International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2016)
D. Reiners, D. Iwai, and F. Steinicke (Editors)

# Synchronized Scene Views in Mixed Virtual Reality for Guided Viewing

I. Vazquez and S. Cutchin

ikervazquezlopez@boisestate.edu, stevencutchin@boisestate.edu
Boise State University
1910 W University Dr, Boise, ID, USA 83725

**Abstract**

*Virtual Reality devices are available with different resolutions and fields of view. Users can simultaneously interact within environments on head mounted displays, cell phones, tablets, and PowerWalls. Sharing scenes across devices requires solutions that smoothly synchronize shared navigation, minimize jitter and avoid visual confusion. In this paper we present a system that allows a single user to remotely guide many remote users within a virtual reality environment. A variety of mixed device environments are supported to let different users connect to the system. Techniques are implemented to minimize jitter and synchronize views, and deal with different fields of view.*

Categories and Subject Descriptors (according to ACM CCS): I.2.10 [Computer Graphics]: Vision and Scene Understanding—Motion I.3.2 [Computer Graphics]: Graphics systems—Remote systems

## 1. Introduction

An expected utilization of virtual reality systems in the classroom is for teachers to provide students with directed tours and navigation through virtual environments. It is expected that students will benefit from being able to virtually browse through a relatively endless environment of virtual locations, including reconstructions of ancient historical sites, photo-realistic immersive captures of important world-wide locations as well as potentially simulated space travel. Classes utilizing virtual reality devices will require effective and comfortable guided navigation so that students can smoothly follow the instructor's view without discomfort or confusion in a synchronized manner. Excess network induced shaking (jitter) can lead to rapid deterioration of shared views and modest lag can lead to nausea for remote students. Because of this it is advisable that shared virtual reality systems utilize view synchronization to minimize these types of extreme events. Additionally, virtual reality devices are available with different fields of view(FOV). Users can simultaneously interact within virtual reality environments on head mounted displays(HMD), cell phones, tablets, and PowerWalls. Sharing scenes across different types of devices requires solutions that both minimize jitter and deal with different FOVs.

In this paper we present a system that allows a user to guide multiple remote users within a Virtual Environment (VE) of stereo panoramas. Specific techniques are studied to minimize jitter between users and to maintain synchronized views across devices with differing FOV. These techniques' results showed how the users prefer smooth transitions over drastic ones. The implemented system provides a collaborative communal space in mixed device environments of HMD, tablet, and PowerWalls.

This paper is structured as follows: in Section 2 the state of the art is presented as well as some techniques that are used for smoothing. Section 3 describes our approaches used to reduce jitter and deal with different FOVs. In Section 4 we describe the implemented system architecture for scene sharing and testing. Different test cases for this system are presented in Section 5. Finally the results of the tests and some discussion about future work are shown in Section 6 and Section 7.

## 2. Prior work

Significant research into Distributed Virtual Environments has already been done for a variety of devices. Problems that were researched were related to synchronizing remote devices in order to provide good Quality of Experience(QoE) and to reduce network traffic to maintain performance. Work has also been done around the physical symptoms virtual environments can cause to the user, such as nausea and confusion.

Prolonged use of virtual environments and it's effects are studied in [LJ00], a investigation of the different symptoms of cybersickness within virtual environments can cause to the users is presented. This study showed that some users using a virtual environment, for a long period of time, exhibited motion sickness which may happen with visual stimulation and no vestibular stimulation. Users which feel some sickness may stop using the system and in





DOI: 10.2312/egve.20161438

the future can be opposed to use any kind of virtual reality devices due to the bad experience. Sickness symptoms chance may increase with wide FOV of the camera and the movements of it. Some devices (like HMD) may not be precise enough to track user's orientation and thus produce a jittering effect. This effect leads to a bad QoE of the system due to the small and fast movements. A network with a high latency causes some cyber-sickness (jittering) symptoms too. This sort of laggy network problems are stated in [PK99], where the study showed how the delay affects the users when completing a collaborative task in a virtual environments. The attention of the humans is focused on stationary things, therefore when there is a movement in a static scene it is easily percepted; with a big amount of visual conflicts in the scene the users may get disoriented.. Tasks with high speed changes in the view need a gradual movement to reduce the disorientation, this is, the user needs to adapt to the movement. The cyber-sickness caused by this changes can be reduced with adaptation strategies which lead to the user to a smooth adaptation instead of a drastic one; this is one of the problems we are going to focus on.

Dealing with differing FOVs of different devices has been studied, in [JK08] a combination of different cameras to provide a bigger FOV is presented and in [BSR*12, FF16] experiments involving blinders, software, and augmented virtual systems to change the FOV. The experiments demonstrated the difference in a user's experience based on the differing sizes.

