# Gesture interactions for Virtual Immersive Environments: navigation, selection and manipulation

Paulo Dias[1,2], João Pinto[1], Sérgio Eliseu[3], Beatriz Sousa Santos[1,2]

[1]DETI/UA- Department of Electronics, Telecommunications and Informatics
[2]IEETA- Institute of Electronics and Informatics Engineering of Aveiro
University of Aveiro
Campus Universitário de Santiago, 3810-193 Aveiro, Portugal
[3]3iD+ / i2ADS - Faculty of Fine Arts, University of Porto
paulo.dias@ua.pt; jhpinto@ua.pt; s.eliseu@ua.pt; bss@ua.pt

**Abstract.** This paper presents an extension to a Platform for Setting-Up Virtual environments with the purpose of allowing gesture interaction. The proposed solution maintains the flexibility of the original framework as well as content association (PDF, Video, Text), but allows new interactions based on gestures. An important feature is the one to one navigational input based on Kinect skeleton tracking. The framework was used to configure a virtual museum art installation using a real museum room where the user can move freely and interact with virtual contents by adding and manipulating 3D models. Two user studies were performed to compare gestures against button-controlled interactions for navigation and 3D manipulation. Most users preferred the Kinect-based navigation and gesture-based interaction despite some learning difficulties and tracking problems. Regarding manipulation, the gesture-based method was significantly faster with similar accuracy when compared to the controller. On the other hand, when dealing with rotations, the controller-based method was faster.

**Keywords:** Virtual Reality, Navigation and Manipulation in Virtual Environments, Gestural interaction, Kinect, 3DUIs, User study.

## 1 Introduction and Motivation

This work aims to update a previously developed framework called pSIVE [1] to support recent hardware and allow gesture interaction in line with the Empty Museum concept [2]. One of the objectives was to create a setup where users can move freely inside an empty room, using a one to one position mapping from the real world to the virtual world, while viewing a virtual scene with the same spatial configuration. In this setup, besides navigation, the visitor had the possibility to interact with art-pieces, browsing contents (such as pdf files, images, textual information and videos), and even modify the museum by adding and manipulating 3D models, therefore creating their own virtual museum. The final setup allows complete immersion of the user in an empty room using several Kinects for tracking and a head mounted display. A virtual museum prototype was developed where the user was able to navigate and

adfa, p. 1, 2011.

modify the environment either hands-free (walking in a real empty room and using gestures), or standing still with a physical controller, while viewing the virtual museum through an Head Mounted Display (HMD). In what follows, we present some related work, and then discuss the framework's architecture, the interaction methods it supports and the virtual museum demonstration used to validate the framework. Finally, the results of two user studies comparing interactions with gestures and controllers are presented.

## 2 Related work

The Empty Museum [2] is the application that most closely resembles ours. It is a multi-user immersive walkable and wireless system where users can navigate a virtual environment with content such as audio, text, images and video. The system includes a laptop computer in a backpack that renders the environment on a HMD according to the user perspective while connected to sensors that captures wireless movement. The only interaction provided is the user position update with no additional support for more complex interactions.

The training School developed for the DCNS group [3] is a virtual training environment, developed in OpenSpace3D, linked to a Learning Management System (LMS). This web solution provides access to training courses and pedagogical progress of students, as well as managing the content of the courses. Users can navigate in the environment, visualise training courses, media and products, check messages and chat with other users (students and teachers). The application allows interaction through a desktop environment, a web browser with mouse and keyboard or in an immersive environment with stereoscopic video projection supporting a Wiimote and kinect camera for content interaction.

The KidsRoom [4] explores computer vision technologies in an augmented reality interactive playspace for children by recreating a child's bedroom with two real walls and two large video projection screens, as well as ceiling mounted coloured lights, speakers, cameras and a microphone, all controlled by a six-computer cluster. Four cameras are pointing at the room, one for tracking, two for action recognition of people in the room, and one to provide a view of the room for spectators.

pSIVE [1] is a platform we previously developed that can be used to set up a virtual scene using a diversity of models associated to variety of content (pdf, video, text). It uses OpenSceneGraph [5] as graphical engine and VRJuggler [6] as a middleware to interpret input from trackers and controllers. Content can be accessed through a 2D linear menu that pops up when the user presses a button on the controller while looking at an object that has been configured with content.

