Mixed Reality TVs: Applying Motion Parallax for Enhanced Viewing and Control Experiences on Consumer TVs

Eduardo Rodrigues, Lucas Figueiredo, Lucas Maggi,
Edvar Neto, Layon Bezerra, Veronica Teichrieb
Voxar Labs
Informatics Center - UFPE
Recife, Brazil
{ehmr, lsf, lom, excvn, ltb, vt}@cin.ufpe.br

João Marcelo Teixeira
Departamento de Eletrônica e Sistemas - UFPE
Recife, Brazil
jmxnt@cin.ufpe.br

***Abstract*—Smart TVs have been popularized in the last years, presenting continued innovation regarding different topics such as connectivity, platform support, and options for interaction. Examples of LG and Samsung present interaction options such as voice and gesture control, showing these emerging technologies that were previously tackled mainly on research investigations, as market alternative solutions for the already established remote control.**

In this paper, we address the interactivity with Smart TVs by using body gestures combined with the visualization modification through the Motion Parallax effect. We show that by tracking the user viewpoint and hand tips, plus applying the Parallax effect, it is possible to fully calibrate the virtual environment demonstrated by the TV with the real world. Once the calibration is performed, the TV becomes a Mixed Reality display, showing its content coupled with the real world, and allowing the user to perceive this coupling through the depth cue provided by the distortions of the Parallax effect. This calibration enables the use of a set of interaction techniques such as enabling the user to visually pinpoint and touch the virtual items of a TV menu. Moreover, we show that by modeling the interaction with the TV content as a 3D virtual world interaction, it is possible to use new metaphors based on physical interactions such as collisions, lights and shadows and magnetic attractions.

***Keywords*-mixed-reality; natural interaction; motion parallax;**

I. Introduction

Since the launch of the Cathode Ray Tube (CRT) Televisions in the first half of the $20th$ century, its concept has been reinvented decade over decade, extending its use over the globe by delivering improved experience and innovation to the end-users [1]. In 2016, a report from Statista showed that the number of TV-equipped households grew since 2010, reaching more than 1.6 billion TV households worldwide [2]. One of the latest innovations brought to consumer TVs is the integration of the screen display with an operating system being called Smart TVs. This integration allows a set of new features for the TVs such as access to the Internet, support to third-party applications and novel ways of interaction. Based on a market report from BI Intelligence [3], Mark Hoelzel states in an article from Business Insider that Smart TVs "will account for 73% of global flat screen TV shipments by 2017".

Smart TVs play a major role considering the future context of Internet of Things (IoT) and Smart Homes. While providing direct visual and auditory feedback for users in living-rooms and bedrooms, Smart TVs can be the center of communication for users homes, providing information in dashboards, displaying conversational avatars, and so on. This future context also presents new concerns towards the interaction with everyday objects. Once all things are connected and provide intelligence, users will need to interact with them at some point. Also, there will be a set of Virtual Reality and Augmented Reality content coexisting with the user home environment, which will also be presented as interactive interfaces. Looking at this scenario, it is unpractical to ask the user to learn a new interaction command for each new interface. Therefore, the interaction with the TV should be revisited, having in mind this place in the future, and seeking for alternative ways of interaction besides the exclusive use of the remote control.

Remote controls are the common ground for interacting with TVs for the last decades. These controls communicate user inputs through infrared coded messages and require the user to point the control towards the TV receiver while pressing a button for the desired command. The available buttons allow commands such as indicating the number of the target TV channel, increasing or decreasing volume, and some menu options such as *opening* the home menu, *navigating* (left, right, up, down), and going *back* to the previous menu screen. By providing input for this set of commands, most TVs can be fully controlled. In addition to standard buttons, the remote control is also a target of innovation. As an example, the "LG Magic Remote Control with Voice Mate" [4] shows additional options for interaction. The improvements include wheel buttons, inertial sensors used to move a cursor on the screen, and microphones coupled to a speech recognition system "capable of recognizing how you naturally speak to change channel, volume or find something to watch".

Another example by Samsung shows gesture control applied for menu navigation among other goals [5]. In this example, the user moves her open hand with the palm facing the TV while a cursor (or selection option) moves across the TV screen accordingly. Such examples demonstrate the interest of the industry on bringing innovation towards the way users interact with TVs, corroborating with a future envisioned scenario in which the interactivity with the TV extends to other input channels (e.g. voice and gestures).

In this paper, we introduce an interface and interaction technique for menus and applications on Smart TVs. Our approach focuses on gestural interaction for 3D content, being reusable for planar cases once 2D content can be mapped to 3D environments. By using hand and head tracking technologies (e.g. Microsoft Kinect), and the Motion Parallax effect[1], it is possible to fully couple the content shown on the TV with the real world in position, orientation, and scale. This means a content on the TV that is supposed to occupy 10 centimeters of size in the virtual world, will also visually occupy 10 centimeters of size in the real world.

By using this coupling capability of the Motion Parallax effect, the TV becomes a Mixed Reality device, displaying 3D virtual content on the user living room. The content can be placed behind, at and in front of the TV screen. Therefore, we simulate that there are virtual interactive objects in the space between the user and the TV. Also, we track the user head and hands and calibrate these tracking results coordinate system with the 3D virtual environment shown by the TV. This way, rather than recognizing specific hand gestures for particular actions (e.g.: open palm to stop a playback), we provide a fully coupled environment, in which user actions coexist with the 3D content regarding the position, orientation and scale. We show that, in this setup, it is possible to explore some interaction techniques and metaphors:

1) *Visual pinpoint:* Provide to the system the understanding about when the user, using her fingertips, is occluding a particular virtual object in her sight. This technique allows the user to rapidly indicate an item of interest in the virtual world (e.g.: a menu item on the TV home screen). Also, it discards the need to map a cursor on the screen, being more direct.
2) *Virtual touch:* Allow the system to recognize when the user actively touches virtual components placed within her arms reach. Touching a virtual object may be the trigger to select it, for example.
3) *Physical simulations:* Apply physical simulations according to the interaction. For example, we can simulate that the user has a magnetic directional power to attract objects. We can also simulate direct solid collisions between the user hand and fingers with the virtual objects, making objects bounce once they are hit. We can also simulate the user has a special flashlight which works only in the virtual world and highlights the direction she is pointing to.

On this paper, we introduce as contributions:

- a tool for the application of the Motion Parallax effect on Unity 3D projects;
- a calibrated environment on the same tool, in which both user body and virtual objects coexist;
- a novel interaction technique for visually pinpointing virtual objects of interest, by coupling the Motion Parallax effect with the tracking of user's viewpoint and hands;
- a system, including a 3D menu interface for Smart TVs, considering the mentioned features of magnetic attraction and flashlight highlighting;
- a Mixed Reality sample application using our setup. The application is a mini-game called "Blocking Blocks" and is built for showing additional entertainment potential on the utilization of the Motion Parallax effect with body tracking and gesture recognition for Smart TVs on the living room environment.