In [Ebe03] the *smoothstep* interpolation function is presented to smooth sharp changes. This function is an *ease in/ease out* function which means that the approach to the maximum and minimum values are smoothed, this behavior is going to be one of our goals for the follower's paths: get an *ease in/ease out* smoothing. It is included as a function in RenderMan® [DNC*01]. This kind of functions help to smooth drastic transitions in the camera movements. In [PBG92] this type of functions are used to smooth camera movements while interacting with 3D objects and provide better QoE to the user. [LaV03] uses a double exponential technique to smooth predictive tracking and orientation; the method is compared against the *Kalman* filter and the *Extended Kalman* filter, showing that the double exponential is more efficient with similar results. [DJB*98] presents a navigation algorithm based on a river flow analogy, following the approximate flow of an anchored river current used for smoothing the navigation path in a virtual environment system. This method allows to predict next movements if the system did not receive the required information; and when it is received a smooth approximation is applied to reach the real position. In [Kha12] the Dead Level Reckoning is used for calculate intermediate positions in a path and reduce the traffic through the network.

[MZP*94] presented a simulator which allows a large number of users at the same time using multicast groups, sharing only crucial information regarding to the area of interest of each user. [GLKP94] is a tutorial for students about how to implement a distributed multi-user system using different techniques like multicasting and broadcasting for communication, and Dead Level Reckoning for user position prediction.

[JCT*09] presents a web-based learning environment called Virtual Laboratories (VL). [ZGL*07] developed a collaborative P2P system which shares a geographic environment to explore complex spatial information and work in a collaborative way. [SBGH09] presents Wonderland, an integration of internet accessible physics experiments (iLabs). [IA07] present a collaborative system to teach students how to use a calligraphy brush with haptic devices. In [GNTH14] a virtual scene navigation system for remote collaboration is presented, where the remote user can insert marks in the scene for the local user. The marks are positioned in the computed 3D scene using the SLAM technique. The remote user can control a virtual camera independently from the local user. The camera is frozen when the remote user wants to add a new mark to the scene, and when finishes the camera jumps to the local user's looking point; if the remote user freezes the camera a considerable amount of times the experience may be uncomfortable due to the inconsistent visual information.

[Sze10] detail a variety of image based modeling and panoramic viewing and manipulation approaches that are utilized heavily throughout the implementation of the viewing and computing components of this work.

Many different types of virtual reality devices and systems in differing physical sizes, configurations and resolutions have been developed [CNSD*92, DDS*09, SPWD13]. Coupled with the development of omnidirection stereo capture of panoramic views [ASS*11, YMDK11, KUDC07, Che95, PCD*12, SBD*10, KW09, Ken08, HH98, PBE99, PBEP01, dTM, PZF10] there is a lot more content available for immersive interaction with stereoscopic panoramas in mixed virtual reality environments and devices.

## 3. Our Methods for Synchronized Viewing

Our approach utilizes a leader and follower model with one virtual reality system acting as the leader (leader) and all others acting as followers (followers). Each system loads the same scene in advance to avoid high data traffic through the network and all systems have identical imagery and user interface tools. The actual appearance of the user interface may differ from physical device to physical device due to the device constraints but the same functionality is present in all.

When users follow another user it can easily become uncomfortable for the followers. Followers do not know what is going to be the next movement of the leader, and thus they may suffer some disorientation due to conflicts between expected and actual visual changes. This conflict leads to a bad QoE. Our different approaches are focused on addressing this problem in an attempt to maintain QoE. We divided the conflict into two parts: the first part deals with jittering transitions of movements and the second one with conflicts related to differing FOVs.

### 3.1. Jitter reduction

The leader user can move freely, therefore he/she knows the flow the system is going to have, and is prepared for it. The followers do not know anything about the leader's next movement; this lack of knowledge combined with large view changes will disorient the user, but the disorientation is user specific. Jitter can occur when the network connection has high latency; packets are received after a long interval of time and thus transitions are very drastic.

Jitter reduction is managed in the smoothing server to provide every follower with the same adjusted coordinates. If this is performed in the follower clients and some packets are lost, the smoothing will be different for some followers breaking the shared group experience. The server receives packets from the leader and transmits the computed adjustments, therefore even though if there are lost packets in the sever-follower communication the system will continue reliably. We implemented two techniques: Dead Reckoning and Weighted Average, but we selected the Weighted Average to use in this study due to similar results and simplicity.

