

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

5,800

Open access books available

142,000

International authors and editors

180M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



3D Computer Graphics and Virtual Reality

Branislav Sobota and Miriama Mattová

Abstract

This chapter is dedicated to the description of 3D computer graphics used for the needs of virtual reality. Virtual reality (VR) is the use of computer technology to create a 3D virtual environment. The chapter presents some graphical features used in an environment as well as an explanation of good design practice. The chapter contains also the description of lighting settings, 3D objects/models and virtualization sequence, camera, and scenes where the wheelchair simulator is used as an example of the implementation environment.

Keywords: 3D graphics, 3D objects, virtual reality, web-based collaborative environments, lighting settings, object geometry, texturing

1. Introduction

Technology is advancing in the directions in which it eases life. In the transport industry, it helps to get from point A to point B faster. In the education industry, it helps to obtain knowledge in a more effective way. In the health industry, it helps to operate or diagnose easier. Those are very few examples of how technology is simplifying life. The process of technology advancing is defined in a few steps. The first one is the curiosity about how things work. The second is the willingness to learn. The third is understanding and the fourth is experimenting. The creation of 3D computer graphics had a similar process. Firstly, there was a curiosity about how people see and percept the world. Secondly, it was a willingness to learn how the visual information from the outside world is getting into person. From that point, people realized that eyes were the ones to blame. Willingness to learn continued in this area and later on there was an understanding. The understanding of how the eyes work helped to create the first photographic camera. Lastly, there was experimenting in this field which led to the first photographs. As it is the nature of humans, we wanted to achieve more. We wanted to make those photographs to be animated and displayed somewhere. The development of this technology has gained rapid momentum. Soon, people were able to watch a short movie on canvas. With an introduction of the first computer, those two areas soonly merged. Before that, there was a need to represent data calculated on computer. Soon, it was represented on a simple display. As computers advanced and were more complex, there was a need to represent calculated data more effectively. Computer monitors were introduced and computer graphic was born. Before it

reached the possibility of 3D displaying, 2D scenes were presented. The most popular one was television. This is the greatest example so-called eye (visual) and computer content. As people realized how effective and entertaining computers can be, they started to wonder if we can simulate our world. Information from the computer area, eyes (visual) information and mathematics combined, and 3D computer graphics was born.

3D graphics opened a new world of experimenting in the area. As we live in a three-dimensional world, it gives us another level of possibilities of how to simulate and improve the virtual world-virtual reality. Representation of virtual reality via 3D technologies gives more authenticity and credibility. Virtual reality (VR) and related technologies are penetrating almost every aspect of our lives [1]. Nowadays, it is one of the most developing technologies that could significantly increase the level of interactivity [2]. This is greatly used in every industry. It has a great impact on health industry as it is possible to create projects like wheelchair simulators for people who are temporarily or permanently immobilized. Those people are provided with a number of courses to gain wheelchair skills but it is often financially unavailable. Virtual reality was proven as an effective tool for disabled people [3].

This chapter is dedicated to the description of 3D computer graphics used for the needs of virtual reality. The wheelchair simulator is used as an example of the implementation environment. Wheelchair simulator was built using web technologies via A-frame framework [4]. It is a simulation of a collaborative environment in VR (developed in authors' home laboratory LIRKIS at Department of Computers and Informatics, Technical University of Košice) where an immobilized patient should learn using an electric wheelchair. Guest can join the same 3D virtual environment and watch patient deal with the obstacles. Whole simulator contains multiple scenes with different obstacles. The pilot project was also implemented in a unique 3D virtual LIRKIS CAVE environment (**Figure 1**).



Figure 1.
Patient on wheelchair in 3D virtual LIRKIS CAVE environment.

2. Related work

There are many works related to 3D computer graphics in virtual reality. Some of the works focus on different virtualization algorithms, others try a new methodology of improving realistic look while maintaining the smoothness of application. Related work below simply focuses on new methodologies or approaches to 3D computer graphics or comparing them.

Chen in [5] describes the rapid creation of photorealistic virtual reality content with consumer depth cameras. Work focuses on the demonstration of a complete end-to-end pipeline for the capture, processing, and rendering of view-dependent 3D models in virtual reality from a single consumer-grade RGB-D camera. The result of this pipeline is a 3D mesh with view-dependent textures suitable for real-time rendering in virtual reality. Progressive feedback point cloud rendering for virtual reality display is the aim of [6]. Work presents a novel approach to a progressive feedback-driven rendering algorithm. This algorithm uses reprojections of past views to accelerate the reconstruction of the current view. The presented method is tested against previous methods, showing improvements in both rendering quality and interactivity. In the [7] is indicated virtual content creation using dynamic omnidirectional texture synthesis. This work proposes optimization that synthesizes a high resolution view-dependent texture map for any virtual camera location. Synthetic textures are generated by uniformly sampling a spherical virtual camera set surrounding the virtual object, thereby enabling efficient real-time rendering for all potential viewing directions.

The paper [8] describes three views of virtual reality: non-immersive virtual reality, and it discusses the advantages and applications of non-immersive VR systems. Immersive and non-immersive VR systems are compared and hybrid possibilities are reviewed.

Kautz et al. [9] present new algorithms and techniques for the acquisition and real-time interaction with complex textured 3D objects and shows how these results can be seamlessly integrated with previous work into a single framework for the acquisition, processing, and interactive display of high-quality 3D models. In addition to pure geometry, such algorithms also have to take into account the texture of an object and its reflectance behavior. The research [10] presents a novel 3D user interface for the immersive design review. This research focuses on the development of a novel immersive user interface, a smart 3D disk with a set of widgets. During the immersive sessions, the user can activate functionalities using cost-effective pointing devices and the conceived 3dUIs projected into the more complex CAD functionalities such as surfaces shape modification.

3. 3D graphic in VR

3D computer graphics are graphics that use a three-dimensional representation of geometric data that are stored in a computer for the purpose of performing calculations and rendering 2D images. Those geometric data are not graphics until they are presented on a screen. Geometric data can be perceived as objects or models. To build up a 3D world in virtual reality, it is needed to include graphic elements such as:

- *Lighting*: Lighting settings give the light in the scene, manage shadows, light sources as well as their intensity.
- *3D objects*: 3D objects or models are a base to build up a scene. It defines, what the scene will contain in the largest spectrum.
- *Camera*: Camera manages the process of visibility, culling, and rendering of 3D objects on the screen.
- *Scenes*: Scene is a space where all elements above are put. It also defines the size of space in a virtual world.

A combination of all mentioned graphic elements can result in scene as shown in **Figure 2** left. Another 3D scene can be seen in drone simulation environment represented in **Figure 2** right.

3.1 Rendering

Rendering can be perceived as the final process of creating the final 2D image from the prepared scene. These images are played at some speed rate. It is called frames per second (FPS). This variable indicates how many final 2D images are rendered in 1 second on the screen. The higher the value, the smoother is playback image. FPS can be used as final control of how great was handled graphics optimization with a given graphics processing unit (GPU). It is important to handle draw calls effectively as it results in smoother applications. Rendering can be divided into real-time rendering and non-real-time rendering.

Real-time rendering is used in interactive media such as games and simulations. It means final 2D images are being processed in real time, according to where the camera is positioned and how it is rotated in the scene. In real life, movement of a person looking around the room can be random as he/she is moving head is not defined speed or rotation. The same logic applies to a camera in media. Therefore, there is no algorithm to predict the exact movement of camera, as it is random input from user, for final 2D images to be calculated beforehand. Using this method, it is needed to keep in mind that human eyes need at least 24 frames per second rendered for a successful illusion of movement. In VR headsets, frames per second must be quite higher, otherwise, it is possible to notice black spots when rotating the head. Black spots around the edges can cause the loss of the illusion of being in a 3D environment.

