PERFORMING 3D MEASUREMENTS IN A VR ENVIRONMENT

D. Kontos, A. Georgopoulos *

Laboratory of Photogrammetry, National Technical University of Athens, Greece
dimitriskontos94@gmail.com, drag@central.ntua.gr

Commission II, WG II/8

KEY WORDS: Virtual Reality, HTC Vive®, Unreal Engine, 3D model

ABSTRACT:

In the context of this paper, a virtual reality application that allows each user to perform basic topographic processes on an already created 3D model inside a virtual environment was developed. Specifically, it is an application that allows the user to perform measurements of distances between two points in three-dimensional space and measurement and extraction of the three-dimensional coordinates of any point inside the virtual reality environment. Furthermore, the created application was evaluated in terms of its functionality, its usability and metric accuracy. Before the developing stage a research was done in order to determine which virtual reality system and which game engine is most suitable to use and finally the HTC Vive® virtual reality system and the Unreal Engine 4 game engine were used. Before all that, the concept of the virtual reality science was defined and also the virtual reality technologies in today's world were analyzed.

1. INTRODUCTION

1.1 Virtual Reality

In 1986, Lanier, in one of his interviews, defined Virtual Reality (VR) as: "*An interactive, 3-dimensional environment, that is generated by a computer and a person can be immersed in it*". Later (Lanier, 2017) he rephrased it to "*The substitution of the interface between a person and the physical environment with an interface to a simulated environment*". Immersion is a basic VR element and is defined as: "*The illusion that the user has regarding his existence in a virtual environment*" (Slater et al., 1994). In addition, Heim (1998) determines the following characteristics as basic for a virtual reality system:

- Interaction
- Immersion
- Density and completeness of information

Furthermore, VR systems can be categorized in three categories according to the degree of interaction they provide:

- Passive, in which the user observes the virtual environment without being able to interact with it.
- Exploratory, in which the user can move independently inside the virtual environment and explore it.
- Interactive, in which the user, in addition to movement, can also fully interact with the virtual environment.

Between a real environment and Virtual Reality, one Augmented Reality and Augmented Virtuality are classified. However, their definitions and characteristics are outside the scope of this paper.

1.2 Virtual Reality in Today's World

Today VR can be defined as a computer science branch that uses headphones and special equipment combined with real or natural spaces and multiple projection environments in order to create and replicate realistic images, sounds and senses to an immersed user. These state-of-the-art virtual reality systems can be:

- Expensive systems, that need a computer with distinct specifications to function properly.
- Low cost systems based on a modern smartphone.
- Autonomous, that can provide VR experience on their own without dependencies on external computing sources.

The selection of a specific system by a developer depends highly on the application he intends to build and on the target user group. Up until now Virtual Reality has been used in a broad variety of applications in different scientific or not sectors, such as Medicine, Education, Tourism, Military, Aeronautics, Space Exploration, Architecture, Virtual Art, Video Games and many more.

Moreover, a field that Virtual Reality technology can provide state of the art solutions and applications is the preservation of cultural heritage. There are already developed applications in that field which can show that the use of VR has enormous potential. Virtual Reality using state of the art technology can contribute to the vast dissemination and preservation of cultural heritage and can provide new tools for researchers.

In this paper, a Virtual Reality application is described, which allows users to conduct basic topographic measurements on an already created 3D textured high resolution model. This application was developed in an immersive virtual reality environment. This kind of application can be useful to the

[*] Corresponding author

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLIII-B2-2020-863-2020 | © Authors 2020. CC BY 4.0 License.

scientific community as no similar application with these capabilities was found in the related literature till the application was complete (Kontos, 2018). So, with this paper targets to highlight the significance of such applications and the room for future research and improvements that they have. The methodology developed is described next. In section 2 the main hardware selected and in section 3 the software environment for the application development are presented. In section 4 the development of the application itself is described and in section 5 its evaluation. Some concluding remarks are presented in the last section.

2. HARDWARE – HTC VIVE® VR SYSTEM

2.1 HTC Vive®

The HTC Vive® is a VR system developed by the HTC and Valve Corporation companies. It was firstly released in April 2016. It is a virtual reality system of high cost, which can work supported by a PC running Windows, Linux or MacOS operating system. HTC Vive® has the following components (Figure 1):

- A virtual reality headset with two full high definition OLED screens, with 90Hz refresh rate for each screen and a 110-degree field of view.
- Two wireless rechargeable controllers.
- Two tracking lighthouses emitting nonvisible light in 8.333ms period to trace the headset and the controllers inside the virtual room.

The above-mentioned gear is the standard equipment for an HTC Vive® to work but there are plenty extra peripherals to cover the extra needs for users or developers. For this application, the basic HTC Vive® setup was used.

