Controlling Virtual Worlds Using Interaction Spheres

Bernd Schwald[1], **Cornelius Malerczyk**[1]

[1]Department of Visual Computing - Computer Graphics Center (ZGDV)
Fraunhoferstr. 5 - D-64283 Darmstadt, Germany

{bernd.schwald,cornelius.malerczyk}@zgdv.de

Abstract. *A growing number of Virtual Reality applications is displayed on large-scale screens and the conventional way of interaction by using a physical device like a mouse, space mouse or some control panel is often not the solution really aimed at. This paper presents a very flexible method to control and interact wireless with virtual worlds by defining a workspace around the user. This workspace consists of virtual spheres, which are connected to certain events, such that the user can interact by just moving his hand into these spheres. Within a short and simple learning procedure, the interaction spheres are defined individually for and by each user. Extensions to perform a virtual mouse click and to take the position of the body into account are included.*

1. Introduction

In the growing number of Virtual Reality (VR) applications the usage of a physical device like a mouse, space mouse or some kind of control panel is mostly the way to interact with the virtual world. While this is satisfying in many cases for an application running on a standard monitor, it is quite often not the desired way for interaction on a large-scale screen or on a virtual table. At least such a conventional way of interaction often does not take advantage of the possibilities to support the immersion of the user in the application.

New and old interaction techniques for VR are studied a lot and can be classified as proposed by Mine [8] into four different categories: movement, selection, manipulation and scaling. The interaction technique based on interaction spheres presented in this paper does fit very well into the categories movement and selection; nevertheless it could also be used in some cases for manipulation and scaling. A closer look to previous work can be found in section 2. The general idea of interaction spheres or in a more general sense interaction regions is to define three-dimensional regions in the surrounding area of an interacting user. In this context, the space around a user, roughly limited by his arm length, is characterized here as surrounding area. Assuming that the user is centre of the world, this area is fix to the user.

The virtual set-up offers two different main forms of interaction: to trigger an event, when the user moves his hand into a certain interaction region or to set a status as long as the user holds his hand in one of the interaction regions. The first solution is typical for selections, the second one is typical for navigating through a virtual world. A more detailed specification of the complete concept is given in section 3.

Understanding the main idea of this interaction method, the question arises, how to determine and to track the pose of the user and his hand. In general our concept is independent from any tracking system, but in order to support a very high level of immersion of a VR application, a wireless solution is the intention behind this method. Therefore we use for the first implementation of the method our optical infrared tracking system EOS. Section 4. includes a small discussion on tracking systems for interaction and a description of EOS.

As an application of the interaction spheres model, a virtual flight through Frankfurt is presented in section 5. Results of our concept can be found in this section as well. The last section contains conclusions and describes further work.

2. Related Work

Interaction with virtual worlds is and was studied from many different viewpoints. To restrict the whole area of interaction we focus on work related to object selection and movement in virtual worlds.

An empirical evaluation of interaction techniques by Poupyrev, Weghorst, Bullinghurst and Ichikawa [13, 12] presented results of an experimental study of two generic egocentric interaction metaphors for object selection and manipulation in immersive virtual environments: virtual hand and virtual pointer. Besides the classical approach to realize a virtual hand by implementing a one-to-one map, other methods, using non-linear mappings, like the Go-Go technique [11] can be applied to. Another solution, offering an additional viewpoint on an extra pad is the Through-The-Lens Technique [16].

A very important class of interaction is the movement through a virtual world. Mine [8, 9] states, that mapping physical motion to virtual motion is very intuitive, but also very limited, even if the mapping is scaled. Basically the idea of the hand controlled flying speed is quite similar to the idea of interaction spheres: At least parts of the area around the user are subdivided and used for interaction by moving the hand into these regions. Evaluations of travel techniques and suggestions for a test environment for different interaction methods are presented by Bowman et al. [4, 3, 2]. Similarly to our intention to avoid a complicated learning procedure for the navigation, Fuhrman et al. [6] developed a head directed navigation system. Movement in virtual worlds is often better described as flying than walking. Pointing into the flying direction [14] is one possibility to fly in a virtual world. For steering, the rotation of the manual input device is common.

