Virtual Table–Teleporter: Image Processing and Rendering for Horizontal Stereoscopic Display

Bruno Eduardo Madeira
Instituto Militar de Engenharia - IME
Rio de Janeiro, Brazil
madeira@ime.eb.br

Luiz Velho
Laboratrio Visgraf
Instituto Nacional de Matemtica Pura e Aplicada
Rio de Janeiro, Brazil
lvelho@impa.br

***Abstract*—We describe a new architecture composed by software and hardware for displaying stereoscopic images over a horizontal surface. It works as a "Virtual Table and Teleporter", in the sense that virtual objects depicted over a table have the appearance of real objects. This system can be used for visualization and interaction. We propose two basic configurations: the Virtual Table, consisting of a single display surface, and the Virtual Teleporter, consisting of a pair of tables for image capture and display. The Virtual Table displays either 3D computer generated images or previously captured stereoscopic video and can be used for interactive applications. The Virtual Teleporter captures and transmits stereoscopic video from one table to the other and can be used for telepresence applications. In both configurations the images are properly deformed and displayed for horizontal 3D stereo. In the Virtual Teleporter, two cameras are pointed to the first table, capturing a stereoscopic image pair. These images are shown on the second table, that is in fact a stereoscopic display positioned horizontally. Many applications can benefit from this technology such as, virtual reality, games, teleconference and distance learning. We present some interactive applications, that we developed using this architecture.**

***Keywords*-3D Stereo; Projective Geometry; Telepresence;**

I. Introduction

The stereoscopic technology is getting more and more common nowadays, as a consequence this kind of technology is becoming cheaper and widely accessible to people in general, [1], [2].

Most stereoscopic applications use simple adaptations of non-stereoscopic concepts in order to give the observer the sense of depth. This is true, for example, in the case of 3D movies where two versions are usually released, one to be watched in a stereoscopic movie theater and other to be watched in a normal theater.

We are exploring the use of stereoscopic technology changing the usual paradigm that tries to give the observer the "Sense of Depth" to the new paradigm that gives the observer the "Sense of Reality". We call Sense of Reality when besides giving a sense of depth to the image, the setting is presented in such a way that it is compatible with real objects in the real world. Normal 3D movies do not implement the "Sense of Reality" because of the following reasons:

- The screen is limited, thus, points in the border can be shown without the stereo correspondence. It is not a problem if the whole scene is "inside" the screen, but it is a problem if the scene is over the screen.
- The objects presented in a movie are usually floating in space, because the scene is not grounded to the real world floor.
- Many scenes usually present a very large range of depth, which cannot be exhibited by the current stereoscopic technology.
- The zoom parameter of the camera is usually chosen in order to capture the scene in the same way as a regular movie, which in consequence magnifies portions of the scene.

The above aspects make it difficult for the observer to believe that the content, although presented in 3D, is actually real. To be physically plausible the content presented in the screen must make sense when viewed as part of the environment that surrounds it. This goal can be achieved by making four changes to the stereoscopic system:

- **Presenting the 3D stereo content on an horizontal support leveling the floor with the screen.**
  It establishes a link between virtual objects and the screen. This link makes the result more reliable compared to the exhibition of virtual objects flying in front of a vertical screen.
- **Not presenting a scene whose projected points in the border of the screen are closer to the observer than the screen.**
  If a 3D point on the left or right border of the screen is closer to the observer than the screen, then one of its correspondent stereoscopic projections will not be exhibited due to the screen limitation. That means that it will generate a stereoscopic pair that does not correspond to a 3D scene. If the stereoscopic projections of an object cross the top border, but do not cross the laterals, then the scene will not be well accepted by the observer either, although the stereoscopic pair corresponds to a 3D scene. In this case, the problem is that the border limitation corresponds to a 3D cut

in the object, that makes the top of the projection be perfectly aligned with the top border of the screen. Besides the fact that the 3D cut makes the scene odd, there is the fact that the alignment between the border and the cut implies that the observer had to be placed in a very specific position in order to be able to see it, it means that the stereoscopic projections are images that do not satisfies the generic-viewpoint assumption [3], that can cause interpretation problems. Finally, if the stereoscopic projections cross the bottom border, then they will suffer from the same problems as those that cross the top border, plus the fact that they will correspond to floating objects.