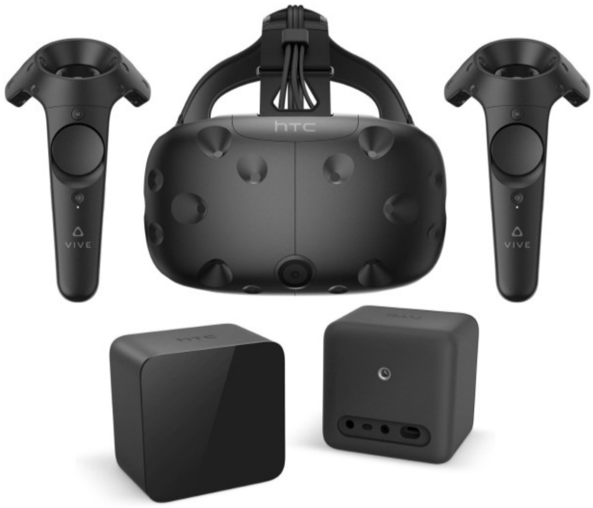

Figure 1. The basic components of the HTC Vive®

Practically, the HTC Vive® system defines a 3-dimensional space inside of which it tracks the headset and the joysticks. That space can be from 2m x 1,5m minimum to 4m x 3m maximum. A thorough research and detailed analysis of the tracking system has been done by O. Kreylos (2016). Furthermore, the PC system requirements for HTC Vive® to function are shown in Table 1:

Table 1. HTC Vive® system requirements.

Component	Recommended System Requirements	Minimum System Requirements
Processor	Intel Core i5-4590/AMD FX 8350 equivalent or better	Intel Core i5-4590/AMD FX 8350 equivalent or better
GPU	NVIDIA GeForce GT X 1060, AMD Radeon R X 480 equivalent or better	NVIDIA GeForce GTX 970, AMD Radeon R9 290 equivalent or better
Memory	4 GB RAM or more	4 GB RAM or more
Video Output	HDMI 1.4, DisplayPort 1.2 or newer	HDMI 1.4, DisplayPort 1.2 or newer
USB port	1x USB 2.0 or newer	1x USB 2.0 or newer
Operating System	Windows 7 SP1, Windows 8.1 or later, Windows 10, Linux or MacOS	Windows 7 SP1, Windows 8.1 or later, Windows 10, Linux or MacOS

2.2 The HTC Vive® Selection

The modern virtual reality systems that are publicly available and fully documented are not many. At the time of the development of this application, there were four systems: HTC Vive®, Oculus Rift, OSVR and PSVR. The first three systems are for developing applications using a computer, while the last one is for use with the PlayStation 4 gaming system.

The capabilities of all four systems are similar, with the HTC Vive® outperforming the others because it can provide virtual reality experience at room scale. In contrast, the other three systems can support applications developed only for standing or sitting experiences, thus greatly limiting the user's mobility.

In addition, prerequisites for the specific system to be selected were the availability of official support in terms of programming and development of an application, as well as the cooperation of the system with the game engine software in which the application was to be developed. To conclude, taking into account all the above, the HTC Vive® was chosen, as it had the advantage of better support and documentation and the provision of virtual reality experiences at room scale.

3. GAME ENGINES

3.1 Game Engines evaluated

The application presented in this paper needed to be developed in a game engine environment. A game engine is a software development environment designed for video game or interactive application building. Specifically, such software can provide rendering engines for 2D and 3D graphics, physics engine, collision detection and response, sound, scripting, artificial intelligence, and more specialized tools for development. The two game engines listed below where tested and evaluated for the development of this application:

1. Unity 3D
2. Unreal Engine 4

The above are free to use, well documented and provide support for VR application developing using HTC Vive®.

3.2 Unity 3D

The Unity 3D game engine was released in 2005 from the Unity Technologies team as macOS exclusive. By now, Unity can be also used in Linux and Windows operating systems. Also, this game engine supports application development for 27 platforms. The core functionalities of this engine are those which were described in section 3.1. At this point it should be emphasized that applications are developed within Unity 3D are object-oriented, as is the C# language it uses. This means that each object of the application is a separate entity, it has its own properties and characteristics and can be configured and programmed uniquely by the developer.

The latest version of Unity 3D available at the time of the development was v.5.5.1. For all the characteristics it has, it was the perfect fit for the specific application. However, the problem with dealing with 3D models by the Unity engine arose after importing the 3D model, which was to be used, in its environment. As it is a highly detailed and complex model, which will be presented and analyzed in the next section and contains 35,414,290 three-dimensional triangles, the engine could not treat it as one entity. In particular, the version used had the ability to treat three-dimensional models with a total number of triangles of 65,535, i.e. 16 bits size of information. Therefore, Unity arbitrarily divided the 3D model into 541 independent sections. No problems were reported with this model's visualization. In contrast, after testing Unity's spatial entities detecting system, called, Raycasting, it was observed that due to this arbitrary separation of the model into smaller models the detection system did not detect them correctly due to their "peel" shape (Figure 2). To conclude, this problem would not allow for the further development and coding of the application so it was decided to use the Unreal Engine platform that could provide a better manipulation for complex 3D models.

