# KAVE: Building Kinect Based CAVE Automatic Virtual Environments, Methods for Surround-Screen Projection Management, Motion Parallax and Full-Body Interaction Support

AFONSO GONÇALVES and SERGI BERMÚDEZ i BADIA, Madeira Interactive Technologies Institute and Faculdade de Ciências Exatas e da Engenharia, Universidade da Madeira

While CAVE Automatic Virtual Environments (CAVE) have been around for over 2 decades they remain complex to setup, unaffordable to most, and generally limited to data and model visualization applications for academia and industry. In this paper, we present a solution to create a monocular CAVE using the Unity 3D game engine by adding motion parallax and full-body interaction support via the use of a Kinect V2 low-cost sensor. More importantly, we provide a functional and easy to use plugin for that effect, the KAVE, and its configuration tool. Here, we describe our own low-cost CAVE setup, a range of alternative configurations to CAVE systems using this technology and example applications. Finally, we discuss the potential of such an approach considering the current advancements in VR and gaming.

CCS Concepts: • **Human-centered computing → Virtual Reality**; Displays and imagers; • Hardware → Sensor applications and deployments

## KEYWORDS

CAVE; Unity, Kinect; Virtual Reality; Low-cost; Multi-Display

**ACM Reference format:**

Afonso Gonçalves and Sergi Bermúdez i Badia. 2018. KAVE: Building Kinect Based CAVE Automatic Virtual Environments, Methods for Surround-Screen Projection Management, Motion Parallax and Full-Body Interaction Support. *PACM on Human-Computer Interaction,* Vol. 1, EICS, Article 10 (June 2018), 15 pages.

https://doi.org/10.1145/3229092

## 1 INTRODUCTION

In [4], the CAVE (CAVE Automatic Virtual Environment) was described for the first time, a stereo multi-screen virtual reality system where the images are projected on the inside walls and floor of a cubic structure. The system used head-tracking and off-axis projections to simulate visual presence in the virtual world. In their work, the authors refer the 8 visual cues that enable us to perceive depth: 1) Occlusion, 2) Perspective projection, 3) Atmospheric effects, 4) Lighting and Shadows, 5) Binocular disparity, 6) Convergence, 7) Motion parallax, and 8) Eye Accommodation (eye lens control reflex for focusing light from

This work was supported by the Fundação para a Ciência e Tecnologia through the AHA project (CMUPERI/HCI/0046/2013), by the INTERREG program through the MACBIOIDI project (MAC/1.1.b/098), and LARSyS – UID/EEA/50009/2013.

Author's addresses: A. Gonçalves and S. Bermúdez, Madeira Interactive Technologies Institute, Polo Cientifico e Tecnologico da Madeira, floor -2, Caminho da Penteada, 9020-105 Funchal – Portugal.

different distances). Of these, cues 1, 2, 3 and 4 can be provided by conventional displays, 5 and 6 can be achieved with stereo graphics (such as with the use of stereo glasses or Head Mounted Displays (HMD)), and 7 with head tracking. Their system, as most Virtual Reality (VR) systems, could produce all of them except 8.

The possible applications of the CAVE in data visualization were first explored by inviting experts and professionals from different fields to experience the system, this included the visualization of architecture, cosmic, fractal, weather, and molecular dynamics data and models; the visualization of jobs being executed on the multiple parallel processors of a supercomputer, for algorithm execution optimization; and visualization of real-time brain activity overlaid on a digital head model [3]. Some other CAVE systems were used for visualization of geophysical simulation data [12], underground cave systems [16], computational fluid dynamics and neuroscience [10]. However, the use of CAVEs has been mostly constrained to data visualization roles, in laboratories or companies and their potential to be used in other VR applications such as gaming has been rarely explored. In this paper, we present the specific methods we employed to add monocular motion parallax effect, through head position dependent perspective projection, to Unity applications using the Kinect V2 as the head tracking sensor. The addition of this effects to computer graphics increases the number of depth cues it can provide and it is a fundamental building block in CAVE systems. These methods were bundled in an open-source plugin, the KAVE, which represents a solution, for VR developers and researchers, in the development and conversion of virtual environments (VE) and games to immersive multiple large-screen projection systems. Using the KAVE, the presented methods were employed in several proof of concept configurations to showcase their flexibility, additionally a few example applications are presented to demonstrate what a user can expect from an VE powered by this tool.

## 2 RELATED WORK

The methods for generating the motion parallax effect and head position dependent perspective projection have been used consistently in computer graphics to design different three dimensional displays such as HMDs, head coupled displays, and CAVEs [4, 18, 20], our own technical implementation of these methods can be consulted in Section 3.1. Although these methods haven't change through the years, many recreations of the original CAVE have been developed, focusing mainly on technical differences, for both scientific and commercial purposes. Yet, the costs for such a system remain very high. The CAVE [4] and CAVE2 [5] systems are estimated to have cost 2,000,000$ and 926,000$ respectively [5]. While low-cost solutions exist, the implementation of a low-cost CAVE including the physical construction and software can still be prohibitive (19,300€) [9]. In [9], they created a CryEngine2 game engine mod for games using that same engine. However, because it lacked user tracking it assumed a static head position of the user and supported neither motion parallax nor stereo vision. Another low-cost CAVE was presented in [17], based on rolling screens suspended from the ceiling with a cost of 9,500$, excluding the computer and the necessary software for their custom setup. Their configuration is easy to use and does not permanently occupy floor space. For tracking sitting users it employed marked glasses and infrared cameras. The immersive virtual content was provided by adapting individual Unity applications to their visualization paradigm. Unfortunately, the code was not made available. Another CAVE made using off-the-shelf hardware and the author's own software revealed to be an alternative low-cost solution for fire brigade training in simulation [15]. This solution was developed for Unity using a Kinect as sensor, however, it lacks the desired replicability as it is not clear whether the software can be used in different setups other than the one for which it was developed.

While building a CAVE, a large part of the budget is generally allocated to the tracking technology, ranging from 8,000$ for a 6 low-end IR camera motion capture setup up, to 40,000$ for high-end cameras, from suppliers such as NaturalPoint Inc. OptiTrack or Vicon Motion Systems Ltd UK. Additionally, middleware software to render across multiple displays, integrate sensors and applications is complex and must either be bought for several thousands of dollars from specialized companies, such as MiddleVR [11] or needs to be custom developed in-house. Like the Uni-CAVE [19], a plugin for managing surround-screen projection in Unity applications, with the drawbacks of lacking full-body tracking and requiring the CAVE setups to be defined before the VE (Virtual Environment) compile time.