- **Constraining the scale of the scene based on some physical reference.**
  It can be achieved by changing the cinematography technique. For example, 3D stereo movies adopt the classic film language used for 2D films. As a consequence, it employs different framing techniques, such as close-ups, medium and long shots that cause the objects in a scene to change size relative to the screen. This practice impairs the sense of reality with the physical world. such problem is avoided by establishing a fixed scaled correspondence between the displayed scene and the real environment.
- **Restricting the field of view to encompass the objects in the scene.**
  In standard 3D stereo movies, the fact that the cameras are positioned parallel to the ground implies in a wide range of depth, including elements far from the center of interest of the scene. Conversely, in stereoscopic images produced for display over a table the camera will be oriented at an oblique angle in relation to the ground, which limits the maximum depth of the scene and favors the use of stereoscopic techniques.

Devices that uses horizontal stereo have already appeared in the patent literature, such as presented in [4], [5] and [6], and also has been explored by the computer graphics community, as can be viewed in [7] and [8]. The method that we use to generate synthetic stereoscopic pairs is similar to the one used in these works, and corresponds to the problem of generating the image on the floor of a CAVE [9], this problem will be explained in section II. Our work differs from them in the method used for exhibiting stereoscopic pairs captured by cameras. For example, in [7], they reconstruct a 3D model of the object from its silhouette, and then they use it to render the stereoscopic pair. The problem with this technique is the low quality of the result. In our system, we solve the problem by using image processing, more precisely, we apply a homography, previously estimated by a Computer Vision process. The details about this process will be explained in section III.

The main contribution of this paper is to introduce a new architecture, composed by software and hardware, that works as a "Virtual Table – Teleporter". The system displays a 3D stereoscopic scene over a horizontal viewing table. It can also capture the stereoscopic appearance of a set of objects disposed over a table, and transmits this content in real-time to the stereoscopic viewing table. This kind of setting can be very useful in applications such as teleconference, allowing a group of people to share virtual representations of objects positioned over a table. The technology presented can also be used to capture and display (in a scaled down fashion) a theater play, a sports match (such as tennis, basketball, etc.) or any other event that takes place in a horizontal field.

II. Rendering horizontal stereoscopic images

There are two issues that should be taken into account when rendering stereoscopic pairs that will be presented as objects over horizontal displays:

- The cameras do not have to point to the object to be captured, instead, their view directions must be orthogonal to the planar surface that supports the object.
- The intrinsic parameters of each camera must be chosen in such a way that makes the projection plane be coincident to the horizontal display, and makes the view frustum encompass the object, although the camera is not pointed to it.

This camera setup makes the virtual object stand over the display for a user whose eyes are in the same position as the optical centers of the cameras, because it makes the rays emitted by the object and passing through each eye have the same color as the correspondent pixel in the horizontal display, thus each eye sees the same image whether it came from the real object or from the display. Figure1-a illustrates it.

The fact that the camera is not pointed to the object can be non intuitive. It happens because the view direction always represents the orthogonal direction to the projection plane, that usually is close to the object direction. It is so common that OpenGL standard uses the "look at" expression as part of the name of a function used for defining the view direction and the others extrinsic parameters of a camera. As a consequence, if one intends to use gluLookAt function for rendering virtual objects over horizontal displays, he or she must keep in mind that the function will miss the "look at" sense, since the camera will not point to the object.

Besides the view frustum skewing due to the non-coincidence of the view direction and object direction, presented in Figure 1-a, there is another skewing in the orthogonal direction, as presented in the Figure 1-b. This skewing occurs as a consequence of using the same rectangular region over the projection plane as border for the image captured by both cameras. This skewing is not exclusive for horizontal stereo, it exists whenever stereoscopic images have to be presented superimposed in the same screen.