The paper structure is organized as follows. Section II presents the related works regarding the use of the Motion Parallax effect and gesture controls and 3D menus on TVs and other similar display setups. On section III we discuss the main techniques which are related to the conception and development of both visualization and interaction systems. Section IV illustrates the main concepts of our design and interaction, and on section V we present the implementation details of our work. At last, section VI shows our main results and on section VIII we draw our conclusions and future works.

II. Related Works

While we found no work that directly uses the Motion Parallax technique on consumer TVs for visual pinpoint and touch of the virtual objects, a set of works tackle these related topics separately.

A. Motion Parallax effect on TVs

3D TVs provide a greater sense of presence and immersion than ordinary two-dimensional images. They also aim to output extremely realistic and dynamic 3D pictures by taking full advantage of its high resolution and wide-screen features. However, these stereoscopic images not rarely lead viewers to feel tired after hours of watching. Through a subjective evaluation test, Ide et al. [7] found that the term "sense of presence" is often correlated to strong depth perception, while 2D images - qualified as "flat" - were placed on the opposite side of the line. Additionally, the term "spreading out back from screen" - a consequence of the parallax effect - was also related to "easy of viewing" factor.

Fehn [8] describes a novel data representation format, which associates monoscopic color video to per-pixel depth information. This format allows introducing depth perception into the existing 2D digital TV. This system provides an additional *extra-stereoscopic* depth cue through head-motion parallax that does not suffer from the well-known "shear-distortion" usually experienced with stereoscopic or auto-stereoscopic 3D TVs. This same feature could be used to increase the sensation of depth on a regular monoscopic 2D TV.

A TV-like Fishtank VR effect is evaluated by [9]. Their solution applied the parallax effect on a 50-inch screen

[1]The Motion Parallax effect is applied by a displacement and distortion of the virtual camera, through the use of a technique called Fishtank VR. The technique name "Fishtank VR" was coined in 93 by Ware and colleagues [6], and massively popularized earlier in 2008 by Johnny Lee's TED Talk available at https://goo.gl/Xeh4qz

displayed by a projector simulating a gaming experience in a living room. In this case, the user's viewpoint was tracked using a Microsoft Kinect to simulate a display behaving like a glass window. They argue that users were amused at the first moment in this scenario, standing up and moving around to explore the virtual scene according to their gameplay needs. In the experiment, users usually moved the viewpoint by crouching to see better the horizon, which was later identified as a negative behavior. Despite that, users immediately felt a different, and less joyful experience when they played the game with no parallax effect.

Heinrichs and McPherson [10] use this concept to simulate a sense of physical presence in a telepresence system. To do so, they mapped head and body movements to real camera (at the other side of the teleconference) movements, taking into account the distance between the viewer and the screen. The setup is arranged by a robotic arm mechanism to move the camera in the three axes, a head tracking system powered by a Microsoft Kinect sensor and video processing to straighten and crop the image. Although the result is quite functional, image resolution suffered as a consequence of cropping the video input and objects appeared to be magnified.

Comparing Fishtank Virtual Reality Displays with a CAVE, Damiralp et al. [11] pointed out that for looking in tasks - in which users view and manipulate a virtual world from the outside-in and interact with a virtual object that is smaller than their body - Fishtank VR displays are more conducive. Such displays separate the user's frame reference, which forces the user to look into the virtual world. However, in tasks that user's peripheral vision must be filled CAVE-like displays are more appropriated than Fishtank VR.

B. Gesture Controlled TVs

Vision-based natural interaction systems for face tracking and gesture recognition are also studied as solutions for interaction with Smart TVs. One of the key advantages of using Computer Vision for recognizing gestures is that it allows the user to interact with the system without the need to wear or attach any accessory or device.

Lee et al. [12] proposed a vision-based system capable of recognizing user face and hand gestures for Smart TVs. They used the face recognition to associate personalized services for each user. They also delivered recognizers for five different hand gestures associated to the metaphors of moving *left, right, up, down* and a *push* gesture. The gestures were used to perform commands like changing channels and volume.

Watanabe et al. [13] present a gesture controlled cursor as shown in the Remote Touch Pointing technique. By tracking user body and hand they create a ray direction, which starts at a base point on the user's chest and passes through the user's hand. Once the ray crosses the screen (which is previously mapped), they use that intersection to set the cursor position. They also perform a comparative analysis of the procedure of using TV systems while using a remote control versus using their gesture control solution as shown in table I.

Table I
COMPARISON OF STEPS REQUIRED FOR SPECIFIC PROCEDURES ON THE WATCHING TV ACTIVITY WHILE USING REMOTE CONTROL AND WATANABE ET AL. TECHNIQUE [13]

Procedure	N Steps (Remote)	N Steps (Gestures)
Startup TV	5	1
Change contents	7	4
Stop viewing	5	1

Kim et al. [14] designed a solution for this scenario and called it Ambient Wall. Using a Microsoft Kinect sensor for hand tracking, they proposed an interactive wall that can control a digital TV, getting a program list or even browsing through the TV channel list. They highlighted the usefulness in controlling surrounding devices in a living room but also emphasized how this interaction method could be helpful for the disabled people who have difficulty in moving their bodies without their guardians.

In parallel, computing devices are already pervading into everyday objects. The interaction with these smart things is supposed to be guided by their affordance, i.e. the ability of the artifacts to express how they should be manipulated. Gestures can also be considered a movement or state of any physical object resulting from human manipulation. Therefore, a gesture can be application-independent, i.e. devices may be controlled by gesticulation with an independent object. Considering this line of research, Ferscha and Resmerita [15] propose an application independent Gesture Library (GLib). They describe a smart home environment in which a solid cube is employed to control a TV. Despite this graspable interface being considered to be intuitive to the researchers, this proposal seems to come from the efforts of standardizing embodied interaction.

Notwithstanding all these technologies seem to need an entirely new set of devices to get functional. On the opposite side, to interact with any existing TV through hand gestures, Ionescu et al. [16] propose a set that uses an infrared blaster to emit the commands typical of a physical remote, making a bridge between the 3D camera and the TV. The system reconstructs the user's hand to recognize a gesture from the viewer. The TV received commands such as turn on, turn off, volume up and down, and mute, being all based on user-defined gestures. This kind of approach provides freedom to users for choosing their gesture and interacting with the TV with their bare hands.

Concerned about the gestures choice acceptance and use, Wu and Wang [17] did a study on eliciting intuitive gestures from the users to interact with TV-based applications in a living room environment. After analyzing the collected gestures, the authors proposed a taxonomy of body gestures that includes physical and interactive characteristics. Among their findings, the authors pointed out that subjects preferred simple, easy-to-perform gestures, once gestures require the involvement of more muscle groups and longer movement than a remote control-based interaction.

C. 3D Menus

The concept of 3D menus has been applied in the game scenario in a very long time ago, but they usually consist in exposing 2D information and interaction in a 3D scene. Even this basic concept was never applied in scale in scenarios such as television menus, DVD and Blu-ray player menus and similar home devices.

There is yet no established set of interaction techniques for 3D menus, especially for the research areas of Augmented Reality and Virtual Reality. For this reason, developing 3D applications is still challenging, often leading to the conclusion that established techniques primarily developed for 2D interaction are not suitable. This challenge is magnified by the increasing use of freehand gestural input, which represents a different paradigm compared to traditional desktop (mouse and keyboard) interaction. Dachselt and Hubner [18] provide a survey of graphical 3D menu solutions for these areas and provide a complete taxonomy that is available on a website and may help the choice of appropriate widgets and techniques according to selected criteria.