Beyond CAVE setups, Microsoft Research has been using consistently Kinect cameras to implement spatial augmented reality (SAR) application through projection mapping in arbitrary scenarios. In LightSpace [21] the authors report their interactive installation that combines multiple depth cameras and projectors registered to a single 3D space. The use of this unified space across projector/camera units (procam) units enables them to project graphics into the depth mapped surfaces and the users to interact with the projected elements with their bodies, through the bodies' 3D mesh obtained from the Kinect sensor. With MirageTable [1] a similar but scaled down system, single Kinect and projector, added support for non-flat surface projection and head tracking for head position dependent perspective projection. In this system, the use of a single Kinect for both tracking the user, online, and mapping the surface, offline, requires that during calibration the Kinect has to be first installed in a position and orientation that can capture the whole projection surface and then installed in a way that it can capture both the user and part of the projection. Again with a single Kinect and projector, IllumiRoom [7] shows how the same technology can be used to augment the arbitrary space surrounding a television using the depth data from a Kinect. The combination of these projects led to RoomAlive [8]. The RoomAlive toolkit joins together tools for the automatic calibration of procam and tools for authoring augmented content in Unity. This tool supports full-body tracking for interaction and head position dependent perspective projection, and projection mapping to arbitrary spaces using Kinect sensors, capabilities that enable developers to create spatial augmented reality (SAR) experiences in any space, theoretically in CAVEs also. The system chains together the information from multiple procam units to automatically calibrate and register the whole setup in the same 3D space reference frame, requiring one Kinect per projector, that their fields of view overlap with each other and include the indented user tracking area.

The works stated above provide tools to create VR experiences tied to a specific setup since development and/or focus on depth mapping of arbitrary spaces, which require elaborate setups for calibration and to run. In this work, we hope to contribute in a significant way to improve two different "situation, task and users" (STU) contexts [13] in the development and use of VEs for CAVEs. First, our contribution can separate VE developers from users of a VR CAVE application, creating the STU of programmers (tool users) developing VEs (task) for CAVE users (end users) that will use the VEs for their own purposes (end task). A separation that is hard to make with the related work tools, where setup configuration must be known at compile time or obtained live from depth mapping. Starting with the second context, concerning CAVE users, when compared with alternative solutions the applications built with our tool have the capability of reaching a much larger population due to the simplicity of the required setup, taking exclusivity of CAVE-like system away from large commercial or research institutions, the fact that users don't have to have access or create the source of the application further increases this population. As for the first context, of VE developers, our contribution gives them the flexibility of developing and compiling an application that can be deployed to a diverse range of user-dependent setups.

In the following sections, we describe the KAVE, our own flexible CAVE software for the Unity 3D game engine (Unity Technologies, San Francisco, USA). Our free open-source software, in the form of a Unity plugin, supports up to 8 displays (both projectors and screens) in any physical configuration; its versatile regarding hardware as it supports different screens, projectors models, resolutions or aspect ratios at the same time with arbitrary positions and orientations; and combined with the use of a low-cost tracking sensor makes entry-level CAVE-like systems much more affordable and easy to use by the community of game, simulation, VR developers and researchers. KAVE relies on a 140$ Kinect V2 (Microsoft, Redmond, USA) tracking sensor, which has a remarkable accuracy of landmark movements [14] and provides full-body tracking of up to 6 simultaneous users, which enables unobtrusive full-body interaction in VR. Additionally, to complement our Unity plugin we make available a versatile tool for the easy setup and calibration of CAVEs and describe our own CAVE physical setup.

## 3 METHODS

### 3.1 KAVE Plugin, Adding Motion Parallax and Whole-Body Support to Unity Using Kinect

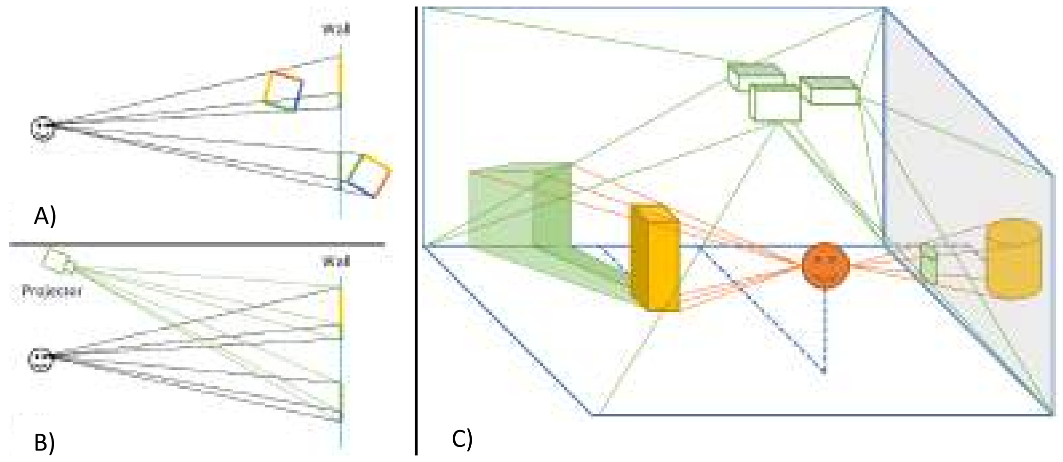

**Fig. 1. A) Finding out how virtual objects (colorful squares) should be seen by the user; B) Projecting on the wall what the user should see. C) Projections (solid green) of virtual elements (yellow) in real surfaces (blue) adjusted to the perspective of the user (orange) point-of-view by the CAVE projectors.**

An important feature of a CAVE is the motion parallax effect produced. This is achieved by keeping continuous tracking of the user's point of view and the automatic adjustment of the projection of images consistent with that perspective. Figure 1 A) & B) illustrates the phenomenon in a 2D monocular diagram. By knowing the positions of the user viewpoint, projection plane, and virtual elements to represent (Figure 1 A), we can project images on that plane coherent with what the user would see if the virtual elements were in the real world (Figure 1 B). By generalizing this effect to 3D and to the use of multiple projection surfaces, we obtain the basic functioning of a CAVE, as showed in Figure 1 C) where the blue lines represent the walls and floor of a CAVE (projection surfaces), the orange icon the user head position and the yellow elements represent the virtual objects in their virtual positions relative to the user. As a result, the CAVE must display the green images (planar images on the projection surfaces) to emulate the visual effect that the virtual objects would have if they were real.

The creation of the parallax effect is thus dependent on the precise knowledge about the size, position, and orientation of the projection surfaces, body tracking coordinate system, and correct mapping of the CAVE projectors images to the surfaces. A calibration process must provide this information as it is necessary to mathematically model the CAVE in Unity. Hence, the calibration process enables the correct mapping of the projections into the physical CAVE walls and to get an accurate location of the user's point of view relative to them.