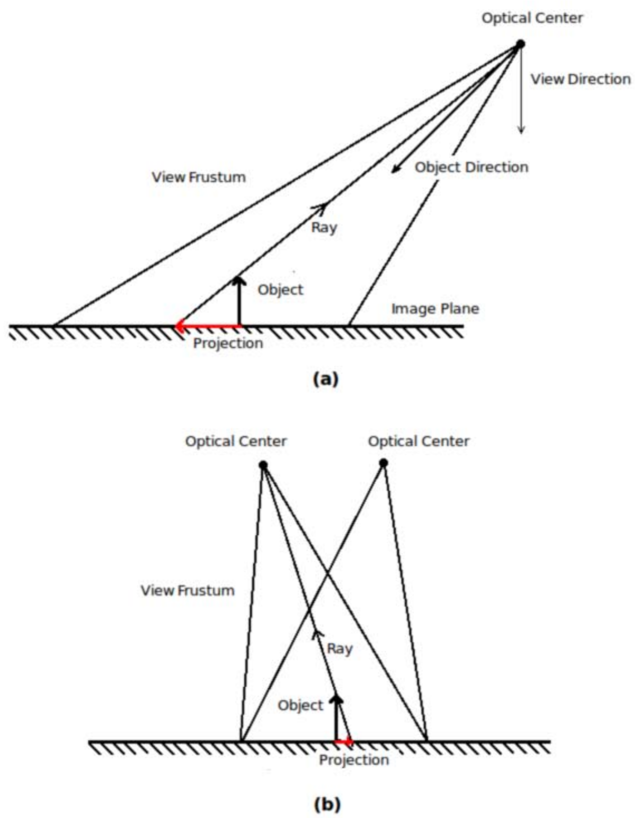


Figure 1. (a) Shows the lateral view of the frustum used for rendering an image to be presented horizontally. (b) Shows the frontal view of the setup.

III. Building horizontal stereoscopic images by deforming

Section II presented how to define camera models for rendering stereoscopic pairs prepared for being exhibited horizontally. This section explains how to build the stereoscopic pair by using real world cameras.

It would be possible to adapt the same approach presented in section II to the case of using real world cameras, Figure 2 illustrates it. We would point the cameras to the ground direction, making their view directions orthogonal to the planar support, and then, we would use a very large field of view in order to encompass the object to be captured. It would be necessary to enlarge the frustum because ordinary real world cameras do not have skewing control. The result would be an image that differs from the image to be presented over the display by a scale factor.

A problem with this approach is that most of pixels captured by the camera pair are far from the projection of the object, thus they would not be used. More precisely, just the portion containing the information to be exhibited by the horizontal display would be used. Another problem is that we would need a camera with a very large field of view.

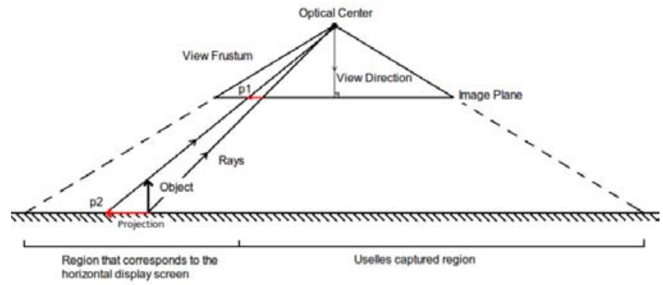


Figure 2. The projection p2 is a scaled version of the projection p1. We can notice that the camera must have a very large field of view, and most of pixels in the image plane are not used.

A better approach consists on pointing the cameras toward the object, and deforming the captured images in order to make them equal to images that would be captured by cameras defined such as in section II. Figure 3 illustrates it.

We will use Projective Geometry to show that this deform is a homography, and then, we will explain how it can be calculated by a well known Computer Vision process.

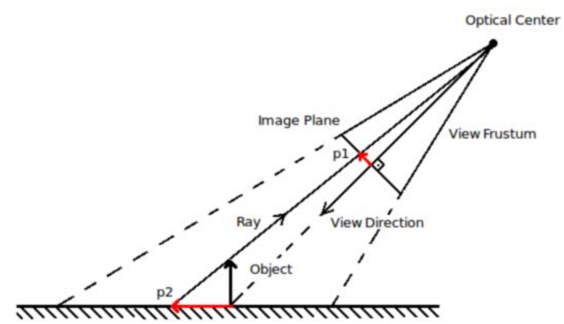


Figure 3. The projection p2 is related by an homography to the projection p1, captured by a camera whose view frustum is not skewed.

A. Projective Geometry Basics

In order to solve the problem at hand we need some mathematical notions from Projective Geometry. We list these concepts here [10]:

***Definition** 1:* The Projective Plane $\mathbb{P}^2$ is the set of lines in $\mathbb{R}^3$ that passes through the origin, excluding the origin.