Despite the use of bare hand techniques that enhance the immersion, users interaction with 3D menus in VR environments can be tracked and recognized by other means such as data gloves. It is also possible to use peripheral devices hand held by the user. Even considering the time to setup and discomfort of such setups, users can get amazed during experiencing a 3D environment once the illusion of immersion could be satisfactory. In this kind of environment, the management of the spatial information is critical, and the ergonomics of interaction must be taken into account to provide a convenient experience [19]. Regardless of using peripheral devices or bare-hand tracking techniques, the design of interactions that require a high accuracy single action could be not appropriated for gestural selection. Therefore, separating the interaction into several connected low demand operations could be more suitable [20].

Providing a haptic feedback can be a way to improve the user experience in interacting with 3D menus, bringing more accuracy to touching feeling, for example. Feedback in a general manner makes users feel more comfortable during the experience and makes the interaction more intuitive, quicker and precise. Essert-Villard and Capobianco [21] proposed a technique of haptic guidance for virtual environments. Called HardBorders, the technique haptically simulates the collision of a pointer with the borders of a polyhedral-shaped menu, making it glide towards the items on the menu.

III. Core Techniques

Depth perception is one of the key elements to create immersive environments on computer systems. There are many types of information that the brain interprets as depth cues to estimate the depth of a scene or the distance between objects. In this session, we discuss these depth cues and their relevance to the interaction techniques explored in our work.

A. Depth Cues

Depth Cue Theory is the main theory of depth perception. It states that different sources of information, or depth cues, are combined to give a viewer the 3D layout of a scene [22].

Alternatively, the Ecological Theory takes a generalized approach to depth perception. It states that the human visual system relies on more than the image on the retina; it requires an examination of the entire state of the viewer and her surroundings [23].

1) Pictorial Depth Cues: Pictorial depth cues are a 2D source of information that is interpreted to create the sense of depth. Because the pictorial depth cues are 2D information, they can present ambiguity. Many optical illusions are based on this type of ambiguity. Nevertheless, once pictorial depth cues are combined with other depth cues, they can present good results for the human depth perception. Some examples of pictorial depth cues are occlusion, relative size, shadow and foreshortening and linear perspective.

2) Oculomotor Depth Cues: Oculomotor depth cues include convergence and accommodation. Convergence defines the motion of both human eyes to converge lines of sight to the same point in space. Accommodation is the adjustment of the eye lenses to focus the acquired image at a distance of interest. These cues are difficult to be reproduced by computational solutions because the point in space which renders the visual result is located on a display (eg.: screens, projections or lenses) at a fixed distance from the user. This way, for example, if the user wants to perceive the rendered image clearly, her focus distance will be biologically defined as the distance between her eyes and the screen.

3) Binocular Depth Perception: Binocular depth perception (or Stereoscopy) describes the processing of the two images obtained by each of the user's eyes (from different angles). These images are combined to compute the depth of the picture points. This is the depth cue tackled by 3D TVs, usually requiring the user to wear glasses to filter a different image for each eye, then reproducing the stereoscopy conditions.

4) Depth from Motion: Motion cues provide depth information about the location, velocity, acceleration and direction of movement of both the viewer or an object [24]. Motion parallax is a depth cue based on the difference between the relative speed of the observed objects given that the user viewpoint is moving. The speed of the object is greater when the object is closer to the viewer and is slower when the object is far. This effect is applicable to computational solutions once the system tracks the user viewpoint and can modify the virtual camera.

This is the main depth cue used on our system. Despite relying on the viewpoint motion for an increased depth perception, the concept of moving and distorting the virtual camera based on the current viewpoint position is also applied when the user stands still. [9] shows a comparison of a 3D virtual environment experienced with and without the effect, as illustrated in figure 1. This effect provides an additional bonus of maintaining consistency of the coupling between virtual and real worlds. As shown in figure 1, while the effect is activated, if the user looks at the 3D cube from the left side, she will see the left side of the cube, if she looks from the right, she will see its right side. Such consistency can not be achieved by other depth cues, such as stereoscopy.

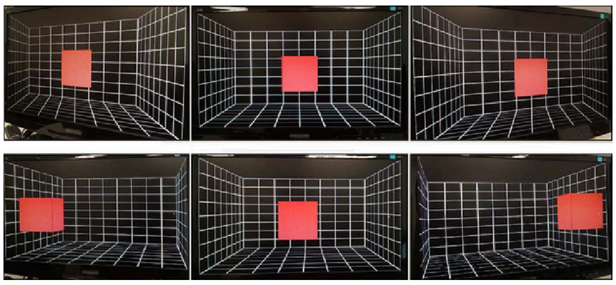

Figure 1. Comparison by [9] of a 3D environment visualized with and without the Motion Parallax effect. The first line shows the visualization without the Motion Parallax effect and the second line shows the visualization with the Motion Parallax effect.

For instance, considering that stereoscopy is being explored on 3D TVs, the 3D content does not change its perceived position regardless of where the user viewpoint is placed. This is why a person on the last row of a 3D movie on a cinema perceives the rendered objects as near to her as another person on the first row. Although this effect may benefit the cinema experience; it does not attain the goal previously mentioned, it is not possible to simulate a consistent coupling between virtual and real content.

As downside, to provide this effect, the output image must be rendered specifically considering the user viewpoint. This way, the effect will not favor a second user unless the display is capable of rendering a second image for that user. For instance, some 3D TVs provide such feature, allowing two different users to see completely different images on the same screen at the same time by wearing glasses [25]. In addition to the Motion Parallax, in our solution, we also make use of some pictorial cues such as lights, shadows, and fog. Our solution is also easily extensible to provide the stereoscopy effect on 3D TVs; the extension requirements are limited to render a second image from a slightly different viewpoint for the user's other eye. A comparison between the Depth cues effectiveness as a function of distance to the screen is shown in figure 2.

B. Tracking and Interaction

We use the Kinect V2 sensor and embedded tracking technology to track the positions of user head and hands. Kinect API provides good performance for real-time head and hand position tracking for up to six bodies. This suffices our tracking needs, which allows the system to combine the 3D tracked information with the 3D virtual world coordinate system. Additionally, using these positions, we can explore other features such as velocity, acceleration, angles, and direction. Beyond that information, we can

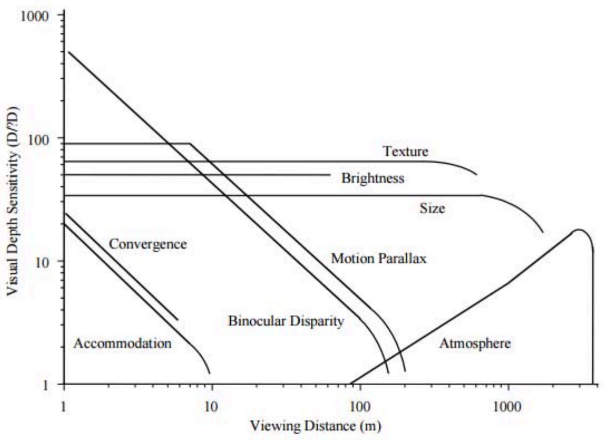

Figure 2. Depth cues effectiveness as a function of distance presented by Pfautz [24].

trigger some useful gestures for interaction, eg.: the "Back Gesture" as we discuss later.