Our calibration process creates a CAVE virtual replica description file with its origin located on the CAVE floor center. Consistent with this calibration file the following elements are created at runtime by a Unity application using our plugin (Figure 2):

- CAVE Kinect V2: A Unity game object with the position and orientation matching the real sensor, it reads up to 6 users' 25 joints skeleton positions and orientations from the Kinect SDK in its own coordinate system. It instantiates a transparent (by default) avatar for each user, these avatars can have their colliders activated, thus allowing users to interact with virtual elements through Unity's physics simulator;

- CAVE Screens or Surfaces: Invisible Unity quads/planes representing flat-panel displays (such as TVs or computer screens) or the CAVE projection surfaces' position, orientation, and size, they serve as reference rectangular targets for the User View Cameras;
- User View Cameras: Unity cameras attached to the user's head, the head position is provided by the CAVE Kinect V2 object. In the case of multiple users, the closest person to the sensor is chosen. One camera per CAVE Surface/Screen is created, each of them with its principal axis perpendicular to that surface/screen. The camera projection matrix values are calculated in real-time to ensure the coincidence of the camera frustum with the target surface/screen edges, meaning that each camera image is exactly framed by the surface/screen rectangle. Depending on the target of the camera (screen or projector) the views from each camera are either sent directly to the screen display or turned to render textures and sent to the corresponding CAVE Projector for further warping;
- CAVE Projectors: Unity cameras that receive the images from the User View Cameras and warp them to achieve a correct mapping into the desired real-world projection surfaces. This warp is done by mapping each of the 4 corners of the image to new viewport locations that will correspond to the real projection wall corner. This makes possible configurations where the projectors are badly aligned with the walls, such when they are in an angle or when they project on more than one wall (usual when the aspect ratios of the projector and wall are different). The projection outside of the new image boundaries is black thus eliminating any overlapping projections between adjacent surfaces. A projector can receive multiple User View Cameras at the same time and apply a different warping mask to each, this enables a projector to display on several surfaces at once. This warping process removes the burden of physical projector calibration and turns into an easier software calibration.

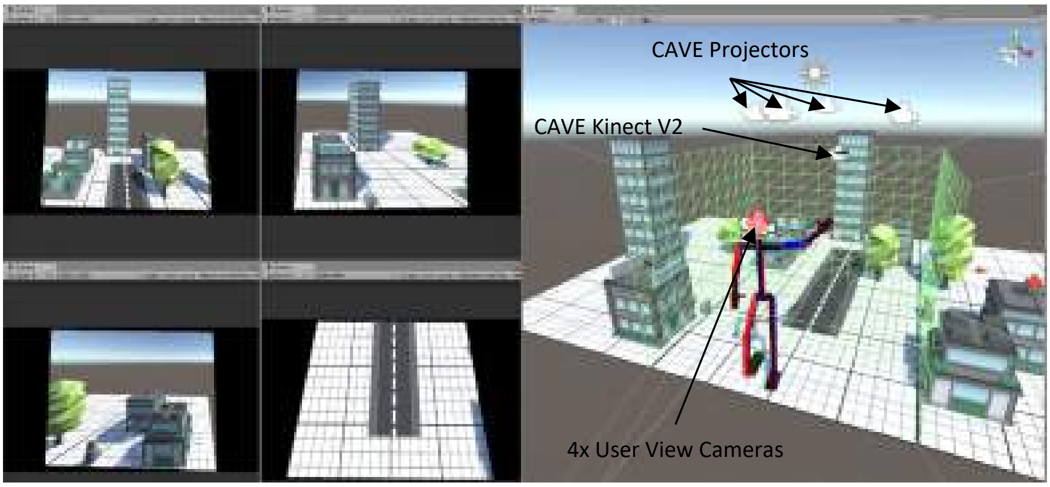

**Fig. 2. The virtual KAVE generated at runtime in the Unity editor. CAVE Surfaces are highlighted in green, the virtual equivalents of the 4 ceiling mounted real projectors are seen as cameras on top, the 4 user view cameras (and their frustums) centered on the user's head, and the Kinect sensor is seen as a black bar above the central wall. On the left are shown the 4 views that the CAVE Projectors display, these are already warped to match the real CAVE walls.**

While there are multiple ways of setting the User View Camera projection matrix to fit our goals, we took advantage of Unity existing camera parameters and functions. At each new frame, the position of the User View Cameras is updated per the CAVE Kinect V2 sensor data, while the orientation remains perpendicular to the corresponding CAVE Surface/Screen. Then, after resetting the camera projection matrix, the vertical field of view (FOV, α) is calculated per equation (1), followed by the manipulation of the camera principal point offset by setting the unity camera matrix elements $m_{1,3}$ and $m_{2,3}$ as described in (2) and (3).

$$\tan\frac{\alpha}{2} = \frac{h/2}{dy} \equiv \alpha = 2\tan^{-1}\frac{h/2}{dy} \tag{1}$$

$$m_{1,3} = \frac{dz}{h/2} \tag{2}$$

$$m_{2,3} = \frac{dx}{w/2} \tag{3}$$

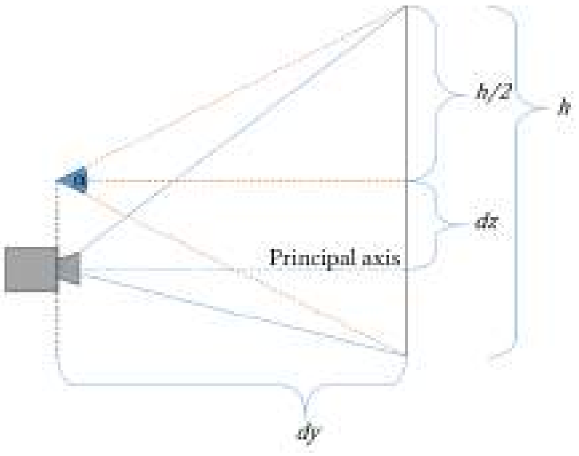

**Fig. 3. Setting a User View Camera frustum (solid blue) coincident with a CAVE surface (solid black), side view.**

Here *h* and *w* are the height and width of the surface/screen, *dy* is the distance from the camera to it, and *dz* and *dx* are the camera's principal axis vertical and horizontal offsets relative to the surface/screen center. In Unity, this results in having a camera frustum that passes through the edges of our projection planes/screens as shown in Figure 3.