**Weighted Average**

This approach averages the leader's coordinate points based on a buffer over the number of coordinate moves in order to return smoothed coordinates to the followers, see equation 1, where $n$ is the buffer size and $w_i$ the weight of the $x_i$ element (the coordinate value) in the buffer. The leaders is constantly sending point coordinates he/she is looking at; the server receives these coordinates and pushes back into a buffer with a maximum size to compute the average. The maximum size of the buffer allows for fine tuning of the shape of smoothed movement path: with a bigger buffer size yielding smoother motion and a smaller buffer size yielding less smooth motion. When the buffer is full the oldest coordinate is deleted from it and the new one is inserted in the back. The buffer size is adjusted based on application requirements to achieve the level of smoothness required by the participants.

$$x = \frac{1}{n} \sum_{i=0}^{n} w_i x_i \qquad (1)$$

Each coordinate in the buffer has a weight associated with it. Weights imply an importance of each element when computing the average: if the first elements have a lower weight than the last ones, the average will tend to be closer to the last elements of the buffer. This will lead to more drastic transitions because the new leader coordinates will pull more than the old ones and the jump between the coordinates sent to the followers will be bigger.

The configuration of the weights depend on the behavior we want to offer, similar to the buffer size, depending on the application and the QoE we want to provide to the users. If the application needs a very close following of the leader, the buffer size and the weights should be configured to provide a minimum smoothing to reduce the jitter and follow the leader's movements very closely.

## 3.2. FOV control

In addition to the jittering effect different type of devices can connect to the system and these devices will have different constraints. We focused on the difference of the FOV. Even though it is possible to adjust each device's FOV to provide a single FOV for all devices, the potential of some of them will be considerably reduced. Narrow FOVs in PowerWalls and wide ones in tablets will affect the user experience negatively. We have developed three methods to study which is the better approach to deal with different FOVs. These methods are computed in the follower client to deal with each device's specific FOV.

The methods for FOV control have the same behavior: while the FOV of the leader is inside the FOV of the follower the camera is still (see Figure 1), once it exits one of the three methods is applied to provide a smooth transition until it reaches the leader's looking point (see Figure 2).

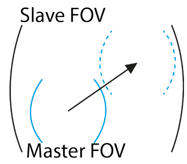


**Figure 1:** *Master's FOV moves inside follower's.*

### 3.2.1. Instant transition for FOV control

Since this method is not a smoothing technique it is used as a baseline for the study. This method relays on matching the leader's looking point in one movement. The current looking point of the follower is matched with the leader's immediately, the same way it is done in [GNTH14].

### 3.2.2. Functional smoothing for FOV control

When the leader's FOV exits the follower's the difference of the $x$ and $y$ coordinates are calculated. These two values are used to create an adaptive function for each coordinate to interpolate the coordinates from the initial point to the final in a smooth manner. We studied four different functions for the interpolation: *gaussian* function, *sinusoidal* function, *Bezier* curve and *inverse* function; our goal with these functions was to have an *ease in/ease out* behavior.

**Gaussian function**

The Gaussian function is the curve that fits better with our goal of smooth interpolation due to its shape. From the difference of the $x$ and $y$ coordinates the parameters for the function are calculated as it is shown in equation 2, where $m$ is the leader's view point coordinate, $s$ is the follower's and $x$ is the actual intermediate coordinate value we use to calculate the next Gaussian value. We selected specific denominators for the fractions based on the experimental testing to get the desired adaptive curve, but they can be adjusted to represent another function which fits better with any desired performance. The value the Gaussian function returns depends on the difference of the leader and follower, with higher difference a higher value is returned and a faster transition is achieved.

$$f(x) = \frac{|m-s|}{16} \times e^{-\frac{\left(x-\frac{m+s}{2}\right)^2}{2\left(\frac{|m-s|}{4}\right)^2}} \qquad (2)$$

This process is an iterative backward feeding process until it snaps to the leader's view point. In each iteration the actual coordinates of the follower are fed to the Gaussian function and the returning values are added to them to get the new coordinates of the follower, see Figure 3. Since the equation 2 is an adaptive function the first and last values are low while the values in the middle are higher, this ensures the *ease in/ease out* behavior.