Based on the concept of Natural Interaction [26] that states that *the system should embed the task of understanding the user intentions* while interacting (and not the opposite way around), our proposed concept introduces a direct selection interaction. This concept allows the user to interact with the virtual world by pointing and touching the virtual content using hand gestures. We aim to diminish the abstraction level of interaction with the virtual content by reusing the user knowledge about how she physically interacts with the real world. Also, we use different metaphors such as magnetic attraction showing the possibilities of empowering the user over the virtual content (e.g. telekinesis).

By pointing to a desired virtual object (which is a menu element), the object is magnetically attracted to the user hand tip. As closer as the pointing direction gets to the object, the closer the virtual object will be placed to the user. The object will stop at a distance of 0.1 meters to the user's hand tip. This mechanic allows the user to rapidly pinpoint an item of interest and make it appear within reach of her touch. Once the object is placed near the user, she is capable of moving her hand and touch it to perform the selection.

IV. Mixed Reality TV Concept

A. Overview

Figure 3 summarizes the main concept of our solution, displaying the interaction from the side on the top part and the Home Menu on the bottom part. The concept requires the positions of the user body to be described in the same coordinate system like the one used in the virtual environment. It also requires the virtual world to be rendered through a virtual camera that is placed exactly on the user viewpoint and has its frustum cutting planes distorted to match the screen boundaries. By satisfying such requirements, it is possible to deliver the Motion Parallax effect and place the 3D virtual objects in the real world regarding the user visual perception.

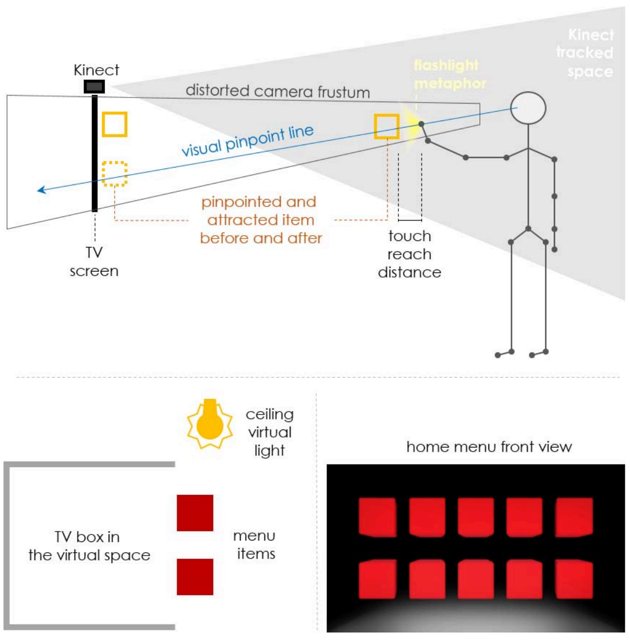

Figure 3. On the top part the *visual pinpoint* interaction concept. This interaction becomes possible only through the coupling of real and virtual environments in real-time. On the bottom part the home screen concept, with selectable items placed in a virtual space, with one spotlight on the top to resemble the behavior of the ceiling light on user living room, also adding sense of depth to the scene.

B. Visual Pinpoint

This setup also enables the system to recognize if the user *pinpoints* and *touches* an item. The *pinpoint* action only becomes effective because both the pinpoint line and the virtual camera have the same origin point (user viewpoint), and are directed to the same canvas (the screen). If either of these assumptions is wrong, the technique does not work properly, not respecting the goal that the display part that is pinpointed by the user is the one that is currently occluded by her hand tip. In our concept of the menu for Smart TVs, we used the *pinpoint* action to attract the item close to user's hand at the minimum of a touch-reach distance, and then allow the user to touch that item. Nevertheless, the same interaction technique could be used in different ways, such as suggesting a destiny point for navigation in the virtual world, and casting a spell or shooting an enemy in a game.

C. Flashlight

We also placed a directional light at the user's hand tip, always directed towards the user eye of sight (shown in figure 3). This light simulates the use of a flashlight by the user, which only illuminates the virtual world. By tackling this metaphor, we reuse the user knowledge about how light behaves in the real world, and use it to provide feedback about which object the user is pinpointing at the moment by highlighting it.

D. Stationary Use

The application of the Motion Parallax effect is acknowledgedly favored by the movement of the user viewpoint, providing an effective depth cue for these cases. Nevertheless, *our technique does not require the user to move around to interact* with the virtual content. We perform the necessary displacements and distortions (as described in figure 4) with the main goal of coupling the virtual objects position, orientation and scale with the user viewpoint and hand tips position. The user is free to both move her viewpoint or use our system in a stationary stance. In both cases, the interaction will work as expected. Therefore, it is suitable for the use on TVs, the user can stay on the couch, while the virtual objects come within her touch reach while using the pinpoint interaction.

V. Implementation

We developed our solution on top of the Unity engine [27], which allows easy prototyping and debugging of 3D virtual environments. Unity is also multi-platform, which primarily allows the developed solutions to be deployed to Windows, Mac OSx, Android, and others. Therefore, the part of our solution that delivers the Motion Parallax effect is also multi-platform. Since we used the Kinect V2 for tracking user viewpoint and hand tips, this part of the solution is exclusive for Windows. Nevertheless, once a tracking algorithm is designed for this task and works on other platforms through Unity, our solution can be used along as well.

A. Motion Parallax

To implement the defined concept, we applied the Motion Parallax effect to our context. On our solution we consider that a previous virtual camera is being used in the 3D application. By making this assumption, we can generalize our solution and make it work for any 3D application which allows the change of the virtual camera parameters. Figure 4 specifies our most important steps to implement the effect as described below:

A first the user's eye position must be determined on the screen coordinate system;
B then a translation is calculated to move the virtual camera position to the user's eye position;
C the virtual camera is moved;
D the frustum boundaries are translated to the opposite direction of the translation vector to be attached back to the screen boundaries.

1) Calculation of Television Dimension: This process begins with the calculation of the Television dimension in meters. For that, the program needs the television aspect ratio and dimension in inches as a user's input.

$$RD_m = RD_i * 0.0254 \tag{1}$$

$$RS_w = RD_m * Sin(Atan(ar)) \tag{2}$$

$$RS_h = RD_m * Cos(Atan(ar)) \tag{3}$$

Where 0.0254 is the conversion factor from inch to meter, RD_i is the real screen size in inches, RD_m is RD_i converted to meters, ar is the aspect ratio of the real screen, RS_w is the real screen width in meters and RS_h is the real screen height in meters.