Our plugin uses the methods described above to make existing Unity 2017 projects CAVE compatible. The plugin is versatile regarding the physical setup and hardware, being able to deal with sets of heterogenous projectors and/or flat screens in arbitrary positions and orientations. While originally developed to create CAVEs, it supports any other configuration of up to 8 displays, such as interactive walls, floors or any other combination of flat surfaces, always providing parallax effect. It works by instantiating at runtime a virtual equivalent of the real CAVE parameterized in an XML configuration file. The virtual CAVE is composed of a set of prefabs from the types described above. In short, cameras that are attached to the user's head, thanks to the Kinect tracking, view the virtual world while having their frustum continuously adjusted to match the correspondent CAVE wall/screen edges. Then their images are either streamed directly to display screens or to the virtual CAVE projectors that warp the images to match the real ones, these warped images are projected into the real-world CAVE walls, resulting in the correct perspective projection.

After importing the KAVE plugin into a Unity project, existing cameras components should be deactivated and the prefab "CAVE Manager" added as a child of the player game object. This game object will take care of everything, including loading the calibration file and generating all the KAVE elements at runtime. The "CAVE Manager" position inside the game matches the reference frame origin of the CAVE calibration, usually on the floor. It is also possible to customize the cameras that are now instantiated at runtime by editing the parameters of the "User View Camera" prefab, such as culling mask and clipping planes or to add visual effects like it is possible in any Unity camera. The addition of this plugin to an existing unity project to make it CAVE compatible takes less than five minutes. The KAVE plugin, source code and demonstration project are fully available in its development repository at https://bitbucket.org/neurorehablab/kave.

## 3.2 KAVE Calibrator

A standalone tool to calibrate the KAVE was developed in Unity. The purpose of this tool is to allow the user to generate a configuration file that represents his physical CAVE setup, which the KAVE plugin can then load to recreate it in virtual space. It provides a 3D virtual environment (see Figure 4 A), similar to the Unity editor where up to 8 surfaces and projectors can be dynamically added, individually adjusted in position and orientation, and resized to match the real-world configuration. Surfaces are associated with projectors to ensure that there is no projection overlap or to allow a projector to project on several walls at once (such as a corner). The mapping of the projections into the surfaces is done visually on the CAVE walls themselves, a 4-sided red polygon is projected by each projector over the wall they are facing, the user then drags with the mouse each of the corners of this mask to match the corners of the real wall (see Figure 4 B), the viewport coordinates of the 4 corners are saved. Once these steps are done, the tracking sensor is added. Its horizontal position and azimuth angle relative to the CAVE reference frame are set on the calibrator. The height to the floor, pitch, and roll angles are automatically obtained from the Kinect SDK estimation of floor plane. The software saves the information in an XML file which is later loaded by the KAVE plugin or by the calibrator again for further adjustments.

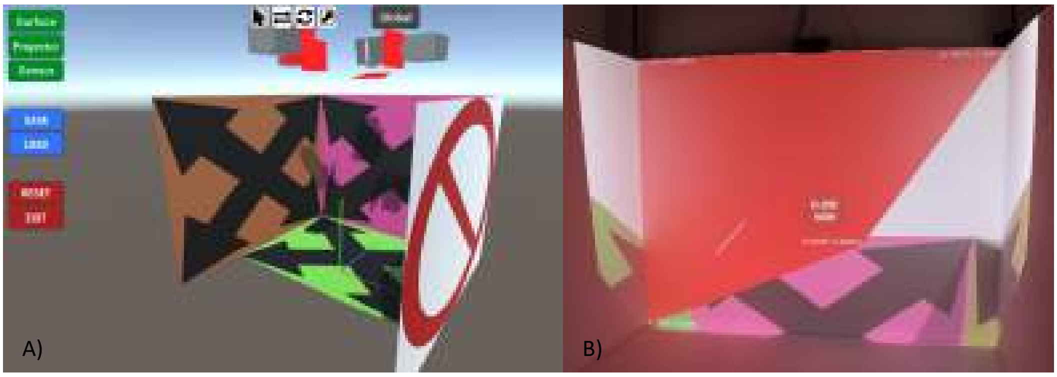

**Fig. 4. A) KAVE Calibrator being used to create a CAVE with 3 walls and floor along 4 projectors and their projection masks. B) Using the calibrator to find the wall corner coordinates in the projector viewport reference frame, the bottom right corner is still missing.**

## 4 KAVE CONFIGURATIONS AND EXAMPLE APPLICATIONS

Although the KAVE plugin was originally developed with a CAVE paradigm in mind, its configuration versatility makes it able to support many other setups. This section presents a limited list of example configurations and applications implemented with the KAVE. The diverse examples of supported configurations showcase the flexibility of the tool [13], as the same compiled application can be deployed in a multitude of setups simply by providing the corresponding calibration file at runtime. The example applications demonstrate the power it provides to the users, ranging from supporting simple VR visualization, by adding motion parallax, to UI and VR interaction through full-body tracking.

### 4.1 NeuroRehabLab CAVE

The KAVE is currently being used to power applications in our CAVE. The NeuroRehabLab CAVE has a configuration of three adjacent walls at 90 degrees with each other and a floor, the walls are 2.8 meters wide by 2.2 meters tall and the floor 2.8 by 2 meters, allowing up to four projections simultaneously (Figure 5).

The projectors are four Optoma GT1080, with 1080p image resolution and throw ratio of 0.5:1. The low throw ratio helps to reduce user shadows inside a CAVE of this dimensions, and the price (around 800€) is on par with market values. For tracking, it uses a single Kinect V2 sensor located on the top of the center wall. In our effort to reduce the costs associated with CAVE ownership we custom made our own structure to support the projectors, sensors and projection surfaces. The metallic structure is made of galvanized iron, allows adjustments of projectors to floor and wall distances trough sliding supports as well as individual orientation. Each wall is made of 3 wooden sheets (1.2 cm thick) which were plastered and painted white, while the floor is made of a white hard vinyl sheet. Besides providing support to the Kinect sensor, the structural beams can also support other types of sensors. In our case, a set of two HTC base stations for the HTC Vive head-mounted display are also installed. The price of the CAVE (structure, projectors, and Kinect) was just under 3,800€ (projectors costing 3,200€), while the accompanying computer (Quad Core 3.4GHZ processor, 8GB of RAM, and Radeon RX 580 8GB graphics card) cost was 1,200€. Calibration of this setup using the KAVE Calibrator described in Section 3.2 is straightforward.

