Navidget for Immersive Virtual Environments

Sebastian Knödel* Martin Hachet† Pascal Guitton‡

INRIA - Université de Bordeaux

Abstract

We present a novel interaction technique called *Immersive Navidget* for navigation in immersive virtual environments (VE), see Figure 1. This technique, based on Navidget, allows fast and easy 3D camera positioning from simple controls. In this paper, we focus on the technical issues that are induced by the VR setups and we propose solutions to adapt Navidget to the immersive context. We show that this new approach has many advantages for navigation in immersive VEs.

CR Categories: I.3.7 [Computer Graphics]: Three Dimensional Graphics and Realism—Artificial, augmented and virtual realities;

Keywords: Virtual Reality, Interaction Techniques

1 Introduction

Controlling the position and the orientation of the viewpoint is a common task in immersive virtual environment (VE). For the focus areas, which are close to the user, one can adjust the view by moving physically around the virtual objects. On the other hand, the same task becomes much harder to perform as soon as the areas of interest are not directly reachable. This requires the user to move the current viewpoint in the 3D environment by way of a travel technique. Examples of such techniques are the well-known flying technique and gaze-directed steering technique[Mine 1995] in which the users move in the direction they are looking at. The movements applied by the users are directly incorporated in the camera motions. The problem with such direct-controlled techniques is that the users have to adjust the position and the orientation of the camera in real-time, which requires attention and expertise. Moreover, this direct approach becomes inefficient as soon as long distances have to be traveled.

We propose a new approach for camera positioning in VE. This approach called *Immersive Navidget* is adapted from Navidget [Hachet et al. 2008]. It allows the users to easily indicate what they want to see and how they want to see it. The users can comfortably specify the target position and orientation of the camera before executing any movement in the VE. Then the viewpoint automatically moves towards the destination through a precomputed smooth trajectory.

Navidget has initially been designed for desktop and pen-based systems. This 3D user interface is totally based on 2D inputs. It allows fast and easy interactive camera positioning from simple gestures. As a result, it is appropriate for a wide variety of visualization systems, from small handheld devices to large interactive displays. On the other hand, it does not support 3D immersive virtual environments with stereo projection and tracking technologies. In this paper we explore the use of the Navidget philosophy in immersive VEs. We state that the benefit of this technique may help the users in their interactive tasks.

In Section 2, we remind the main functionalities of Navidget as well as the main related work. We highlight the problems resulting from the immersive setup for Navidget in Section 3 and we propose adapted solutions. In Section 4, we give directions to future work and, finally, we conclude.

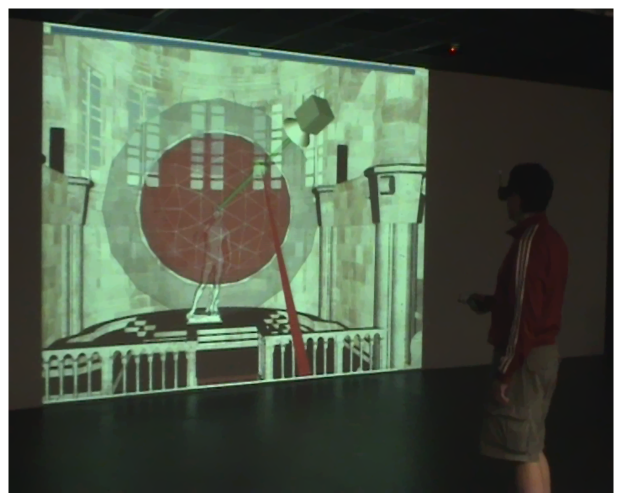


Figure 1: The *Immersive Navidget* for camera positioning in immersive Virtual Environments.

2 Previous Work

2.1 Navidget

Our work is mainly based on Navidget[Hachet et al. 2008], which is an interaction technique for camera positioning in 3D environments. Navidget derives from the Point-of-Interest (PoI) technique introduced by [Mackinlay et al. 1990]. Compared to a standard PoI technique where the user only selects a target, Navidget allows you to control the distance to the target as well as the viewing direction with which you want to visualize it. This is possible by way of adapted controls and feedback. To control the distance to the target, the user encircles the area he or she wants to focus on. The size of the focus area can be adjusted by way of some size actuators,

*e-mail:knoedel@labri.fr
†e-mail: hachet@labri.fr
‡e-mail:guitton@labri.fr

too. A widget which is mainly composed of a half-sphere allows the user to adjust the viewing direction. By moving a 3D cursor on the surface of this half-sphere, the user controls the position of a virtual camera around the focus area. Once, the virtual camera is positioned, a smooth trajectory from the current viewpoint to the selected destination is computed and the corresponding motion starts. Figure 2 illustrates Navidget for pen-based systems.

