5* Magic Wand: a RGBD camera-based 5 DoF pointing device for 3D interaction

Alexandre M. F. de Sousa, Carlos H. Morimoto
Department of Computer Science
University of São Paulo
São Paulo, Brazil
{*alemart, hitoshi*}*@ime.usp.br*

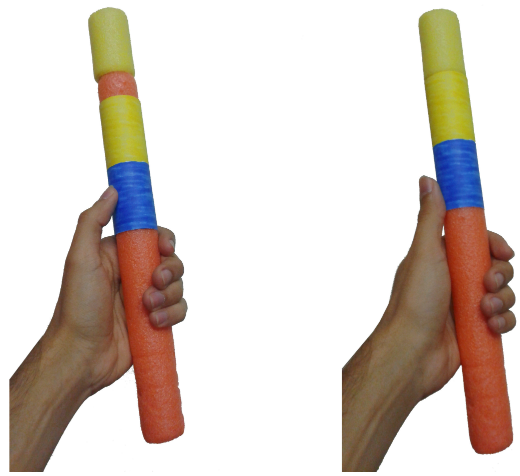
***Abstract*—This paper introduces the 5* Magic Wand, an inexpensive pointing device for 3D interaction. As users play the role of a "wizard", the 5* Magic Wand allows them to interact with the computer by pointing to places, performing gestures, and casting "spells". We have developed computer vision techniques to track the wand with 5 degrees of freedom, and built a real-time prototype consisting of a standard PC, a RGBD camera, and a simple stick. Finally, in order to demonstrate the applicability of the system for navigating in virtual environments, we have created a 3D application called "Magic Carpet".**

***Keywords*-3D user interface; magic wand; RGBD sensor; virtual reality; ubiquitous computing.**

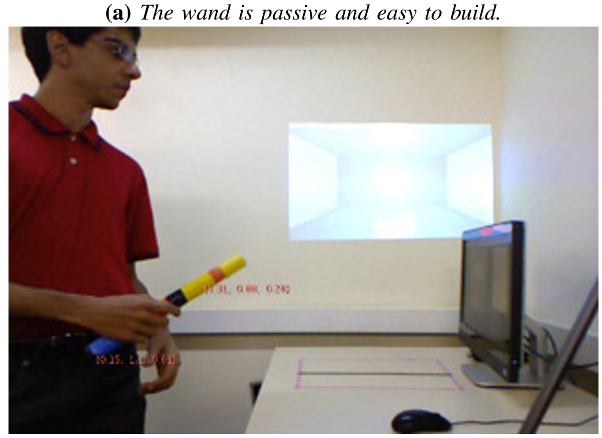
I. Introduction

As technology drive us away from office spaces, interacting with computers like we do with the traditional desktop environment is no longer enough. In Human-Computer Interaction (HCI) research, advances in hardware and software have stimulated the development of a new generation of user interfaces called post-WIMP. Post-WIMP interfaces contain at least one interaction technique that is not based on classical 2D widgets such as menus and icons [1]. The motivation behind these interfaces is to leverage the mundane, pre-existing knowledge of users for reducing the gap between human goals and the actions required to accomplish them [2].

One important application of post-WIMP interfaces is the interaction in virtual environments (VEs). In particular, travel in VEs is the most common interaction activity in 3D user interfaces [3]. Such interfaces should be easy to learn and easy to use, becoming "second nature" to users. As our interaction with computers expand beyond the standard desktop setting, it becomes desirable to explore different input settings. One possible approach is to use a wand. The affordances of that physical tool can be exploited in a variety of ways; it can be mapped to different interaction tasks in VEs. Using the notion of a magic wand for interaction makes it easy for the regular user to interact with the virtual world, as it is unobstructive and removes the learning curve of less intuitive equipment such as the 3D mice [4]. In a recent study, Khan *et al.* [5] have shown that adopting a wand is an efficient manner to move within a VE, compared to relying on the position of the user in the physical space. Additionally, in comparison to using gloves or a 3D mouse, wand users tend to adopt more comfortable postures (they tend to find postures they like rather than postures they feel are required) [6].



(a) *The wand is passive and easy to build.*



(b) *A user holding the wand.*

Figure 1: *The 5* Magic Wand.*

Wands also find application in diverse areas such as smart-home environments [7], education and interactive art. A magic wand could be used in a smart-home to turn a light on/off, control a television and so on. A teacher could use a magic wand in a classroom to point to and manipulate an object in 3D (e.g., a cell, a chemical molecule or a machine) in order to aid an explanation, if such an input device was cheap and easy to deploy. Similarly, for

ISSN: 2236-3297

storytelling purposes, an artist could use a magic wand as part of an artistic performance, mixing both the real and the virtual worlds.