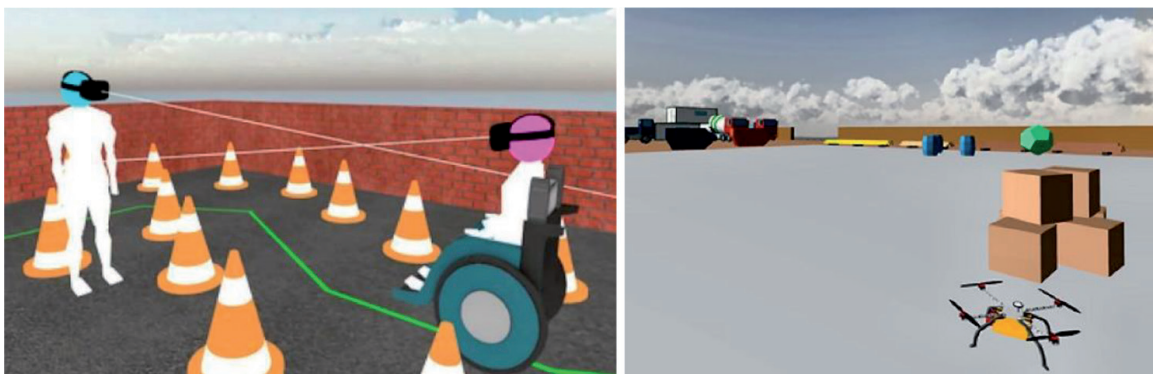


Figure 2.
Example representation of the wheelchair simulator scene and the drone simulation scene.

Non-real-time rendering is used for such media in which it is possible to predict the sequence of images. These are non-interactive media such as film or video. It can take much more time to render in order to obtain higher image quality.

Speed of rendering depends not only on effective optimization of application but also on hardware specification of device like GPU, CPU, and memory, for example. This is a reason why the same application can be smooth in desktop computers but not in VR standalone headsets. Therefore, there is a need for 3D graphics knowledge and hardware specification knowledge to optimize application if it is built for multiple platforms.

3.2 General optimization

General optimization in 3D applications consists of multiple steps. But firstly, it is important to realize what is the desired result. To call application optimized, it is desired to achieve a count of draw calls as small as possible, limit the unnecessary calculations or optimize complexity. This can be achieved through several steps, for example:

- *Light optimization*: Uneffective placement, settings, and types of lights sources can greatly impact calculations during rendering which results in lower FPS. Next Section 4 focuses on these parameters as well as implemented lights in wheelchair simulator.
- *Simplifying objects*: It includes creating lots of details (LOD), having an object with no unnecessary polygons, and using adequate resolution of textures (Section 5).
- *Camera settings*: Since camera view is the final view of what user will see, it is important to pass to the renderer only information that needs to be handled in real time and to prevent all the unnecessary information to be calculated (Section 6).
- *Scene logic*: If it is not necessary for application logic, scenes should be divided into smaller scenes. A wide scene with open field can cause computational load, therefore it is recommended to reconsider scene logic. Section 7 focuses on scene design in wheelchair simulators.

4. Lighting setting

Lights in 3D virtual environments are necessary for objects to be visible. The same logic applies in real world. Without light, it is not possible to see your surroundings. There are several types of lights, but most common are *directional light*, *point light*, *spotlight*, and *ambient light*. The difference between these lights is ways of lighting up a specific area, where a light source and lighting direction is located. Lighting itself works basically the same. Light has defined its source, direction, intensity, and area, then this parameter defines the vectors which are colliding and bouncing from the objects in dependence on how close the light source is, how strong light intensity is, and area of light up space. Wheelchair simulator uses ambient light in all scenes. **Figure 3** represents scenes with ambient lightning.

One of the scenes is designed to be at night. Scene is called crosswalk-night. As is mentioned above, ambient light is everywhere and it illuminates every object evenly from every side. **Figure 4** shows the scene.

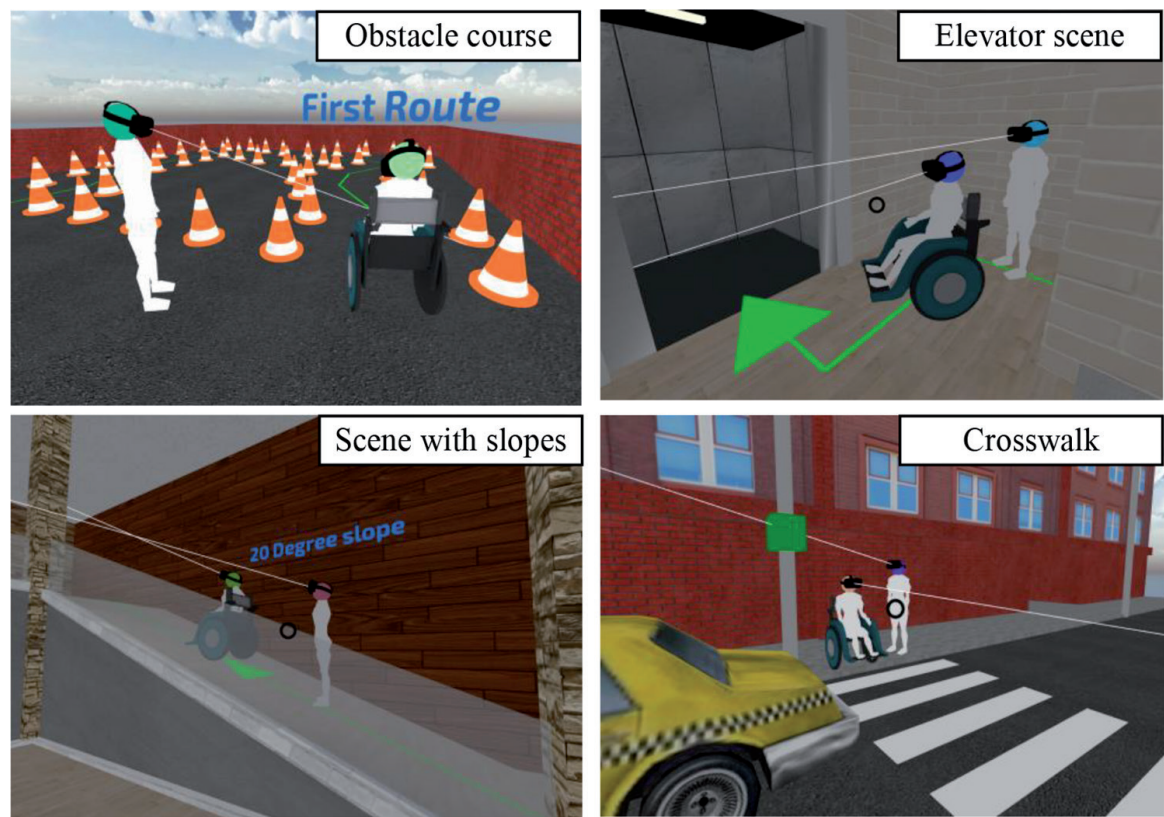


Figure 3.
Four out of five scenes with ambient lighting.



Figure 4.
Representation of crosswalk-night scene.

It is possible to observe that besides ambient light, fog is implemented as well. Fog allows to shade the objects at a certain distance in a predetermined color. It is implemented on camera of the user and not as a light object. Therefore, fog is not perceived as a lighting setting but as a shading setting. In this case, distance of shading is few meters from the user, and color of shading is black. This way it is possible to simulate first-person view at night, samely as in real life. From **Figure 4**, it is also possible to see the red traffic light, which signals that patient cannot cross the crosswalk as cars are moving. The red color on a box may seem a lot lighter than surroundings. It is because of emissive intensity of material on a traffic light was set on a higher value to make it

seems lit. This way, it is not necessary to add the point light in the environment and it will lighten the computational core. Details are explained in Section 3.3. But it is necessary to keep in mind, if priority is to develop application with high rate of frames per second, it can affect the realistic look of the scene. Especially, for virtual headsets as their computational cores are not so advanced yet. **Figure 5** shows the traffic light with an implementation of point light in red color and with shadows settings.

4.1 Shaders

Thanks to specific light definition, it is possible to observe illuminated and unlit parts of an object. Light has an exact border where the illumination of object ends. Therefore, it is needed to shade the transition between illuminated and unlit parts to make it appear more natural. Shader will make a transition by adjusting the color of the nearest pixels between these two sections. **Figure 6** shows the illuminated part, shaded part, and unlit part of the cone illuminated by point light only.



Figure 5.
Traffic light with point light in red color and shadow settings.