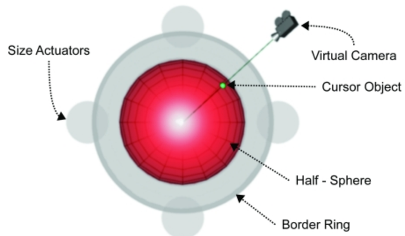


Figure 2: Navidget

In addition to the described functionalities, Navidget integrates a border ring which allows fast access to perpendicular views. Navidget allows you to spin the half-sphere to reach the backside of the focus area, too. This is done by moving the pen out of the widget and back in.

2.2 Related Work

Other techniques like Unicam [Zeleznik and Forsberg 1999], Hovercam[Khan et al. 2005] and StyleCam[Burtnyk et al. 2002] support a 2D mouse or a stylus input device that is based on gestural interaction and use surface information and predetermined trajectories to control the camera position and orientation especially to navigate around 3D objects at close proximity.

Since control of the virtual camera in 3D environments requires at least 6 degrees of freedom (DOF), [Ware and Osborne 1990] propose the scene-in-hand, eyeball-in-hand, and flying vehicle metaphors to control virtual cameras in VEs directly from 6-DOF input devices.

Several approaches have been developed for traveling through VEs by defining the camera's point of view. One possibility is the WIM metaphor, presented by [Stoakley et al. 1995]. It naturally offers multiple points of view to place a camera in the scene. The users can change the view point in the miniature and will be sent to the selected location. [Pierce and Pausch 2004] present an interface for large-scale VEs that consists of a combination of visible landmarks and place representations. Furthermore, the user can select a landmark to travel directly or relatively to the desired place. In contrast to that approach we do not propose any location to the user where he can travel to, but we let him choose by himself where he wants to position the camera. For traveling through large multi scale environments [Kopper et al. 2006] proposes a target-based navigation technique using a magnifying glass metaphor. In this case the user is automatically moved to the center of a selected level of scale (LoS).

As [Bowman et al. 1997] stated in his categorization of motion control and travel in immersive VEs, pointing techniques like the virtual ray are a good choice for interaction in immersive VEs due to the fact that they provide accuracy and speed at the same time. Therefore we decided to use the standard virtual ray [Hinckley et al. 1994][Mine 1995] to interact in the scene .

3 *Immersive Navidget* for Virtual Environments

Immersive Navidget follows the Navidget philosophy, where complex trajectories can be obtained from simple controls. However, what has been designed for pen-based and desktop systems is not well suited to immersive VE configurations anymore. Consequently, we have adapted Navidget to the VR specificities. In our setup, the user is able to control a virtual ray by way of a tracked flying mouse. In addition to the 6DOF provided by the device, the user can trigger mouse buttons and wheel.

We divided *Immersive Navidget* into two different states. In the first state, the users indicate what they want to focus on. In the second state, they adjust the viewing direction. The users can easily switch from one state to another by pressing a mouse button.

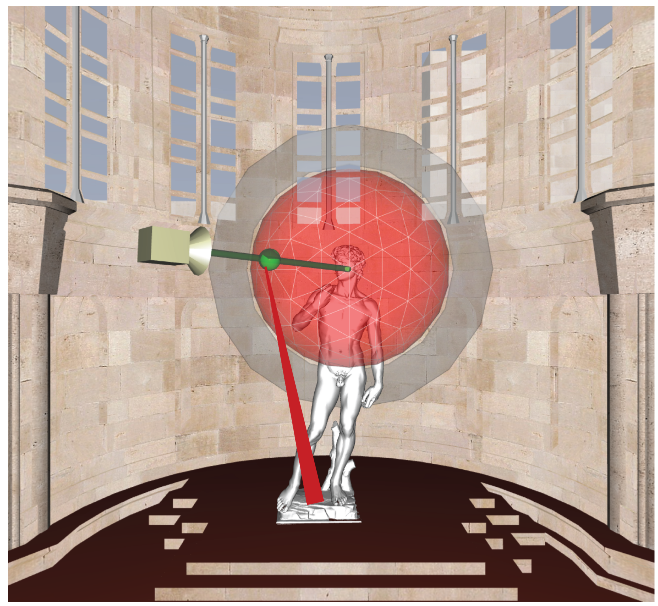
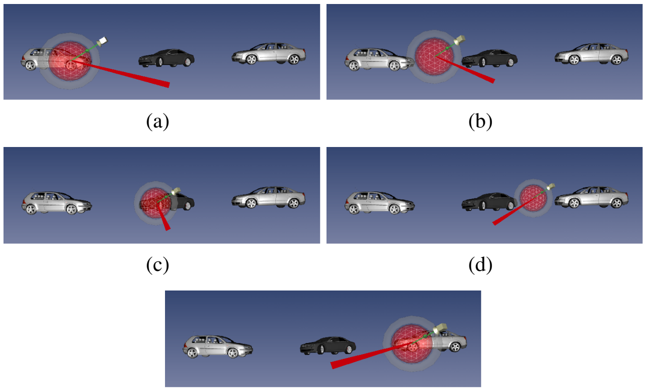