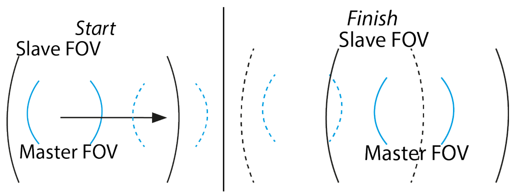


**Figure 2:** *Master's FOV exits follower's.*

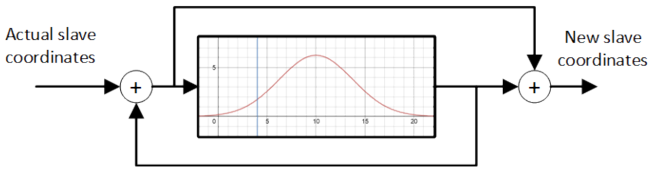


**Figure 3:** *Gaussian backward feeding smoothing system.*

#### Sinusoidal function

The *sin* function shape is similar (see Figure 4) to the Gaussian function, the difference between the two functions is minimal for our purpose if we adjust the parameters to match them. Since the Gaussian function is an exponential function the computation of it is more expensive than the *sin* function, thus we studied the results the *sin* function can provide. The same way as the Gaussian function, we used the parameters of the function to adapt the curve and get the transition speed we need to use in each step of the interpolation. In each iteration the value from the sinusoidal function is added to the current coordinates and in the next iteration a new value is computed from these new coordinates. In this case adapting

**Figure 4:** *Difference between a Gaussian function and the* sin *function. Where red line is the Gaussian function, the blue line is the* sin *function and the green line is the difference between them.*

the parameters to get the adaptive curve is very complex and after some experiments we did not reach to a solution to this problem. The parameters of the function are too high to use them to change the shape, because with a small variation of them the curve change is considerably high.

#### Bezier curve

Using the Gaussian and sinusoidal functions the final result was very similar to a *sigmoid* function, but this function is an exponential function, therefore the cost of computing it is similar to the Gaussian function. Bezier curves can be used to represent a similar *sigmoid* function using the *Bernstein* polynomials. Adjusting the control points of the curve we created individual curves for each coordinate. The values for the transition are calculated based on the Bezier curve generated by the slave's coordinates and master's coordinates as control points, but the calculation is computed independently for the $x$ and $y$ axis. Combining the two coordinates' result the behavior of the transition from the slave's viewing point to the master's one is the *ease in/ease out* movement.

#### Inverse function

Instead of using a Bezier curve we considered using a dual inverse function to get an *ease in/ease out* shape. When the leader exits follower's FOV the average point of the difference in each coordinate is computed. For the coordinate values lower than the average one function is applied and for the higher ones another one is used as it as shown in the equation 3, where $t$ is the time step like in the Bezier curve.

$$f(x) = \begin{cases} -5\frac{1}{t-5} & if\ t_{<}5 \\ 5 & if\ t = 5 \\ -5\frac{1}{t-5} + 10 & if\ t_{>}5 \end{cases} \tag{3}$$

When dealing with different coordinates the shape of the dual function is hard to maintain, so we scaled the coordinates to map into a range in which the shape is as desired and we can control in a simple manner. In our case we mapped into the range $[0, 10]$ and the asymptote is located in the average of that range, in this case 5. The shape of the mapped function looks is shown in Figure 5. Once we have extracted the value we convert it to the previous scale.

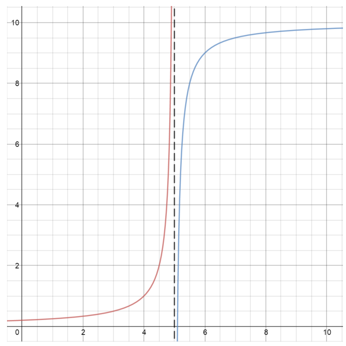

**Figure 5:** *Resulting inverse dual function in the range* $[0, 10]$.

### 3.2.3. Step sizes transitions for FOV control

This method uses linear interpolation. When the leader's FOV exits follower's the follower starts following the leader until it snaps to the leader's coordinates. Each time the follower receives the coordinates from the leader a direction vector to the leader is computed and the follower moves a fixed percentage of the length of the vector towards the leader, see Figure 6. If the leader does not move a certain amount of distance the vector will still be the same. The follower will get closer step by step through the vector until it reaches the leader. If the leader moves while the follower is approaching, a new vector is computed from the follower's looking point overriding the previous one and changing the movement direction. This method often yields the desired *ease-in/ease-out* behavior.

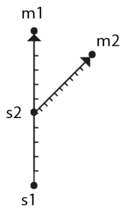


**Figure 6:** *Step sizes approach to leader's looking point.*

**Figure 7:** *iPad Web Client.*

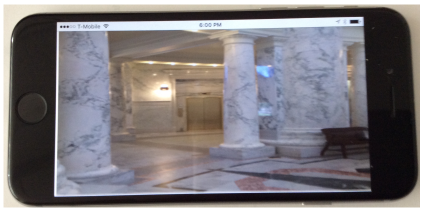

**Figure 8:** *iPhone Web Client.*


**Figure 9:** *GearVR Android Client.*

## 4. System Architecture

Our testing platform is a stereo panorama viewing application. Multiple users view high-resolution stereoscopic panoramas simultaneously. They may associate their client with a leader application and this leader leads them on a guided tour. The clients follow the leader's view direction, position, and panorama changes. Stereo viewing is supported in GearVR and PowerWall systems, while the WebGL version is limited to monoscopic views. See snapshots in Figures 7,8,9,10.