Nowadays, RGBD cameras are becoming inexpensive and widespread. These cameras provide color and estimated depth data for each pixel. In November 2010, Microsoft released the Kinect, a popular RGBD sensor that is an order of magnitude cheaper than earlier similar devices [8]. More recently, Intel has announced the RealSense[1], a RGBD sensor that may be embedded in everyday devices such as laptops. RGBD cameras have a wide range of areas of applications, including: HCI, augmented reality, 3D reconstruction and robotics [9]. The potential ubiquity of such sensors in the future make them an attractive choice for implementing an interface such as a magic wand for 3D interaction.

This paper presents the 5* Magic Wand: a minimalist 5 degrees of freedom (DoF) 3D user interface that can be built inexpensively and that draws upon users' knowledge carried over from childhood (e.g., fantasy movies). Unlike similar works, the 5* Magic Wand includes an on/off switch to enhance its interaction capabilities and is tracked using one RGBD camera, making it resilient against background noise. A novel computer vision technique to track the wand is described. In order to show how the wand may be used for navigating in virtual environments, we introduce a 3D demo application called "Magic Carpet".

The rest of this paper is organized as follows. Section II presents related work. Section III discusses the design of the 5* Magic Wand. Section IV discusses the system implementation, the computer vision algorithms to track the wand with five degrees of freedom and presents a working prototype. Section V presents a demo application and section VI concludes the paper.

II. Related Work

Using magnetic technology, Ciger *et al.* [4] have presented "The Magic Wand", a user interface for interaction with a VE displayed on a large projection screen. The user flies on a "magic carpet", exploring the landscape. The wand is used to point to regions of the scene, and a small vocabulary of voice commands ("spells") directs the simulator. Although the system doesn't use position data, it recognizes four basic postures/orientations of the wand. The hardware setup demands the presence of wired sensors attached to the wand.

The XWand [7], which works in an intelligent environment, is a wireless sensor package used to control devices by pointing at them and performing simple gestures. The hardware prototype of the proposed handheld device brings together a combination of sensors and other components, such as an infra-red (IR) LED and a microcontroller. The authors combine the output data of an accelerometer and of a magnetometer to recover the 3D orientation of the wand. A pair of calibrated cameras equipped with IR pass filters is used to determine, by triangulation, the 3D position of the wand. Although the solution provides an output of 6 degrees-of-freedom, an important drawback of the discussed implementation is its high complexity.

Wand-like devices may also be used in everyday contexts such as "sofa interaction". Interactive televisions are becoming more widespread and feature-rich. On the other hand, to accomodate such an increasing number of functions, remote controls get frequently cluttered with buttons. That said, Bailly *et al.* [10] have proposed the use of mid-air gestures to augment and improve remote control interaction. Their system employs a Wiimote, which includes a few buttons and provides orientation data, thus enabling gesture recognition capabilities.

Cabral *et al.* [11] have developed a collaborative application that works in a CAVE system equipped with a Optitrack System featuring four infrared cameras. One of the users of the application, called the explorer, uses a Wiimote coupled with a reflective marker to point where he wants to navigate to (the Wiimote is tracked for its position and orientation). Then, he pushes a button to pull himself to that place. That interaction metaphor, called "point and go", was used in a closed VE (i.e., no sky).

Using a pair of calibrated stereo cameras, Guo *et al.* [12] have designed the "Featured Wand", a passive wand with two colored end markers and a spiral marking in between. A computer vision system tracks the position and the orientation of the wand in 3D space, at 9 frames per second. The authors have subsequently used the wand for navigation in VEs: tilting up or down controls the inclination of the viewpoint, and rotating left or right controls the heading. Raising or lowering the wand controls the elevation, and lateral movements change the panning.

Researchers have also employed computer vision techniques to explore the use of standard laser pointers as input devices. In a recent work, Kang and Yang [13] track a laser point by capturing a computer screen projected in a room and performing a color-based tracking after a background subtraction. A standard rectification procedure is employed to identify, in the screen coordinate system, the location pointed by the laser. Although the solution has the advantage of low cost, the input is restricted to two dimensions only, and issuing commands to the computer can be a daunting task, since there are no buttons or similar mechanisms. Still, the authors have used hybrid techniques combining dwell time and gesture recognition to control the flow of a presentation.

Unlike previous works, the 5* Magic Wand enables real-time 3D interaction using a passive stick coupled with a "spell casting" mechanism. Our method employs one RGBD camera to track the pointing device.

[1]https://software.intel.com/en-us/realsense/. Last access: March 11th, 2015.

III. Design

The design goals of the 5* Magic Wand can be ennumerated as follows:

1) it should be tracked for its position and orientation in 3D space;
2) it should enable users to point to elements on a screen;
3) it should present a mechanism for enabling the execution of "spells" (gestures), hence giving the wand its magic touch;
4) it should be built easily and inexpensively;
5) it should be wireless.

The proposed design for our wand brings features that enable its tracking by a RGBD camera. We start with a plastic rod with length of about $30cm$. Colored markings placed along the wand (see Fig. 2a) enable its tracking by a computer vision system. This particular configuration allows 5 DoF pose estimation (position in 3D space plus two rotation angles: yaw and pitch). One can then define a 3D ray starting from the head of the wand pointing to the direction of the rod. Once the coordinates of the screen are known, it's easy to use the wand as a pointing device.