Figure 3: *Immersive Navidget* operated with a virtual ray.

3.1 Positioning the focus area

In the first state, the user is able to position the focus area by directly moving the virtual ray in the scene. The half-sphere is attached to the virtual ray. If the ray intersects an object during the dragging movement the half-sphere is placed at the intersected position, as illustrated in Figure 3. The Navidget "snaps" onto the intersected surface. If no object is intersected, the sphere sticks with the ray within the distance of the last intersection. A similar method of a so called "sticky ray" was presented by [Steinicke et al. 2004]. If no intersection occurs the scroll wheel of the mouse can be used to move the half-sphere along the ray. Figure 4 and Figure 5 present an interaction sequence where the user drags the sphere through the scene and attaches it onto different objects.

Picking a small object with a ray can be a challenging task. In some cases the virtual ray cannot be positioned accurately enough to intersect an object, as for example if objects are very small or far away. To overcome this problem an aperture approach can be used as presented in [Forsberg et al. 1996]. A conical volume is created starting from the users point of view in ray direction. The closest point to the camera of all objects that intersect the cone volume defines the focus depth.



(a) (b) (c) (d) (e)

Figure 4: Positiong the *Immersive Navidget* on the left car (a) and moving it through 3D space (b),(d). If the ray intersects another car the *Immersive Navidget* will be positioned at the point of intersection (c),(e).

3.2 Controlling the virtual camera

In the second state, the users are able to specify how they want to visualize the focus area. The scroll wheel allows the users to modify the size of the half-sphere, so the distance to the target. We do not use the size actuators anymore since they had been introduced in the original Navidget system due to the fact that the pen-based approach does not support any mouse wheel or button input. Similarly, the circling metaphor, which was very well suited for pen-based devices, is not adapted anymore. Indeed circling with a virtual ray in immersive VEs is not convenient from an ergonomic point of view. Moreover, the display of the drawn circle on the screen plane affects the immersion of the user.

To modify the position of the virtual camera around the focus area, the user moves the 3D cursor on the surface of the half-sphere. This can be done very accurately on pen-based or desktop computers. On the other hand, accuracy is a crucial problem in immersive VEs as soon as the projected size of the sphere is small. Indeed, a small rotation of the user's hand can induce a large modification of the virtual camera location. To overcome that problem we propose two solutions. The first one uses the absolute orientation of the tracked flying mouse. In the second approach, a second larger transparent half-sphere allows improving the accuracy of the technique.

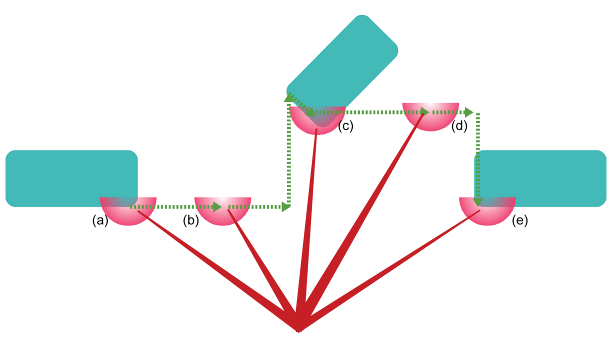


Figure 5: The scheme shows a birds eye view of the interaction sequence presented in Figure 4(a) - (e). One can clearly see that the half-sphere stays in the same distance on the virtual ray until another intersection occurs.