The panorama application presents a partial view of the full panorama with FOV determined by device characteristics. The user may look at any portion of the panorama via device specific controls. The mechanism varies by device: keyboard, mouse, touchpad, or gaze. Once a client followers itself its view and panoramas are completely controlled by the leader. Whenever the leader adjusts its view or changes panoramas commands are transmitted to the followers.

Communication is done via multicast UDP with a jitter server used to smooth motion between leader and followers.

The mixed device environment consists of a collection of clients for several platforms and communication servers to provide smoothing and interfacing services across the interacting group.

The different servers and applications implemented:

**Smoothing** Server acts as a buffering server that receives packets from the master controlling the overall view of the current collaboration, smooths them based on system defined window parameters and forwards them on to all slaves. It receives packets via UDP, and broadcasts to all slaves via multi-cast UDP, ensuring that all the slaves will receive the same data. Slaves with high latency will have high delay on the viewing path of the master.

**GearVR** Android Client is an Android application implemented using Samsung's GearVR framework that runs on the Samsung Galaxy Note S5 phone within the GearVR headset. The user view is controlled by the direction that the leader is looking and is streamed via UDP into the smoothing server. The system uses a separate thread to handle received packets when in follower mode and updates the view based on the received commands. The clients control of the view is disabled when in follower mode and only the remote orientation is available.

**PowerWall** Client is a C++ implementation of the panorama stereo viewer that runs on top of OpenSceneGraph and OpenGL that can display stereo panoramas at 100 Megapixel resolution on large multi-panel displays using mutli-channel graphics cards. This client when driving five simultaneous displays has an extremely wide FOV. The testing of mismatched FOVs was done between this system and clients on singular display clients.

**WebClient** The WebClient viewer is implemented using Javascript and WebGL. It can load 16K panoramas but only supports mono-panoramas. The user interface is handled via arrow keys on desktop systems and device physical orientation when running on mobile devices without a keyboard. Currently web browsers do not support UDP messaging except in an experimental mode. This required that the communication networking layer be implemented using WebSockets and the introduction of a separate repeating WebServer interface that forwards instructions to all of the UDP based clients.

**WebServer** acts as the intermediary between the web clients and the UDP based clients. It accepts connections from all WebSocket based clients and sends out UDP packets to the group network for every received WebSocket command sent as well as forwarding received commands to the other WebSocket clients. For received UDP packets it forwards the packets over the set of WebSockets to connected web based clients.

**Figure 10:** *PowerWall Stereo Viewer.*

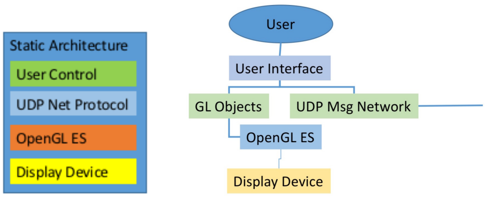


**Figure 11:** *Static and Runtime Architecture.*

### 4.1. Hardware Environment

Implementation and testing was done using the following hardware:

**PowerWall** A PowerWall system using five LG 65" Passive Stereo display systems in portrait mode driven by a single Dell workstation using an ATI Radeon graphics card for shared immersive stereo panorama viewing.

**GearVR** Two GearVR head mounted displays driven by Samsung Galaxy S5 phones were used for private head mounted display interaction using android apps with pre-loaded panoramic imagery.

**iPad** Apple iPad tablets were available but were not used in the testing protocols. However the WebClient implementations run effectively on them in mono stereo mode for viewing panoramas.

**Dell PC** WebGL implementations running in Chrome browsers on Dell workstations with keyboard viewing interfaces were used for desktop interaction and group viewing within the mixed device environment.

All devices and systems used within the environment were located on the same local area network. Simple experiments were performed with wide area networks but multicast UDP packets were being lost too frequently to be useful in initial experiments.

## 5. Test Cases

We selected five individuals to take tests evaluation the implemented techniques. All members of the testing group were volunteers from the same program as the authors. The age range of the evaluation group was between 24 and 27. Some participants had prior experience with head mounted displays but not all. The recorded details of each participant are shown in the Table 1.

