MeasureVR: Project Technovation

August 3, 2021

Authors:	Saad el Morabit and Ruben van Erp
Project Name:	Immersive technology for design, prototyping and training
Research Institute:	FNWI
Research Group:	Visualization Lab
Supervisor:	Robert Belleman

Contents

1 Introduction

Virtual reality (VR) technology has seen an increase in both usage and interest in the past few years. It has been used both as a recreational and professional tool. Virtual reality enables a greater level of immersion for users in a variety of applications by using both motion as an input, and taking over the whole field of view of the user using a VR headset. The application in this report, MeasureVR, takes the perspective of VR as an alternative user interface (UI) whose properties can be used to simplify tasks that are difficult to execute with more traditional UI.

The application in this paper does one thing; allow users to take measurements of 3D-models in a 3D-space. Traditional UI is designed around 2D-monitors, a mouse, and a keyboard, which is in most cases sufficient for handling 2D-data. However, when 3D-data must be processed the monitor has to project the object from a 3D-space onto a 2D-screen, "reducing" the amount of information the user has access to.

This projection is mathematically well understood and used in professional data science, computer graphics, and entertainment such as movies or video games. However, this projection that takes a dimension away from the user can make working with 3D-data difficult. In practice the user loads a 3D-model into their preferred choice of software, such as Blender, and then has to fiddle around with a virtual camera to get a good view of the area of interest of the 3D-model. From there the user either uses a mouse or tablet with their 2D-monitors to select a location in 3D-space, effectively eyeballing their selection which can lead to frustration as the user can not estimate depth well using a 2D-monitor.

This takes new users to 3D-programs a good amount of time to get used to, especially as different 3D-programs use different methods to work in 3D-space[1]. MeasureVR aims to solve this by removing the mouse, the keyboard, and the 2D-monitor all together. Moving through the real world's 3D-space is already intuitive enough for human beings to the point that mentioning it is trivial and obvious.

This intuition for 3D-space can be used by VR: the headset is used to take over the whole field of vision of the user, thereby "transporting" them to the 3D-space rather than showing them a slice of a 2D-projection of their 3D-model. If a user wants to see another part of the 3D-object, they can move their heads rather than move around an abstract virtual camera. The hands use a motion controllers whose buttons help the user move through menus, and whose movement through 3D-space allows the user to "directly interact" with objects in the 3D-space.

Interacting with the model by both showing the model to the users as if it were truly in front of them, and by having the users rely more on their hands and their motor function, the user has an easier time getting used to the program and becomes more efficient in taking manual measurements.

[1] Compare a program such as ZBrush with Blender. The camera moves around 3D-space using different keys and the two programs have different focuses when it comes to additional tools to help the user view their 3D-model.

In addition, a user who wishes to take measurements of a real world object can now scan the object and put it into a 3D-space by for example using photogrammetry. Not only does this remove the necessity of putting in every measurement using a keyboard and mouse, but it also allows the user to modify the mesh to have an easier time taking measurements compared to the real world objects. Imagine taking measurements of a 3D-model of a landscape rather than the landscape itself, to take large scale measurements or measurements at awkward angles the user can simply transform the 3D-model rather than waste time on complicated measurement techniques or keeping massive amounts of notes. The ability to take and keep records of measurements this easily would not occur in the wildest dreams of the Cassini maps, and is faster than traditional 3D-software as the user does not need to be wary that their measurements of a patch of grass is elevated because their virtual camera obscured the fact that their measuring points were raised off of the ground slightly.
The interest in using more of the hands range of motion is due to them being a highly expressive body part. The keyboard and mouse are a modern invention that only exploits a small part of the user's motor function. What motion controllers allow the program to do is to take in far more of the user's motor inputs, allowing for nuanced and complicated operations to happen more organically and efficiently. The hands are known to be the largest part of the sensory and motor homunculus which display how much of our somatosensory system is activated by our body parts, using more of our most expressive body part is the principal appeal of MeasureVR.