***Definition** 2:* We call homogeneous coordinates, representatives of the set of lines in $\mathbb{R}^3$ that passes through the origin, excluding the origin.

***Definition** 3:* A homography is an invertible mapping h from $\mathbb{P}^2$ to itself such that three points $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ lie on the same line if and only if $h(\mathbf{x}_1)$, $h(\mathbf{x}_2)$ and $h(\mathbf{x}_3)$ do.

***Theorem** 1:* A mapping $h : \mathbb{P}^2 \rightarrow \mathbb{P}^2$ is a homography if and only if there exists a non-singular 3×3 matrix H such that for any point in $\mathbb{P}^2$ represented by a vector $\mathbf{x}$ it is true that $h(\mathbf{x}) = Ht\mathbf{x}$.

B. The solution by using homographies

It is easy to notice, by examining Figure 4, that if a set of points in a scene is projected by a camera over a set of collinear projections, then they keep collinear if we maintain the optical center in the same place and change the position of the projection plane. It happens because the rays whose intersection generate these projections must be coplanar, and if the optical center is unchanged they still have to be used for defining the projections over the plane in the new position. Since the rays are coplanar, the intersection of them with any plane must be collinear.

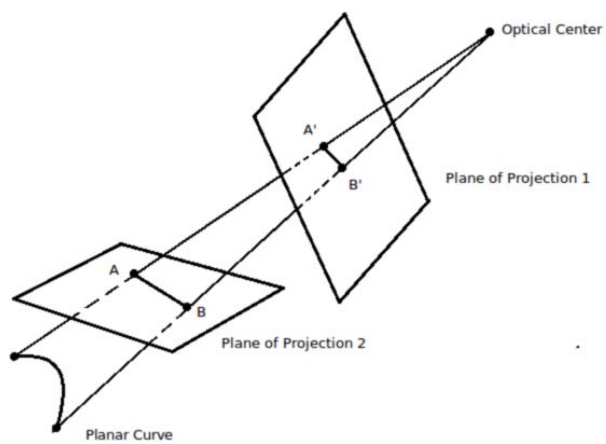


Figure 4. This example shows a curve whose projection over a projection plane is collinear. As a consequence, it is also collinear if we change the projection plane and keep the optical center unchanged.

This result implies that there is an homography relating the coordinates of projections, measured over the images captured by the cameras pointed to the object to be captured, and the coordinates of the projections, made by using the same optical center as center of projection and using the planar support as projection plane. This fact explains why the projections p_1 and p_2, presented in Figure 3, are related by a homography.

From the Theorem 1 follows that a homography can be represented by a 3×3 matrix that acts as a projective mapping over $\mathbb{P}^2$ to itself, and from the Fundamental Theorem of Projective Geometry follows that these mappings are completely defined by a set of four correspondences between elements in the domain and in the range. So, if we establish the correspondence between the coordinates of four known markers over the planar support and their respective coordinates over the images captured by the camera pair, then the homographies will be defined.

Since the coordinates over the planar support are being measured in spatial units, such as centimeters, the homographies cannot be used for finding the deformed images directly, because they are measured in terms of pixels. This problem can be easily solved by rescaling the deformed image by the the pixels per unit of length relation, that represents how many pixels of the horizontal display are in each unit of length used for defining the markers coordinates.

IV. Homography estimation

The homography estimation from correspondences is a very well known problem in Computer Vision, and is usually solved by using many more than four correspondences, since the measures are in general corrupted by noise, and doing this we can improve the accuracy of the estimation.

The set of corresponding points can be defined with the help of a checkerboard, whose square corners can easily be detected by image processing, allowing the correspondence process be automatic. In this case, we use a coordinate system over the checkerboard for defining the position of each corner.

We describe here the solution presented in [10] for estimating homographies from correspondence of points.

A. The Direct Linear Transformation Algorithm

Suppose that we are given a set of at least four correspondences $x_i \leftrightarrow x'_i$. We need to estimate the homography H such that

$$H\mathbf{x}_i = \mathbf{x}'_i$$

for every pair.