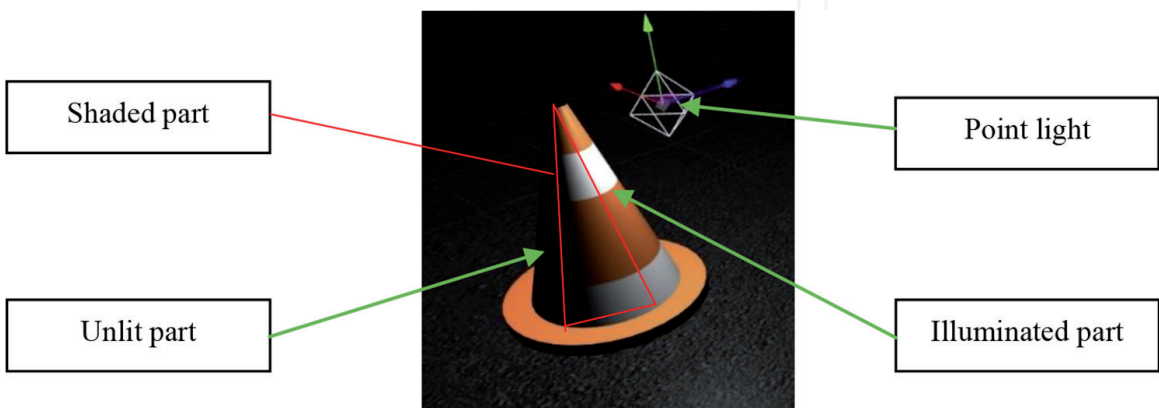


Figure 6.
Shader represented on a cone with a point light.

4.2 Lights combinations and their complexity

All mentioned light types can be used at a same time and multiple times in a scene. **Figure 7** shows implementation of spot light, which is red, point light which is blue, and directional light which is white.

All lights presented in **Figure 7** are casting shadows. All objects are casting and receiving shadows as well. This is an example of how light can simulate the real-world behavior of lighting. Even though it looks natural and realistic, it is needed to keep in mind that there is a lot of calculation done in the background. It is possible to observe that blue light is illuminating the shadowed part on a terrain from the red light as well as the not shadowed part. Directional light illuminates area of red light and blue light at once but also creates shadows on a left from the cones. All lights are penetrating areas of other light, therefore, renderer needs to consider priority of which light it will display. It is possible to bake these lights at the same time, but calculation complexity will greatly improve. Vectors from lights are crossing paths, which mean they are affecting each other. Therefore, they need to be recalculated to give a result similar, if not same, to a real world. Simply said, the more lights are used in a scene and the more they are penetrating into areas of other lights, the higher complexity lighting setting will have. If renderer needs to take shadows into account, it needs to calculate position, rotation, and size of an object for every light separately as well. And lastly, renderer needs to compare position of the object which is receiving the shadows to an object which is casting the shadow. When creating a 3D scene, it is advised to consider lighting logistics depending on the strength of the computational core as it can cause computational load which can result in small FPS.

Wheelchair simulators were using few point lights for better authenticity. But after profiling of performance was done, FPS was lower than 60 in virtual headsets. The main goal of simulator was to teach disabled person using the wheelchair and overcame obstacles. To teach that, realistic graphics were on a lower priority than smoothness of application. Due to this fact, all unnecessary lights were removed from the scene and ambient light was implemented. Ambient light is considered as the light with the

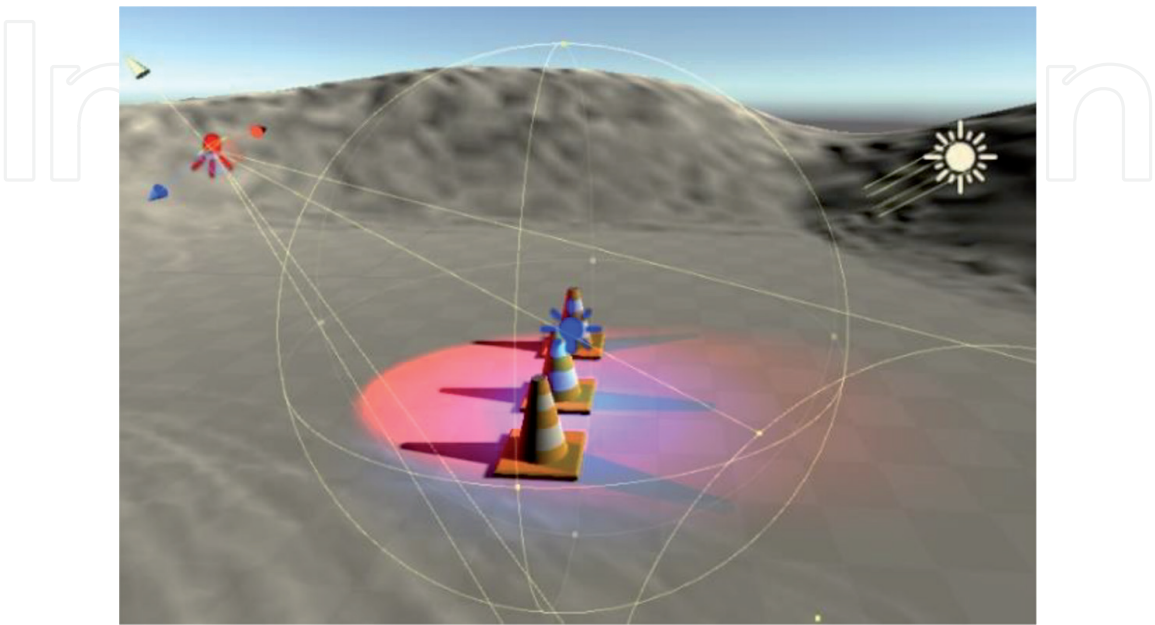


Figure 7.
Combination of spot light, point light, and directional light in a scene.

lowest cost for the computational core. Therefore, shadows were not implemented at all, as ambient light cannot cast them. If shadows cannot be cast, there is no need for a shader. Result is a simple lighting setting but a lot of computational space is saved.

5. 3D objects and virtualization sequence

The whole process of object processing and virtualization is called the virtualization sequence. A simplified 3D virtualization sequence scheme is shown in **Figure 8**. Of course, each of these steps contains a number of other sub-steps naturally. Many sub-sequences of this sequence including 3D objects modeling and editing are already standardly used in practice, and they have steady conventions and standards. The virtualization sequence is important in defining and creating VR-based systems. In principle, it determines all the virtual world objects, their way of use, and of course, the semantics (meaning) of individual objects [11].

As mentioned in the chapter, 3D computer modeling is the creation or modification of a 3D graphical virtual model using a specific software tool—3D modeling application (e.g. Blender, Autodesk 3D Studio Max, Maya, Sketchup, etc.). The modeling process of preparing/editing graphical data is similar to sculpture creation. 3D objects modeling is not an exception in this case and it is based on a four-step principle: model/object preparation, polygonal modeling, texture preparation, and texture application and filtering. The properties of these steps depend on the model/object type. The created model/object can be used for visualization, simulation, or for 3D printing.

Each model consists of *points*, *edges*, and *faces*. These are primitives that create geometry of any model. **Figure 9** represents geometry of wheelchair using primitives.