2 Empathizing

Whilst working on the original concept we discussed the multitude of applications that this program can help with such as the analysis of coral structures or measuring the proportions of 3D-scanned architecture. However seeing both authors were inexperienced in C# and that the application could accept whatever 3D-model the user puts in, the project ended up becoming more of a general measurement tool rather than being designed around a specific use.
The focus was therefore more on making sure the user had the tool set to straightforwardly analyze any non-pathologic 3d model, rather than be focused on a single set of 3D-models such as medical or archeological scans of crania. A basic tool set is given to the user to show them how powerful the program can be and helping users who want to take simple measurements. The program is also to be designed so that adding a new tool requires little effort for the modder. Seeing as we ended up making the program open source[2] at the end, this broader appeal can help in attracting a wider audience and thereby sustain the development of the program for a longer period of time.
It was important to make a good first impression, so we used the mathematical fact that almost every geometric property of interest can be measured with a set of points. Therefore with the first vertical slice the program was made to

[2]Specifically using the MIT license and posting the code on Github.

import data, export data, and have the user be able to set down two points from which in the program they can see the distance between the points.
The export file has the interesting property that not only does it return this distance, but it returns the coordinates of the two points as well which is not visible in the program during runtime. This was chosen to be done this way because showing both the distance and the locations of the points at the same time would have caused too much information to be displayed to the user. This way the program balances clarity and completeness, the user gets feedback to intuitive values such as the distance, area or angles, and if they want to see all the information they can see the export file.
To obtain a better idea of what VR users like and what they do not like we looked at both applications that allow the user to interact with a self-made 3D-object such as Tilt Brush, and AR measuring tools. Looking through the reviews and the way that these applications handle problems such as how the user puts down measuring points and handles complex geometry was a large help in deciding what would be useful for MeasureVR.
For example we found that people enjoyed Tilt Brush's menu where one hand is dedicated to the menus and the other to interacting with the menu to be good. From the AR apps we found that users really value a plug and play experience so we decided to maximize the speed at which a user can take a measurement over them getting all of the information handed or tools enabled at the start.

3 Definition

The program must take in an input of the user's VR headset, two controllers, and a 3D-model. Only an OBJ file with no textures is supported. This is due to Unity not having a simple import function for 3D model during run time. We used [3] with no textures (not supported in VR). Other formats such as FBX can be converted to OBJ as it is a generic file type.
The program puts out a generic text file with the measurements taken while using the program. This text file notes down the date and time of when a logged group of measurements have been exported. Every line is a single measurement with the name of the tool used, the location of all the measuring points, and values that were seen in the program such as the distance or the angle.
The program must be able to measure the distance between two points, the distance of multiple points, the area and circumference of a polygon, and the angle between points. This is done with three tools

- Single Distance Tool
- Multiple Distance Tool
- TriAngle (three angles from a triangle)

[3] `https://assetstore.unity.com/packages/tools/modeling/runtime-obj-importer-49547`

The single distance tool is the simplest of them all and lets the user put down two measuring points and returns the distance. TriAngle returns the three angles of a set and the location of the vertices. The multiple distance tool is a line of the multiple points. It returns the distance by default, but when "closed mode" it closes the line. This creates a polygon where the program returns the circumference and area.
These tools are sufficient in showing the multiple uses of MeasureVR and make a competent 1.0 version. The single distance tool will emulate a measuring tape for the user, showing them intuitiveness of the program, the multiple distance tool shows the user a higher level tool that can help in measuring more complicated values. The triangle is a tool which shows how not only distances can be measured, but also angles, as angles are difficult to measure with traditional programs.

4 Ideating

The ability to import 3d models into the program is highly limited due to the Unity engine, the engine of choice for this project, not having any standard method of importing 3D meshes during runtime. This was resolved by using the runtime OBJ importer, but the material import function of this asset did not function in VR.
No viable alternatives were found so a standard and easy to change (during development) material is used to add neutral turquoise material to any imported model. This standard material was chosen through trial and error as the most pleasant color to work with. As the neutral turquoise is easy to distinguish from the background, it is a "cool" color, and it is a relatively unusual color (unnatural) which invites the user in without being alarming.
In future releases it would be beneficial to have the user pick their material in runtime in a manner similar to VR architecture applications that allow the user to change the material on a chair during runtime. Ideally a texture can be imported as for certain application a texture can be informative, such as a model of the Uffizi where some of the details are textured in but can not be seen in the shape of the mesh.
When importing the mesh there are other problems that can vary on a model by model basis: scale and location. For example in applications like Autodesk Maya the default scale is in centimeters while in Unity it is meters, this can mean that models that are imported are either too tiny or too big for the user to reasonably work with. In addition, the model spawns in Unity's origin[4] which in some cases makes the mesh clip through the ground. The space in which the user works in only has a ceiling and floor so as to help the user's depth perception and to have a ground for the user to fall on and sky to limit them. In addition, the floor has a grid texture where the squares are 1 by 1 meters. This size was chosen as smaller squares were either not intuitive or made the floor cluttered and hard to read.