The tests were preformed in the same laboratory room as the development environment. All testers were familiar with the environment and comfortable with the location. Each volunteer took

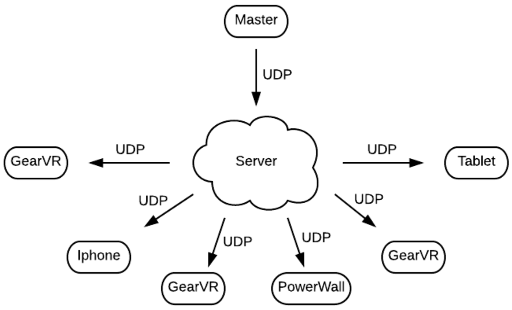


**Figure 12:** *Mixed Device Network.*

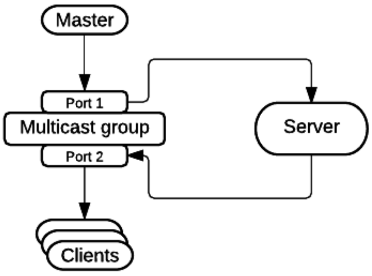


**Figure 13:** *Network Architecture*

three tests, one for each method. In each test the user worked independently within a single virtual environment showing a immersive spherical panorama. All tests utilized the same panoramic image for all users. The individual test duration was one minute per method, as see in Table 2, with another minute between tests to rest and fill out a questionnaire related to it. Every test used the same guided path to maintain a consistent experimental experience and record consistent feedback with the different methods.

Since we have a considerable amount of techniques to apply and some of them have a similar behavior, we selected only some of them for testing. With a bigger amount of test cases to test, the users may get tired due to the long exposure to a virtual environments and may not provide the best feedback for each technique. We skipped some of the techniques of the test cases which are similar to avoid the struggling of the testing group and we selected the ones that, from our point of view, are the best ones to apply.

For jitter reduction we have two techniques: Modified Dead Level Reckoning and Weighted Average. We selected Weighted

| ID | Age | virtual reality experience | Visual problems |
|---|---|---|---|
| U1 | 27 | Yes | No |
| U2 | 25 | No | No |
| U3 | 25 | Yes | Yes |
| U4 | 24 | Yes | Yes |
| U5 | 26 | No | No |

**Table 1:** *Testing group user description.*

| Test | Duration(s) | No. Questions |
|---|---|---|
| Instant | 60 | 3 |
| Step Sizes | 60 | 3 |
| Bezier | 60 | 3 |

**Table 2:** *Test Duration and number of survey questions*

| User | Instant | Step sizes | Bezier |
|---|---|---|---|
| U1 | 1 | 4 | 4 |
| U2 | 1 | 4 | 3 |
| U3 | 1 | 3 | 3 |
| U4 | 1 | 4 | 4 |
| U5 | 1 | 3 | 3 |

**Table 3:** *Results of the different methods.*

Average for testing due to the similar behavior and path adaptation with the weights.

For the FOV control we selected the instant technique as a baseline, one method from the functional techniques, and the step size method. We test the Bezier curve method of our functional techniques. The user interaction goal for the functional techniques is to get a behavior where the initial motion begins smoothly with an increase in speed of motion after movement has begun and a gradual coming to a stop when the remote user has stopped moving. We refer to this as *ease-in/ease-out* . We selected the Bezier curve due to the fast computation of the function and the intuitive nature of controlling the shape of the curve to provide an intuitive understanding over the follower's path.

## 6. Experimental Results

The test group was asked to fill out questionnaires after each method test. For each method the user had to answer two questions: *Were the transitions comfortable?* and *Did you have any disorientation or confusion?*. The user rated their experience with each of the methods on a 5 point scale. All the tests had the same duration (60 seconds) and the same questions as shown in Table 2 .

The results of each test case showed consistent agreement of all test subjects. For instant transitions the group was not comfortable and confirmed that they suffered some disorientation and rated the method as poor, see Table 3. The overall response about their experiences with this method was that it was unpredictable, chaotic and confusing.

The Step Sizes and Bezier curve methods showed an improvement for the test group. They were comfortable with the transitions and did not suffer as much disorientation as in the instant transition. They marked the techniques with acceptable and good scores, see Table 3.

The users preferred the smooth transitions over instant changes over instant changes or jitter movements in the view.

## 7. Conclusions and future work

The implemented mixed device environment supports the interaction of multiple users on devices with differing FOVs. Several techniques were implemented and tested with preliminary results demonstrating a preference for step wise directional smoothing. However the number of test subjects was limited and more testing needs to be done. Initial results indicate promising utility for smoothly synchronized shared views in virtual environments.