Comparing walking, walking-in-place and flying, Usoh et al. [18] realised, that above all for walking as form of interaction, a wireless tracking system is the one of the most important requirements. To overcome the problem to teach the user how to learn, locomotion interfaces were developed [19, 10, 5]. While such a kind of interaction is very attractive to allow walking as interaction technique, the expenses are the weak point in this case.

Obviously, there is not the one and only technique to interact with virtual worlds and the way to realise interaction depends strongly on the application. Nevertheless, summarizing the observations of the results from different approaches, some simple key points can be gained: Wireless solutions are more user-friendly than those with cables and the user should not be stressed by learning a non-intuitive interaction technique. Also

requiring a quite affordable solution, the technique presented in this paper is a suitable approach.

3. Interaction Spheres Model

The keynote of the interaction spheres concept is a discretisation of space. While in general interaction regions of arbitrary shape can be considered, we will restrict to the sphere as interaction volume. Before defining interaction spheres, we outline which coordinate systems and transformations are needed in this context. The user is located in a given world coordinate system. Attached to his body, we define a body coordinate system and a transformation $F_B^W : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ with

$$F_B^W(x^B) := R_B^W x^B + T_B^W = x^W, \tag{1}$$

that maps a point $x^B \in \mathbb{R}^3$ in body coordinates to a point $x^W \in \mathbb{R}^3$ in world coordinates by applying a rotation matrix $R_B^W \in SO(3)$ and a translation $T_B^W \in \mathbb{R}^3$ to it. Furthermore we define a hand coordinate system and a transformation $G_B^W : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ from hand to world coordinates analog to Equation 1. An overview of the coordinate systems is sketched in Figure 1. Resulting from this convention, T_B^W denotes the body translation and T_H^W the

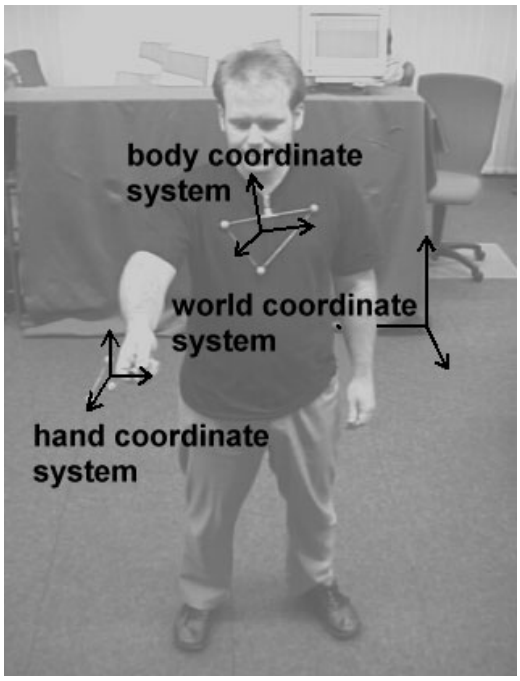

Figure 1: Visualization of the coordinate systems for the interaction spheres model.

hand translation in world coordinates. Since the interaction spheres will be defined fix relative to the body coordinate system, the hand (translation) T_H^B in body coordinates has to be determined by

$$T_H^B = R_W^B(T_H^W - T_B^W) \quad \text{and} \quad R_W^B := (R_B^W)^{-1}, \tag{2}$$

obtained from Equation 1.

Based on these conventions for coordinate systems and the transformations between them, we define interaction spheres as follows:

Definition. An **interaction sphere** S is a tuple

$$(X, d)$$

with a point $X \in \mathbb{R}^3$ and a radius $d \in \mathbb{R}$, whereas X is given in body coordinates. A point $x \in \mathbb{R}^3$ belongs to S, if $|X - x| \leq d$. Replacing $d \in \mathbb{R}$ by $D \subset \mathbb{R}^3$ would lead us to the more general definition of interaction regions.