None of the presented systems supports at the same time content presentation, one to one navigation and gesture interaction in the same framework as we propose in this work.

## 3 Framework

The developed framework took advantage of our previous work on pSIVE. The main features are still easy configuration, support for several contents (videos, pdf files, text and images), and hardware flexibility. However, in this work we have expanded and updated the framework to support recent hardware (Oculus Rift and Kinect) and abandoned the VRJuggler [6] library due to its difficult set-up and lack of recent development. The next sub-sections present details on the new version of the framework as well as a prototype developed to demonstrate its flexibility and illustrate the type of applications it can be used for.

### 3.1 Architecture

The architecture of the framework is presented in Fig. 1. The selected graphical engine is OpenSceneGraph (as used in pSIVE) to benefit of its VR libraries (namely osgOculusViewer for Oculus rift and osgVRPRN, a node kit to integrate the VRPN-Virtual Reality Peripheral Network providing access to a variety of VR devices). A PC based client-server architecture is used in which the client is responsible for all the rendering of the virtual world and handling of the (HMD orientation tracking. Several VRPN servers [7] communicate the hardware input, when using a physical controller, or the user skeleton information (positional data of head, hands and gripping gestures) to the client using one or several Microsoft Kinect devices. Skeleton data is collected with the Kinect SDK 1.8.
The framework is configured through several XML files, namely Config.xml (list of models, physical attributes (size, rotation, location) and available content), Kinect.xml (information to set-up the servers which read data from multiple Kinects) and Controls.xml (for the use and mapping of a physical controller). It receives input information from one or more VRPN Servers, which is interpreted by the osgVRPN library. That information is then handled in one of two ways: interaction with menus or content (Menu Handler), or navigation (Camera Handler). Finally, the scene is rendered for use with the Oculus Rift using the osgOculusViewer library.

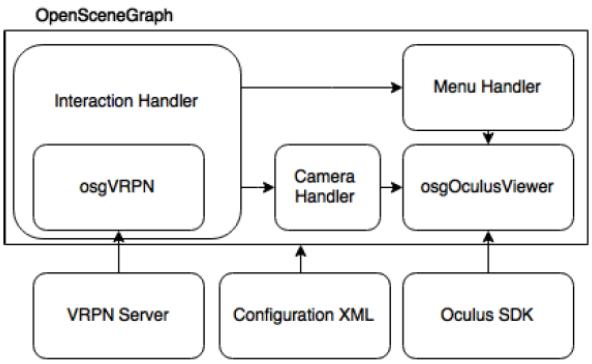


**Fig. 1.** Framework Architecture

### 3.2 Calibration

In order to calibrate the Kinect position regarding real world and allowing one to one navigation, a calibration tool was developed. The visualization Toolkit (VTK) [8] is used due to familiarity and easy access to the Iterative Closest Point (ICP) algorithm. The calibration process captures a depth image from the Kinect and allows the user roughly fit the point cloud within the 3D model using the keyboard. After the frame is positioned manually the user can adjust automatically the depth image with the model using VTK's ICP function (see Fig. 2). Finally, the transformation matrix is exported into a file ready to be used as input in our custom VRPN server for a given room with given Kinect positions.