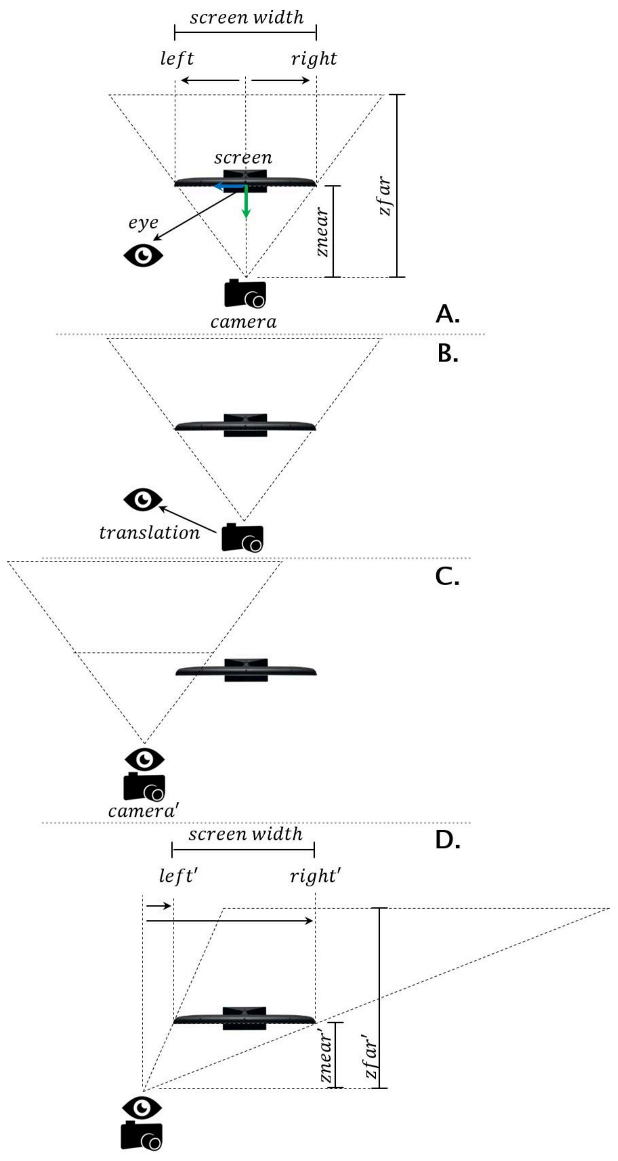

Figure 4. Fishtank VR technique explained in four steps.

2) Tracking: In this step the user's head position is obtained using a Kinect V2 sensor placed on top of the TV, centering its camera to the TV center, and also adjusting the orientation of its Z axis to be perpendicular to the plane of the TV screen. This position represents the user's viewpoint, and from this point, we will refer to it as the user's eye. This position should also be converted to meters. By guaranteeing that all positions are described in the same unit system, we can correctly place the virtual objects and specify their measures in the real world.

3) Calibration: The user's tracked skeleton is incremented with the value of $RS_h/2$. This adjusts the simulation of user coexistence with the virtual world, by calibrating the Kinect coordinate system, which has the $(0,0,0)$ point in its center with the TV coordinate system. In our setup, we consider that the TV center is the center of the virtual world, and therefore once the Kinect tracking results are adjusted as described, the user position is one step further to be correctly placed in that world.

4) Motion Parallax Calculation: The next step is to couple the virtual camera rendering plane with the TV screen. This step is necessary because we must guarantee that the rendering plane of the virtual camera (usually the *near plane*) is perfectly mapped to the TV screen. In this case, our strategy is to perform all distortions in a middle term coordinate system that we call *screen space coordinate system*. This space is related to the screen width on both virtual (camera *near plane* width) and real (TV screen width) environments.

This way, by simulating that the Kinect is positioned on the screen center as previously described, we then divide the value of the eye position by S_w, so that the eyes position is transformed from the *real world coordinate system* to the *screen space coordinate system.*

$$RE_{wc} = E_{tracked} + (RS_h/2) \tag{4}$$

$$RE_{sc} = RE_{wc}/RS_w \tag{5}$$

Where $E_{tracked}$ is the user's eye tracked position, RE_{wc} is the user's real eye position on the world coordinate system with the Kinect already centered on the screen and RE_{sc} is the user's real eye position on the screen space coordinate system.

Now the same process is repeated, but this time for the virtual environment. The virtual camera position and its frustum parameters are obtained from the virtual camera object. That way, the virtual screen dimensions are obtained through the subtraction of the right minus left and top minus bottom parameters from the camera frustum. After that, the virtual camera position is divided by its width, obtaining the camera relative position coordinate on the screen space coordinate system.

$$VS_w = F_r - F_l \tag{6}$$

$$VS_h = F_t - F_b \tag{7}$$

$$VE_{wc}.x = (F_r + F_l)/2 \tag{8}$$

$$VE_{wc}.y = (F_t + F_b)/2 \tag{9}$$

$$VE_{wc}.z = F_n \tag{10}$$

$$VE_{sc} = VE_{wc}/VS_w \tag{11}$$

Where VS_w is the virtual screen width, VS_h is the virtual screen height, F is the virtual camera frustum, F_r is the frustum right coordinate, F_l is the frustum left coordinate, F_t is the frustum top coordinate, F_b is the frustum bottom coordinate, F_n is the frustum near coordinate, VE_{wc} is the virtual eye position in the virtual world coordinate system and VE_{sc} is the virtual eye position relative to the virtual screen.

This configuration can be seen in figure 4 part A. Now it is time to calculate the vector that will represent the translation of the virtual camera to the user's eye position. This vector is obtained by the subtraction of the transformed user's eye position minus the transformed virtual camera position. Then we apply this transformation vector to the virtual camera, as shown in the figure 4 part B and C.

$$T_{vector} = RE_{sc}/VE_{sc} \tag{12}$$

Where T_{vector} is the translation vector.

After applying the translation, it is time to create a frustum distortion so the *near plane* is translated back to the screen and the user can see the same scene window, but from a different point of view. This distortion is obtained by applying the inverted translation to the frustum parameters, and the result can be seen on the figure 4 part D.

$$DF = F - T_{vector} \tag{13}$$

Where DF is the distorted frustum.

This process should be executed for each rendered frame. Two camera objects are used during the process; the first will keep the camera default parameters and the second will be the one that will suffer the distortions and used to render the desired effect for the user. This second camera is reset to the first camera values at the beginning of the process on every frame.

Any camera modification applied during the application runtime that has no relation to the Fishtank distortion process, e.g., walking translations or rotations must be done on the first camera because the second camera will always use the first camera parameters and transformations to create the distortion. If the modification is applied to the second camera, the obtained results can have some inconsistency in the Fishtank effect or the modification results. This way we can attach our solution to any preexisting 3D environment coded using the Unity engine, and respect the previously defined navigation and camera control interactions while adding the Motion Parallax on top of it.

B. Gestural Interaction

To use the tracking information provided by the Kinect API, we implemented a set of gestural interactions which are described as follows.