In addition to the above, we have designed a switch: a movable part near one of the extremes of the wand will indicate when a "spell" begins and when it ends (see Fig. 2b). The switch may be moved using a thumb, and it works like an on/off element. Whenever it overlaps the corresponding extreme of the device (i.e., the extreme sharing the same color), the wand is said to be active. Otherwise, it's said to be inactive.

The blue marking helps to determine the orientation of the wand. Although it may be placed near the switch (Fig. 1a) or at one of the ends of the wand (Fig. 1b), after testing both configurations we have found out that the former is a better choice to reduce occlusions and to help the user move the switch.

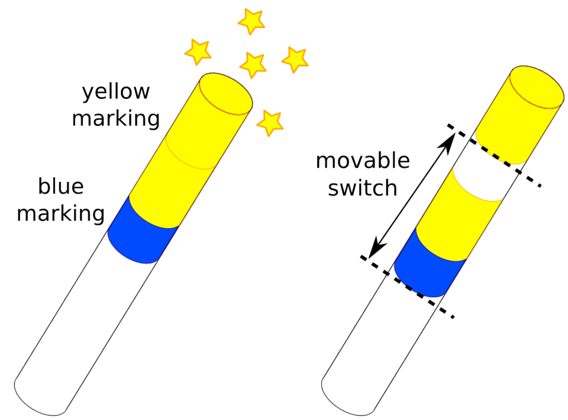
Finally, the design of our passive wand (i.e., no electronic components, no wires, etc.) enables it to be built easily and inexpensively and makes it a wireless interaction device.

IV. Implementation

As shown in Figure 4, the computational implementation of the 5* Magic Wand is divided in three layers:

- **Sensory layer:** a RGBD camera will capture color and depth data from the environment;
- **Tracking layer:** this layer determines the position, orientation and state (active or inactive) of the wand;
- **Application layer:** uses the wand as an input device for some activity (e.g., travel in a VE).

Regarding the sensory layer, a Kinect camera is used. Its availability and affordable cost make it an attractive choice for implementing the system. The tracking layer brings technical challenges that will be addressed in more detail.



(a) *Active.* **(b)** *Inactive.*

Figure 2: *Two modes of operation.*

A. Tracking layer

As depicted in Figure 3, given the RGBD data from the sensory layer, a sequence of steps is employed to extract the position, orientation and state (active or inactive) of the wand.

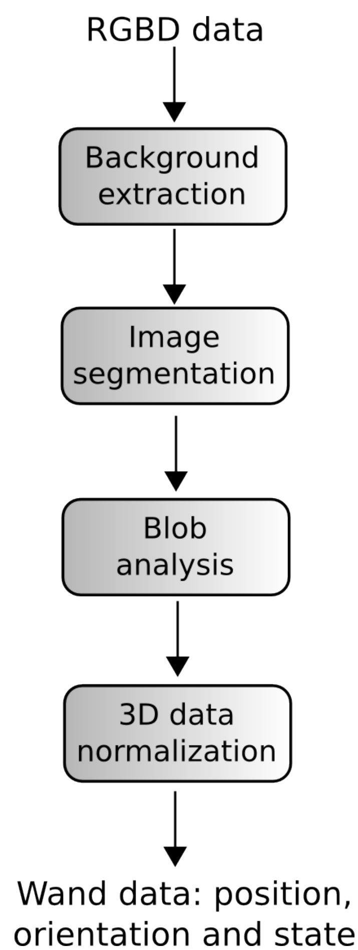


Figure 3: *Steps performed by the Wand Tracker.*

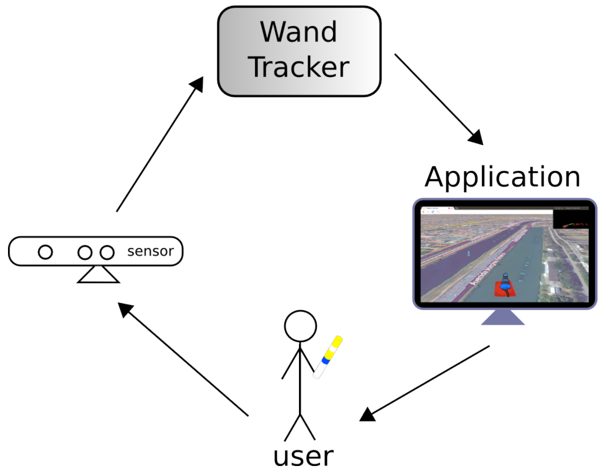


Figure 4: *System overview.*

1) Background extraction: Unlike similar works based on optical tracking, tracking the wand with a RGBD camera make it resilient against background noise, since the background can be extracted using depth data. Additionally, past research has indicated that the depth estimates provided by the Kinect are quite stable [14].