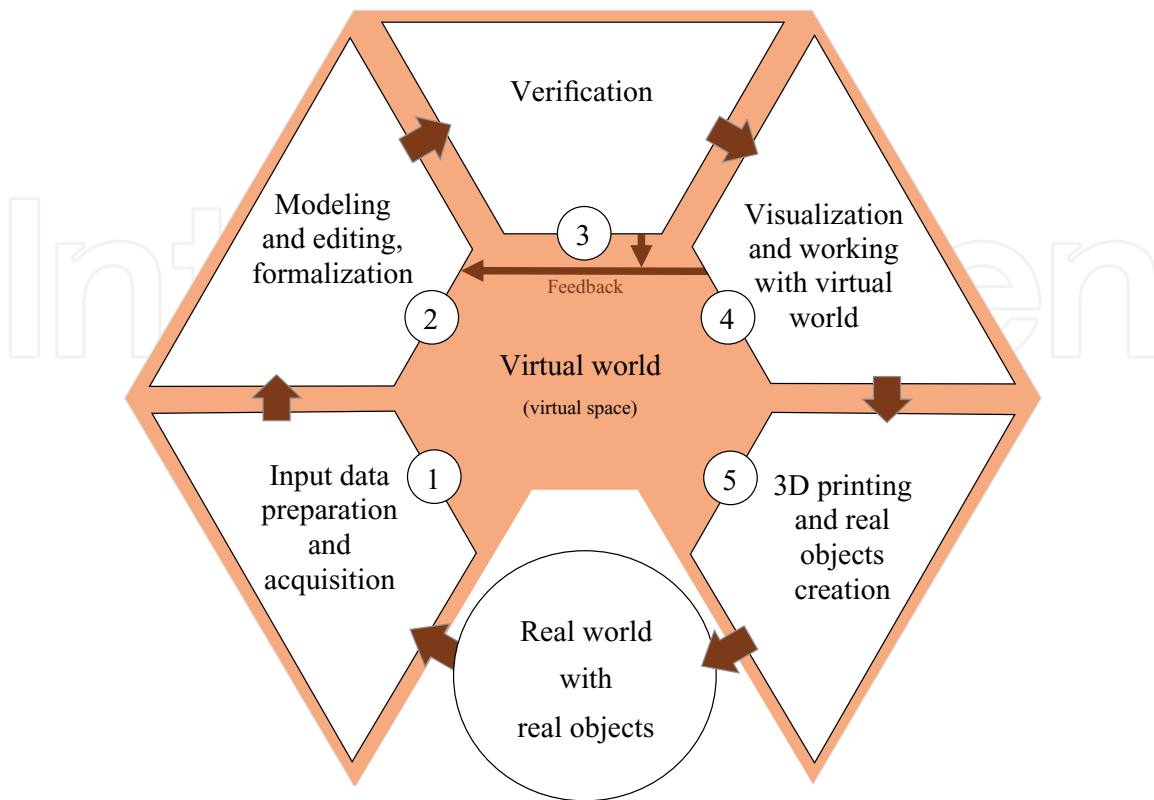


Figure 8.
A simplified 3D virtualization sequence scheme.

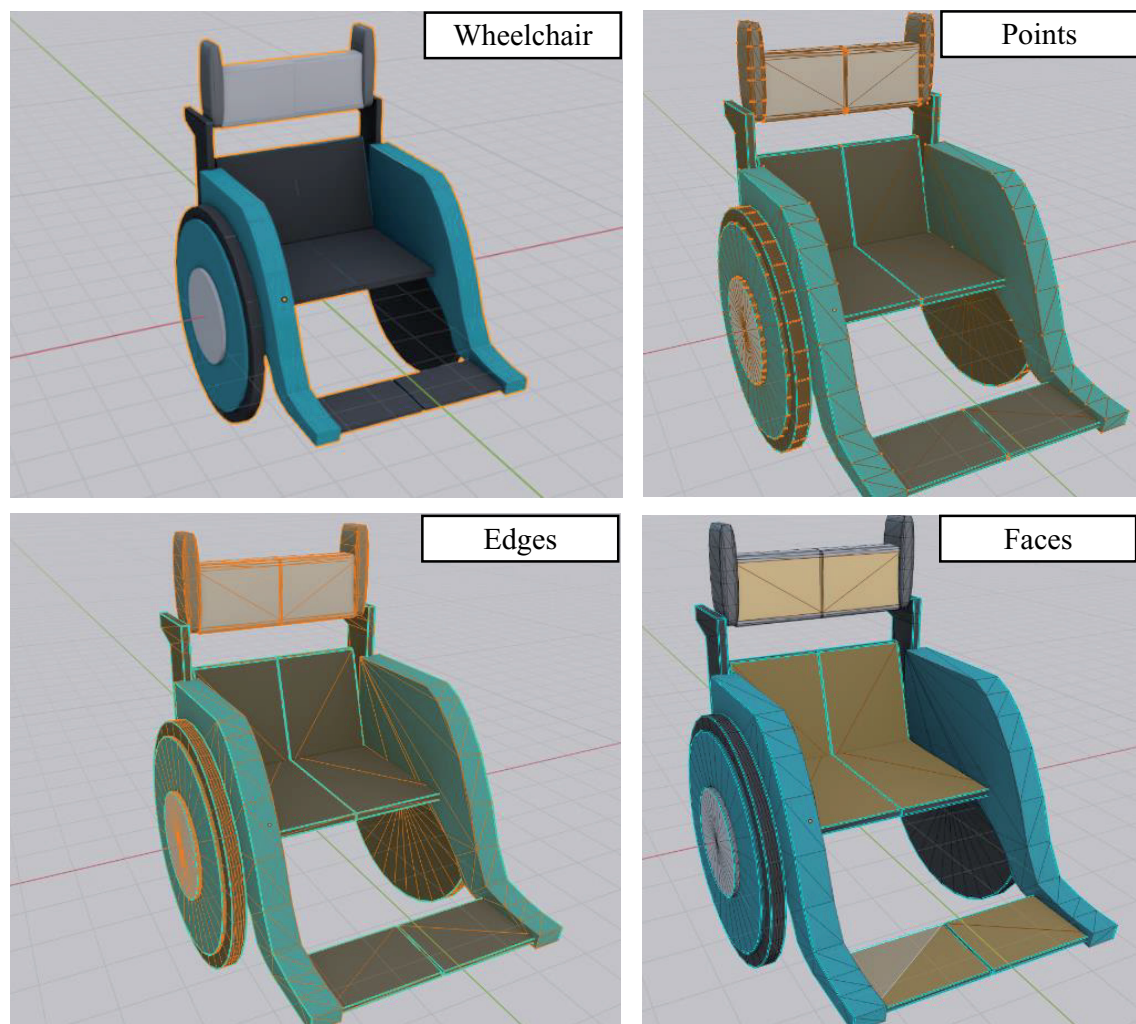


Figure 9.
Points, edges, and faces represented on cone model.

Wheelchair model has assigned base color texture. A number of polygons are in a count necessary to shape the model into a wheelchair look. Although it does not look realistic, it brings a possibility to render such model faster. Therefore, its optimal design solution for VR headsets. All models in wheelchair simulators are designed with low polygons principle. Due to this fact, scenes have united design with fast rendering. As helmets are used commercially, it is generally known that realistic look does not affect the possibility of user being drawn into the environment. The user is drawn into the environment, especially, by united design of models and their texture, interactivity with environment and sound effects. If these elements are designed creatively and are not exaggerated, user can enjoy the virtual world samely as environment with realistic look.

5.1 Objects geometry in the simulator

Target devices for wheelchair simulators are virtual headsets, therefore, it is needed to keep objects geometry as simple as possible. The higher is complexity of an object, the more it is compelling for the renderer to process. Every polygon in the object will burden renderer to calculate it and display it on the final screen. Due to this fact, many applications for virtual reality use low polygon technique of modeling. **Figure 10** shows geometry of some objects used in the simulator.

As **Figure 10** proves, models are build up the way just to represent its necessary shapes, to figure out meaning of the object. Disadvantage of low polygon objects is that neglected folds result in a non-realistic look. But it still keeps the shape to figure out the context. One of the advantages is, that this type of modeling practice does not require creating lots of details—LOD for every object. LOD is also an unnecessary feature in wheelchair simulators as scenes are not that wide for LOD to be applied. It is possible to decimate some polygons on a wheelchair or the car model, for example, those parts of model that are not so visible by a user. But a number of polygons will not change dramatically if it is desired to keep the look so it is an unnecessary step. Such a step would be worth considering if these objects were placed into environment multiple times. Then the polygon reduced will be the number of reduced polygons times the number of duplicated models.