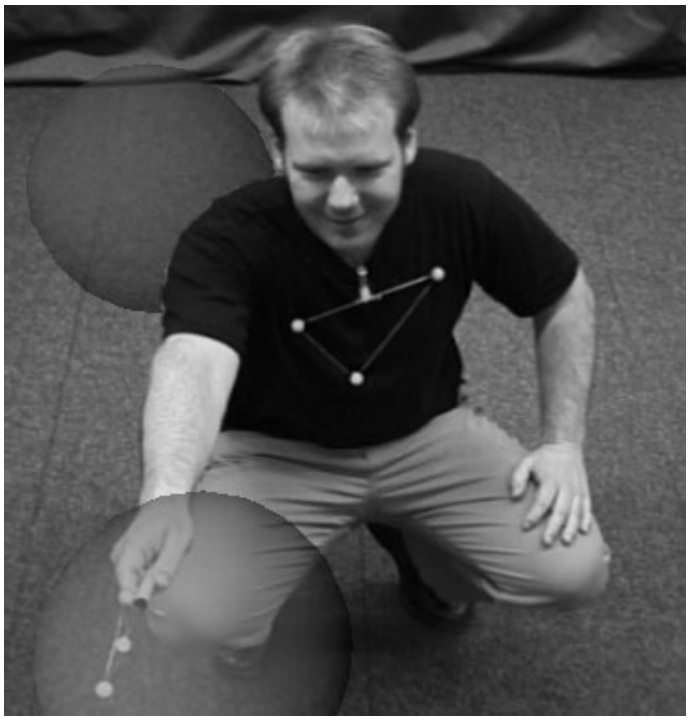

Figure 2: Visualisation of interaction spheres. Selection of one of two spheres.

Using only one interaction sphere is normally not satisfying for a functional interaction, therefore we define a finite number $n \in \mathbb{N}$ of interaction spheres S_n:

Definition. A set of n interaction spheres $S^n := S_1, ..., S_n, \;\; n \in \mathbb{N}$ is **valid**, if for each k and l with $1 \leq k < \; l \leq n$ and $k, l \in \mathbb{N}$

$$|X_k - X_l| > d_k + d_l,$$

with $S_k := (X_k, d_k)$ and $S_l := (X_l, d_l)$. By defining a valid set S^n of n interaction spheres we obtain a discretisation of $\mathbb{R}^3$ into $n + 1$ disjunctive subsets, whereas

$$S_0 := \mathbb{R}^3 \setminus \bigcup_{k=1}^{n} S_k.$$

For we will assign a status to each interaction sphere, S_0 is also called zero status. A visualisation of interaction spheres can be found in Figure 2.

While these definitions can be expressed quite simple in the formulas above, defining the interaction spheres for a real application is quite cumbersome, if the centres of the

interaction spheres would have to be estimated. Therefore a short and simple learning procedure is essential for the whole concept. The learning procedure is composed of following steps:

1. The number of interaction spheres and their radius' has to be defined. Typically for a simple navigation in a virtual room $n = 4$ would be chosen, see Figure 3. One sphere in front of the user to move forward, one on the left and one on the right side to turn around and a fourth e.g. on the shoulder to move backwards.
2. The procedure is started and the user has to hold his hand for few seconds in the position, where he wants to place the first sphere. If the hand was in one position up to a certain tolerance, this position is accepted as centre of the sphere.
3. The last step is repeated for each sphere.
4. After defining all sphere centres the interaction can be started.