A background model is created by taking n snapshots $\{S_1, S_2, \ldots, S_n\}$ of the depth image. Denoting the depth value of the j-th pixel of S_i by d_{ij}, let μ_j and σ_j be, respectively, the mean and the standard deviation of $\{d_{1j}, \ldots, d_{nj}\}$. The background model is said to be the collection of all μ_j. The set of all σ_j is used as a noise profile.

The background extraction is performed by subtracting the newly acquired depth images from the background model.

2) Image segmentation: After the previous step, the color image provided by the RGBD sensor features artifacts belonging to the foreground. The colored markings of the wand are segmented by converting the color image to the HSV space and employing a thresholding method.

3) Blob analysis: The resulting blobs, computed using connected components analysis, indicate the colored markings. In addition to the blue marking, the wand features two yellow markings: one corresponds to the on/off switch and the other indicates one of the ends of the wand. If the switch is on, the markings overlap, meaning that only one yellow blob will be visible in the color image. Therefore, the state of the wand is set to be "active". If there are two yellow blobs, the wand is "inactive".

4) 3D data normalization: A RGBD sensor can provide the 3D position, in its own coordinate system, of any pixel [15]. That said, the yellow markings are used to determine the position of the wand in 3D. The position is taken to be the 3D location of the center of mass of the yellow blob (or the mean of the centers of mass if there is more than one yellow blob).

In order for the wand to be usable in the application layer, its position and orientation need to be normalized. This allows one to account for different placements of the camera. Let R be the cubic region described by:

$$R: \lim_{t\to\infty} \sqrt[t]{\left|x-\frac{1}{2}\right|^t + \left|y-\frac{1}{2}\right|^t + \left|z-\frac{1}{2}\right|^t} \leq \frac{1}{2}$$

We create a transformation T that maps a position from the camera coordinate system to the normalized space described by R. Let (x_k, y_k, z_k) be the coordinates of the wand in the camera system and (x, y, z) be the corresponding normalized position. Given the model parameters $(a, b, c, d, e, f, g, h, i, j, k, l)$, mapping T is defined as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ z_k \\ 1 \end{bmatrix}$$

Expanding the above equation, we have:

$$\begin{cases} x = a \cdot x_k + b \cdot y_k + c \cdot z_k + d \\ y = e \cdot x_k + f \cdot y_k + g \cdot z_k + h \\ z = i \cdot x_k + j \cdot y_k + k \cdot z_k + l \end{cases}$$

A calibration procedure is employed to estimate the model parameters. Given a set of $n \geq 4$ correspondences, denoted by $(x^{(j)}, y^{(j)}, z^{(j)}) \leftrightarrow ({x_k}^{(j)}, {y_k}^{(j)}, {z_k}^{(j)})$ for $1 \leq j \leq n$, the above equations can be rewritten to:

$$\begin{bmatrix} x^{(1)} \\ y^{(1)} \\ z^{(1)} \\ x^{(2)} \\ y^{(2)} \\ z^{(2)} \\ \vdots \\ x^{(n)} \\ y^{(n)} \\ z^{(n)} \end{bmatrix} = \begin{bmatrix} {x_k}^{(1)} & {y_k}^{(1)} & {z_k}^{(1)} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & {x_k}^{(1)} & {y_k}^{(1)} & {z_k}^{(1)} & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {x_k}^{(1)} & {y_k}^{(1)} & {z_k}^{(1)} & 1 \\ {x_k}^{(2)} & {y_k}^{(2)} & {z_k}^{(2)} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & {x_k}^{(2)} & {y_k}^{(2)} & {z_k}^{(2)} & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {x_k}^{(2)} & {y_k}^{(2)} & {z_k}^{(2)} & 1 \\ & & & & & \ddots & & & & & & \\ {x_k}^{(n)} & {y_k}^{(n)} & {z_k}^{(n)} & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & {x_k}^{(n)} & {y_k}^{(n)} & {z_k}^{(n)} & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & {x_k}^{(n)} & {y_k}^{(n)} & {z_k}^{(n)} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \\ j \\ k \\ l \end{bmatrix}$$

A least-squares solution for the model parameters can be found by applying a pseudoinverse matrix, computed using methods such as singular value decomposition or QR factorization [16].

From the end-user point of view, one does not need to manually describe the correspondences. The calibration procedure simply requires that the user moves the wand near the camera (see Fig. 5). Let the curve $\gamma(t) : (x_k(t), y_k(t), z_k(t))$ describe such a movement, where $t \geq 0$ denotes time. We proceed by defining what we call box functions:

$$x_k{}^{\#}(d) = d \cdot \max_{t\geq 0}\, x_k(t) \;+\; (1-d) \cdot \min_{t\geq 0}\, x_k(t)$$

$$y_k{}^{\#}(d) = d \cdot \max_{t\geq 0}\, y_k(t) \;+\; (1-d) \cdot \min_{t\geq 0}\, y_k(t)$$

$$z_k{}^{\#}(d) = (1-d) \cdot \max_{t\geq 0}\, z_k(t) \;+\; d \cdot \min_{t\geq 0}\, z_k(t)$$