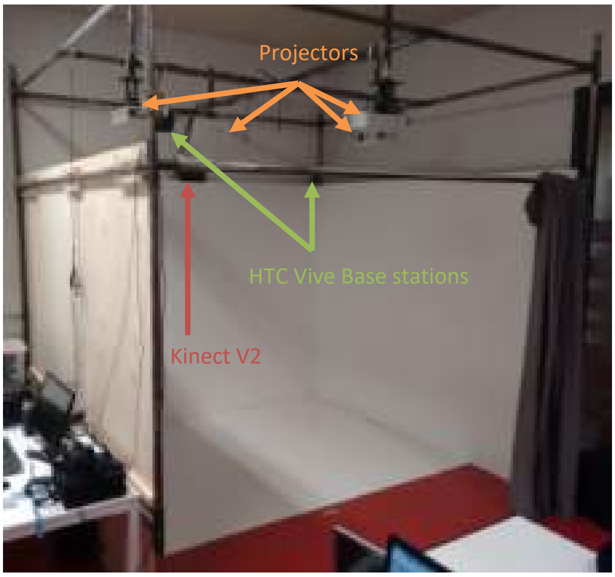

**Fig. 5. NeuroRehabLab CAVE.**

## 4.2 Interactive Floor

A simple configuration of one projector facing a surface can turn it into a large gaming/visualization platform supporting direct interaction and the feeling of depth due to motion parallax. By attaching a projector vertically to a wall or ceiling, a large projection area can be made. An interactive floor can be implemented in this fashion by placing the Kinect sensor facing the user (Figure 6). This configuration has been used for the development of a set of Portuguese themed exergames [6]. Calibrating a one surface/one projector setup such as this one is the simplest calibration possible, requiring only the floor dimensions and orientation, projector viewport coordinates of the 4 floor corner points and the Kinect position, all relative to the center of the floor.

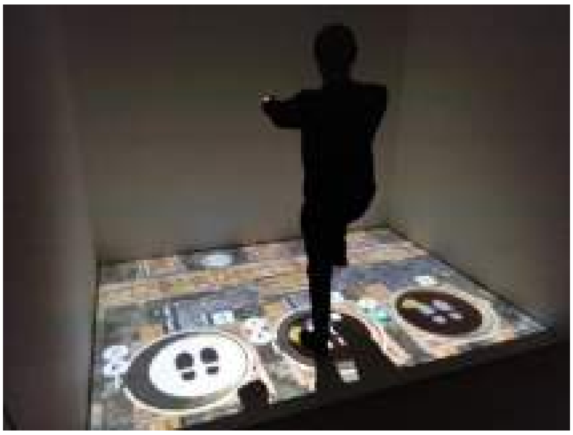

**Fig. 6. Interactive floor being used to play a grape stomping game [6] on the floor of the NeuroRehabLab CAVE.**

## 4.3 Interactive Wall / PowerWall

Multiple projectors can be used to project over a single large surface (such as a long wall). This is possible because the plugin can seam multiple projections into a single view over adjacent areas. Figure 7 shows one example of such setup where 2 Acer S5200 (1024x768, 3000 Lumens) projectors are used to create a 2.84 + 2 m wide by 2.6 m tall projection wall. The parallax effect is provided by setting the Kinect sensor behind the user. Having the Kinect placed behind the user does create a problem for interaction as this sensor is not trained to recognize people from the back and instead assumes they are facing the sensor. Because of that, the accuracy in body joint estimation is reduced and the body seen as inverted by the system, still the head can be tracked. Alternatively, the Kinect can be placed in a better front-facing position. Up to 8 projectors with arbitrary resolutions can be used to create a high-resolution interactive wall with the KAVE plugin in this configuration. The main calibration problem with this setup concern the precise way in which the user defines the boundaries between adjacent projections, the seam. First the user must ensure that 2 adjacent projections overlap, then physically mark the desired corners of each rectangular “virtual wall” on the real wall, finally the calibrator can be used in the same manner as the previous 2 configurations, including to obtain the position of the markers in each projector viewport coordinates, just like they would to map the 4 corners of individual walls in a CAVE.

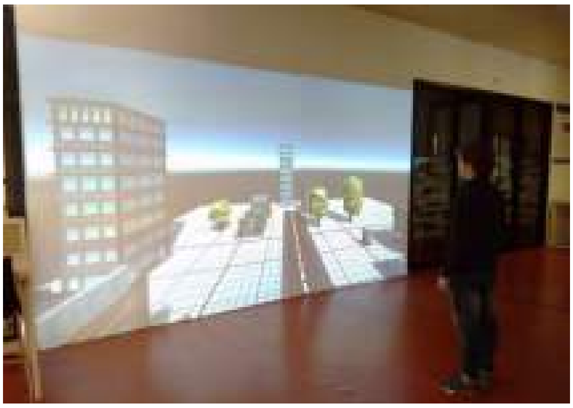

**Fig. 7. Interactive Wall, 4.84 m by 2 m of 2 projectors.**

## 4.4 Corner Projection

A single projector can be used to project into multiple surfaces simultaneously, such as the corner of a room. This could be used to create a CAVE using 2 walls and floor projections with just one projector. An example of this is shown in Figure 8, where one portable LG Minibeam PW800 (1280x800, 800 Lumens) projector creates a 1.2 m x 1.2 m, 0.9 m tall CAVE on a corner. The corner CAVE concept can be made larger by simply using a low throw ratio projector covering a larger area. The main issues regarding this setup are where to place the Kinect relative to the user and the reduced brightness and resolution that results from having a projector facing multiple projection surfaces no perpendicularly, at high angles. Regarding calibration, as in the previous example, temporary markers on the walls are needed to serve as reference for each corner of every wall.

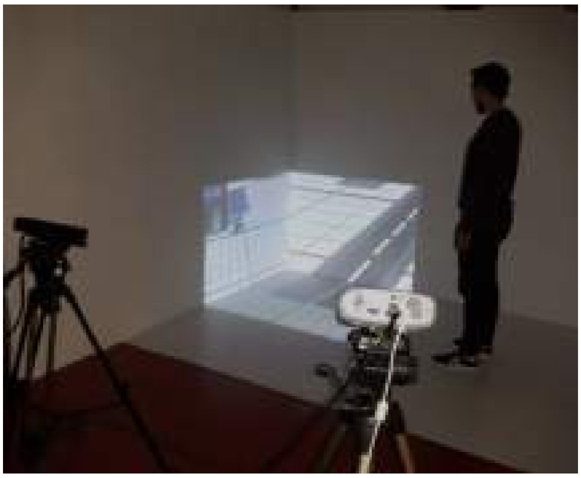

**Fig. 8. Improvising a minimal CAVE with a corner projection.**

## 4.5 Parallax Screens

Since the plugin supports display screens as well as projectors, it can also be used for screen-based CAVEs, or even living room and desktop applications. Here we illustrate it with on desktop screen with parallax effect (Figure 9). By using multiple or larger screens, a more immersive setting can be created. Even though the KAVE plugin supports this configuration, the KAVE calibrator does not yet support screens. This calibration was instead achieved by manually editing the calibration file to match the screen dimensions and orientation as well as the Kinect position relative to it.