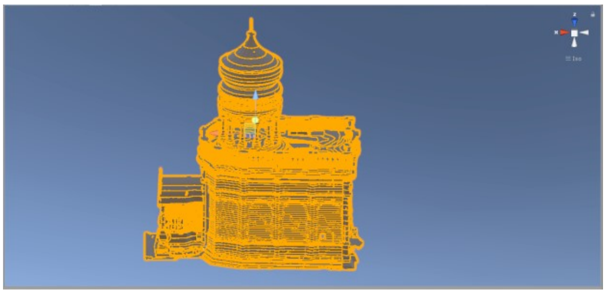

Figure 2. The arbitrary separation of the model into "peel" shaped smaller models.

3.3 Unreal Engine 4

The Unreal Engine game engine was developed by the Epic Games company and its first version became public in 1998. At first, it was developed with the only purpose to build first person shooter games but later it was used for the development of various apps. The latest stable version by the time that this paper's application was developed was 4.18. It also supports app development for 17 platforms and its core functionalities are like those of Unity 3D. Thus, this engine was also considered to be suitable for this paper's development purposes.

Unreal is a fully modernized game engine which provides many parameterization features for the developer. A great advantage is the ability to create objects with artificial intelligence. This means that the behavior of an object can be programmed to react "voluntarily" to the stimuli it receives from its digital environment. This makes it possible to create a fully interactive environment in an application that will greatly improve the end user experience.

Unreal Engine is based on the C++ programming language and an application within it can be coded it two ways. The first way is the classic script code writing and then importing the script in the engine's environment. The second way is the use of visual programming tools, that the engine calls blueprints, to develop an application solely inside the game engine's environment. Blueprints can assist users that are not familiar with programming or this specific language to develop an application. With the Blueprints method each object has its own separate page intended for its programming. On this page, each command is represented by boxes containing the command and its parameters that can be modified. Finally, each command is connected to the next one with the help of simple continuity links, logical or mathematical operators, but also specialized links depending on the case. It should be emphasized that in the second method all the possibilities offered by the classic coding are provided and applicable. Indicatively, functions, iterative processes, embedded processes, logical processes, and other similar functions can be created (Figure 3).

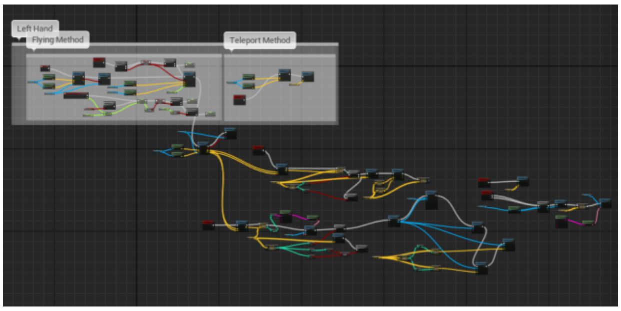

Figure 3. The core coding blueprint of the application. Every separate function that was coded and used has its own blueprint page

To conclude, with Blueprints programming, the user is relieved of the need to use a specific editor and syntax for his code to work, since the Unreal Engine 4 undertakes to translate its virtual programming into real code.

4. APPLICATION DEVELOPMENT

4.1 The 3D model used

The 3D textured model of the Holy Aedicule which was created during its full rehabilitation process carried out by the NTUA's Interdisciplinary team (Georgopoulos et al. 2017, Moropoulou et al. 2017) was used for testing and evaluation of the developed functions of this application. The above-mentioned 3D model was created combining photogrammetric and laser scanning methods therefore it is highly detailed and accurate, containing 35.414.290 triangles and being textured with high resolution (Figures 4 and 5).