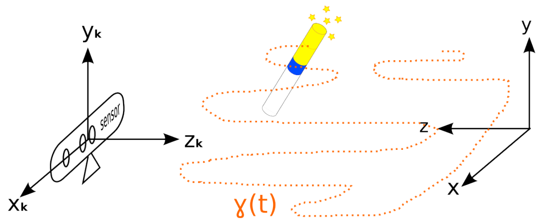


Figure 5: *Calibration procedure.*

For $i \in \mathbb{N}^*$, let us define the family of functions:

$$J_i(j) = \left\lfloor \frac{j-1}{2^{(i-1)}} \right\rfloor - 2 \left\lfloor \frac{j-1}{2^i} \right\rfloor$$

Then, for $j = 1, 2, \ldots 8$, we set the correspondences:

$$x^{(j)} = J_1(j) \;\leftrightarrow\; {x_k}^{(j)} = {x_k}^{\#}\,(J_1(j))$$

$$y^{(j)} = J_2(j) \;\leftrightarrow\; {y_k}^{(j)} = {y_k}^{\#}\,(J_2(j))$$

$$z^{(j)} = J_3(j) \;\leftrightarrow\; {z_k}^{(j)} = {z_k}^{\#}\,(J_3(j))$$

Once the model parameters have been estimated, the normalized 3D position of the colored markings will determine the orientation of the wand. This phase ends with a state (active or inactive) and a pose estimation with five degrees of freedom: the 3D position of the wand plus two rotation angles (yaw and pitch).

B. Application layer

The tracking layer communicates wand data to the application layer. As displayed in Figure 6, an HTML5 application has access to the position, orientation and state of the wand. Additionally, a gesture recognition module enhances the interactive capabilities of our wand, turning it into a "magic" tool.

According to Mitra and Acharya, gestures are meaningful body movements involving fingers, arms, head or the full body in order to convey information or interact with the environment [17]. Ghirotti and Morimoto have pointed out two main reasons for using gestures for interaction with computers [18]:

1) People use a wide range of gestures in everyday life. New gestures can be easily and quickly learned by observing others perform them;

2) Gesture-based interfaces allow the "natural" usage of "gesture phrases". These phrases can be thought of as a way to break the communication dialog in parts. These parts have simple meanings and can be easily interpreted by computers (e.g., the action of moving an object in a VE can be decomposed in three parts: grabbing, translating and releasing it).

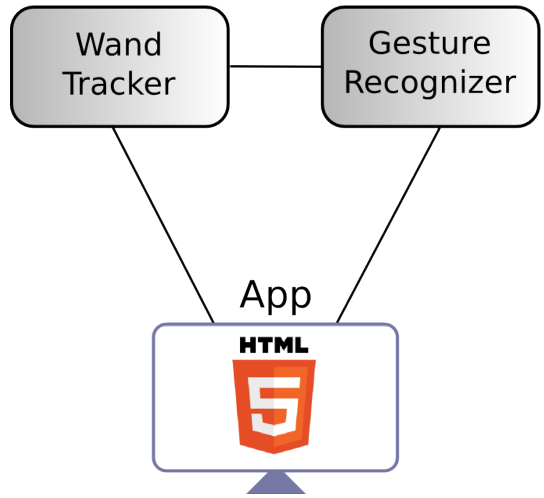


Figure 6: *The wand can control an HTML5 application.*

Gestures can be static (the person assumes a specific pose), dynamic (there is ongoing movement) or a mixture of the two. Automatic recognition of dynamic gestures requires temporal segmentation, and often the user has to specify the beginning and the ending points of a gesture in space and time [17].

The 5* Magic Wand includes an on/off switch. Whenever the user pushes the switch so that the wand becomes active, we'll say that a spell has begun. Whenever the user pulls the switch back to its previous position (i.e., whenever the wand becomes inactive), we'll say that a spell has ended. In this work, a spell is defined to be the trajectory performed by the wand in its active state.

The Gesture Recognition block depicted in Figure 6 receives a spell from the Wand Tracker and assigns it a label ("circle", "hat", and so on). The advent of lightweight and robust gesture recognizers has enabled the rapid prototyping of gesture-based user interfaces [19], [20]. After normalizing the gestures, often adding position, scale and rotation invariance, they employ template matching techniques for recognition. These recognizers usually rely on simple geometry and trigonometry and perform remarkably well with only a few training examples, making them an attractive choice for enabling users to cast spells with our wand. Our prototype uses the $1 Recognizer, a popular single-stroke gesture recognizer [19].

The Wand Tracker, built in C++, sends data to the HTML5 application using TUIO over WebSockets. Although TUIO is a protocol originally designed to support the development of multitouch tangible user interface systems [21], its feature set makes it a fair choice for the transmission of wand data.

Figure 7 shows our prototype implementation running at about 28 frames per second on a Dell All-in-One Inspiron 2330 PC (with 6 GB RAM and an Intel Core i5 CPU at 2.30 GHz).