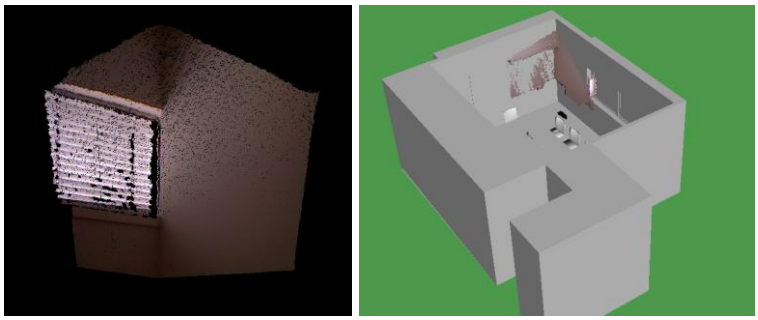

**Fig. 2.** A Kinect point cloud and final alignment with 3D model after ICP

Regarding head orientation, a calibration of Oculus Rift's orientation is necessary to match the direction the user is looking at in the virtual world with the corresponding direction in the real world. This procedure requires the user to stand facing the real world Kinect and grip their right hand in order to overlay the view direction of the 3D world with the direction they are facing in the real world. After these calibrations, the user should be facing a white cube that represents the position of the Kinect and can then freely walk in the room, with one to one positioning into the virtual room.

### 3.3 Custom VRPN server

A custom VRPN Server using the VRPN library with the Microsoft Kinect library was developed to integrate Kinect information into our system. The Kinect 1.8 SDK was used to track the user skeleton and the grip gesture of both their hands. The coordinates are then transformed to real world coordinates using a transformation matrix computed during Kinect calibration. That information is communicated to the client by mapping Kinect information (user's left and right hands as well as global positions) through analogue channels. Left and right hand closing state are mapped as buttons. Head position is used by the client to define the camera position in the scene, while right and left hand positions are used to show visual aid (spheres) representing the user's hands. In a multiple Kinect setting additional information is sent to indicate which Kinect is detecting the user.

### 3.4 Virtual Museum

To validate our framework, we configured a virtual museum art installation: a virtual model of a real room of the museum of Aveiro was created and users were able to navigate and interact inside a virtual representation by just moving their body, while tracked by three Kinects covering most of the working area (Fig. 3). Besides visualization and navigation, users could also activate menus as well as select and manipulate 3D models of contents related to the museum (Fig. 4).

**Fig. 3.** Kinect setup in the virtual museum

**Fig. 4.** User manipulating a model in the virtual museum

## 4 Interaction

An important goal of the developed platform is flexibility in terms of hardware and interactions. In this section, we describe the interaction methods supported by the platform for navigation and manipulation.

### 4.1 Navigation

Three different methods of navigation are supported: the first one, and the main focus of our work, uses the Kinect skeleton information to position the user in the virtual scene, allowing to navigate the world by simply walking. As an alternative, when one to one tracking is not possible, a method based on a "Steering Wheel" metaphor [9] was implemented allowing the user to navigate the world as if driving a car and based on hands tracking. A physical button controller (Wiimote for example) can also be used to control user position and navigation. The forward and backward directions are determined by the user's viewing direction (gaze-directed), meaning s/he will always move towards or away from the direction s/he is looking at.

### 4.2 Manipulation

Regarding object manipulation, two different interaction modes are provided. One method uses a handle bar metaphor inspired in [9] to position, rotate and scale the objects (see Fig. 5). The original implementation was modified to allow positioning of the objects (the original proposal only coped with rotation and scaling).

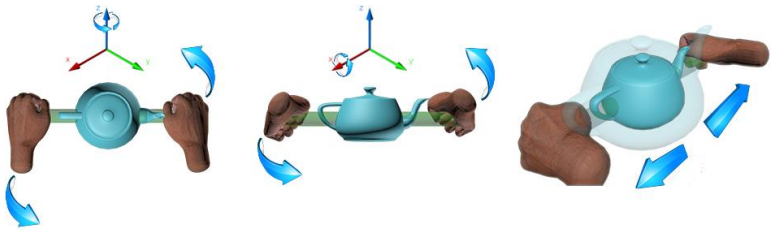

**Fig. 5.** Handle bar metaphor for object rotation and scaling adapted from [9]

Using this metaphor, scale is updated according to the distance between the user's hands. Rotation is given by the angle between the vector calculated from the position of the user's hands and the horizontal/vertical planes. The object position is obtained by the midpoint between the user's hands in the world. Scale and rotation changes are accumulated during the manipulation, meaning that the user may combine rotations to make up for the fact that there is a missing DOF, and scale the object up or down without being limited by the length of their arms.