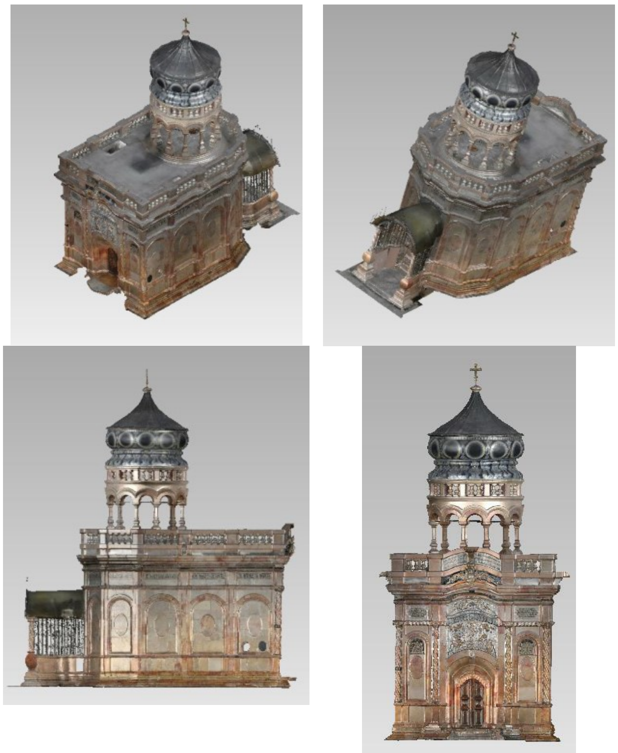

Figure 4. The exterior of the 3D model from various angles

4.2 Procedures and Blueprints

As stated above the application presented in this paper was developed with the Unreal Engine 4 (UE4) game engine. Specifically, the following steps had to be developed:

4.2.1 UE4 project creation

A project file was created in which every aspect of the application will be developed and saved. An UE4 project is a sub catalog with specific folders. Every folder is accessed by the game engine and refers to a different function.

4.2.2 Applications 3D environment creation

A realistic application environment was created. This “real world feeling” environment will make the user feel more immersed inside it and give a more realistic feeling to the application. At first, a terrain surface was created for the user to stand on and move around. This surface levitates inside the virtual environment and was created with dimensions 1x1km and 50m thickness. These dimensions were given to the terrain to make the user feel “safe” to move around and not be afraid of “falling off”. In addition, a neutral texture was given to the terrain with the goal not to disorientate the user from the application's main object of interest. Moreover, to enhance the realism of the environment a sphere-shaped sky texture was installed around the terrain (Figure 6). The center of the terrain was placed on the (0,0,13.415) point of the UE4 project's reference system.

4.2.3 3D model's import and placement

To import a 3D model inside an UE4 project is a very straightforward procedure with easy steps. In this case, though, due to the complexity of the 3D model that was going to be used, there was the need to manipulate it in order to import it correctly. Specifically, two problems had to be tackled. The first problem was the unification of the various 3D model chunks with texture and the second was the transformation of the model's reference system into the local UE project reference system. Specifically the transformation was done by moving each chunk by -80 meters on the X axis, by -110 meters on the Y axis and by -760 meters on the Z axis and moved the model close to the (0,0,0) point of the UE project. In addition, the relative position of each chunk with every other one was maintained. Afterwards, each chunk was imported separately in the project and then all chunks where grouped together to perform as one entity. Furthermore, final position adjustments were made manually through the tools that the UE4 provide to make the building touch the terrain surface (Figure 7). Finally, the unified object's textured was parameterized to be independent of light sources inside the virtual environment because during the initial tests the external light sources downgraded its detail level.

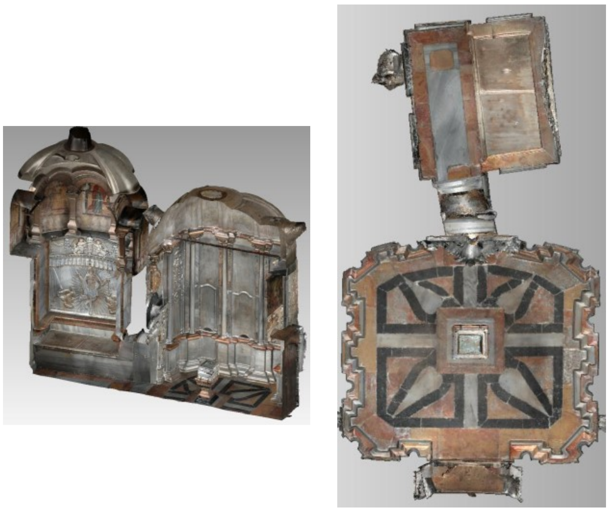

Figure 5. Vertical and horizontal sections of the 3D model showing its interior

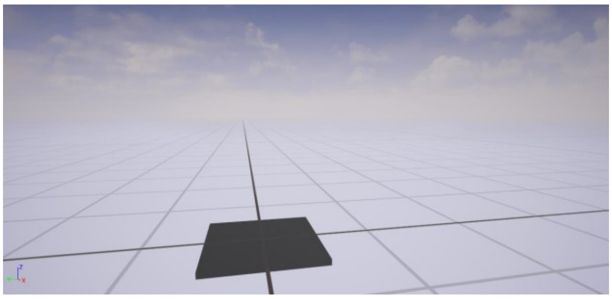

Figure 6. The 3D virtual environment

4.2.4 User entity creation

This entity represents the application's user and is the core of the application as it identifies and locates the virtual reality system and also within it all of the core application functions are coded. It is the physical representation of the user in the digital world. Since the UE4 supports the HTC Vive® the entity was set to identify, track and interact with the specific VR headset and controllers.