Since $\mathbf{x}$ and $\mathbf{x}'$ are given in homogeneous coordinates, then

$$H\mathbf{x}_i = k\mathbf{x}_i$$

for some non negative number k.

This fact can be rewritten in the following way:

$$\mathbf{x}'_i \times H\mathbf{x}_i = 0 \tag{1}$$

If we name $\mathbf{x}'_i = (x_i, y_i, w_i)^T$ and if the j-th row of H is denoted by $\mathbf{h}^{jT}$, then we can build a vector $\mathbf{h} = \left(\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3\right)^T$ with the 9 entries of H, and then, Equation 1 will correspond to the following set of equations:

$$\begin{pmatrix} \mathbf{0}^T & -w'_i\mathbf{x}_i^T & y'_i\mathbf{x}_i^T \\ w'_i\mathbf{x}_i^T & \mathbf{0}^T & -x'_i\mathbf{x}_i^T \\ -y'_i\mathbf{x}_i^T & x'_i\mathbf{x}_i^T & \mathbf{0}^T \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0} \tag{2}$$

Although there are three equations in (2), only two of them are linearly independent. Thus the above equation is equivalent to:

$$\begin{pmatrix} \mathbf{0}^T & -w'_i\mathbf{x}_i^T & y'_i\mathbf{x}_i^T \\ w'_i\mathbf{x}_i^T & \mathbf{0}^T & -x'_i\mathbf{x}_i^T \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0} \tag{3}$$

If a set of $n > 4$ correspondences is given, then Equation (3) is a over-determined system, and thus it can be formulated as the following optimization problem:

$$min \, \frac{\|A\mathbf{h}\|}{\mathbf{h}}, \tag{4}$$

constrained to $\|\mathbf{h}\| = 1$.

As shown in [10], the solution for this problem is the eigenvector related to the smallest eigenvalue of $A^T A$.

B. Least Squares Solution

The optimization problem defined by Equation 4 does not have a direct geometric interpretation. A better solution consists in finding the homography H that minimizes the following sum over all the n correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$:

$$min \sum_{i=0}^{n} \|H\mathbf{x}_i - \mathbf{x}'_i\|^2, \quad (5)$$

This problem can be solved by using the Levenberg-Marquardt algorithm. Since it is an iterative algorithm, it demands an initial estimation for H near to the optimum solution. The homography calculated by the Direct Linear Transform algorithm, explained in the previous section, can be used for this purpose.

V. Scene scale adjustment

An important fact is that we can change the scale of a scene without modifying the image captured by a camera. More precisely, if all the objects of the scene are scaled by a scalar factor $s \in \mathbb{R}$ and the optical center of a camera is scaled by s too, then the scaled point will be projected over the same pixel as the original. The proof of this fact is that:

Let $\mathbf{x} \in \mathbb{R}^3$ be a point in the scene. Lets consider a camera whose orientation relative to the scene reference is given by the 3×3 rotation matrix R, and whose optical center is in the position $\mathbf{c} \in \mathbb{R}^3$.

The position of the point x in the reference defined by the camera is:

$$\mathbf{p} = R^T\mathbf{x} - R^T\mathbf{c}$$

If we multiply $\mathbf{x}$ and $\mathbf{c}$ by the same factor s then the new point $s\mathbf{x}$ will have the following coordinates in the new camera reference:

$$R^T(s\mathbf{x}) - R^T(s\mathbf{c}) = s(R^T\mathbf{x} - R^T\mathbf{c}) = s\mathbf{p}$$

Since we did not change the intrinsic parameters of the camera, the point $s\mathbf{p}$ will be projected over the same pixel as $\mathbf{p}$, because they are in the same line passing through $(0, 0, 0)^T$, which is the coordinate of the optical center in the camera reference.

As a consequence of this scale ambiguity follows that whatever is the distance between the cameras used for capturing the stereoscopic pair, there is always a position for the users head, and a scale for reproducing the images over the display, that allows the user to observe a correct version, in some scale of the reality, of the object whose appearance is being captured. It happens because there is always a rescaled version of the scene that makes the distance between the cameras be equal to the distance between the user eyes.

For instance, considering that the distance between the eyes is about 6.5 cm, if the distance between the two optical centers of the cameras used for capturing the stereoscopic pair is 65 cm, and the distance of them to the captured object is four times it, that means 260cm, then the user must observe the display by the distance of four times 6.5cm, that is 26 cm, and the object displayed will be ten times smaller than the real one.