In the first approach, the orientation of the virtual camera is directly mapped onto the orientation of the flying mouse. The advantage of this method is that resolution and precision are independent of the size of the sphere. Turning the device results in turning the camera around the sphere. Turning the device slightly more than 90°degrees and immediately turning back allows to spin the half-sphere to reach backside views.

In the second approach we use a large semi-transparent sphere superimposed over the initial half-sphere as presented in Figure 6. We maximize the size of the sphere to increase its surface. The larger the sphere's visible surface the easier and accurate is the positioning of virtual camera. This method is illustrated in Figure 7. Without the enveloping sphere the visible surface of the half-sphere can be small. Consequently small movements with the ray result in a large displacement of the virtual camera. An accurate positioning is thus very hard to achieve. Enveloping the half-sphere with a large semi-transparent sphere allows the user to position the camera more comfortably since its surface is bigger.

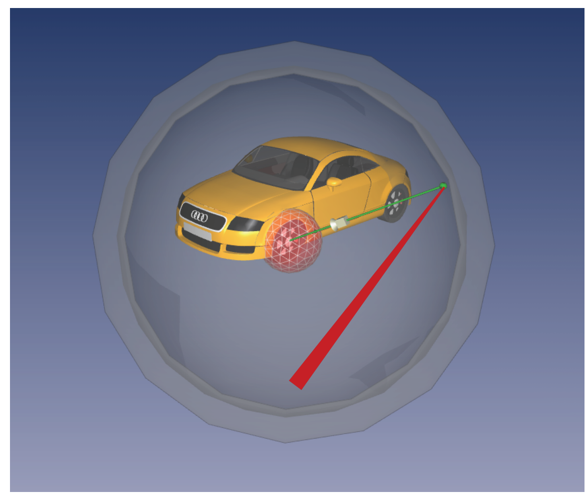


Figure 6: A Large transparent half-sphere around the *Immersive Navidget* initial half-sphere allows precise camera positioning.

By switching between the two states of *Immersive Navidget*, the users are able to easily position and control the virtual camera everywhere in the VEs. Consequently, they can quickly reach any part of the scene within a reduced mental and motor effort.

4 Beyond Camera Positioning

In addition to camera positioning tasks, the Navidget approach allows the users to perform several interaction tasks in immersive VEs.

One example is the observation of distant objects. By adding a preview window close to the virtual camera, the user can observe distant objects without changing his current position. This relates to the magic mirror presented by [GROSJEAN and COQUILLART 1999]. The use of a preview window can also be useful for accurate positioning before starting the motion along the pre-calculated trajectory.

The Navidget approach can be helpful for 3D pointing tasks, too. Indeed, positioning a 3D cursor in a VEs to highlight parts of the scene can be difficult. With *Immersive Navidget*, this task can be performed easily as Navidget intrinsically manages the positioning of a 3D cursor around a focus area.

Finally, the use of multiple *Immersive Navidgets* with associated preview windows opens new directions for immersive applications. For example, in the case of physical simulation processes, such an approach allows the user to define several focus areas to observe multiple parts of the scene.

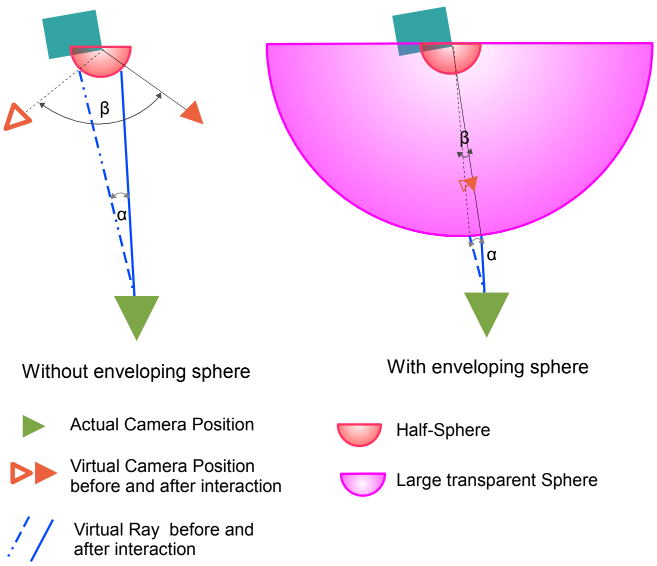


Figure 7: The left scheme shows that already small movements of the ray cause a large displacement of the virtual camera, because the half-sphere is very small. The scheme on the right shows the same interaction with the large semi-transparent sphere with a much smaller and consequently more accurate movement of the virtual camera

5 Conclusion and future work

With the direct camera-controlled techniques, users have to adjust their camera movements at anytime. With these techniques, they have to reach their target areas by themselves. On the opposite, *Immersive Navidget* allows the users to clearly indicate what they want to focus on. The camera motion is then automatically computed, which requires less mental effort, as well as less motor control. This allows exploring immersive VE in a new fashion.