1) Visual Pinpoint Attraction: The first step of the interaction is to extract the user hand tip position from the Kinect skeleton. After this, the position value is converted to the distorted camera viewport coordinates. This allows the system to describe the user hand tip position concerning the projection plane of the modified camera. Then the same conversion is applied to the 3D virtual objects that are selectable. This conversion is important because it guarantees that the objects that aren't in the same Z-axis on the world coordinate system but are on the same Z-axis based on the user viewpoint will be attracted. It happens because they will have close X and Y coordinates on the viewports. Then the application calculates this distance between these converted positions. If this value is lower than the minimal interaction distance (e.g. 10 centimeters), the application calculates a T value that varies between 0 and 1. This T value indicates how close the virtual object will be translated (attracted) from its initial position (T = 0) to the nearest point it can get to the user's head (T = 1), according to the equation 14.

$$T_p = (100 - (D * 100/M_i))/100 \tag{14}$$

Where T_p is the T parameter, D is the distance between the user's finger and the cube that represent the program and M_i is the minimal interaction distance.

If the user starts to put hear hand tip farther from the virtual object, the T parameter will start to fall, and the object will slowly come back to its original position. This process should be performed for all virtual objects shown on the screen and for both user hand tips.

2) Back Gesture: To allow the user to return to the Home Menu after a particular menu item is selected, we added a "Back Gesture" to our solution. The "Back Gesture" was added as a specific gesture which can be performed in any place of the living room visible for the Kinect sensor and is not directly connected to the coupled virtual-real environment.

We assigned the "Back Gesture" to be a *swipe right* gesture. It is triggered based on a comparison of the velocity of the user's hand to determine the direction and intensity of the movement. If the intensity reaches the value of $1.5 meters/second$, we analyze its direction. This value was selected because it is slow enough for the user to perform without harming herself, and fast enough to distinguish from other small gestures that would be potentially triggered otherwise. Once we check that the direction points towards the user left side, we accept the performed gesture.

C. Home Screen Interface

As a first core case of the proposed technology we developed a Home Screen application for Smart TVs. In our home screen, all the applications are presented as cubes in a 3D virtual world. This application gives control to the closest user body tracked by the Kinect API. A screenshot of the home screen application can be seen on the figure 5.

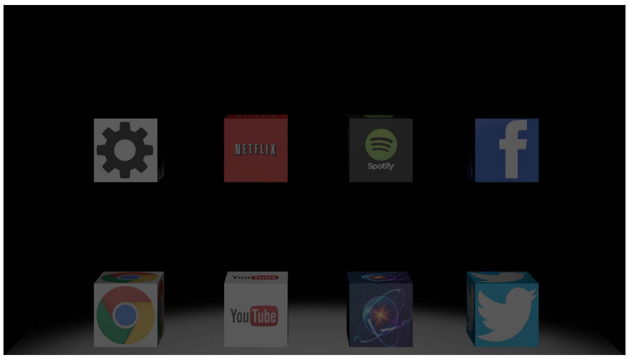

Figure 5. Screenshot of the Home Menu interface for Smart TVs.

To select an application (cube) on the home screen the user just has to point directly to it in a way that the chosen application is partially occluded by the user's finger. When the user is pointing to the screen, the cubes are attracted to his head. The Home Screen Menu is designedly slightly dark to induce the user to highlight it using the flashlight metaphor. The menu also shows the previously mentioned fog simulation, which becomes denser for objects far away from the user.

VI. Results

The main results of the implemented interaction techniques and interfaces are described in the following subsections.

A. Calibration

Before testing our solution to interact with the Home Menu and the sample application (mini-game), we developed a calibration scenario to test if the intended effect was working properly. Our calibration experiment consisted of displaying a virtual environment, and render the avatar of the tracked user skeleton (part A of figure 6). If the effect is successfully working, the user would not be able to see her skeleton because it would be occluded by her body.

The results of the calibration explained in section V-A were successful, showing that the rendered skeleton was not visible from the user viewpoint. In other words, the rendered skeleton was fully occluded by the user's real body as shown in the part B of the figure 6.

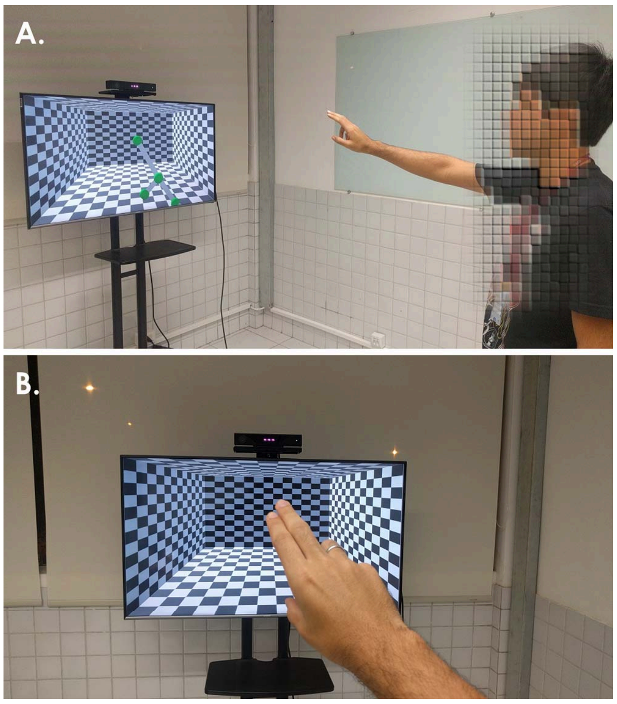

Figure 6. Fully occluded skeleton as a result of the calibration between the user's tracked body and the virtual world. On part A we show the TV from a diagonal viewpoint, which reveals part of the user avatar, showing green spheres for the joints of the user's hand, thumb and hand tip. On part B we show a take from the user viewpoint, which precisely occludes the same avatar, meaning the effect is properly applied.

B. Interaction

The results of the *visual pinpoint* can be seen on the figure 7. Part A of the figure shows the interaction being used with the user positioned on the right side of the television, while part B shows the interaction being used with the user positioned directly in front of the television.

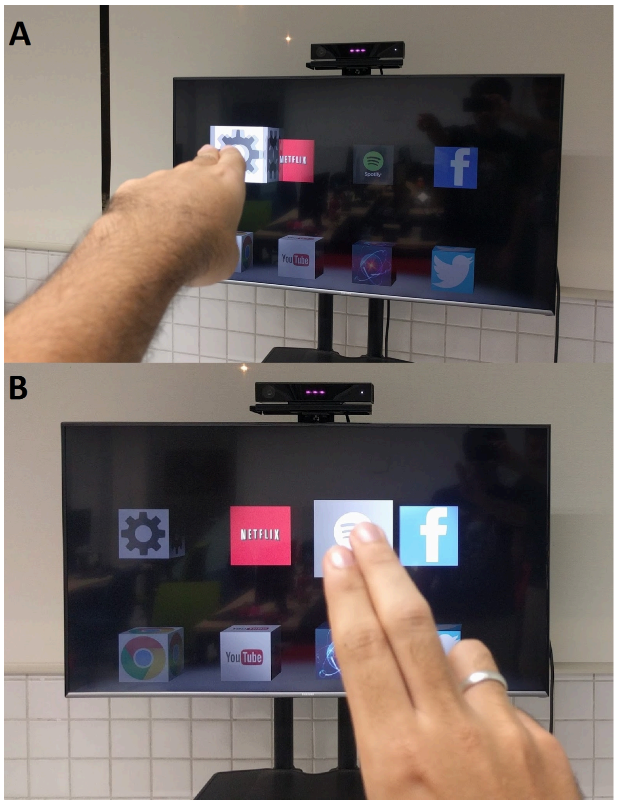

Figure 7. Point and touch interaction with the addition of the motion parallax effect shown by two different points of view. On part A of the image the user is displaced to the right of the television and on part B the user is in front of the television.