VI. System Architecture

We built prototypes for capturing and for presenting stereoscopic images. The ones that present images are the Virtual Tables, and when combined to capture devices they build what we call Virtual Teleporters.

The capture devices are plane surfaces that we point a pair of stereo cameras. We use as plane an ordinary table for small objects and the floor for large ones.

Before using capture devices, the homographies related to each camera are estimated by a software that establishes correspondences between the square corners of a checkerboard and their respective projections over the images captured by the camera pair. It is done as described in section IV.

While the system run, another software is used for applying the homographies previously estimated and for rescaling the images accordingly to the distance between the cameras, as described in section V. The result is a pair of images that are prepared to be shown horizontally.

We have built two capture Devices. The first one, presented in Figure 5-a, is used for capturing small objects. It is composed by a pair of small cameras whose distance between them are the same between the eyes of a human being. These cameras are pointed to an ordinary table. The second one, presented in 5-b, is used for capturing large objects and people. It is composed by two HD cameras fixed to a structure placed near the ceiling. Since the distance between the cameras in the second device is greater than in the first one, it makes the result be a stereoscopic pair that corresponds to a reduced size version of the captured objects.

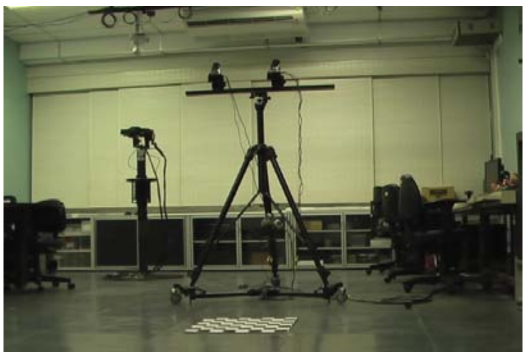
The Virtual Table is the device that shows the stereoscopic image pair. It has a stereoscopic screen positioned horizontally, that is connected to the computer by a NVIDIA Quadro card, which allows us to use quad-buffering in OpenGL.

We have built three Virtual Tables, the first one, presented in Figure 6-a, is made of a CRT monitor horizontally positioned over an iron and wood structure, and stereoscopic glasses.

The second one, presented in Figure 6-b, consists on an LCD monitor that supports 120Hz refresh rates positioned horizontally. This refresh rates allows it to display high quality stereoscopic images for shutter glasses.

The third one, presented in Figure 7, has not been designed to be part of a Virtual Teleporter, since it has lots of peripherical equipments that are useful only in the case

(a) Capture device used for small objects



(b) Capture device used for big objects

Figure 5. Virtual Teleporter Stereo Images

(a) (b)

Figure 6. Virtual Tables. (a) the CRT version and (b) the LCD version.

of presenting rendered content. Currently installed at the VISGRAF Laboratory. It is composed by:

- a stereoscopic projector, fixed to the ceiling of the laboratory, and its respective 3D glasses.
- a table that receives the stereoscopic projection.
- a camera, also fixed to the ceiling, that is used for capturing fiducials in interactions performed by the AR Toolkit.
- a Wii video game controller, that can be used as a head tracking system, by tracking an infrared LED on a cap worn by the user, and also as a controller, depending on the application.
- wireless mouse and keyboard, used for conventional interaction.

(a) (b)

Figure 7. In (a) it can be seen the camera and the stereoscopic projector fixed to the ceiling; and in (b) the table, 3D glasses, keyboard, mouse and the Wii control.

VII. Results

A Virtual Teleporter prototype has been built. Figure 8 illustrates users interacting with a 3D scene using both the Capture Table for acquisition of real object's images and the Stereo Viewing Table for showing them in real time.

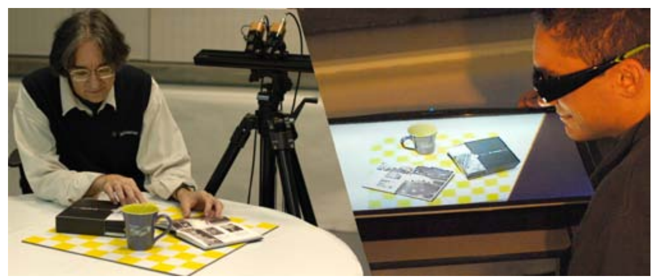


Figure 8. Interacting with the Virtual Teleporter.