In this paper, we have shown how the Navidget approach could be adapted to immersive VEs. We have proposed adapted controls and we have addressed problems that are directly linked to the immersive context. Now, the same approach can be used from mobile devices to immersive VEs.

One important additional work we have to do is to conduct a comprehensive user study. There, we have to prove that the *Immersive Navidget* is compatible to existing camera positioning control techniques for immersive VEs. Moreover there are still open questions how to provide the user with a suitable interface to define a certain point of view in a scene that contains many small and distant objects. There we have to develop a technique that goes beyond the classic aperture selection mechanism that may support gesture-based interaction, like circling.

References

BOWMAN, D. A., KOLLER, D., AND HODGES, L. F. 1997. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. *IEEE Proceedings of VRAIS'97*, 7, 45–52.

BURTNYK, N., KHAN, A., FITZMAURICE, G., BALAKRISHNAN, R., AND KURTENBACH, G., 2002. Stylecam: Interactive stylized 3d navigation using integrated spatial and temporal controls.

FORSBERG, A. S., HERNDON, K. P., AND ZELEZNIK, R. P. 1996. Aperture based selection for immersive virtual environments. In *ACM Symposium on User Interface Software and Technology*, 95–96.

GROSJEAN, J., AND COQUILLART, S., 1999. The magic mirror: A metaphor for assisting the exploration of virtual worlds.

HACHET, M., DÈCLE, F., KNÖDEL, S., AND GUITTON, P. 2008. Navidget for easy 3d camera positioning from 2d inputs. In *Proceedings of IEEE 3DUI - Symposium on 3D User Interfaces*. best paper award.

HINCKLEY, K., PAUSCH, R., GOBLE, J. C., AND KASSELL, N. F. 1994. A survey of design issues in spatial input. In *UIST '94: Proceedings of the 7th annual ACM symposium on User interface software and technology*, ACM, New York, NY, USA, 213–222.

KHAN, A., KOMALO, B., STAM, J., FITZMAURICE, G., AND KURTENBACH, G. 2005. Hovercam: interactive 3d navigation for proximal object inspection. In *I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games*, ACM, New York, NY, USA, 73–80.

KOPPER, R., NI, T., BOWMAN, D. A., AND PINHO, M. 2006. Design and evaluation of navigation techniques for multiscale virtual environments. In *VR '06: Proceedings of the IEEE conference on Virtual Reality*, IEEE Computer Society, Washington, DC, USA, 175–182.

MACKINLAY, J. D., CARD, S. K., AND ROBERTSON, G. G. 1990. Rapid controlled movement through a virtual 3d workspace. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 171–176.

MINE, M. R. 1995. Virtual environment interaction techniques. Tech. Rep. TR95-018, 4,.

PIERCE, J. S., AND PAUSCH, R. 2004. Navigation with place representations and visible landmarks. In *VR '04: Proceedings of the IEEE Virtual Reality 2004*, IEEE Computer Society, Washington, DC, USA, 173.

STEINICKE, F., ROPINSKI, T., AND HINRICHS, K. 2004. Object selection in virtual environments using an improved virtual pointer metaphor. In *Computer Vision and Graphics International Conference, ICCVG*.

STOAKLEY, R., CONWAY, M. J., AND PAUSCH, R. 1995. Virtual reality on a wim: interactive worlds in miniature. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 265–272.

WARE, C., AND OSBORNE, S. 1990. Exploration and virtual camera control in virtual three dimensional environments. In *SI3D '90: Proceedings of the 1990 symposium on Interactive 3D graphics*, ACM, New York, NY, USA, 175–183.

ZELEZNIK, R. C., AND FORSBERG, A. S. 1999. Unicam - 2d gestural camera controls for 3d environments. In *Symposium on Interactive 3D Graphics*, 169–173.