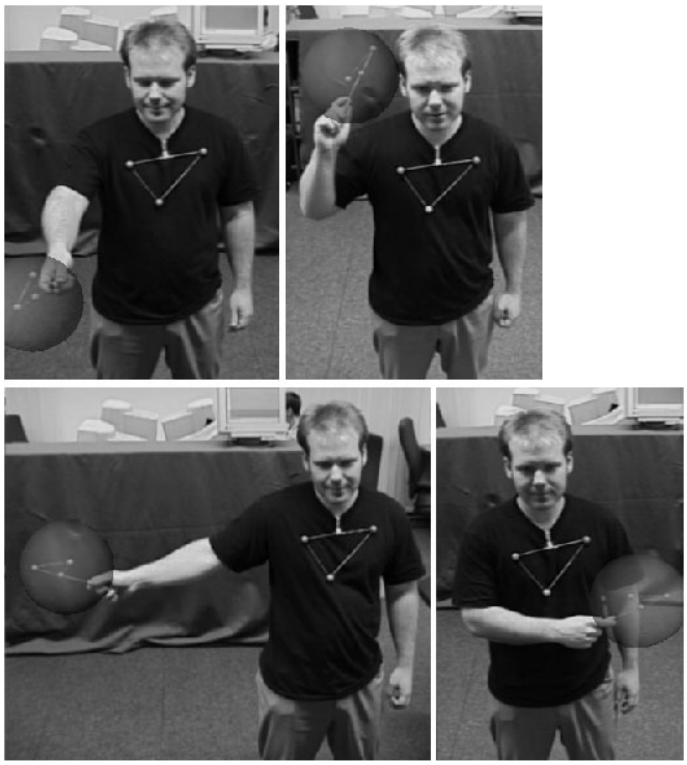

Figure 3: Learning four interaction spheres (in each image just the relevant sphere is visualised due to clarity). First row pointing foreward and backward, second row right and left.

This procedure is individually for each user and takes different anatomies of the users into account. Gestures like pointing forward also may vary from user to user and can be interpreted personalized for each user. After this learning procedure is completed, the interaction phase can start. Each interaction sphere is connected to a certain (hand) status, canonically the interaction sphere S_k will activate status k. Whether events are triggered if a certain hand status is set all the time or if status are changed only, depends on the application. A hand status k is set, if the hand is in the sphere S_k. Expressed formally, hand status $k \neq 0$ is set, if the hand translation T_H^B in body coordinates fulfills $|T_H^B - X_k| \leq d_k$. If $|T_H^B - X_k| > d_k$ for all $k \in \{1, ..., n\}$, hand status is set to zero.

This is the basic concept of the interaction spheres model. Before presenting some extensions of the basic concept - making the whole model more elaborate - the meaning of the body rotation R_B^W is discussed.

If the body rotation is tracked and taken into account, the spheres move around the user, even when he does not change his position, but only turns his body. Testing the interaction spheres model it turned out, that taking the body rotation not into account is more convenient to some users. In this case R_B^W is just the identity matrix E and equation 2 for determining the hand translation in body coordinates simply becomes

$$T_H^B = \underbrace{R_B^W}_{E}(T_H^W - T_B^W) = T_H^W - T_B^W .$$

Comparing interaction in 3d space with the familiar mouse interaction, the typical question arises: How to perform a mouse click. Some applications won't need a mouse click, but for many of them, it is necessary to have the possibility to trigger an event that depends on a status, such that the status does not have to be changed.

One possibility to solve this problem is multi-modal interaction, here especially the combination of the interaction spheres model with speech recognition. This is a suitable way to solve this problem, when speech recognition itself plays an important role in this application. But if it is needed only for triggering a mouse click event, the usage of this technology is quite excessive.

We designed other methods to implement what we called a virtual mouse click within the interaction spheres model. The first solution is an in-out-in movement with the hand. To make for example a mouse click in status k, this means following process:

1. The hand has to be in status k, or more precisely in interaction sphere S_k.
2. The hand is moved out of the interaction sphere; status is zero.
3. Within in a certain time interval, the hand has to be moved back into status k.

If any other status then k or zero is set during this time, the click fails. Obviously, the in-out-in method has two disadvantages: First, this interaction leads to a lot of fast movements, making the interaction less smooth and secondly, the in-out-in movement becomes very difficult, if spheres are defined quite closely.