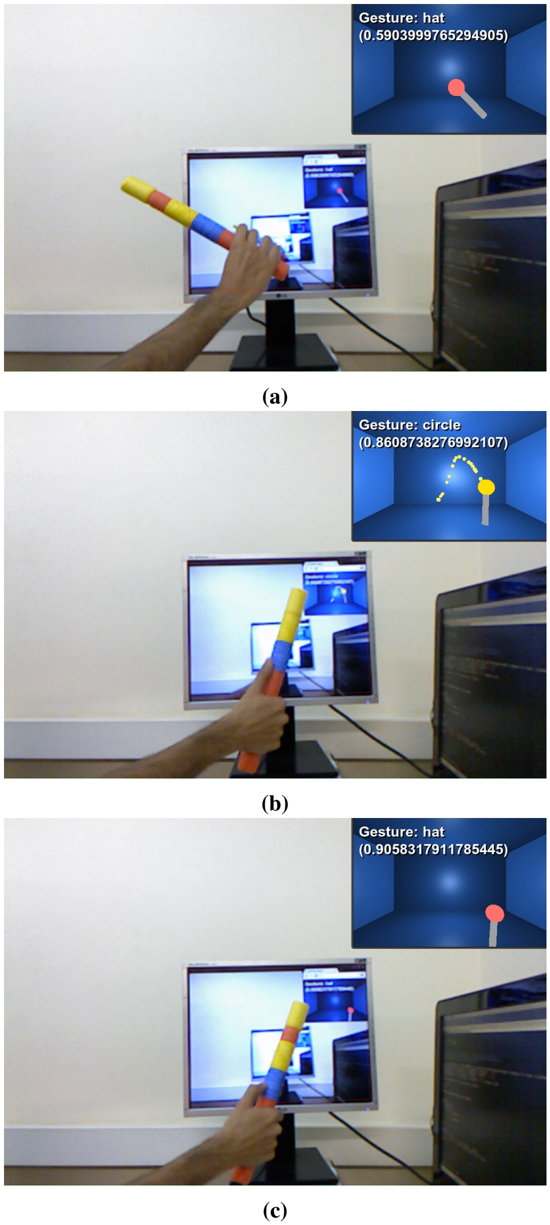


Figure 7: *A prototype implementation featuring gesture recognition capabilities.*

V. Demo Application

We have built a simulator in order to demonstrate how wand based input may be used in a 3D navigation activity. The user, playing the role of a "wizard", will fly around on a "Magic Carpet" that is controlled by the wand alone. The desirable features for the simulator are the following:

1) The user may travel anywhere around the Earth;
2) The viewpoint is controlled in three dimensions by the wand (see Fig. 9);
3) The carpet always runs parallel to the ground, in fixed altitudes (see Fig. 10).

The orientation of the wand controls the viewpoint. More precisely, let $\omega = [\ \omega_x\ \omega_y\ \omega_z\ ]^t$ be the orientation of the wand (see Fig. 8), where $\|\omega\| = 1$. Then, at each framestep, the simulator proceeds as follows:

1) set the panning of the camera to $\varphi + \varphi_0$, where $\varphi = atan2(\omega_z, \omega_x)$ and φ_0 is a predefined offset[2];
2) move the viewpoint if $|\,tan(\theta)\,| \leq tan(\theta_{lim})$, where $\theta = sin^{-1}(\omega_y)$ and θ_{lim} is an empirical constant[3].

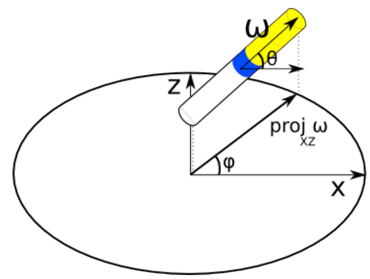


Figure 8: *Orientation of the wand.*

An obvious consequence of the proposed design is that the viewpoint will move forwards if the wand is laid down (i.e., $\theta \approx 0°$), but it will stop if the wand is put in a standing position ($\theta \approx 90°$). Additionally, changing the direction of the wand will change the direction of the camera.

An important feature to be considered when designing user interfaces is the rest pose. The rest pose is important to establish a situation where there is no interaction with the system [18]. In order to rest, the user may put the wand in a standing position, so that the system is instructed to stop the flight.

In addition to using the orientation of the wand to navigate in 3D, the user may also cast spells in order to issue commands to the Magic Carpet. Three spells are supported by the simulator: $\wedge$, $\vee$ and $\bigcirc$. Table I associates the spells to the corresponding actions performed by the simulator.

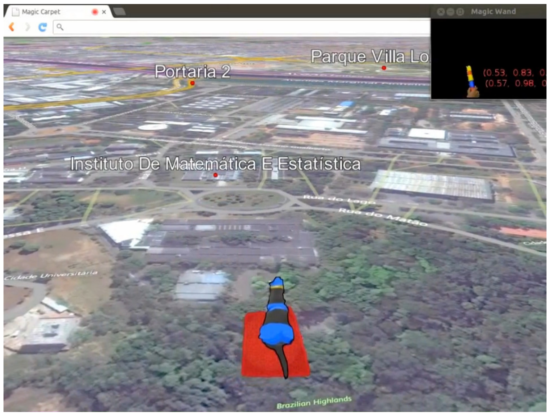
Spell	Resulting action
$\wedge$	Increases the altitude.
$\vee$	Decreases the altitude.
$\bigcirc$	Rotates the carpet by 180°.