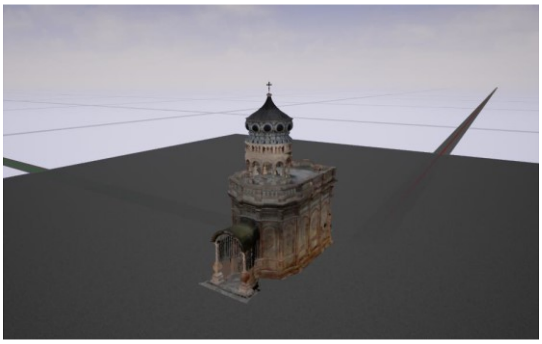

Figure 7. The final placement of the 3D model inside the virtual environment

4.2.5 Controllers visualization

A very important element for achieving a great degree of immersion, when using the application is the realistic visualization of the HTC Vive's® controllers so that the user can see them in the digital environment, can understand their buttons and interact with them. A realistic free to use model of the controller was obtained on-line (www.sketchfab.com). Also, because the right and left controllers are identical, the texture of the right one was colored green and the left one red (Figure 8).

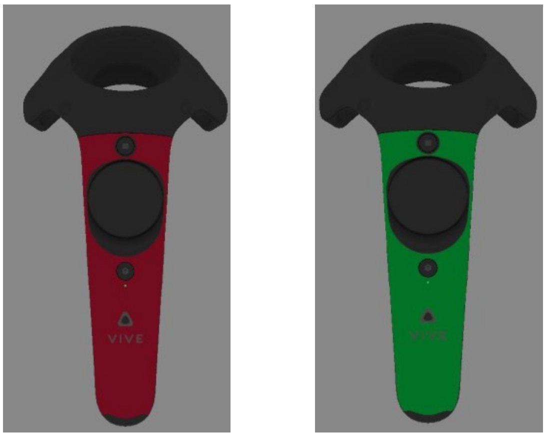

Figure 8. The final model of each controller

4.2.6 Coding the application's functions

As mentioned above, all the functions of the application were programmed using virtual programming in the form of Blueprints provided by the Unreal Engine 4 game engine. The core functions that were coded are briefly described below:

i. Aim Assist Function: For this purpose, a laser function was created in Blueprint, with which a laser beam is emitted from the center of gravity of each controller. This beam also works as a point selector for both controllers. For the left-hand controller, the user selects his teleportation spots, while for the right one the user selects the points with which he will perform the measurement procedures. Also, the laser beam is red when there is no intersection with an object, i.e. the 3D model, and green otherwise.

ii. Movement Functions: There are two ways a user can move around the virtual environment. These are a ground teleportation method and a flight method. The flight method is the one proposed to be used, because it allows the user to explore the virtual environment and the 3D from any perspective. It was observed that for some users, flying in a VR environment could be an unpleasant experience. For this reason, the other teleportation method was also coded that is considered a more classic approach (Kersten et al. 2017) concerning locomotion in a virtual reality environment.

iii. Measurements Functions: The measurement procedures that a user can conduct in the 3D virtual environment are distance measurements in the three-dimensional space between a pair of points, measurement of cross section points and exporting their coordinates and, finally, measuring and exporting the coordinates of any given point in the three-dimensional space. Moreover, to assist the user during the measurements a table was designed, appearing above the right controller which is used for measurement functionalities. It displays how many points have been measured and the last distance measured (Figure 9). Furthermore, the selected points in the virtual environment are highlighted with a small red sphere and connected with a blue line when used for distance measurement and with a yellow line when used for section measurement.

iv.

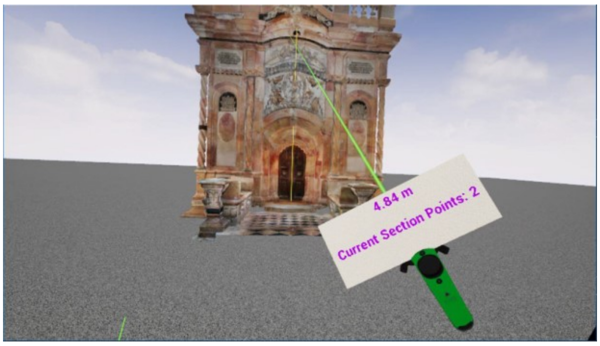

Figure 9. The virtual environment of the final application

4.3 Final Application

The application created serves its purpose as it enables the user to perform all the functions for which it was developed. This section will provide a brief overview of how the application works to make it easier to understand the capabilities of the final product.

The functionalities offered by the application, are divided into moving tools in the three-dimensional virtual space and into measuring tools. All movement capabilities are controlled by the red (left) controller, while all measurement capabilities are controlled by the green (right) controller. In terms of mobility, the user can either teleport from one part of the ground to another by aiming the left laser beam at the desired point and pressing the thumbstick button of the controller or flying freely in space by starting and interrupting the flight process with the trigger key of the controller. In addition, the flying speed is adjusted by touching the vertical axis of the thumbstick key and the direction of the flight is controlled by moving the controller in the 3D space and it actually matches the direction of the controller at all times (Figure 10).