An improved method to perform a virtual mouse click takes the hand rotation R_H^W into account. The idea is to measure strong changes in rotation of the hand. If an infrared optical tracking system is used, as described in section 4.1., the rotation of the hand corresponds to the rotation of the hand device. A virtual mouse click in status k by rotating the hand is defined as follows:

1. The hand has to be in status k during the whole process.
2. The hand rotation R_B^H in body coordinate system has to be determined by solving

$$R_H^B = (R_B^W)^{-1} R_H^W .$$

3. The variation of two successive rotations has to be measured, and if it exceeds a certain threshold, the click is accepted.

Measuring the variation of two rotations has to be expressed more clearly. It can and has to be done differently, depending on the application, in case of the usage of the already

mentioned infrared optical tracking system and a hand device, consisting of three markers, it can be described easily:

The three markers define the hand coordinate system. Two of them are placed on the axis, which will be the rotation axis for the virtual mouse click. Knowing, that the rotation matrix R_H^B consists column by column of the basis vectors of the hand coordinate system in body coordinates, changes in the three axes can be considered separately. While one axis stays in case of a virtual mouse click constant up to a certain threshold, the other ones rotate around the first axis and the angle between the rotated axis has to extend a certain threshold to trigger the event. Other methods based on rotation presentations as quaternions or axis and radians are considerable as well.

Another way to use the hand rotation for interaction is to control speed by the hand rotation within a navigation application. For example turning the hand to the right could mean to move or turn around faster and turning the hand to the left to slow down.

Up to now, the only way to trigger events was caused by hand translation or rotation relative to the body of the user. To include the position T_B^W of the body in this concept, the space is again split into different parts, but this time fix relative to the world coordinate system and a so called body status is set, depending on user's body's position. If for example a plane parallel to the ground is defined, it can be decided, if the user is standing or kneeling, depending if the reference for the body is above or below the plane. In combination with the hand status, such a simple extension doubles the number of possible events. Depending on the application more complicated regions for the body can be defined.

A further possible extension would be the usage of both hands for interaction. They can share one set of interaction spheres or they can both have their own set of interaction spheres, such that spheres from different sets even can intersect.

4. Tracking Systems

Though the interaction spheres model relies on operating together with a tracking system, it is independent of any kind of the tracking system. The only condition is, that it has two 6 DOF sensors and is submitting translation and rotation of body and hand. The simplicity of learning and interacting using this concept should be continued, when choosing a suitable tracking system. Therefore electro-magnetic tracking systems take the disadvantage, that they have sensors directly connected to a computer by cables, reducing the freedom of action of the user. Furthermore those systems are susceptible to interferences.

In contrast, optical tracking systems allow a maximum of freedom of action to the user, because the devices to be tracked are wireless. This allows an easy and intuitive manipulation of virtual worlds, even for first-time users.

4.1. Infrared Tracking

We use our tracking system EOS for implementing and proving the concept of controlling virtual worlds with interaction spheres. EOS is an optical tracking system, based on a stereo reconstruction of rigid bodies from grey scale images. The system is set up of two progressive scan cameras synchronised by a frame grabber card in a standard PC. The

cameras are equipped with infrared lenses to block out the visible light and the scene is illuminated with infrared light beamers attached to the cameras, see Figure 4. EOS is able to detect and to track several different models, consisting of retro-reflective markers to determine the centres of gravity of the devices, simultaneously. Therefore a simple intensity based segmentation algorithm is used to separate regions containing markers from the background.

Figure 4: Selection of interaction sphere 2 (infrared light filtered image from EOS). Visualisation of models and spheres are painted directly on the camera image.

The described hand and body devices to track are models consisting of three reflecting spheres with measured distances. Devices of three or more markers allow not only the determination of the 3d position but also of the orientation of the model, which is necessary to calculate as well the rotation of the hand for mouse clicking events as the body rotation, as described in the previous section.

For the pose estimation of the models, the stereo system has to be calibrated. We use a self-calibration method as described by Azarbayejani[1], that requires just little technical knowledge on user's side. During the calibration phase the user only has to sway a single marker device for a few seconds to record measurement data. The outcomes of this process are two corresponding clouds of 2d points in image planes representing a set of points in 3d space.

