertices are visible, and vice-versa. Fig.9 shows the same solid as in Fig.5 but with a different approximation value. The approximation value in Fig.5 is l_1=0.25mm and in Fig.9 - l_2=5mm, 20 times greater than l_1. As a result, surface B for l_1 is described by 64 triangles. The 30 triangles represent the same surface for l_2 (Fig.8).

Therefore, increasing the approximation value by a factor of 20 cuts twice the number of triangles. At the same time, in Fig.5, the whole surface is smooth; in Fig.9, the same surface is rough, with visible vertices.

Thus, the approximation value should express the excellent balance between the number of triangles and the quality of the visualization. The critical parameter for choosing the value of l is the Z_{max} - maximal zooming rate coefficient of the scene, defined in STEP#01 of the simplification. There are no typical numerical values for the dependency l=f(Z_{max}). The visualisation application developers define them according to the user's requirements.

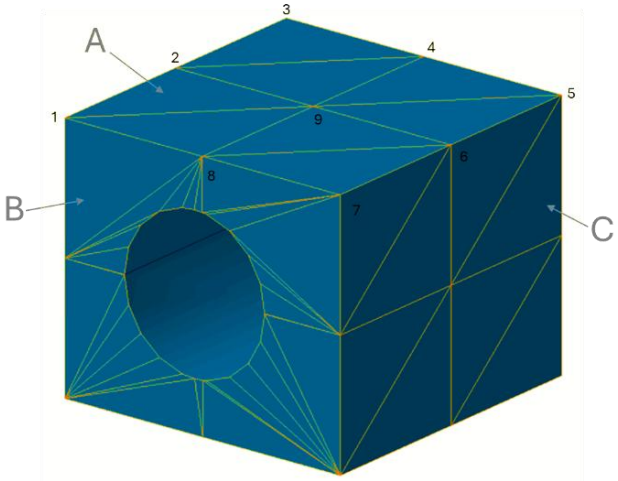

Fig. 8. Hole with approximation l_2=5mm

4 Browser-Based VR Application

The geometry simplification method was implemented for the development of the browser-based VR application Tracer/VR (https://tracer-vr.web.cern.ch). Four VR scenes were developed from the CATIA as-built geometry passing through the chain described in section 2. The parameters of scene 2 considered above in Table 1 were modified and presented in Table 2.

Table 2. *Scene-2* before and after simplification

	As-built	Simplified
BIS Small Chambers	149'600	22’697
BIM Middle Chambers	825'344	29’864
BIO Outer Chambers	45'962'224	23’152
Feet	9'760'647	9’660
Cavern	140'000	9’108
Total:	**56'837'815**	**94’481**

Tracer/VR uses Google Cardboard virtual reality headsets, enabling them to start the application on the mobile phone and accommodate it. The application causes moderate loads on the 3D scenes and smooth movement on the average power phone. The camera in the VR scene moves on a fixed path, and users can control the scene by head movement using the gyroscopic control of the phone.

Selection of the VR scene is possible before starting the VR experience in the headsets (Fig.9). After that, the application divides the mobile device screen into two synchronized screens for the headsets.

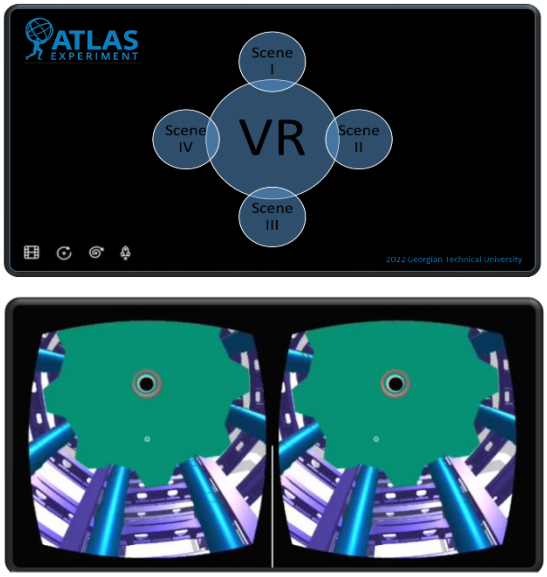

Fig. 9. Interface of the Tracer/VR application

Conclusion

1. Geometry is an important factor in finding a good balance between the quality of the VR scenes and the performance of the application.
2. Geometry simplification implies the transformation of the as-built mesh geometries into meshes with minimal required, from a scene quality point of view, the number of triangles.
3. The geometry transformation chain from the CAD platform includes several graphical platforms and steps for mesh corrections.
4. Geometry simplification methods enable us to receive a good VR experience on the average power of mobile phones and cardboard headsets.

References

1. Daniel Carson, WebGL: Graphics and Animation. ResearchGate April 2020 DOI:10.13140/RG.2.2.18538.95683
2. Mert Bal et al., Using WebGL in Developing Interactive Virtual Laboratories for Distance Engineering Education. American Society for Engineering Education (2017)
3. A. Sharmazanashvili, Geometry Modelling in HEP. Book, CERN library **504**, (2022) ISBN 978-9941-8-5034-9
4. G. Aed et al. (The ATLAS Collaboration), ATLAS Pixel Detector Electronics and Sensors. Journal of Instrumentation, vol.3 July (2008)
5. G. Aed et al. (The ATLAS Collaboration), Operation and Performance of the ATLAS Tile Calorimeter Run 1. European Physics Journal C78 **987** (2018)
6. J. Snuvernink, The ATLAS Muon Spectrometer: Commissioning and Tracking. University of Twentem (2009) ISBN 13:978-90-365-2912-9
7. Jei Lee Jo, KeyShot 3D Rendering. Packt Publishing Ltd, 2012 ISBN: 978-1-84969-483-4