Table I: *Spells command the Magic Carpet.*

The spells cast by the user are projected onto a plane and then passed on to the gesture recognizer. The recognizer receives the spell, normalizes it and then matches it against the set of supported gestures, as described in [19].

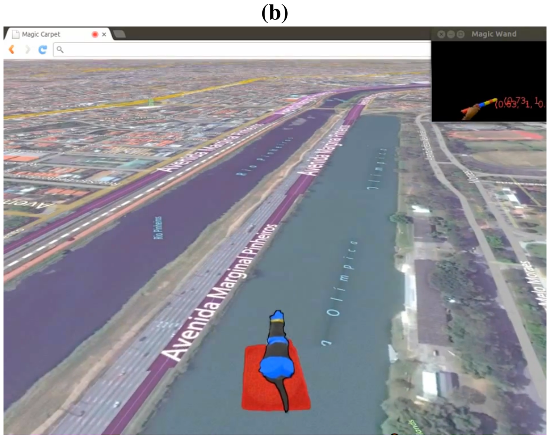
The simulator has been built using Cesium, an open-source WebGL-based Virtual Globe & Map Engine[4].

[2]in our simulator, $\varphi_0 = 1.23\pi$.
[3]in our simulator, $\theta_{lim} = 45°$.
[4]https://cesiumjs.org/. Last access: March 15th, 2015.



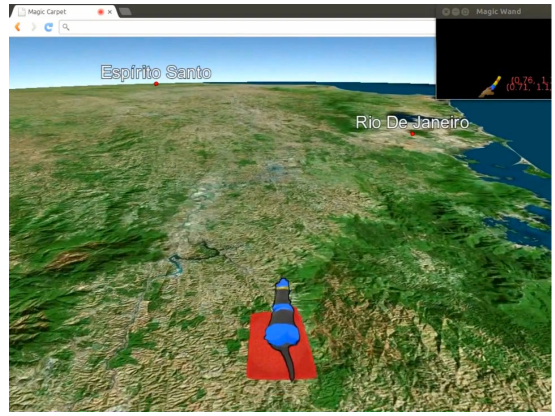
(a)

(b)

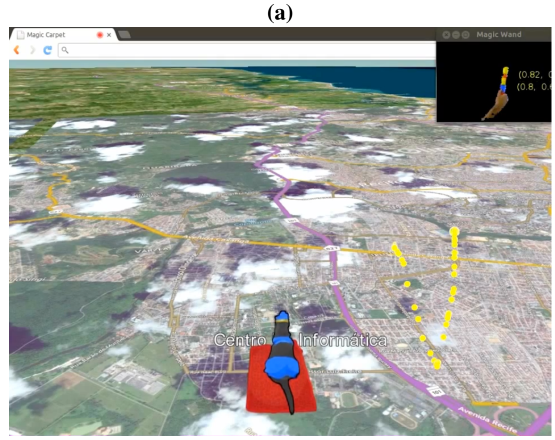


(c)

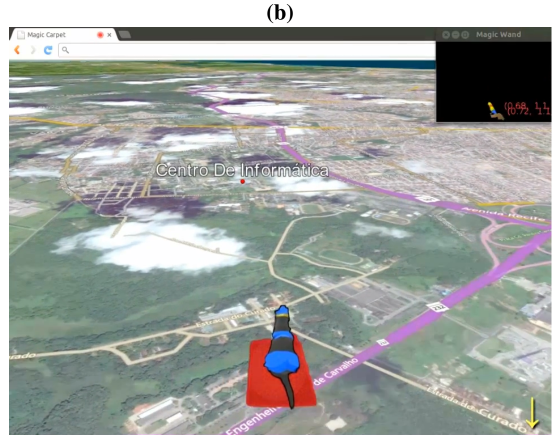
Figure 9: *Flying over the university campus.*



(a)



(b)



(c)

Figure 10: *Casting spells to modify the altitude.*

VI. Conclusion

This paper has presented the 5* Magic Wand: a user interface for 3D input that provides position and orientation. Its physical shape makes it suitable for pointing, gesturing and casting magical "spells". The presence of the on/off switch expands the interactive capabilities of the device. The simple hardware setup of the wand eliminates the need of wires/electronic components, hence it is unobstructive and can be built quickly and inexpensively.

Computer vision techniques to track the wand in 3D space with a RGBD camera were presented. After extracting the background and finding the colored markings of the wand, a linear model is employed to determine its position and the orientation in a normalized space. A calibration procedure that we have developed is used to estimate the model parameters.