The principle of the calibration method is a minimization of the sum of residuals of a transformation in a least-squares sense, which takes both extrinsic (translation and rotation of the camera) and intrinsic (such as focal length and lens distortion) parameters into account. The residual of two points in corresponding images is calculated by back projecting the point from the left camera image with an estimated depth into 3d space.

This new 3d point is projected to the right camera image and the residual is calculated as the Euclidean distance between the projected point and the measured centre of

gravity from the segmentation process. Variation of the extrinsic and intrinsic parameters of the cameras leads to a diminution of the sum of residuals over several iterations. We use the Levenberg-Marquardt method [7] to find the minimum of this residual function. This approach takes advantage of a wide flexibility in modifying the function to be minimized. It allows an easy switching between different calibration functions, e.g. if only intrinsic camera parameters need to be determined, while the extrinsic parameters are already known.

The calibrated system allows to estimate 3d coordinates of corresponding image points, if two points P_1 and P_2 in the left and right camera image are known, after having them transformed depending on lens distortion, scaling and a shifted principle point [17]. Ideally the rays through the camera centre point C_1 and P_1 and through the camera centre point C_2 and P_2 should intersect. Due to image discretisation and calibration errors, this normally does not occur. A good approximation is the midpoint P of the shortest connection line between the two rays. Let s_1 be the direction of the ray through C_1 and P_1 and s_2 the direction of the ray through C_2 and P_2. Ray 1 is then given by

$$C_1 + t \cdot s_1, \quad t > 0,$$

with $s_1 = P_1 - C_1$. Calculating the normal vector $n_1 = s_1 \times (s_1 \times s_2)$ of a plane containing ray 1, the intersection of this plane with ray 2 is given by

$$n_1 \cdot (C_2 + t \cdot s_2 - C_1) = 0.$$

A second point can be evaluated by intersecting a plane containing ray 2 with ray 1. The midpoint P of both intersection points then obtains the estimation of the reconstructed 3d coordinates.

Accordingly the model detection in EOS is realised by searching estimated 3d points, whose distances are known from measuring the model once. For the 3d point estimation the centres of gravity of the segmented regions are used weighted by their intensities. The matching of the centre points between left and right images is solved by applying epipolar constraint up to a certain threshold, which reduces the search area for a corresponding point to a small tube around the epipolar line. This increases the operational speed of the system immensly, such that EOS is tracking several devices in real time with up to 25 frames per second, limited only by the technical restrictions of the frame grabber card.

5. Application and Results

As an application of the interaction spheres model we have chosen a virtual model of Frankfurt in Germany. The user's body and hand is tracked by EOS, sending the triggered events to a rendering system via a network connection. The virtual world is presented on a large-scale screen with side lengths of 1.4 by 1 meters approximately. Therewith the user is put in a position to take a virtual flight through Frankfurt, see Figure 5. He shall be able not only to navigate within the world, but also to mark points of interest for later use by performing mouse click events.

Our tracking system uses two different models to differentiate between body and hand: The body device with side lengths of 137, 190 and 210 mm is attached to the