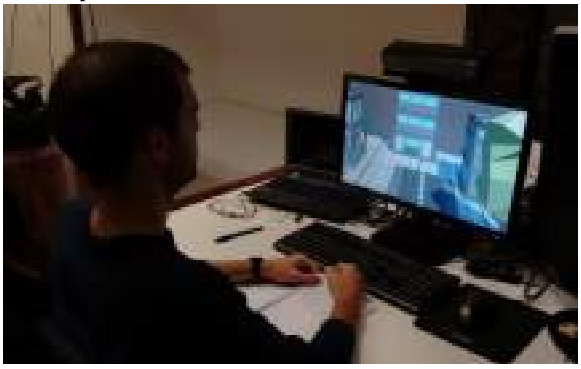

**Fig. 9. The Parallax Screen acts as a window into a virtual scene, or virtual elements can pop out from it.**

## 4.6 Exclusive Use of the Plugin for Body Tracking

By creating a calibration file deprived of screens and projectors/surfaces instances, the plugin will only generate the Kinect V2 interface instance, responsible for avatar creation and control. Although this foregoes all the perspective control and projection management that the plugin provides, it can still be useful for users who have other means of visualization in mind and are only interested in having their full-bodies represented and/or interacting with a virtual scene through Kinect tracking. An example of such an application would be the integration with HMD to provide virtual bodies to its users, allowing players to see, in 1st person, their own VR bodies and others around them, as shown in Figure 10. This is possible just by making sure that both HMD and KAVE reference frames are coincident.

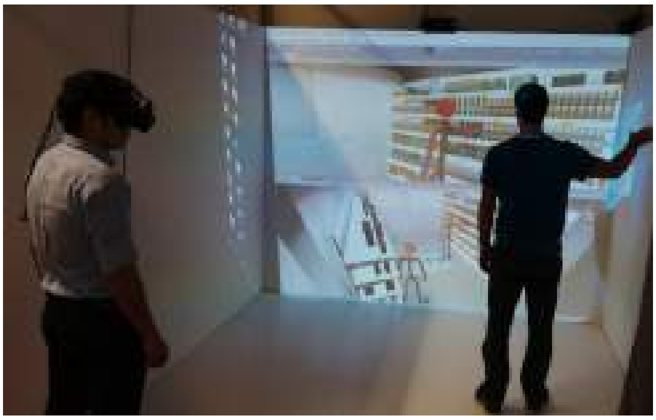

**Fig. 10. The KAVE plugin enabling an HMD wearer to see his own virtual body and others tracked by the Kinect. The HMD view and perspective view of the scene are being exceptionally projected on the front wall, top and bottom half respectively.**

## 4.7 Example Applications

The NeuroRehabLab CAVE powered by the KAVE plugin gave the authors' research institute a powerful yet affordable VR system. A simple application was used during the plugin development as a way to internally test the motion parallax effect generated. This application consists in a set of static 3D low polygon count assets and the KAVE plugin prefab "CAVE Manager" placed among them. The machine ran Windows 10, had an Intel processor I7-6700 3.4GHZ, 8GB of RAM, and a Radeon RX 580 8GB graphics card connected to a CSV-5400 quad monitor Multi-Stream Transport (MST) Hub. This setup achieved a stable performance of 60fps while displaying 8.2944Mp (4x 1920x1080p), averaging 4 ppcm, and can be seen in Figure 11.

The KAVE plugin was integrated into 2 already existent games, developed by institute collaborators, and by incorporating it in an embodied interaction course project.

"Explornesia: The Battle" is a real-time multiplayer sea battle game, featuring age of sail ships controlled from a 3rd person point of view. The conversion to a CAVE playable version was as easy as replacing the standard Unity game camera with the KAVE plugin game object. By editing the position and scale of the KAVE relative to the player's ship two prototypes where made. One where the player controlled the real size ship from the deck in first-person, see Figure 12 A), and another maintaining the original 3rd person perspective where the player's ship was displayed as a 1-meter long ship floating on the CAVE floor.

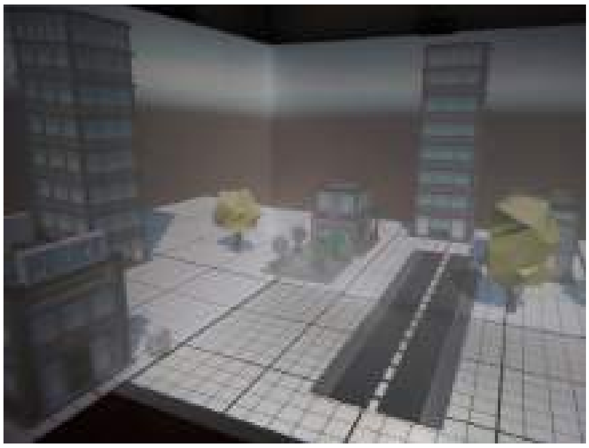

**Fig. 11. User perspective of the KAVE demo project running in the NeuroRehabLab CAVE.**

The second game was "Keeper of Intheris", an online, turn-based strategy game designed to be played on a computer screen with keyboard and mouse from an angled perspective. The developers modified the game and added a non-playable proof-of-concept first-person observer view scene with the KAVE plugin (Figure 12 B). It supports navigation through a handheld thumb joystick. Additionally, full-body interaction with some elements of the game environment was enabled through physics simulation. The observer can interact with them via body motion, such as being able to physically kick virtual rocks. Development of this scene required: creating a standalone non-playable scene of the game, where the game characters are in idle animation; adding rigid body and collider components to rocks; removing the scene original camera and adding the "CAVE Manager" to the scene; activating the Kinect body colliders; and add joystick support.