Manipulation can also be done with a controller. In our tests, a Wiimote was used, but any controller with buttons can be configured. In this case, manipulation is split into three modes that can be toggled using the controller: translation (x, y, and z axis), rotation (x, y, and z axis), and uniform scaling. The current mode is depicted by an icon on the display and manipulation takes effect with button presses.

## 5 User studies

A preliminary study with 12 volunteers and a controlled experiment involving 28 participants have been performed to compare gestures and commands for navigation and manipulation. During both studies, execution times and accuracy of the movements were logged.

### 5.1 Preliminary study

A preliminary study was performed with 12 volunteers participating in the University's summer academy (ages between 15 and 17) to compare the two methods of interaction and navigation in a virtual room. The input variables are the navigation and the menu selection methods. In one method, the user walks in the room to navigate, and is able to select an item by moving their hand and performing the grip gesture. In the other method, the user uses a controller with buttons to navigate in the room and interact with the menu. Both methods display radial menus and use an Oculus rift DK2 HMD.

The output variables were the time the participant took to get to the interaction position (interaction area shown as a sphere in Fig. 6) and activate the menu, the time necessary to select the correct option, and the number of incorrect selections.

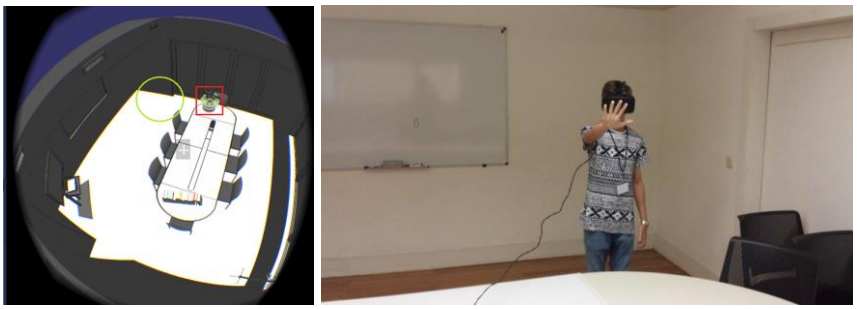

**Fig. 6.** Experimental environment (green circle: interaction area) and user performing a gesture

The time differences to reach the interaction position are negligible between the two setups (35.25 seconds average for controller-based and 35.41 seconds for Kinect-based). Regarding interaction with the menu, users were faster to activate the correct option with the controller-based method (with an average 16.6 seconds) when compared with the Kinect-based method (30.3 seconds) that also presented more selection errors. Participants filled a questionnaire regarding navigation and interaction preferences and preferred Kinect interaction despite its worse performance for menu selection.

### 5.2 3D manipulation study

This study was used to compare the manipulation interaction styles. As such, we have gathered data regarding the manipulation times and accuracy, performing a comparison between gestures and a controller. Two different studies were performed. One regarding object position and the other regarding rotation/scale.

**Method.**
This experiment aimed to verify if the two 3D manipulation methods were equally usable in our demonstration environment. Our experiment had two input variables, namely the two different manipulation methods: gestures or controller-based. The experiment was divided in two separate stages: the first consisted of a manipulation regarding the positioning of an object. In the gesture-based method, users performed a grip gesture with the two hands to start interaction and then move the hands to position the object. In the controller-based method, they used buttons to move the object. In both methods, the user's task was to position the object as such that it matched a ghost model of that object as closely as possible. The experiment continued to the next phase once the user verbally signalled that they were satisfied with their placement of the object. In the second stage, we evaluated the manipulation of the object's rotation and scale. Given some initial difficulties of the first users in this task, we decided to set two and a half minutes of training before starting the test and gathering data. Similarly to the manipulation, the users had an object centred inside the ghost model, with a different orientation and scale (Fig. 7), and they must use gestures or the controller's buttons to rotate and scale the object to match the ghost model as closely as possible. The experiment ended when the user verbally indicated that they were satisfied with the matching between the two models.