As for the measurements, the user can measure the three-dimensional distance between two points with the trigger button while with the thumbstick button of the controller he can measure the three-dimensional coordinates of successive points and then extract them with the shoulder button of the red (left) button. It is noted that the process of measuring three-dimensional coordinates of points can be used as well for measurements of horizontal or vertical sections of each object. In addition, the already measured points can be deleted with the shoulder button on the green (right) control. To conclude, with the above application, the user is given the opportunity to navigate around and inside the 3D model and to perform basic measurement procedures. In addition, by extracting the data, it is possible for them to be processed in any desired way.

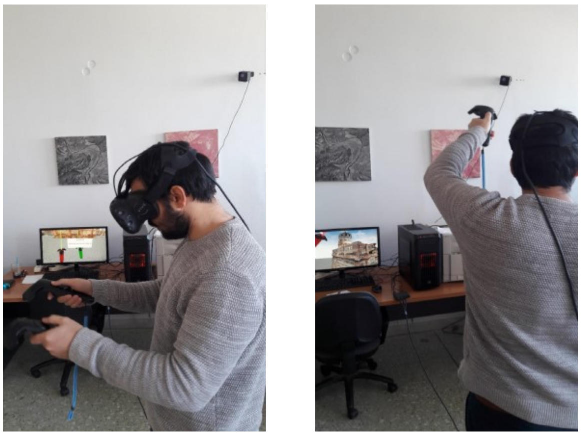

Figure 10. Photos during the use of the application

5. EVALUATION

In this section, the application will be assessed and evaluated in terms of its functionality and usability. For this purpose, it was considered appropriate to check the application for the accuracy of its measuring tools and to have it evaluated it by different users.

5.1 Coordinate Systems Transformation

Checking the accuracy of the measurement tools in the application reflects the accuracy with which the measured points correspond to points on the surface of the 3D model used. It needs to be emphasized that as the reference systems of the application and the primary 3D model were differentiated during the creation of the application the measured points had to be transformed from the application's coordinate system to the original system.

At first, for the coordinate system transformation it was known that the model was moved to a specific position and also mirrored over XZ plane. The above transformation is described with equations (1):

$$X_O = X_{VR} + X_T \; ; \; Y_O = -Y_{VR} + Y_T \; ; \; Z_O = Z_{VR} + Z_T \qquad (1)$$

where: X_O, Y_O, Z_O = coordinates of the original model
X_{VR}, Y_{VR}, Z_{VR} = coordinates of the VR model
X_T, Y_T, Z_T = coordinates of the transformation point

Consequently, in order to find the best transformation point, 11 distinct points, known in the original model, were measured inside the VR application and the most accurate one was selected. It is obvious that the need for this transformation introduces errors throughout the process since the measurements of the fixed points in the two reference systems are subject to the errors of the measurement of the tool used, to the error of the user when finding the identical point etc..

5.2 Distance tool accuracy

To check the accuracy of the distance measurement tool 30 fixed distances were measured using the VR application and compared with their true values, as determined in the original 3D model. It needs to be stated that the same fixed distances were measured directly onto the 3D model, using the Geomagic Wrap software (v.2017) and are considered as the true values.

The result of the above described comparison was a -0.001m arithmetic mean of all differences and a 0.012m standard deviation. In conclusion, the accuracy with which each user can measure distances on a three-dimensional model through the application implemented is estimated as 0.012m, i.e. the value of the standard deviation. In detail, since the numerical average of the other measurements has a value of -1mm, which is close to zero, it is concluded that systematic errors are not introduced in the measurements. Thus, the calculation of the standard deviation is enough to estimate the error of the population from which this sample of measurements is derived and consequently the error of the sample. If the arithmetic mean was much larger than zero then in order to estimate the error of the total population it would be necessary to calculate the average square error of the measurements (RMSE) (Qassim, 2013).

Of course, it should be emphasized that the specific result emerged with a relatively small number of observations and using as true values those performed on the three-dimensional model in the primary reference system. Thus, the relative accuracy between these two systems has been calculated above. In order to calculate the absolute accuracy of the system for the distance measurement tool, a different procedure should be followed which is not within the scope of this paper.

5.3 Point measurement tool accuracy

To calculate the accuracy of the point measurement tool 13 predefined sections on the original 3D model where also measured inside the developed application. These sections were also measured in the original 3D model using the Geomagic Wrap (v.2017) software.