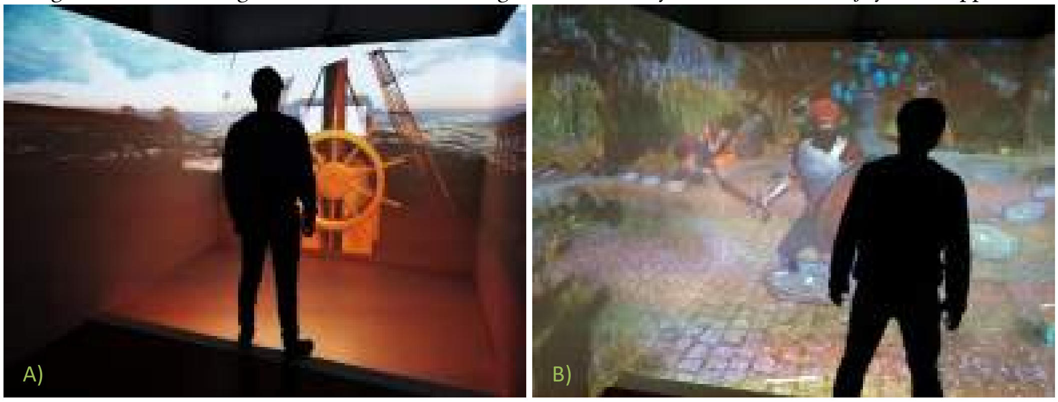

**Fig. 12. Using the KAVE to visualize games from the "inside": A) "Explornesia" being played from the deck of a ship. B) An observer inside "Keepers of Intheris".**

Over a college course of embodied interaction, a group of students developed a proof of concept prototype for atmospheric and oceanic data visualization. By observing the work and interviewing researchers from the "Oceanic Observatory of Madeira" (OOM) a set of ideas to facilitate natural data visualization was put in practice with the KAVE. The resulting prototype featured a VR presentation of the topographical data in the center of the CAVE, which the user could explore by moving around it. The interaction was done on the wall-sized UI, either through buttons or sliders, all activated solely by the hand position of the users provided by the KAVE.

## 5 DISCUSSION

Even though CAVEs have been around for a considerable amount of time they remain hard (and costly) to implement and maintain. To this end, we developed an easy to use plugin for Unity developers that adds CAVE support and motion parallax effect to virtual environments across multiple displays. This, together with our configuration and calibration tool provide all the required elements for a functional monocular CAVE system. A summary of strengths, weaknesses, opportunities, and threats is presented in Table 1.

**Table 1. SWOT Analysis of our Contribution**

| **Strengths:** | **Weaknesses:** |
|---|---|
| • Free and open source software<br>• Low-cost tracking<br>• Versatile in multiple hardware and screen configurations<br>• Space geometry calibration independent from VE development and compilation, all KAVE elements created at runtime<br>• Unobtrusive interaction<br>• Software easy and fast to use | • Limited tracking accuracy<br>• Monocular<br>• Price of the projectors<br>• Tracking area limited to the field of view of one sensor |
| **Opportunities:** | **Threats:** |
| • Large community of Unity developers<br>• Very competitive compared to regular CAVE cost<br>• Emergence of new game modalities supporting full-body interaction in immersive environments<br>• Alternative middleware solutions for CAVEs are expensive<br>• Popularization of VR | • Dependency on Kinect V2 for Windows<br>• High investment in hardware for the average developer<br>• Dominance of immersive VR by Head-Mounted Displays<br>• Lack of dedicated content for CAVEs |

Compared to the Uni-CAVE [19], noncommercial software with similar goals of powering CAVEs, the KAVE distinguishes itself from it for supporting full-body tracking and interaction, and runtime instantiation of all its elements. This means that the developer of the VE can compile a KAVE project independently from its users and their projection setups. This single built project can be distributed to different users, which only need to add their own calibration files to experience the same VE correctly mapped to their setup. With the KAVE, contrary to the Uni-CAVE, there is no need for the users to ever touch the VE development project or for the developer to build a custom project for every user individual setup. This difference is also applicable to RoomAlive [8], a toolkit for SAR, which requires the static room geometry automatically obtained from its calibration tool to be loaded into the Unity editor project, thus tying VE development to a specific space geometry. Although RoomAlive also supports the dynamic surfaces to be obtained online from the depth stream of Kinect, this is intended for projection mapping on moving physical objects. This means that in both the Uni-CAVE and RoomAlive, the application user and the application developer are the same, while with the KAVE the can remain independent. Also, to power a CAVE with RoomAlive, multiple sensors (each running on a dedicated PC) would be needed to cover the whole projection setup and user interaction space, dramatically increasing the cost and complexity of what would be otherwise a very simple setup. It is of note that calibrating RoomAlive for planar surfaces, such as the ones in CAVEs, requires the procam units to be temporarily oriented towards non-planar surfaces due to the nature of their autocalibration procedure. Lastly, neither of these two tools support screen displays.

Some difficulties to the adoption of this system exist. Among them, the Kinect V2 is less accurate than the gold standard IR marker-based sensors, with a 1 cm mean error and standard deviation for the "head joint" [14]. However, it has been reported that human head movements can average up to 2cm when standing and looking at static images in screens [2], this might indicate that such accuracy can be acceptable. Regardless of our solution, a CAVE still requires a minimal investment in projectors and physical structure, but it can be as low as 5,000€ as shown by the NeuroRehabLab CAVE. Our low-cost KAVE solution presents a good alternative to the traditionally high cost of these systems, namely by reducing the software, hardware and sensor costs by using low-cost off-the-shelf equipment. In particular, our solution can have high acceptance by the industry and academia, as it offers most of the features of high-end commercial CAVE systems with a ready to use software bundle for Unity. While the chosen platform and the ease of setup potentiate the adoption by the large community of Unity developers, it will also contribute to the promotion and growth of CAVEs as research, professional and gaming platforms, and the creation of novel content exploiting both full-body interaction and immersive VR. The fact that the Kinect V2 sensor provides tracking of 25 body joints per user is an additional clear advantage over traditional marker-based systems with the potential of developing new non-obtrusive interaction modalities in VR and games.

Finally, alongside CAVEs, the plugin in its more distilled configuration form, of just one projection surface, can also be used to provide simple interactive experiences for public spaces, such as imaginary windows to virtual worlds, interactive walls, floors or ceilings, facilitating and opening opportunities to develop applications for these settings. Thus, the extreme versatility of this plugin makes it go beyond the original purpose of CAVE creation and allows both developers and users to easily explore and set up different projection mapping scenarios. Besides the addition of motion parallax to Unity projects, the most interesting aspect of this plugin is that it takes less than 5 minutes to convert a Unity game into an immersive CAVE experience with the potential of natural user interaction. By having a virtual skeleton matching the user's own body inside the virtual environment, the whole body becomes an interface in Unity.