Figure 7 also shows the flashlight metaphor being used, highlighting the item of interest by applying a directional light towards its position. While the other objects remain with the original illumination conditions, the region to which the user is pinpointing becomes brighter. At last, the figure also shows the magnetic attraction property, simulating the user has telekinesis powers and can pull virtual objects while pointing to them. This metaphor is especially interesting because all menu objects can remain on the screen and reach the user's touch once the user intends to interact with them. Figure 7 depicts the moment which the pinpointed objects are being attracted to the user's reach. The result of the *virtual touch* interaction can be observed in figure 8.

C. Sample Application: Blocking Blocks Mini-game

A mini-game was implemented to show a potential application for a Smart TV, providing an extra entertainment option while demonstrating the interactive capabilities of our solution setup. This mini-game is based on classic

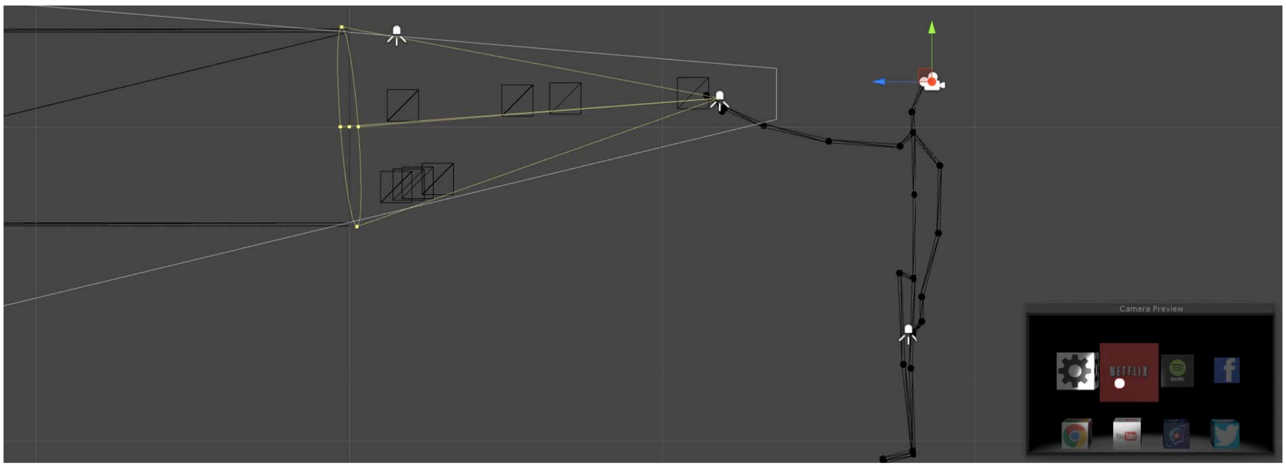

Figure 8. Screenshot of the virtual environment being debugged. This scene shows: the TV menu box as the larger rectangle on the left; the menu items, as squares spaced between the user and the screen; the user skeleton, and the user hand tip intersecting (touching) the nearest menu items; the camera is placed at the user viewpoint, and its distorted frustum is shown by light gray lines; the flashlight illuminated region by yellow lights; on the bottom right part a preview of the rendered scene is shown.

Pong game. In our concept, the player has to block arriving cubes coming from television and passing through the player. The player loses points for each cube that passes and sums points each time a cube is successfully blocked. Blocked cubes return to the screen.

The scene can be described as a room with walls, floor, and ceiling coinciding with the television screen limits as shown on figure 9. The walls present textures with stripes, which helps the player to identify the depth of arriving cubes positions through it visually are shadows on the walls, floor, and ceiling. The cubes are thrown from the back of the screen, away from the player, but with a speed that is slow enough to make the user see the cube and reposition the hand to block it while it is arriving. This game is independent of hand preferences, working with both user's hands.

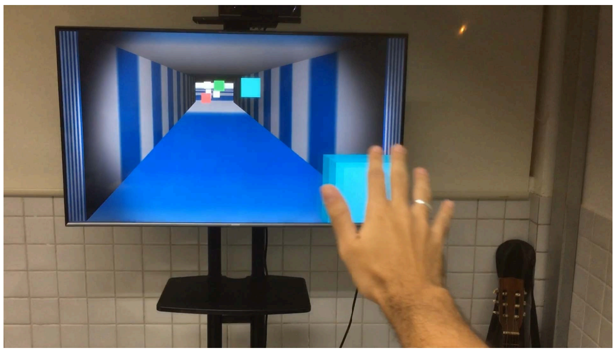

Figure 9. User playing the Blocking Blocks mini-game.

VII. Discussion

A. Additional Uses

The new interaction environment proposed on this work brings new possibilities for interface design and usability. As an example, we can use each face of the application cube to show different pieces of information about the same application. For example, on the Netflix cube, each face can show a specific movie or a film series episode of interest for the user, and the user can select the face and go directly to the desired content by touching the virtual face. This interaction can significantly reduce the number of steps for the same task, skipping the needing of opening the application, visually searching for the content, navigating to it and finally opening it.

Several metaphors could be added. Applications could be rearranged through pushes and pull from one side to the other, dismissed with a fast push to the side, and excluded with a clap. The possibilities are strongly tied to how we interact with solid objects in front of us in the real world and are still to be explored. Diverse games could also exploit the parallax effect combined with gestures. One application example is a game where the user plays as a sniper and uses the motion parallax effect to look at different sides of a building through its window to find an enemy and use gestures to shoot them.

One of the key limitations regarding the application of the motion parallax effect through the Fishtank VR technique is that it distorts the screen image based on a single user viewpoint. This way, if there are no extra options that make a single TV render a different image for a second user (such as Dual Play from LG 3D TVs), the interaction will only work for the nearest user.

B. Limitations

There was also some perceived latency, however with no impact on the interaction. On preliminary tests users successfully selected the desired menu item without effort. We also notice some jitter on the tracking of the user hand tip, which we used to detect the touch (select) gesture. The tracking result provided by the Kinect SDK uses as input the depth frame given by the Kinect (RGBD) sensor, and therefore the noise of the depth data can directly compromise the tracking result. We understand that once the hand tip presents small versatile shapes if compared to other body parts, the jitter of this tracked joint is also a common issue. Nevertheless, once we set up the size of the virtual objects and the space between them to be 10 centimeters or more, no selection errors were detected.

Since the focus of our proposal wasn't to create a body tracking system, the sample application has a limitation the use of the Kinect sensor and needs a computer or an Xbox One to run. Nevertheless, the proposed solution was implemented on Unity Engine, and once it's community releases a body tracking solution which do not require the use of a Kinect, the application can be exported to any of the supported platform, including televisions.