Specifically, these sections included 3 outside vertical, 3 outside horizontal, 3 inside vertical and 4 inside horizontal sections. In detail, a user guided by these predefined sections, measured them in the application and 13 sparse clouds were exported after the measurements, each corresponding to a different section. Then, in the way analyzed in section 5.1, these sparse clouds were transformed from the application's coordinate system to the original system of the 3D model for their direct comparison. Finally, with the help of CloudCompare® software, the distance of each measured point, from the reference surface, was calculated. Thus, the surface of the original reference system was defined as the reference surface. The conclusion is that, the outside vertical sections have errors that far exceed those of the other sections. Specifically, standard deviation values calculated for them exceed 0.02m and reach up to 0.055m. In addition, in the 2nd and 3rd the arithmetic means are far from zero.

In contrast, for the other sections the calculated standard deviations have values from 0.007m to 0.009m with the arithmetic mean being either zero or 0.001m at absolute value. Thus, for all sections other than the outside-horizontal ones, it is safe to conclude that there is no systematic error. (Qassim, 2013). However, the errors that occur in the vertical external sections are not accidental and are most likely due to comparable points that are close to the surface but do not actually belong to it and were inadvertently measured by the user in the application. Ultimately, a safe conclusion is that the points measured with the developed tool belong to the reference surface with an ultimate value error of 0.01m.

5.4 Application's evaluation by users

In the context of assessing the usefulness, good operation, and ease of handling of the application, it was decided to evaluate it using an appropriate questionnaire. Specifically, 48 users of different ages, various levels of education, different specialties and different experience levels for using similar applications were asked to use the application developed and then evaluate it by answering a set of predefined questions.

The general conclusion of the evaluation is that the VR application created is easy to use for a user familiar with this type of technology and applications (Figures 11 and 12). However, some users with less relevant experience or ones that are older found it difficult to adapt and interact with the virtual environment. Moreover, users were sufficiently immersed to be able to enjoy the virtual reality experience. In terms of movement inside the virtual environment, teleportation was understood and used by everyone (Figure 13). In contrast, the flight mode made it difficult for some users who were completely inexperienced in video games and technology.

The application's measurement tools were considered easy to use. In particular, the distance measurement tool was characterized, based on the results, as easier than the tool for measuring and extracting points or sections which was difficult for many users (Figure 14). Thus, it is concluded that the tool for measuring and extracting points needs to be improved (Figure 15). Also, it is important to state that the application can be improved in all of its aspects and for that reason the opportunity was given to test users, through the questionnaire, to propose possible improvements.

Finally, the real fears and insecurities of each user are reflected in virtual reality environments. Thus, each developer must put himself in the place of such users to eliminate as many sources of discomfort as possible when using the application. For example, many test users expressed a fear of heights while using the flight movement tool while others expressed dizziness due to the immersion to the virtual environment.

6. CONCLUDING REMARKS

The development and implementation of the innovative application described in this paper leads to the general conclusion that the use of virtual reality technology can support many scientific disciplines and push them towards technological development. The engineering sector can make the most of virtual reality in terms of visualization and many more fields. Specifically, engineers related to 3D data of any kind can use this technology as an alternative and, in many cases, more complete method of displaying their data, but also as a way to interact with them when using the right tools. The depiction of three-dimensional models in a virtual reality environment is very realistic, which is not the case with other two- or three-dimensional imaging methods. In addition, virtual reality technology can contribute to the study of cultural heritage, to its preservation and its promotion.

The virtual reality system which was used was HTC Vive® due to its advantage of offering room scale tracking. Specifically, the user is able to use his physical movement in a predetermined space in order to move in the virtual environment of each application. In addition, the user can comfortably use the system by having his back turned to the computer as the sensors are not affected by the position and orientation of the user or the objects of the system. Furthermore, the game engine selected for the development of the application was Unreal Engine 4 due to its better handling of complex 3D models, better communication with the HTC Vive® and the blueprint virtual coding system.