## 6 CONCLUSIONS

In this paper, we present and describe the KAVE Unity plugin and the principles that make it work. The plugin is a publicly available a tool for the creation of surround-screen projections, supporting up to 8 projectors or screens in any combination or geometrical arrangement. Motion parallax and full-body tracking are featured using affordable off-the-shelf components. It differentiates itself from the commercial related work by being free and open-source, and from the academic related work by its versatility. By untying the physical setup calibration phase from the VE development we distinguish the KAVE from the previous software solutions where each application was tailored before compile time to a specific lab setup. We provide specific examples of the KAVE versatility in both applications and screen setups. Two examples of VEs being converted into CAVE compatible demos are provided and five other variations in terms of hardware elements configuration, all of them accessible without requiring VE source code access. Given its ease to use and low requirements we expect the plugin to be a valuable tool for research laboratories and VR enthusiasts.

## ACKNOWLEDGMENTS

This work was supported by the Fundação para a Ciência e Tecnologia through the AHA project (CMUPERI/HCI/0046/2013), by the INTERREG program through the MACBIOIDI project (MAC/1.1.b/098), and LARSyS – UID/EEA/50009/2013. The authors would like to acknowledge Luis Vélez for the development of the KAVE Calibrator and José Camacho for the construction of the CAVE setup. Further acknowledgment is given to the developers of "Keepers of Intheris", Clumsy Games Studio; and the developers of "Explornesia: The Battle" MadFox Studios, for the availability to integrate the plugin in their games.

## REFERENCES

[1] Benko, H., Jota, R. and Wilson, A. 2012. MirageTable: Freehand Interaction on a Projected Augmented Reality Tabletop. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2012), 199–208.

[2] Ciria, L.F., Muñoz, M.A., Gea, J., Peña, N., Miranda, J.G.V., Montoya, P. and Vila, J. 2017. Head movement measurement: An alternative method for posturography studies. *Gait & Posture.* 52, (Feb. 2017), 100–106. DOI:https://doi.org/10.1016/j.gaitpost.2016.11.020.

[3] Cruz-Neira, C., Leigh, J., Papka, M., Barnes, C., Cohen, S.M., Das, S., Engelmann, R., Hudson, R., Roy, T., Siegel, L., Vasilakis, C., DeFanti, T.A. and Sandin, D.J. 1993. Scientists in wonderland: A report on visualization applications in the CAVE virtual reality environment. *Proceedings of 1993 IEEE Research Properties in Virtual Reality Symposium* (Oct. 1993), 59–66.

[4] Cruz-Neira, C., Sandin, D.J. and DeFanti, T.A. 1993. Surround-screen Projection-based Virtual Reality: The Design and Implementation of the CAVE. *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1993), 135–142.

[5] Febretti, A., Nishimoto, A., Thigpen, T., Talandis, J., Long, L., Pirtle, J.D., Peterka, T., Verlo, A., Brown, M., Plepys, D., Sandin, D., Renambot, L., Johnson, A. and Leigh, J. 2013. CAVE2: a hybrid reality environment for immersive simulation and information analysis. (Mar. 2013).

[6] Gonçalves, A., Muñoz, J., Gouveia, É., Cameirão, M. and Bermúdez i Badia, S. 2017. Portuguese Tradition Inspired Exergames for Older People. (Funchal, Portugal, 2017).

[7] Jones, B.R., Benko, H., Ofek, E. and Wilson, A.D. 2013. IllumiRoom: Peripheral Projected Illusions for Interactive Experiences. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2013), 869–878.

[8] Jones, B., Sodhi, R., Murdock, M., Mehra, R., Benko, H., Wilson, A., Ofek, E., MacIntyre, B., Raghuvanshi, N. and Shapira, L. 2014. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-camera Units. *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2014), 637–644.

[9] Juarez, A., Schonenberg, W. and Bartneck, C. 2010. Implementing a low-cost CAVE system using the CryEngine2. *Entertainment Computing.* 1, 3–4 (Dec. 2010), 157–164. DOI:https://doi.org/10.1016/j.entcom.2010.10.001.

[10] Kuhlen, T.W. and Hentschel, B. 2014. Quo Vadis CAVE: Does Immersive Visualization Still Matter? *IEEE Computer Graphics and Applications.* 34, 5 (Sep. 2014), 14–21. DOI:https://doi.org/10.1109/MCG.2014.97.

[11] MiddleVR | MiddleVR: Improve reality! *http://www.middlevr.com/home/*. Accessed: 2018-01-19.

[12] Ohno, N. and Kageyama, A. 2007. Scientific visualization of geophysical simulation data by the CAVE VR system with volume rendering. *Physics of the Earth and Planetary Interiors.* 163, 1–4 (Aug. 2007), 305–311. DOI:https://doi.org/10.1016/j.pepi.2007.02.013.

[13] Olsen, D.R., Jr. 2007. Evaluating User Interface Systems Research. *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2007), 251–258.

[14] Otte, K., Kayser, B., Mansow-Model, S., Verrel, J., Paul, F., Brandt, A.U. and Schmitz-Hübsch, T. 2016. Accuracy and Reliability of the Kinect Version 2 for Clinical Measurement of Motor Function. *PLOS ONE.* 11, 11 (Nov. 2016). DOI:https://doi.org/10.1371/journal.pone.0166532.

[15] Poschner, F. 2014. Fire fighting and related simulations in a CAVE using off-the-shelf hardware and software. *Proceedings of SIGRAD 2014* (Göteborg, Sweden, Jun. 2014), 33–40.

[16] Schuchardt, P. and Bowman, D.A. 2007. The Benefits of Immersion for Spatial Understanding of Complex Underground Cave Systems. *Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology* (New York, NY, USA, 2007), 121–124.

[17] Stuerzlinger, W., Pavlovych, A. and Nywton, D. 2015. TIVS: temporary immersive virtual environment at simon fraser university: a non-permanent CAVE. *2015 IEEE 1st Workshop on Everyday Virtual Reality (WEVR)* (Mar. 2015), 23–28.

[18] Sutherland, I.E. 1968. A Head-mounted Three Dimensional Display. *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I* (New York, NY, USA, 1968), 757–764.

[19] Tredinnick, R., Boettcher, B., Smith, S., Solovy, S. and Ponto, K. 2017. Uni-CAVE: A Unity3D plugin for non-head mounted VR display systems. *2017 IEEE Virtual Reality (VR)* (Mar. 2017), 393–394.

[20] Ware, C., Arthur, K. and Booth, K.S. 1993. Fish Tank Virtual Reality. *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems* (New York, NY, USA, 1993), 37–42.

[21] Wilson, A.D. and Benko, H. 2010. Combining Multiple Depth Cameras and Projectors for Interactions on, Above and Between Surfaces. *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2010), 273–282.

Received January 2018; revised March 2018; accepted June 2018.