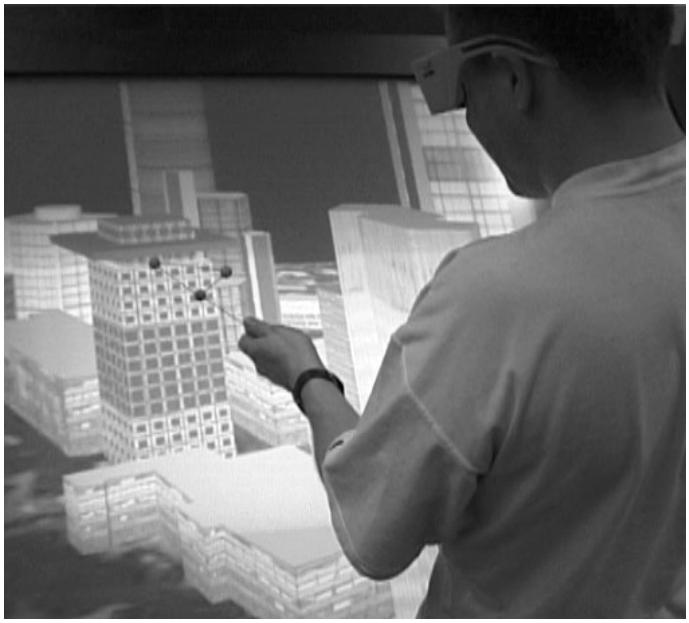

Figure 5: The interaction spheres model enables a flight through virtual Frankfurt.

collar of the user. The hand device with side lengths of 48, 82 and 110 mm is equipped with a handle, making interaction comfortable. Figure 4 shows both devices visualized as overlaid triangles with their centres of gravity.

Starting the application the user is asked to define four different interaction spheres in a learning procedure. The user places the spheres, where he wants to have forward, right, backward and left according to his perception. Holding the hand device for about three seconds in one place defines the interaction sphere. If the user sets two spheres, that would intersect, the system refuses the definition of the second sphere. This procedure takes about 20 seconds. After finishing the learning procedure, the interaction phase starts automatically. If the user wants to redefine the spheres during the application, he can do this, whenever he wants to.

Now the user starts his flight through virtual Frankfurt from a certain distance and controls the flight by moving forward, when the device is put into sphere one, turns to the right when he intersects with sphere two and analogous to the left, when intersecting with sphere three. Sphere four allows him to move backwards. It is intuitively clear, that this is not like the interaction for a real flight, where e.g. moving backwards would be impossible. Rotating the hand device during the flight allows storing the actual viewpoint to the scene without interrupting the current flight, such that this viewpoint can be reconstructed at a later date.

One of the main aspects of the evaluation of the interaction spheres model is the easy handling of the learning procedure. The system was tested by several people as well with knowledge and experience in using tracking systems and manipulating virtual worlds as people, who were new to these kinds of techniques. No one had major difficulties in understanding and accepting the new techniques, both the learning procedure and the moving within the virtual world by triggering events. All finished the learning phase within less than one minute (up to 20 seconds for some people experienced in VR

technologies) after a short introduction, how to use the system.

The most frequently occurring problem during the training phase was the definition of the interaction sphere, which lies on the opposite side of the body in comparison with the acting hand. Due to our camera setup a right-hander often covered parts of the body attached model, when pointing to the left and vice versa.

The completely wireless interaction verifies, that in combination with our optical tracking system EOS, we have an appropriate system at hand to make the idea of controlling virtual worlds with interactions spheres accessible to a variety of users.

6. Conclusion and Future Work

The concept of the interaction spheres model is quite simple in terms of theory and implementation. On the other side, it is a very good mapping from fuzzy terms like ”pointing forward”, interpreted from every user slightly different, to an exact set of events. The included learning procedure is essential for the implementation, which is very flexible, but not too complicated even for a first-time user, as the results in the previous section show. However, this kind of interaction is not thought for VR applications presented on standard monitors, but for large-scale screens.

The interaction spheres model, as described here, is independent from any tracking system with the minimum requirement of two 6 DOF sensors for translation and rotation of body and hand. Our implementation is based on an infrared optical tracking system, with the big advantage in contrast to e.g. electro-magnetic trackers, that this technology is wireless. But there is still the drawback that active or passive markers have to be used, i.e. the user has to hold a device in his hand. Therefore using tracking systems, based on gesture recognition, like presented in [15] is a planned step to make the interaction even more intuitive.

While the basic concept with only few interaction spheres, two different body status only and mouse click by rotation is implemented and tested well, the step to more complicated configurations, maybe also including interaction regions for the feet, still has to be done, to prove the benefit of this concept as a manifold interaction.

References

[1] A. Azarbayejani, A. Pentland. Camera self-calibration from one point correspondence. *Media Lab Technical Report* 341, 1995.