The Figure 9 shows the image pair captured by each camera (9-a) and their respective deformed version (9-b) .

It can be noticed that, even when the floor is aligned to the screen, the limitation of the cameras frustum can generate points over the plane surface that are presented in one image that do not appear in the other one, Figure 10 illustrates it. It means that they should not be displayed, otherwise they may cause an interpretation problem, compromising the sense of reality. It must be clear that there are many cases that points do not have both stereoscopic projections visible due to occlusion. It is not a problem because the human visual sense is used to deal with it.

One way to deal with the limitation of the camera frustum consists in excluding the portion of the support of one image that does not appear in the other. It corresponds to reduce the supports of both images to the intersection of them. A better solution is to choose the distance between the cameras in such a way that makes the region exhibited by the display be inside the frustum of both cameras. It makes the quadrilateral border of both stereoscopic images does not appear, as is shown in Figure 11.

We used the OpenCV library for doing all the image processing. A consequence of this is that the algorithm has real-time performance on the CPU in the main memory. We

(a) Captured image Pairs

(b) Distorted images

Figure 9. Virtual Teleporter Stereo Images

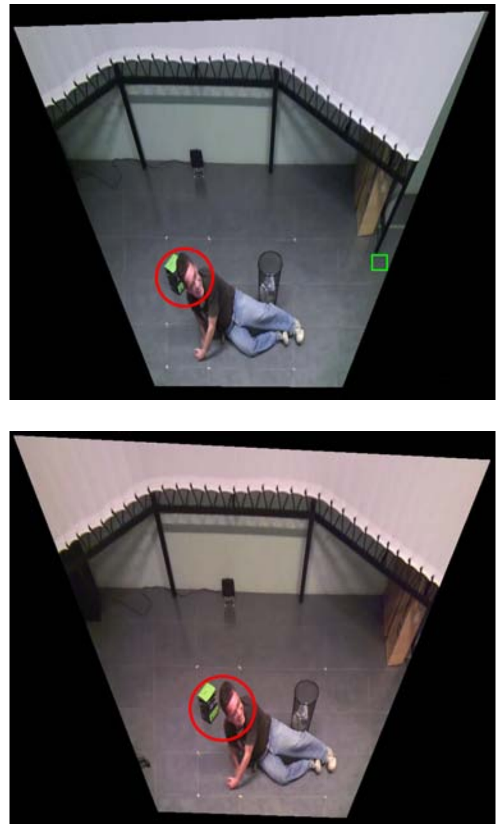


Figure 10. This Figure shows a stereoscopic image pair. The region inside the red circle present points on the box that do not have correspondence due to the occlusion with the head. The region inside the green square present points without correspondents due to the camera frustum limitation.

tested it on an Intel Core i7 computer, where we achieved a very good interactive response for images captured by ordinary NTSC cameras. It is possible that for FULL HD images a GPU implementation may become necessary, but we have not evaluated this yet. All the HD tests were off-line.

Besides the development of the Virtual Teleporter architecture that uses Image Processing for generating the stereoscopic pair, we developed applications that generates them synthetically by the use of Computer Graphics.

We adapted ordinary 3D applications for running over the Stereo Table, more specifically we adapted open source games, such as Warzone2100 and Cannon Smash. The choice of these games was not arbitrary. The game Warzone2100 is a 3D Real-Time Strategy Game that present various combats over hills. When this scenario is presented in an stereoscopic horizontal way the user has the idea that the combats are taking place over a miniature set, which is more natural than the sensation given by the original version (See Figure 12).

Although the modified version is interesting because of the mountain reliefs, nonetheless it presents two inconveniences:

- There are problems of absence of stereo correspondence at the edges of the image, because the mountain reliefs is not in the same level as the display at the edges.
- The game needs that the set scrolls, because the scene is much bigger than the area exhibited within the field of view. The scroll of 3D objects over the screen does not correspond to any natural process in the real world.

In order to test the stereoscopic effect without these problems we selected the game Cannon Smash (Figure 13), that represents a table tennis game, to be adapted. The above problems are eliminated because the tennis table can be kept static, without scrolling and the floor can be adjusted to match the screen level.