5.2 Textures

Wheelchair simulator does not use special texturing features, such as normal mapping or Dot 3 bump mapping. It uses only simple base color textures and image textures. If the quality of the image texture is restricted to smaller size and resolution, rendering will be optimized to desired preferences. Wheelchair simulator is using

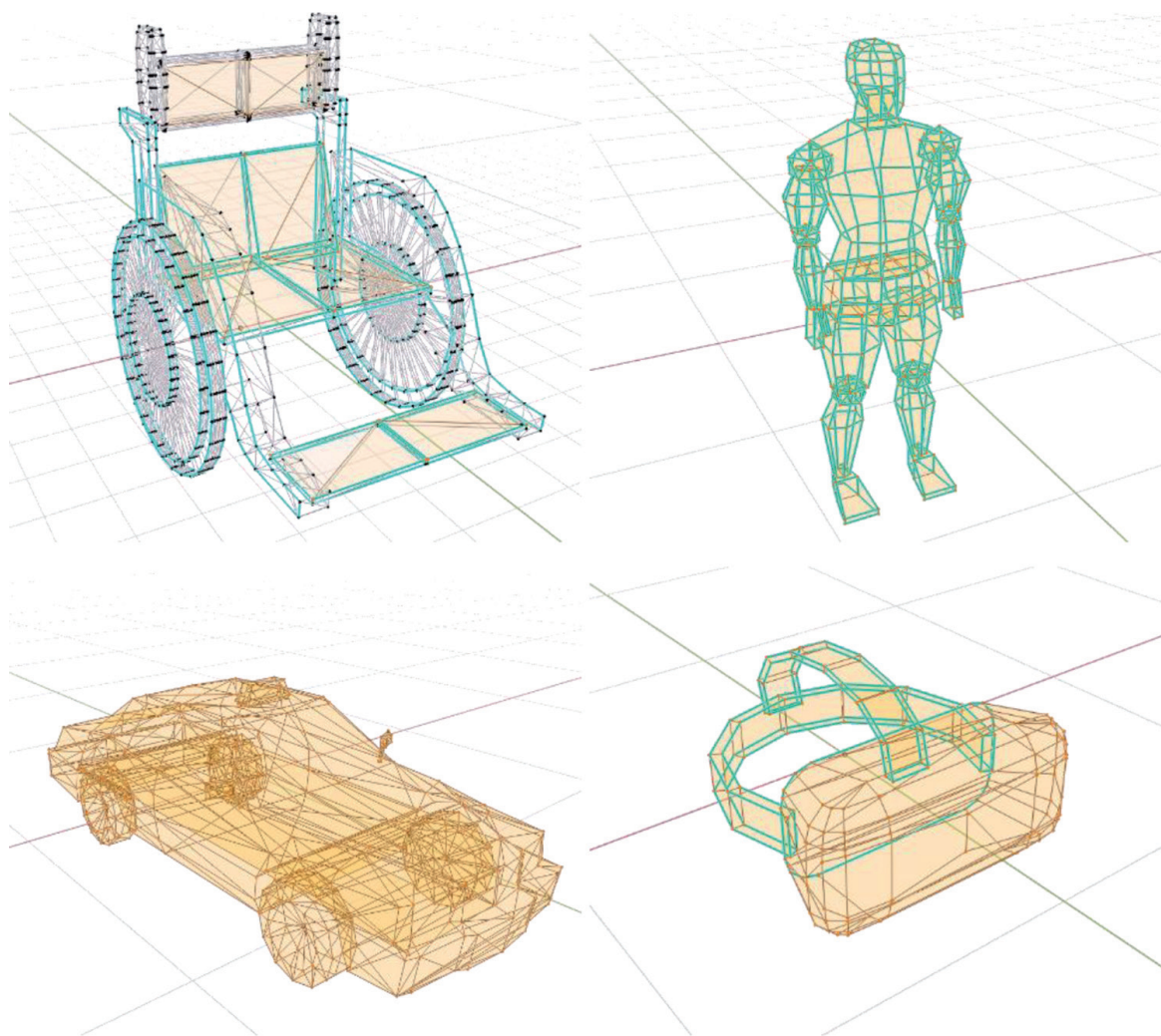


Figure 10.
Geometry representation of wheelchair, avatar, headset, and car models.

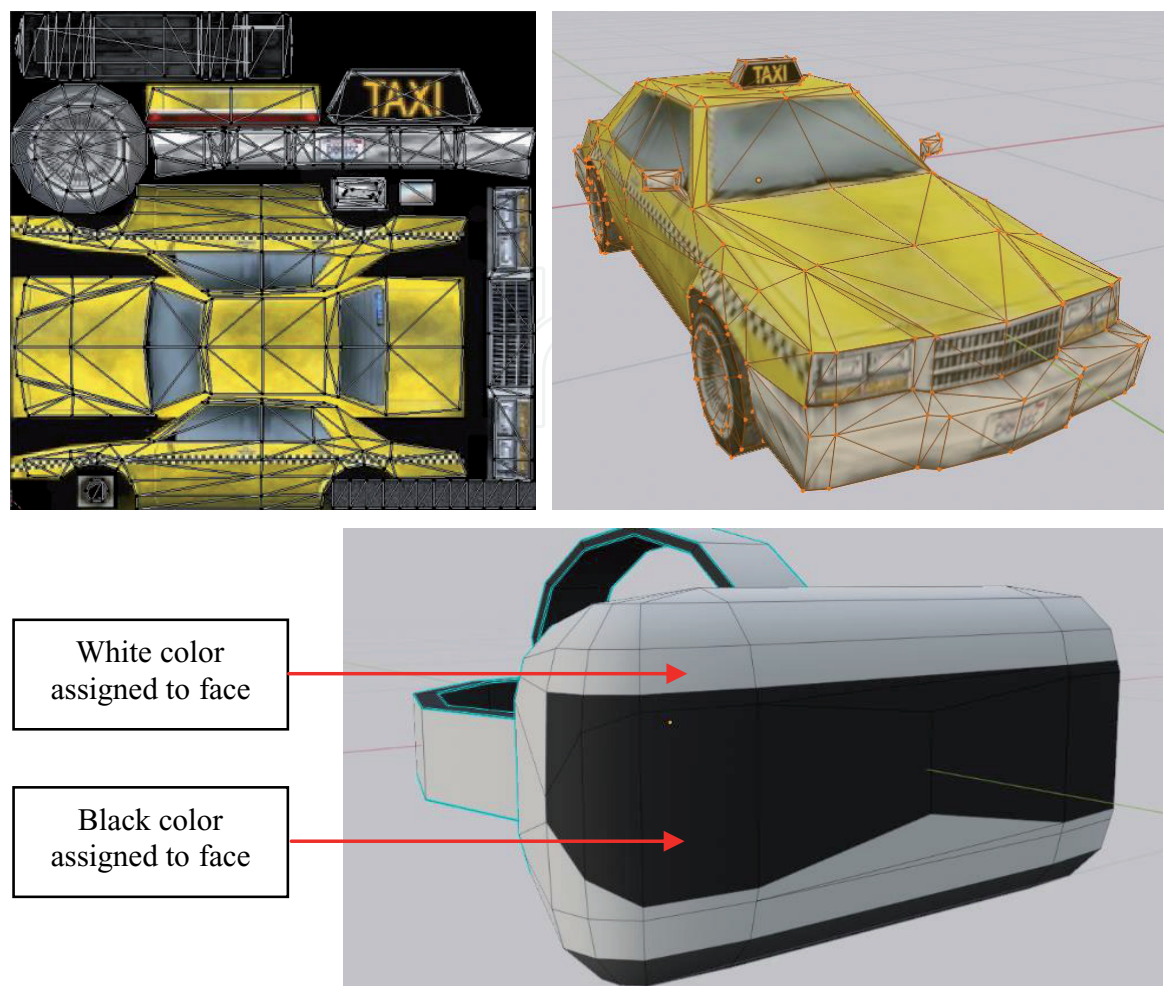


Figure 11.
Texture representation on taxi car and VR headset models.

resolution and size up to 5 megabytes. But the average sizes of image textures are 300–500 kb. Following **Figure 11** represents an example of image texture of car with a size of 387 kb and virtual headset object with assigned based colors. Image texture of a car also shows the mapping of the taxi car object. Mapping is represented with points primitives of an object. Their position defines the position of the texture on the object.

Resolution of a texture is important to be optimized. If the texture has for example resolution of 4000×4000 pixels, but texture coloring and details can fit into resolution 500×500 pixels without any pixel compressing, it is highly recommended to change the resolution of the texture. Every time, when an object is in a field of view, piece of texture is rendered in dependence on the face shape and size of an object. Therefore, if the simple cube with 6 walls-6 faces have assigned full-size texture with resolution 4000×4000 pixels on each face, and top side, right side, and left side of a cube are in a field of view, the texture will be rendered 3 times with defined resolution.