VIII. Conclusion

In this paper, we present a technique for coupling the camera displacement and distortion from the Motion Parallax effect on VR systems with the tracking data of user's head and hand tips obtained from a Microsoft Kinect V2 sensor. This coupling can transform a consumer TV into a *Mixed Reality display* that enables the user to *pinpoint* and *touch* 3D virtual objects, with her bare hands. We apply this technique on a living room environment, providing a 3D menu system for Smart TVs. Our solution adds enhanced depth perception and gesture control for this context. Also, we show some possibilities to use interactions based on *physical metaphors* found in the real world, such as magnetic attraction, illumination, and fog. At last, we introduce a mini-game as sample application showing additional interaction and entertainment options using the interaction techniques previously discussed.

This work presents a ground base to explore the provided interaction technique and therefore leaves space for a set of future works. At first, we want to expand the set of interaction options given that the TV is a Mixed Reality display. For instance, the "back gesture" can also be part of the physical simulation of the 3D environment, allowing the user to use a metaphor such as "slap to dismiss," meaning the user can hit the virtual object to throw it away. Another possible interaction is to "push" the current content forward to get back to the home screen for example. A set of different animations and simulations can also be applied to enhance the 3D menu. For example, the selection of an item could trigger the opening of its box, the interaction with two hands or a grasping gesture could make it bigger or rearrange its position, lifting a box could place it in a favorites bar, and so on. At last, we plan to perform an extensive user experiment to compare these interaction options between each other, as well as with another established interaction techniques such as the control of the application cursor through hand gestures.

Acknowledgment

The authors wish to acknowledge that this work was supported in part by FACEPE and CNPq.

References

[1] M. Bellis. (2016) The invention of television: Timeline. URL = https://www.thoughtco.com/the-invention-of-television-1992531.

[2] S. T. S. Portal. (2016) Number of tv households worldwide from 2010 to 2021 (in billions). URL = https://www.statista.com/statistics/268695/number-of-tv-households-worldwide/.

[3] M. Hoelzel. (2014) Smart tvs are on pace to take over the entire tv market. URL = http://www.businessinsider.com/smart-tvs-are-on-pace-to-take-over-the-entire-tv-market-2014-7.

[4] LG. (2017) Magic remote control with voice mate for select 2014 smart tvs. URL = http://www.lg.com/us/tv-audio-video-accessories/lg-AN-MR500-magic-remote.

[5] Samsung. (2016) Samsung smart tv: Tv gesture book. URL = https://goo.gl/IAT9Zh.

[6] C. Ware, K. Arthur, and K. S. Booth, "Fish tank virtual reality," in *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*. ACM, 1993, pp. 37–42.

[7] S. Ide, H. Yamanoue, M. Okui, F. Okano, M. Bitou, and N. Terashima, "Parallax distribution for ease of viewing in stereoscopic hdtv," in *Electronic Imaging 2002*. International Society for Optics and Photonics, 2002, pp. 38–45.

[8] C. Fehn, "Depth-image-based rendering (dibr), compression, and transmission for a new approach on 3d-tv," in *Electronic Imaging 2004*. International Society for Optics and Photonics, 2004, pp. 93–104.

[9] L. S. Figueiredo, E. V. Neto, E. Arruda, J. M. Teixeira, and V. Teichrieb, "Fishtank everywhere: Improving viewing experience over 3d content," in *International Conference of Design, User Experience, and Usability*. Springer, 2014, pp. 560–571.

[10] C. Heinrichs and A. McPherson, "Recreating the parallax effect associated with fishtank vr in a real-time telepresence system using head-tracking and a robotic camera," in *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 283–284.

[11] C. Demiralp, C. D. Jackson, D. B. Karelitz, S. Zhang, and D. H. Laidlaw, "Cave and fishtank virtual-reality displays: A qualitative and quantitative comparison," *IEEE transactions on visualization and computer graphics*, vol. 12, no. 3, pp. 323–330, 2006.

[12] S.-H. Lee, M.-K. Sohn, D.-J. Kim, B. Kim, and H. Kim, "Smart tv interaction system using face and hand gesture recognition," in *Consumer Electronics (ICCE), 2013 IEEE International Conference on*. IEEE, 2013, pp. 173–174.

[13] K. Watanabe, Y. Miyake, N. Nakamichi, T. Yamada, and T. Ozeki, "Remote touch pointing for smart tv interaction," in *Consumer Electronics (GCCE), 2014 IEEE 3rd Global Conference on*. IEEE, 2014, pp. 232–235.

[14] H.-J. Kim, K.-H. Jeong, S.-K. Kim, and T.-D. Han, "Ambient wall: Smart wall display interface which can be controlled by simple gesture for smart home," in *SIGGRAPH Asia 2011 Sketches*. ACM, 2011, p. 1.

[15] A. Ferscha, S. Resmerita, C. Holzmann, and M. Reichör, "Orientation sensing for gesture-based interaction with smart artifacts," *Computer communications*, vol. 28, no. 13, pp. 1552–1563, 2005.

[16] D. Ionescu, B. Ionescu, C. Gadea, and S. Islam, "An intelligent gesture interface for controlling tv sets and set-top boxes," in *Applied Computational Intelligence and Informatics (SACI), 2011 6th IEEE International Symposium on*. IEEE, 2011, pp. 159–164.

[17] H. Wu and J. Wang, "User-defined body gestures for tv-based applications," in *Digital Home (ICDH), 2012 Fourth International Conference on*. IEEE, 2012, pp. 415–420.

[18] R. Dachselt and A. Hübner, "Three-dimensional menus: A survey and taxonomy," *Computers & Graphics*, vol. 31, no. 1, pp. 53–65, 2007.

[19] P. Lemoine, F. Vexo, and D. Thalmann, "Interaction techniques: 3d menus-based paradigm," in *AVIR 2003*, no. VRLAB-CONF-2007-030, 2003.

[20] G. Ren and E. O'Neill, "3d selection with freehand gesture," *Computers & Graphics*, vol. 37, no. 3, pp. 101–120, 2013.

[21] C. Essert-Villard and A. Capobianco, "Hardborders: a new haptic approach for selection tasks in 3d menus," in *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*. ACM, 2009, pp. 243–244.

[22] E. B. Goldstein and J. Brockmole, *Sensation and perception*. Cengage Learning, 2016.

[23] J. J. Gibson, *The ecological approach to visual perception: classic edition*. Psychology Press, 2014.

[24] J. D. Pfautz, "Depth perception in computer graphics," University of Cambridge, Computer Laboratory, Tech. Rep., 2002.

[25] T. Pendlebury. (2012) Lg's dual play gets serious with split-screen gaming. URL = https://www.cnet.com/news/lgs-dual-play-gets-serious-with-split-screen-gaming/.

[26] A. Valli, "Notes on natural interaction," *retrieved from on Jan*, vol. 5, no. 2012, p. 80, 2005.

[27] U. G. Engine, "Unity game engine-official site," *Online][Cited: October 9, 2008.] http://unity3d. com*, pp. 1534–4320.