[2] D. Bowman, L. Hodges. Formalizing the Design, Evaluation, and Application of Interaction techniques for Immersive Virtual Environments. *The Journal of Visual Languages and Computing*, vol. 10, no.1, pp. 37-53, 1999.

[3] D. Bowman, D. Johnson, L. and Hodges. Testbed Evaluation of Immersive Virtual Environments. *Presence: Teleoperators and Virtual Environments*, vol. 10, no. 1, pp. 75-95, 2001.

[4] D. Bowman, D. Koller, L. F. Hodges. Travel in Immersive Virtual Environment: an Evaluation of Viewpoint Motion Control Techniques In *Proceedings of IEEE VRAIS'97*, 45-52, 1997.

[5] R. P. Darken, W. R. Cockayne, D. Carmein. The Omni-Directional Treadmill: A Locomotion Device for Virtual Worlds. *Proceedings of UIST 97*, 1997, pp.213-221.

[6] A. Fuhrmann, D. Schmalstieg, M. Gervautz. Strolling through Cyberspace with Your Hands in Your Pockets: head Directed Navigation in Virtual Environments. In *Virtual Environments'98 (Proceedings of the 4th EUROGRAPHICS Workshop on Virtual Environments)*, Springer-Verlag, 216-227, 1998.

[7] D.W. Marquardt. *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, pp. 431-441, 1963.

[8] M. R. Mine. Virtual Environment Interaction Techniques. UNC Chapel Hill CS Dept., *Technical Report TR95-018* (1995)

[9] M. R. Mine, F. P. Brooks Jr., C. Sequin Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction. In *Proceedings of SIGGRAPH 97, Los Angeles, CA* (1997)

[10] H. Noma, T. Miyasato. Design for Locomotion Interface in a Large Scale Virtual Environment. ATLAS: ATR Locomotion Interface for Active Self Motion. *Proceedings of ASME-DSC-Vol.64*, 1998, pp.111-118.

[11] I. Poupyrev, M. Billinghurst, S. Weghorst and T. Ichikawa. Go-Go Interaction Technique: Non-Linear Mapping for Direction Manipulation in VR. *Proceedings of UIST 96*, 1996, pp. 79-80.

[12] I. Poupyrev, T. Ichikawa. Manipulationg Objects in Virtual Worlds: Categorization and Empirical Evaluation of Interaction Techniques. *Journal of Visual Languages and Computing* 10(1), 1999, pp. 19-35.

[13] I. Poupyrev, S. Weghorst, M. Billinghurst and T. Ichikawa. Egoncentric Object Manipulation in Virtual Environments: Empirical Evaluation of Interaction Techniques, *Computer Graphics Forum, EUROGRAPHICS'98 issue*, 17(3), 1998. pp.41-52.

[14] W. Robinett and R. Holloway. Implementation of Flying, Scaling and Grabbing in Virtual Worlds. *Proceedings of 1992 Symposium on Interactive 3D Graphics*, Cambridge MA, March 1992. pp 43-52.

[15] V. Sa, C. Malerczyk, M. Schnaider. Vision Based Interaction within a Multimodal Framework. *10th Potuguese Computer Graphics Meeting*, Lisbon, ISCTE, 2001.

[16] S.L. Stoev, D. Schmalstieg, W. Strasser. Two-handed through-the-lens-techniques for navigation in virtual environments. *Proceedings of the Eurographics Workshop on Virtual Environments*, 16-18 May 2001.

[17] R.Y. Tsai. An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision. *Proceedings CVPR '86*, Miami Beach, Florida, pages 364–374. IEEE, June 1986.

[18] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, F. P. Brooks, Jr. Walking¿Walking-in-place¿Flying, in Virtual Environments. *Proceedings of SIGGRAPH 99* (Los Angeles, CA, 1999), Computer Graphics Proceedings, Annual Conference Series, 1999, 359-364.

[19] H. Yano, H. Noma, H. Iwata, T. Miyasato. Shared Walk Environment Using Locomotion Interfaces. *Proceedings of ACM CSCW200*, 2000.