6. Screen-camera

Camera and screen settings can facilitate render calculation as well. To do so, it is needed to reconsider, for example, if it is more effective to use *single pass* or *multipass*

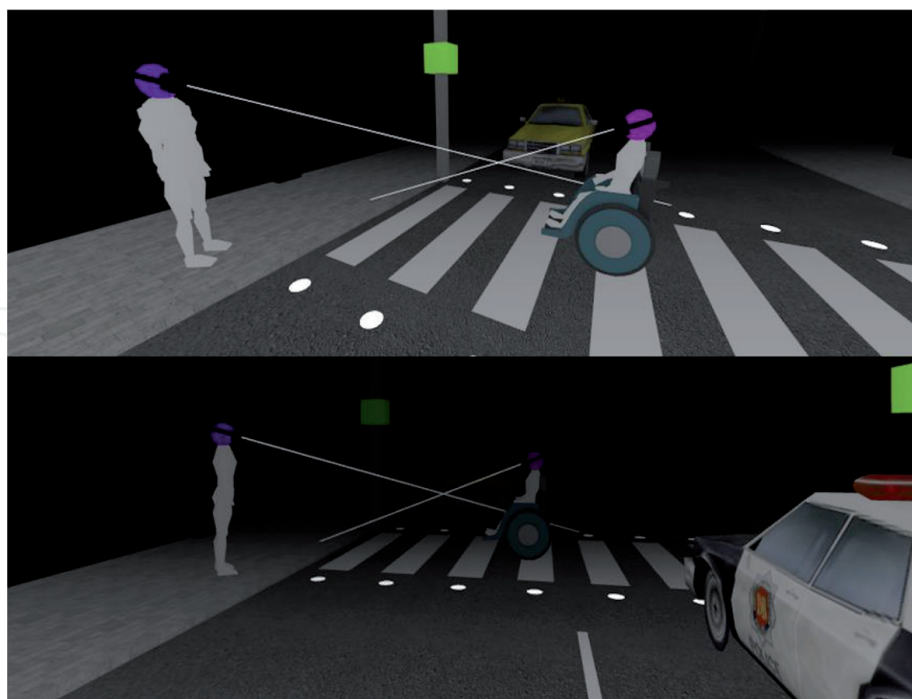


Figure 12.
 Night implementation in wheelchair simulator.

rendering. Object visibility based on camera distance is one of the features to consider as well. For example, occlusion *culling* is a camera feature that defines if objects should be rendered when they are behind another object. There are many effective rendering techniques that help lower the run time cost.

Besides saving computational cost, camera has some features to manipulate the graphics and visualization in scenes as well. One of the features was presented in Section 3. And in **Figure 3**. As it was mentioned before, scene called crosswalk-night is situated at night. In real world, when a person is not in a completely dark environment, he/she can see objects within few meters radius. Visibility, colors, and sharpness of the objects depend on the intensity of light. To simulate such environment, scene contains ambient light of certain intensity—but lower than the ambient light in other scenes. This will secure objects to be visible. To implement logic of seeing things in some radius around a user, a feature called fog need to be applied. Fog has a parameter, which defines what color fog will be. To simulate the night, color needs to be black. Other colors, such as red for example, will be seen in a distance as ambient light is applied all around the scene. Result is shown in **Figure 12**.

Figure 12 shows, as user goes further from the crosswalk, other objects will gradually shade as they are not in a radius of fog-free area. It can be observed, that taxi car on an upper part of the figure is visible, but as user went further back, taxi car is not visible anymore.

Others performance improvement techniques such as single pass, multi-pass, occlusion culling are predefined in used framework—A-frame.

7. Scenes-spaces

When creating a 3D application, it is good to know whether the size of the scene will be wide or small in advance. Wider scenes are used mostly when creating an

outdoor environment. In this case, wheelchair simulator is using small scenes. Simulator has a scene where a disabled person needs to go around cones or cross the crosswalk. This activity is placed outside, but scenes were designed with a wall limitation on terrain edges. It is easier to solve graphic limitations with a smaller scene as it does not contain many objects. Polygons' number is lower which result in faster rendering. Following **Figure 13** presents the scenes from the distance together with the top view.

7.1 Scene with slopes

One of the wheelchair simulator's goals was to teach a disabled person to go through elevated terrain. Such obstacles are mostly situated inside buildings in real life. Mostly staircases, which have barrier-free platform placed on them, have the steepest ascent. Therefore, scene with slopes was created. Whole scene consists of one big room with two slopes on the edges of the room and patient can go through

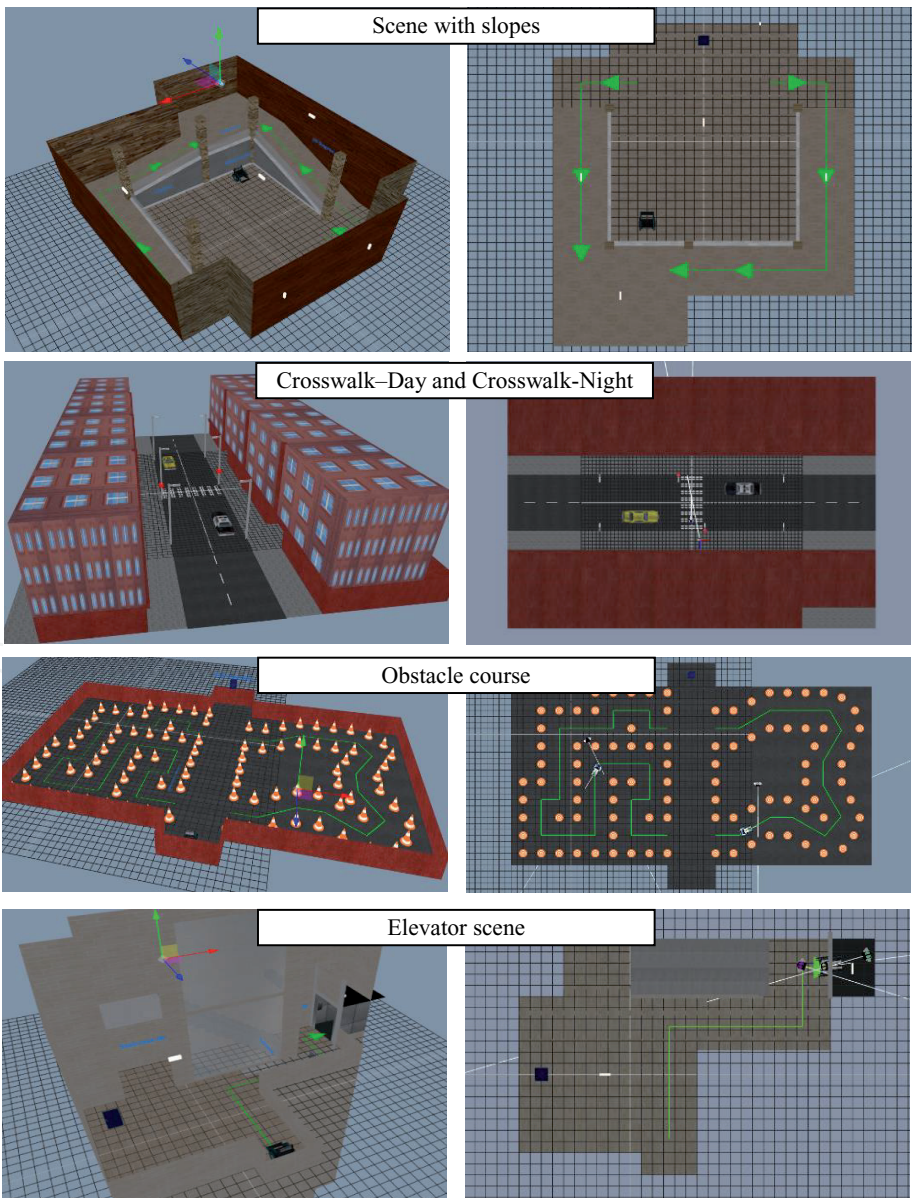


Figure 13.
Scenes representation in wheelchair simulator.

them. Scene is categorized as a small scene because its visible width is a few meters. **Figure 14** shows scene details as well as its top view with explanation.