In order to demonstrate the proposed device and the presented techniques, we have built a prototype featuring a standard PC, a RGBD camera and a simple stick. Our solution works in real-time, making it a suitable choice for adding wand interaction to virtual reality systems. That said, we have created a 3D application called "Magic Carpet" to show how the wand may be used to navigate in virtual environments. Additionally, a gesture recognizer enables users to cast magical "spells", thus enhancing the capabilities of the simulator.

Given the potential high availability of RGBD cameras in the near future (e.g., embedded in everyday computers) and the minimalist nature of the 5* Magic Wand, we believe that this work may help bring 3D wand interaction to everyday settings such as education.

Possible improvements upon this work include: evaluate the wand and the proposed interaction technique with user studies, include a 6th degree of freedom (twist) on the wand and use the push-pull mechanism of the switch as a source of continuous data rather than an on/off one. Additionally, the passive and inexpensive nature of the wand enables it to be used for controlling games.

The complete source code of this work is available at https://github.com/alemart/mestrado. A video showing the simulator can be found at https://youtu.be/57htwAJ4Im4.

References

[1] A. Van Dam, "Post-wimp user interfaces," *Communications of the ACM*, vol. 40, no. 2, pp. 63–67, 1997. 1

[2] R. J. Jacob, A. Girouard, L. M. Hirshfield, M. S. Horn, O. Shaer, E. T. Solovey, and J. Zigelbaum, "Reality-based interaction: a framework for post-wimp interfaces," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2008, pp. 201–210. 1

[3] D. A. Bowman, E. Kruijff, J. J. LaViola Jr, and I. Poupyrev, *3D user interfaces: theory and practice*. Addison-Wesley, 2004. 1

[4] J. Ciger, M. Gutierrez, F. Vexo, and D. Thalmann, "The magic wand," in *Proceedings of the 19th spring conference on Computer graphics*. ACM, 2003, pp. 119–124. 1, 2

[5] V. Khan, M. Pekelharing, and N. Desle, "Efficient navigation in virtual environments: A comparative study of two interaction techniques: The magic wand vs. the human joystick," in *Intelligent Human Computer Interaction (IHCI), 2012 4th International Conference on*. IEEE, 2012, pp. 1–5. 1

[6] M. Henschke, T. Gedeon, R. Jones, S. Caldwell, and D. Zhu, "Wands are magic: a comparison of devices used in 3d pointing interfaces," in *Human-Computer Interaction–INTERACT 2013*. Springer, 2013, pp. 512–519. 1

[7] A. Wilson and S. Shafer, "Xwand: Ui for intelligent spaces," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2003, pp. 545–552. 1, 2

[8] K. Litomisky, "Consumer rgb-d cameras and their applications," *University of California, Riverside. Ano*, 2012. 2

[9] L. Cruz, D. Lucio, and L. Velho, "Kinect and rgbd images: Challenges and applications," in *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2012 25th SIBGRAPI Conference on*. IEEE, 2012, pp. 36–49. 2

[10] G. Bailly, D.-B. Vo, E. Lecolinet, and Y. Guiard, "Gesture-aware remote controls: guidelines and interaction technique," in *Proceedings of the 13th international conference on multimodal interfaces*. ACM, 2011, pp. 263–270. 2

[11] M. Cabral, G. Roque, D. dos Santos, L. Paulucci, and M. Zuffo, "Point and go: Exploring 3d virtual environments," in *3D User Interfaces (3DUI), 2012 IEEE Symposium on*. IEEE, 2012, pp. 183–184. 2

[12] F. Guo, D. Kimber, and E. G. Rieffel, "Featured wand for 3d interaction." in *ICME*, 2007, pp. 2230–2233. 2

[13] S.-H. Kang and C.-K. Yang, "Laser-pointer human computer interaction system," in *Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1–6. 2

[14] A. D. Wilson, "Using a depth camera as a touch sensor," in *ACM international conference on interactive tabletops and surfaces*. ACM, 2010, pp. 69–72. 4

[15] J. Kramer, M. Parker, D. Herrera, N. Burrus, and F. Echtler, *Hacking the Kinect*. Apress, 2012. 4

[16] L. N. Trefethen and D. Bau III, *Numerical linear algebra*. Siam, 1997, vol. 50. 4

[17] S. Mitra and T. Acharya, "Gesture recognition: A survey," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 3, pp. 311–324, 2007. 5

[18] S. E. Ghirotti and C. H. Morimoto, "Um sistema de interação baseado em gestos manuais tridimensionais para ambientes virtuais," in *Proceedings of the IX Symposium on Human Factors in Computing Systems*. Brazilian Computer Society, 2010, pp. 159–168. 5, 6

[19] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes," in *Proceedings of the 20th annual ACM symposium on User interface software and technology*. ACM, 2007, pp. 159–168. 5, 6

[20] Y. Li, "Protractor: a fast and accurate gesture recognizer," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2010, pp. 2169–2172. 5

[21] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza, "Tuio: A protocol for table-top tangible user interfaces," in *Proc. of the The 6th Intl Workshop on Gesture in Human-Computer Interaction and Simulation*, 2005, pp. 1–5. 5