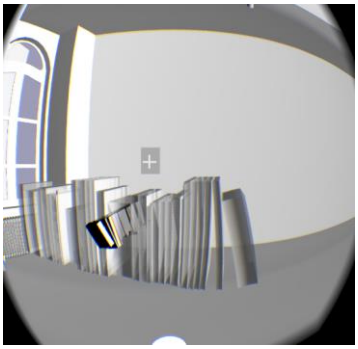

**Fig. 7.** Ghost model and rotated model used in the controlled experiment

In the positioning study, we logged three variables during user interaction: time elapsed until the objects was at a position below 0.002 units from the final position, time to the final position and final distance between the ghost and the final model in final position.
In the rotation/scale study, four key values were monitored: first time that the object reached our minimum requirement (a solid angle below 5º between the two models),

time to the final position, and the angular and scale difference between the two models at the end of the interaction.
28 users participated in this experiment. All users performed the tasks with both experimental conditions (a within group experimental design was used). Half of the users performed the experiment using the Kinect first and the others using the controller first, in order to attenuate possible bias due to learning effects. The participants were observed while performing the experiment, and were asked to answer a questionnaire regarding their satisfaction, difficulties and preferred method.

**Results.**
**Table 1** presents the average results of the positioning tests. It shows that users take roughly half the time to achieve a distance below the threshold with the Kinect when compared with the controller. On the other hand, after achieving the minimum distance, the adjustments required until the best position were much faster with the controller. The accuracy in the final position was similar between both methods. Preferences were similar with 13 users preferring the Kinect, 14 preferring the controller and 1 having no preference.

| | Kinect | Controller |
|---|---|---|
| Time to first position below threshold distance (s) | 23.4 ± 13.0 | 48.1 ± 18.3 |
| Best position time (s) | 38.7 ± 29.0 | 54.8 ± 20.3 |
| Best position (units) | 0.00015 | 0.00013 |

**Table 1.** Positioning test results

**Table 2** presents the average results of the rotation/scale tests. The scaling difference between models is not presented since its variation was negligible between both methods. Users achieved similar accuracy with both methods, with the Kinect-based method being 30% slower than the controller-based method. However, despite being slower to reach a < 5º error, the Kinect-based method was faster after that point in reaching the minimum angle error (17s vs 22s). This difference might be explained by the necessity to accumulate rotations in the Kinect-based method (only 2 degrees of freedom in rotation were available), where the controller-based method had 3 degrees of freedom in rotation. As previously, users' preferences were split between the two methods, with 12 users preferring the Kinect-based method, 13 users preferring the controller-based method and 3 having no preference. It is noteworthy that there were 11 failed attempts either with the controller-based (7) or the Kinect-based method (4), corresponding to users not able to reach an error below 5º within a reasonable time.

| | Kinect | Controller |
|---|---|---|
| Time to first angle below threshold (s) | 75.5 ± 53.6 | 53.5 ± 37.0 |
| Time to best angle (s) | 88.3 ± 57.3 | 75.8 ± 41.4 |
| Angular difference (º) | 1.19 | 1.03 |

**Table 2.** Positioning test results

## 6 Conclusions

The main goal of this work was to provide a framework that allows the creation of an interactive virtual world. The users can either interact with the environment by accessing content such as videos and pdf files within models in the scene, or by inserting and manipulating directly objects in the environment. The developed framework was tested in a live scenario, in the museum of the city of Aveiro. An art installation was created where users could navigate/move around a virtual/real room in the museum and experience virtual content, as well as setting up their own virtual exposition by adding models of monuments through a grid-like menu and manipulating them by positioning, rotating and scaling them in the virtual environment.