7.2 Crosswalk-day and crosswalk-night

People living near roads will encounter crosswalks almost on daily basis. It is an unwritten rule, that crosswalk needs to be crossed as quickly as possible. Disabled person has a speed limitation. Due to this fact happens, some of them do not have confidence doing activities where they or other objects should move fast. Scene with crosswalk was designed for a person to gain confidence in a wheelchair. Scene was designed even in night mode as visibility of objects, such as cars, is limited. It is considered a small scene, even though it is situated outside because terrain is a size of a few meters. **Figure 15** shows scene details as well as its top view with explanation.

Scenario of crossing is simple. Cars are moving at a certain speed when traffic light is red. The traffic light will, after a few seconds, lights up in orange, signaling that a person can prepare for crossing. Then it lights up green signaling that person can cross the road. Cars will stop in front of the crosswalk. They do not have a collider to make sure, that if the patient is still on the road, and traffic light will light up red, cars will not move him/her out of the map. This can cause discomfort and dizziness.

7.3 Obstacle course

A person in a wheelchair must learn to control it in a first place. To learn the lever manipulation, how it affects the wheelchair and how to rotate the wheelchair, the scene with cones was created. Obstacle course scene has two routes, where a disabled person can practice its wheelchair skills. That way he/she will gain agility to continue

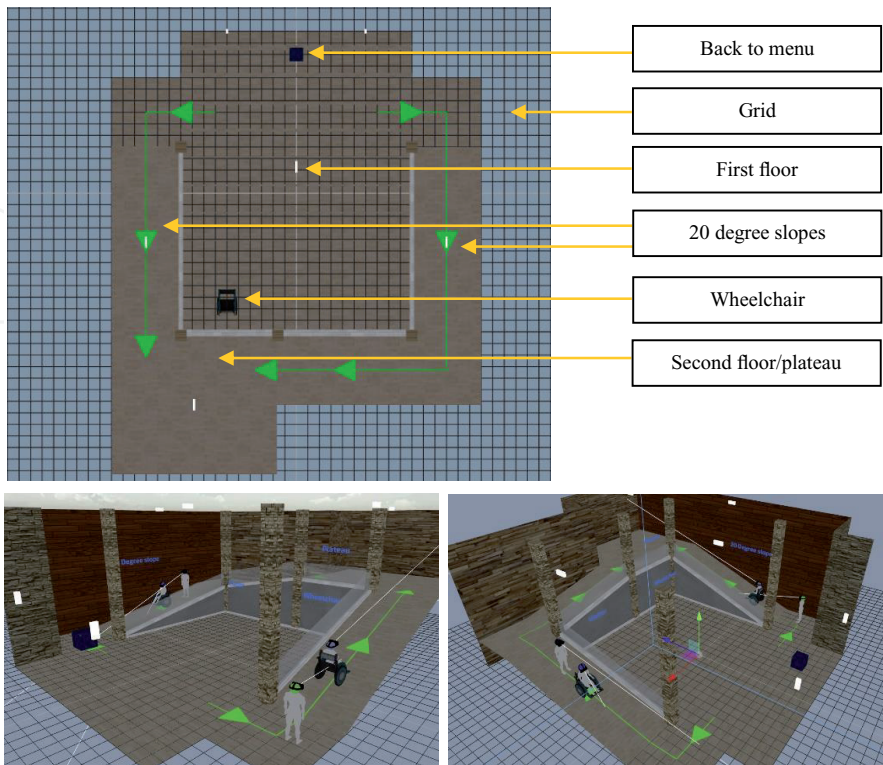


Figure 14.
Scene with slopes.

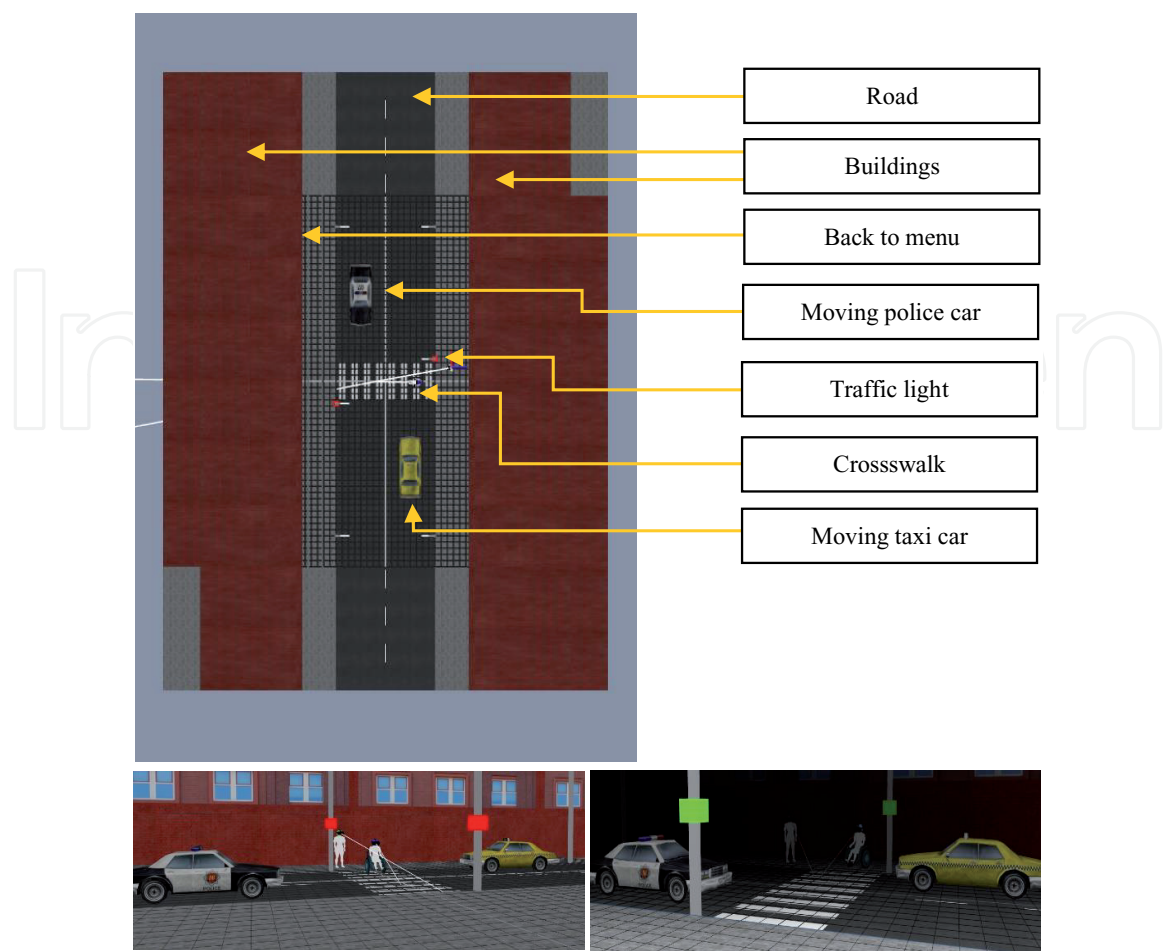


Figure 15.
Scene crosswalk-day and crosswalk-night.