[4]The location (0, 0, 0) in Unity's 3D-space.

To solve these issues, and give a practical feature to the user, the user can scale the model and translate it in height. This is beneficial when handling meshes that are too small or big, for instance, the user can measure distances more conveniently with a small scale Sistine Chapel than the real scale. The user can also flip the whole mesh in all three axes if they scale it down enough[5], amounting to a parity transformation along all three axes. The translation in height is also useful as the user can not move a great deal in this direction in the real world without physically standing up or kneeling, so having the world translate to chest or face height is a work around this issue.

For the tools there were a large variety of options but three major values were chosen: angles, distances, and area. Adding in additional values such as volume can be incredibly useful as well, however trouble emerges in how to exactly define a volume in 3D space and how to make this intuitive to the user at run time. This problem can be seen to a smaller scale with the area calculation in MeasureVR: the area becomes ill defined when the polygon[6] is self-intersecting. There are many ways to solve this: warning or stopping the user from measuring the area of a self-intersecting polygon, or visualize the polygon so the user knows exactly what they are measuring. These solutions come with their drawbacks that have to be balanced for, such as obnoxious pop ups or obscuring their view with the polygon.

We also want users to be precise in their measurements, but need to take account of the imperfections of current VR technology. Allowing the users to change the scale of their mesh helps a great deal in that but also a snapping tool and fly tool have been added to improve the user's ability to take precise measurements. The flying tool is effectively a no-clip mode for the user where they use their controller to move around the camera, giving them complete freedom from the other more specific tools. The snapping tool is useful as it uses a ray path which detects the mesh and allows the user to stick a measuring point precisely on the surface of the mesh. The snap tool will fail with pathological geometry, but this is exception rather than the norm so the user can use the snap tool on regular geometry and do the more difficult parts manually.

The first version of the program will have all these options in front of the user as there are not too many tools shown to the user. However in future versions of the software it is of importance that a new UI is made where only the most used tools are in front of the user and the rest is chosen by switching menus. A solution may be inspired by Google's Tilt Brush, where the menus are always on the left side of the user's hand and can be moved through by rotating the hand. This is useful as it reduces the amount of menu clicking.

Accessibility is also of importance. A colorblind mode would be useful when the

[5] A scaling factor of 0.5 means the mesh is reduced in size by a half, and a scale factor of 2 means it has grown by a factor 2. Now as the scale decreases by a constant amount, the scale eventually becomes 0 and then turns negative. A negative scale being tantamount to a positive scale except the mesh gets mirrored along all three axes simultaneously.

[6] The polygon is made using the points from the multi-distance tool. The area function calculates the center of the polygon, simplifies the polygon into a set of triangles, and then calculates the area of the whole polygon by calculating the area of all triangles separately.

user base becomes both large enough for us to expect colourblind persons, and when the texture problem for meshes is fixes so the program includes more than three colors. Taking a system similar to that used in the video game Mirror's Edge where a filter is added for the most common forms of colour blindness.
In addition, allowing users to change the left hand to the right hand can be very useful to people who are left handed or ambidextrous who prefer the menu on their right. There is also the issue of language, the buttons are currently signified using English words which can be awkward for people who can not read English or can read the language but don't want to read through VR glasses. This could be solved by adding an audio mode, adding more languages, or by using abstract icons for the buttons.