We adapted a previously developed framework to work with one or several Kinect cameras, tracking the user in a physical room and mapping their movements to the virtual scene. To do this we developed tools to calibrate the Kinect cameras with the real room and its virtual model. The use of a depth sensor also provided the opportunity to add gesture-based interaction for manipulating content.

Two user studies were performed, a preliminary one to compare input methods (controller *versus* gestures in a radial menu), and movement methods (tracked by the Kinect *versus* buttons on a controller). From this study, we concluded that users were interested in gesture-based controls and one-to-one mapped navigation, and while the navigation timings with the Kinect tracking method were comparable to the timings when using the controller, menu selection was not as good. We also performed a controlled experiment to compare gesture and controller-based manipulation. Both methods fare about the same when it comes to accuracy in the position, rotation and scale of the object, with the Kinect-based method being faster in positioning and the controller-based method faster when it comes to rotation.

From the results of these user studies, as well as the experience from developing and testing the framework, we consider that both controller-based and gesture-based methods have their place in interaction inside immersive virtual environments. Controller-based methods have the advantage of providing clearer and more accurate actuation when compared with gesture-based methods, where we found some cases of false-positives and false-negatives. On the other hand, gesture-based interaction methods provide us with a more natural type of interaction with the virtual environment, particularly in the case of object manipulation. We believe that a hybrid method of interaction could prove to be beneficial, where a physical controller might be used to activate menus and select options, while combined with depth and movement sensors to provide 6 degrees of freedom on manipulation tasks.

**Acknowledgements.** The authors are grateful to all volunteer participants. This work was partially funded by National Funds through FCT - Foundation for Science and Technology, in the context of the projects UID/CEC/00127/2013 and Incentivo/EEI/UI0127/2014.

## References

1. Souza, .D, Dias., P., Sousa Santos, B.: Choosing a Selection Technique for a Virtual Environment. Human-Computer Interaction: Virtual, Augmented and Mixed Reality. Proceedings of 16th HCI International, Heraklion, Crete, Greece. Vol. 8525 of Lecture Notes in Computer Science, Springer, pp. 215-225 (2014).
2. Hernandez, L., Taibo, J., Seoane, A., López, R., López, R..: The empty museum. Multi-user interaction in an immersive and physically walkable VR space. Proceedings of International Conference on Cyberworlds. IEEE Comput. Soc, pp. 446–452 (2003).
3. DCNS Training School - A virtual environment interfaced with an e-learning platform. http://www.openspace3d.com/lang/en/2013/03/19/dcns-training-school (2013).
4. Bobick, A.B., Intille, S.S., Davis, J.W., Baird, F., Pinhanez, C.S., Lee, Campbell, L.W., Ivanov, Y.A., Schütte, A., Wilson, A.: The KidsRoom: A Perceptually-Based Interactive and Immersive Story Environment. Teleoperators and Virtual Environments, Vol. 8, No. 4, pp. 367-391 (1999).
5. Wang R., Qian, X.: OpenSceneGraph 3.0: Beginner's Guide. Packt Publishing, (2010).
6. Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A., Carolina Cruz-Neira., C.: Vr juggler: A virtual platform for virtual reality application development. Proceedings of the Virtual Reality 2001 Conference (VR'01), VR '01, p.89, Washington, DC, USA. IEEE Computer Society (2001).
7. Taylor, R.M.II, Hudson, T.C., Seeger, A., Weber, H., Juliano, J., Helser, A.T.: Vrpn: A device-independent, network-transparent vr peripheral system. In Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST '01, pages 55–61, New York, NY, USA. ACM (2001).
8. Schroeder, W., Kenneth, M.M, Lorensen, W.E.: The Visualization Toolkit (2nd Ed.): An Object-oriented Approach to 3D Graphics. Upper Saddle River, NJ, USA: Prentice-Hall (1998).
9. Cardoso, J. "3D manipulation and navigation methods with gestures for large displays". Master thesis. Universidade de Aveiro, Portugal (2015).