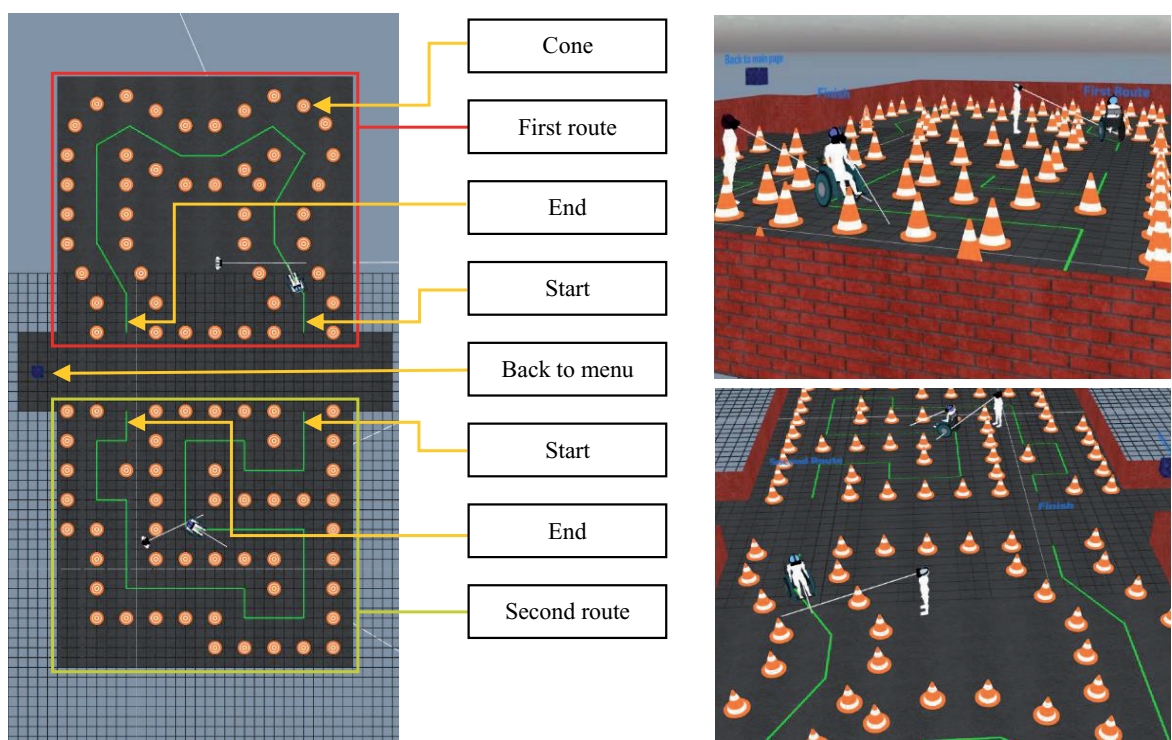


Figure 16.
Obstacle course scene.

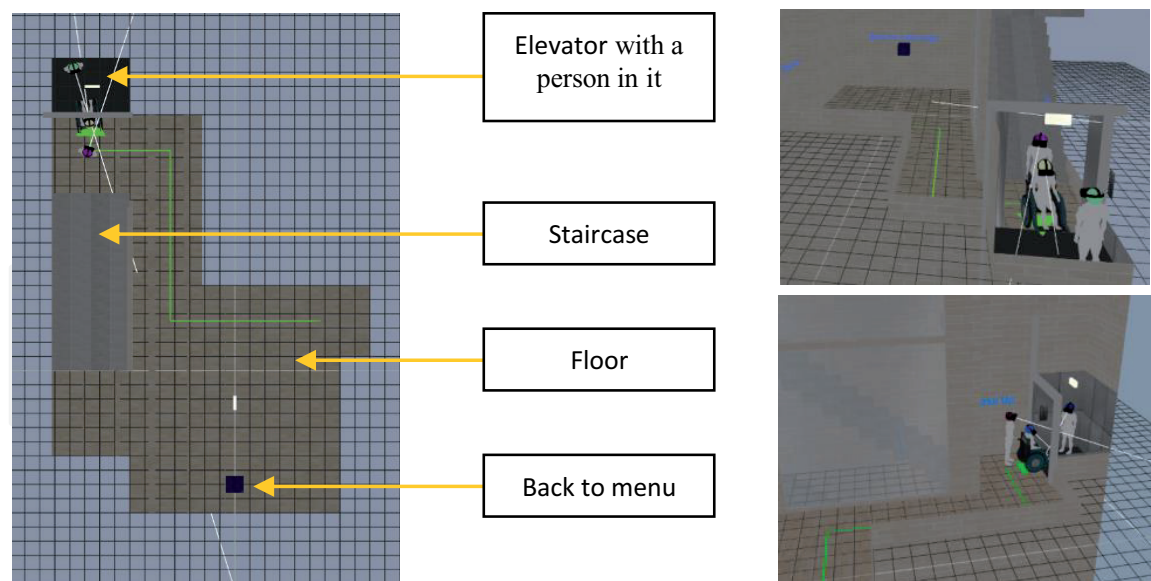


Figure 17.
Elevator scene.

with more difficult obstacles. Cones are static and have colliders so the person in wheelchair does not move them if he/she will bump into them. Same as other scenes, obstacle course is considered as a small scene. Following **Figure 16** shows scene details as well as its top view with explanation.

7.4 Elevator scene

The elevator scene was designed to teach the person in a wheelchair to go through a narrow hallway or learn how to fit in an elevator. Scene is situated in a building with a staircase. It is impossible to go through the staircase as it has a collider. Therefore, proposed scenario is that person in a wheelchair wants to go upstairs. **Figure 17** shows scene details as well as its top view with explanation.

8. Conclusion

3D computer graphics gave the possibility to create a 3D world with scenarios, where immobilized persons can learn to handle the most common obstacles. System is supported for virtual headsets, therefore, immobilized persons can experience more authentic environment. Chapter represents 3D computer graphics used in an environment together with an explanation of why they were implemented that way. Virtual reality is a technology that supports such environments. As VR technology is relatively young and it is still evolving, it was needed to implement such 3D graphics, which will not compromise the smoothness of the simulator. Therefore, simulator is implemented with prioritization of smoothness over a realistic look. Chapter also focuses on such 3D graphics combinations to give the user the best experience.

Acknowledgements

This work has been supported by the Operational Programme Integrated Infrastructure in a frame of the project: Intelligent systems for UAV real-time

operation and data processing, code ITMS2014+: 313011 V422 and co-financed by the European Regional Development Fund and also it has been supported by the KEGA grant No. 048TUKE-4/2022: “Collaborative virtual reality technologies in the educational process”.

IntechOpen


IntechOpen

Author details

Branislav Sobota* and Miriama Mattová
Technical University of Košice, Košice, Slovakia

*Address all correspondence to: branislav.sobota@tuke.sk

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Sobota B, Korečko Š, Hrozek F. On building an object-oriented parallel virtual reality system. *Central European Journal of Computer Science*. 2012;2(3):261-271
- [2] Hudák M, Korečko Š, Sobota B. Enhancing team interaction and cross-platform access in web-based collaborative virtual environments. In: *Proceedings of 2019 IEEE 15th International Scientific Conference on Informatics*. IEEE; 2019. pp. 160-116
- [3] de Ipina JML, et al. Virtual reality: A tool for the disabled people labour integration. In: *Proceeding of Challenges for Assistive Technology*. IOS Press; 2007. pp. 141-145
- [4] Boguščiak J. Wheelchair simulator in web virtual reality [diploma thesis]. Košice: TU; 2021. p. 62
- [5] Chen C, Bolas M, Rosenberg ES. Rapid creation of photorealistic virtual reality content with consumer depth cameras. In: *2017 IEEE Virtual Reality (VR)*. 2017. pp. 473-474. DOI:10.1109/VR.2017.7892385
- [6] Tredinnick R, Broecker M, Ponto K. Progressive feedback point cloud rendering for virtual reality display. In: *2016 IEEE Virtual Reality (VR)*. 2016. pp. 301-302. DOI:10.1109/VR.2016.7504773
- [7] Chen C, Rosenberg ES. Virtual content creation using dynamic omnidirectional texture synthesis. In: *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 2018. pp. 1-2. DOI:10.1109/VR.2018.8446410
- [8] Robertson GG, Card SK, Mackinlay JD. Three views of virtual reality: Nonimmersive virtual reality. *Computer*. 1993;26(2):81. DOI: 10.1109/2.192002
- [9] Kautz J, Lensch HPA, Goesele M, Lang J, Seidel H. Modeling the world: The virtualization pipeline. In: *Proceedings 12th International Conference on Image Analysis and Processing*. 2003. pp. 166-174. DOI:10.1109/ICIAP.2003.1234044
- [10] Martini A, et al. A novel 3D user interface for the immersive design review. In: *2015 IEEE Symposium on 3D User Interfaces (3DUI)*. 2015. pp. 175-176. DOI:10.1109/3DUI.2015.7131757
- [11] Hrozek F, Korečko Š, Sobota B. Solutions for time estimation of tactile 3D models creation process. In: *Advanced Machine Learning Technologies and Applications: Second International Conference (AMLTA 2014)*. Cairo: Springer International Publishing; 2014. pp. 498-505