Moving forward we intend to expand the number of test subjects and to introduce a more rigorous testing program. Particularly performing testing on Wide Area Networks in addition to the current LAN based testing. Comparing these additional test results more directly against other researchers scene sharing systems would be useful as well.

Additional useful features that would enrich the collaboration environment include visual and audio cues between group members, simple audio chat and highlighting for a participant the active view point for all other group members so that an individual can see specifically what any other group member is looking at at any given moment of collaboration. Finally adding visual tracing of participant motion and a monitoring viewer mode that shows all participants view orientations could also be useful for collaboration and further research.

## References

[ASS*11] AINSWORTH R. A., SANDIN D. J., SCHULZE J. P., PRUDHOMME A., DEFANTI T. A., SRINIVASAN M.: Acquisition of stereo panoramas for display in vr environments. In *IS&T/SPIE Electronic Imaging* (2011), International Society for Optics and Photonics, pp. 786416–786416. 2

[BSR*12] BOWMAN D. A., STINSON C., RAGAN E. D., SCERBO S., HÖLLERER T., LEE C., MCMAHAN R. P., KOPPER R.: Evaluating effectiveness in virtual environments with mr simulation. In *Interservice/Industry Training, Simulation, and Education Conference* (2012). 2

[Che95] CHEN S. E.: Quicktime vr: An image-based approach to virtual environment navigation. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), ACM, pp. 29–38. 2

[CNSD*92] CRUZ-NEIRA C., SANDIN D. J., DEFANTI T. A., KENYON R. V., HART J. C.: The cave: audio visual experience automatic virtual environment. *Communications of the ACM 35*, 6 (1992), 64–72. 2

[DDS*09] DEFANTI T. A., DAWE G., SANDIN D. J., SCHULZE J. P., OTTO P., GIRADO J., KUESTER F., SMARR L., RAO R.: The starcave, a third-generation cave and virtual reality optiportal. *Future Generation Computer Systems 25*, 2 (2009), 169–178. 2

[DJB*98] DIGIOIA A. M., JARAMAZ B., BLACKWELL M., SIMON D. A., MORGAN F., MOODY J. E., NIKOU C., COLGAN B. D., ASTON C. A., LABARCA R. S., ET AL.: Image guided navigation system to measure intraoperatively acetabular implant alignment. *Clinical orthopaedics and related research 355* (1998), 8–22. 2

[DNC*01] DISSANAYAKE M., NEWMAN P., CLARK S., DURRANT-WHYTE H. F., CSORBA M.: A solution to the simultaneous localization and map building (slam) problem. *Robotics and Automation, IEEE Transactions on 17*, 3 (2001), 229–241. 2

[dTM] DĂĆÂĂÂŹANNIBALEA E., TASSETTI A., MALINVERNI E.: From panoramic photos to a low-cost photogrammetric workflow for cultural heritage 3d documentation. 2

[Ebe03] EBERT D. S.: *Texturing & modeling: a procedural approach*. Morgan Kaufmann, 2003. 2

[FF16] FERNANDES A. S., FEINER S. K.: Combating vr sickness through subtle dynamic field-of-view modification. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)* (2016), IEEE, pp. 201–210. 2

[GLKP94] GOSSWEILER R., LAFERRIERE R. J., KELLER M. L., PAUSCH R.: An introductory tutorial for developing multiuser virtual environments. *Presence: Teleoperators & Virtual Environments 3*, 4 (1994), 255–264. 2

[GNTH14] GAUGLITZ S., NUERNBERGER B., TURK M., HÖLLERER T.: World-stabilized annotations and virtual scene navigation for remote collaboration. In *Proceedings of the 27th annual ACM symposium on User interface software and technology* (2014), ACM, pp. 449–459. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.678.803&rep=rep1&type=pdf. 2, 3

[HH98] HUANG H.-C., HUNG Y.-P.: Panoramic stereo imaging system with automatic disparity warping and seaming. *Graphical Models and Image Processing 60*, 3 (1998), 196–208. 2

[IA07] ISHIBASHI Y., ASANO T.: Media synchronization control with prediction in a remote haptic calligraphy system. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology* (New York, NY, USA, 2007), ACE '07, ACM, pp. 79–86. URL: http://doi.acm.org/10.1145/1255047.1255063, doi:10.1145/1255047.1255063. 2

[JCT*09] JARA C. A., CANDELAS F. A., TORRES F., DORMIDO S., ESQUEMBRE F., REINOSO O.: Real-time collaboration of virtual laboratories through the internet. *Comput. Educ. 52*, 1 (Jan. 2009), 126–140. URL: http://dx.doi.org/10.1016/j.compedu.2008.07.007, doi:10.1016/j.compedu.2008.07.007. 2