Additionally, we have developed our own interactive applications that can generate synthetic objects and present them over the Stereo Viewing Table. We developed solutions in C using OpenGL and in Python using the Panda 3D library. Examples of those applications are shown in the Figure 14.

We have also modeled a scene in the Autodesk Maya software adjusting the intrinsic and extrinsic parameters as explained in Section II. Then, we rendered the animation using global illumination and we displayed the videos over the Stereo Table, as shown in Figure 15.

VIII. Conclusion and Future Works

We presented a system that generates horizontal stereoscopic images, and we also explored the process to use it as a Virtual Teleporter, by presenting a virtual version of an object that is positioned over a surface.

Figure 11. The Virtual Table is showing an image whose scale was chosen in such a way that the quadrilateral border generated after applying the homography is outside the display.

(a) (b)

Figure 12. In (a) original version of the game Warzone. In (b) the modified version of the game being exhibited over the Stereo Table.

Figure 13. In (a) original version of the game Cannon Smash. In (b) the modified version of the game being exhibited over the Stereo Table.

In the case of synthetic images we used a head tracking to deform the image in order to allow the user to move his head, and we experimented many kinds of interactive mechanisms.

In the case of captured images we applied homographies in order to deform them appropriately, the disadvantage when compared to the method used in the synthetic case is that the user cannot move his head while observing the virtual scene, one the other hand, we achieved a very high visual quality. Besides that, the computer vision approach used to estimate the homographies gave us some freedom to set the pose of the cameras used to capture the non deformed stereoscopic pair.

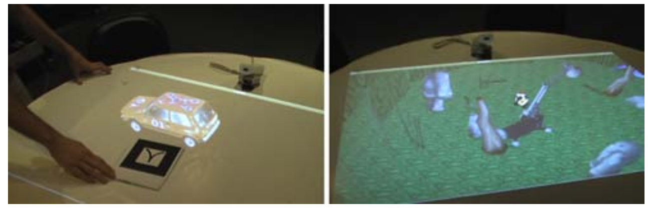


Figure 14. Interactive applications developed with Panda 3D.

Figure 15. 3D Animation of a chorus line.

We have used OpenCV to apply the homography, which makes the process heavy, since it is being done by software. We intend to implement all the processing with Cuda and OpenGL using texture map resources. It will move the problem to the GPU for increased performance. We also intend to adapt the technique presented here to the case of theater plays.

REFERENCES

[1] J.-B. de la Rivire, “3d multitouch: When tactile tables meet immersive visualization technologies,” *SIGGRAPH Talk*, July 2010.

[2] J. R. Yoshiki Takeoka, Takashi Miyaki, “Z-touch: A multi-touch system that detects spatial gesture near the tabletop,” *SIGGRAPH Talk*, July 2010.

[3] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York, NY, USA: Henry Holt and Co., Inc., 1982.

[4] N. C. Michael A. Vesely, “Horizontal perspective representation,” *United States Patent*, no. US 2006/0126926 A1, June 2006.

[5] O. C. Wester, “Anaglyph and method,” *United States Patent*, no. 6,389,236, February 2000.

[6] S. Aubrey, “Process for making stereoscopic images which are congruent with viewer space,” *United States Patent*, no. 6,614,427, February 2000.

[7] B. Leibe, T. Starner, W. Ribarsky, Z. Wartell, D. Krum, J. Weeks, B. Singletary, and L. Hodges, "Towards spontaneous interaction with the perceptive workbench, a semi-immersive virtual environment," *IEEE Computer Graphics and Applications*, vol. 20, pp. 54–65, 2000.

[8] L. D. Cutler, B. Fröhlich, and P. Hanrahan, "Two-handed direct manipulation on the responsive workbench," in *Proceedings of the 1997 symposium on Interactive 3D graphics*, ser. I3D '97. New York, NY, USA: ACM, 1997, pp. 107–114. [Online]. Available: http://doi.acm.org/10.1145/253284.253315

[9] C. Cruz-neira, D. J. Sandin, and T. A. Defanti, "Surround-screen projection-based virtual reality: The design and implementation of the cave," 1993, pp. 135–142.

[10] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.