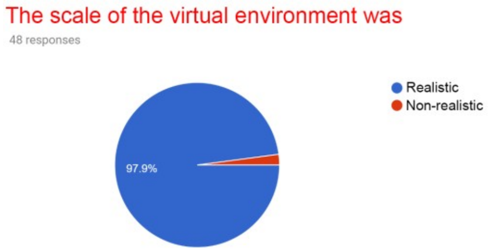

Figure 11: User response on the scale of the environment

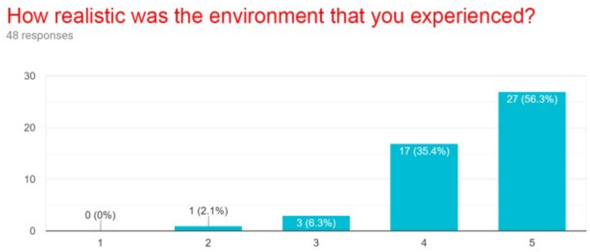

Figure 12: User response on the reality of the experience

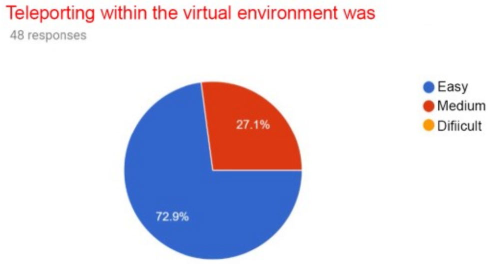

Figure 13: User response on the ease of teleporting

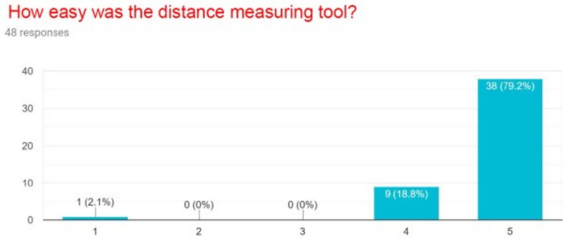

Figure 14: User response on the usability of the distance tool

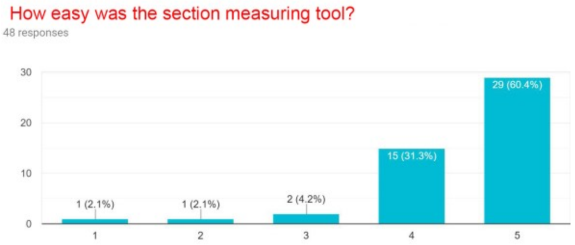

Figure 15: User response on the usability of the section tool

Eventually, the application was developed with two measuring tools: the distance measurement tool and the tool for measuring and extracting sections and three-dimensional coordinates of points. Using the 3D model of the Holy Aedicule as an object to be measured, which is a very detailed and accurate model, it is concluded that the application can work with any 3D model and the only limitation in its import are the capabilities of each computer that supports the whole system.

As stated in the section 5, the accuracy of both measurement tools compared to the reference surface is estimated to about 1cm. Thus, having as reference surface the original three-dimensional model basic measurement work, of supervision, control and study of a three-dimensional object that does not have very high accuracy requirements, can be carried out with sufficient accuracy. This metric error is traced to the coordinate system transformation which takes place as the fixed points have been selected by the user both in the application and in the reference surface and their identification depends on the accuracy of their selection. In addition, the accuracy with which the application selects and measures three-dimensional coordinates is unknown and may introduce errors throughout the process. Finally, the targeting error of each user while using the application plays a significant role. Finally, the evaluation process concludes that this is an unprecedented experience for most users who are interested in it. But using it can be improved a lot in order to make the user experience even smoother.

.

REFERENCES

Georgopoulos, A., Lambrou, E., Pantazis, G., Agrafiotis, P., Papadaki, A., Kotoula, L., Lampropoulos, K., Delegou, E., Apostolopoulou, M., Alexakis, M., Moropoulou, A., 2017. Merging Geometric Documentation with Materials Characterization and Analysis of the History of the Holy Aedicule in the Church of the Holy Sepulchre in Jerusalem. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLII-5/W1, 487-494, 2017 https://doi.org/10.5194/isprs-archives-XLII-5-W1-487-2017.

Heim, M., 1998. Virtual Realism, Oxford University Press, ISBN-13: 978-0195138740, pp. 263.

Kersten, T. P., Büyüksalih, G., Tschirschwitz, F., Kan, T., Deggim, S., Kaya, Y., Baskaraca, A. P., 2017. The Selimiye Mosque of Edirne, Turkey – An Immersive and Interactive Virtual Reality Experience using HTC VIVE®, GEOMATICS & RESTORATION – Conservation of Cultural Heritage in the Digital Era, 22–24 May 2017, The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII-5/W1, 403-409 DOI: 10.5194/isprs-archives-XLII-5-W1-403-2017.

Kontos, D., 2018. Development of a Measuring System in a Virtual Reality Environment. Diploma Thesis, Lab of Photogrammetry, School of Rural & Surveying Engineering, NTUA, pp. 189 (in Greek).

Kreylos, O. 2016. Lighthouse Tracking Examined. http://doc-ok.org/?p=1478 (last visit 24.04.2020)

Lanier, J., 2017. Dawn of the new everything: encounters with reality and virtual reality. Henry Holt and Company, ISBN 9781627794091, pp. 425.

Moropoulou, A., Korres, E., Georgopoulos, A., Spyrakos, C., Mouzakis, Ch., 2017. Faithful Rehabilitation of the Holy Tomb of Christ. American Society of Civil Engineers, Nov. 2017 (pp. 54-61,78).

Qassim, A.A., 2013. Mapping Matters, Photogrammetric Engineering and Remote Sensing, April 2013, pp. 319-320.

Slater, M., Usoh, M., & Steed, A., 1994. Depth of Presence in Virtual Environments. Presence: Teleoperators and Virtual Environments, 3(2), 130–144. doi:10.1162/pres.1994.3.2.130.