[JK08] JEO S., KIM G. J.: Providing a wide field of view for effective interaction in desktop tangible augmented reality. In *Virtual Reality Conference, 2008. VR'08. IEEE* (2008), IEEE, pp. 3–10. 2

[Ken08] KENDERDINE S.: The irreducible ensemble: place-hampi. In *Virtual Systems and Multimedia* (2008), Springer, pp. 58–72. 2

[Kha12] KHARITONOV V. Y.: Motion-aware adaptive dead reckoning algorithm for collaborative virtual environments. In *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry* (New York, NY, USA, 2012), VRCAI '12, ACM, pp. 255–261. URL: http://doi.acm.org/10.1145/2407516.2407577, doi:10.1145/2407516.2407577. 2

[KUDC07] KOPF J., UYTTENDAELE M., DEUSSEN O., COHEN M. F.: Capturing and viewing gigapixel images. *ACM Transactions on Graphics (TOG) 26*, 3 (2007), 93. 2

[KW09] KWIATEK K., WOOLNER M.: Embedding interactive storytelling within still and video panoramas for cultural heritage sites. In *Virtual Systems and Multimedia, 2009. VSMM'09. 15th International Conference on* (2009), IEEE, pp. 197–202. 2

[LaV03] LAVIOLA J. J.: Double exponential smoothing: an alternative to kalman filter-based predictive tracking. In *Proceedings of the workshop on Virtual environments 2003* (2003), ACM, pp. 199–206. 2

[LJ00] LAVIOLA JR J. J.: A discussion of cybersickness in virtual environments. *ACM SIGCHI Bulletin 32*, 1 (2000), 47–56. 1

[MZP*94] MACEDONIA M. R., ZYDA M. J., PRATT D. R., BARHAM P. T., ZESWITZ S.: Npsnet: a network software architecture for largescale virtual environments. *Presence: Teleoperators & Virtual Environments 3*, 4 (1994), 265–287. 2

[PBE99] PELEG S., BEN-EZRA M.: Stereo panorama with a single camera. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* (1999), vol. 1, IEEE. 2

[PBEP01] PELEG S., BEN-EZRA M., PRITCH Y.: Omnistereo: Panoramic stereo imaging. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 23*, 3 (2001), 279–290. 2

[PBG92] PHILLIPS C. B., BADLER N. I., GRANIERI J.: Automatic viewing control for 3d direct manipulation. In *Proceedings of the 1992 symposium on Interactive 3D graphics* (1992), ACM, pp. 71–74. 2

[PCD*12] PIRK S., COHEN M. F., DEUSSEN O., UYTTENDAELE M., KOPF J.: Video enhanced gigapixel panoramas. In *SIGGRAPH Asia 2012 Technical Briefs* (2012), ACM, p. 7. 2

[PK99] PARK K. S., KENYON R. V.: Effects of network characteristics on human performance in a collaborative virtual environment. In *Virtual Reality, 1999. Proceedings., IEEE* (1999), IEEE, pp. 104–111. 2

[PZF10] PISA C., ZEPPA F., FANGI G.: Spherical photogrammetry for cultural heritage: san galgano abbey, siena, italy and roman theatre, sabratha, libya. In *Proceedings of the second workshop on eHeritage and digital art preservation* (2010), ACM, pp. 3–6. 2

[SBD*10] SARGENT R., BARTLEY C., DILLE P., KELLER J., NOURBAKHSH I., LEGRAND R.: Timelapse gigapan: Capturing, sharing, and exploring timelapse gigapixel imagery. 2

[SBGH09] SCHEUCHER B., BAILEY P. H., GÜTL C., HARWARD J. V.: Collaborative virtual 3d environment for internet-accessible physics experiments. *iJOE 5*, S1 (2009), 65–71. 2

[SPWD13] SCHULZE J. P., PRUDHOMME A., WEBER P., DEFANTI T. A.: Calvr: an advanced open source virtual reality software framework. In *IS&T/SPIE Electronic Imaging* (2013), International Society for Optics and Photonics, pp. 864902–864902. 2

[Sze10] SZELISKI R.: *Computer Vision: Algorithms and Applications*, 1st ed. Springer-Verlag New York, Inc., New York, NY, USA, 2010. 2

[YMDK11] YAMAOKA S., MANOVICH L., DOUGLASS J., KUESTER F.: Cultural analytics in large-scale visualization environments. *Computer 44*, 12 (2011), 39–48. 2

[ZGL*07] ZHU J., GONG J., LIU W., SONG T., ZHANG J.: A collaborative virtual geographic environment based on p2p and grid technologies. *Information Sciences 177*, 21 (2007), 4621–4633. 2