5 Prototyping

First a vertical slice of the program is made where the user can grab two points with ease, a simple UI template is made for them to interact with, and the absolute basics of the import and export tools are made. The program lacks most of the features but is sufficient for the most banal of measurement tasks.
When this prototype was made some problems of design were found and bugs. First there was an obscure bug in the syntax where 3D-models that were imported did not have textures as discussed earlier, but also that it would lag out the VR application due to improperly importing the mesh multiple times. There was also the issue of where the OBJ goes and where the export .txt files go on the device of the user. The user's documents folder was possible but was changed to the data folder that the program comes with. Rather than changing the program to a standard .exe it was decided to keep the standard format of Unity projects when they are build. This was useful for we added two folders, an import and export folder, where the user can move their OBJ file into and find their measurements in respectively.
Users who put in multiple OBJ files can expect only the first model found by Unity to be loaded in, and users get a single .txt file with all measurements with timestamps added in so we do not have to worry about spamming the user with a large amount of text files.
The first vertical slice, except for the broken import function, was a success but we were unable to find testers for it as the COVID epidemic made it difficult to set-up a VR headset somewhere and ask curious potential users to test it out.

6 Testing

Now that the program has been completed to a sufficient 1.0 state it is considered ready for release on itch.io. The program was handed to two friends of the developers who tested it out. One of them used their person Valve Index and is a casual user with a lot of experience using their headset. The program clicked quickly except for the pin tool which was awkward to use. A problem that was

early on expected was that the OpenXR library that was used did make the program compatible with headsets other than the Oculus Quest 2 as originally planned, but that differences in controllers could cause problems. This can be helped by creating future patches where the OpenXR library sees which VR system is being used and then changes to an alternative control scheme.
An interesting result however was that when handing it to one of the users the user having "exhausted" their interest in the Stanford Bunny and Arc de Triomphe models that were given to them, started grabbing OBJ files from their own computer. They had an installation of the videogame Half Life Alyx and wanted to use the opportunity with the program to explore the models of the game out of their own interest. The user had a layman knowledge of computer graphics but by manipulating and moving through the game assets quickly understood where and what the models were for. We believe it is unlikely that this will become a primary use of MeasureVR, but would not be surprising if the simple drag and drop method of MeasureVR is used by future users as a way to quickly analyze 3D-models that are interesting to them.

7 Discussion and Conclusion

The program is, even in the context of the programmers their inexperience, and the Covid pandemic complicating testing, a success. The program is both usable and stable, and does a great job of being easy to jump into and quickly take measurements.
Where it is still lacking however is doing more large scale measurements as the user is unable to see the export log without exiting the program. In addition, the user can only see a handful of logged values without being able to scroll back or remove/modify measurements they have made. The pin tool was also problematic as the user can return the measuring point back to their hand which is inconvenient in many scenarios. The user can learn how to best handle this and it does not break the measuring progress but it is a high priority fix as the snap tool suffered from bugginess due to the OpenXR toolkit lacking certain functionality.
With the program finished and the problems listed for future updates to fix and optimize, the project is finished. The program is to be first released on itch.io for easy access. The storefront is well known for hosting more experimental and smaller works so the feedback possible through it will be useful.
As the program gets updated it will be offered to researchers at the University of Amsterdam and other users of VR so as to ascertain their experience for future developments. If the program becomes well developed enough it can be uploaded to bigger or more specific fronts such as the Sidequest store or even the Oculus store.
This requires additional improvements on the program and making the program work on the Oculus Quest 2 without a pc, as the Oculus Quest 2 does not allow easy access to users who want to put in a 3D-model or output a text file. For this a net solution would be possible or finding a manner in which to save and

put in files into Quest 2's public folders[7].
The UI can also use some improvements but these are preferably based on user experience. As the program expands additional documentation ought to be made for new users and users who want to create their own tools. For the time being the program can be displayed as a proof of concept which we hope to be useful for future researchers handling 3D-data and as a method for users to take down the proverbial "monitor wall" and directly interact with their 3D-model.
In summary, the program has been a success with some downsides and limitations for more advanced use. The program is open to the public and the creators will watch over it, with a consumer installation on itch.io and MIT Licensed Github repository for the code. Future updates will try to iron out the issues listed in this report and try to improve the code based on user feedback.

[7]The folders viewable from a USB connection with a computer.