**M. Fjeld, N. Ironmonger, F. Voorhorst, M. Bichsel, & M. Rauterberg (in press): Camera control in a planar, graspable interface.
17th IASTED International Conference on Applied Informatics (AI'99), pp. 242-245.**

CAMERA CONTROL IN A PLANAR, GRASPABLE INTERFACE

MORTEN FJELD(1), NICHOLAS IRONMONGER(2), FRED VOORHORST(1),
MARTIN BICHSEL(2) & MATTHIAS RAUTERBERG(3)

(1) Institute for Hygiene and Applied Physiology (IHA)
ETH, Clausiusstr. 25, CH-8092 Zurich, Switzerland
phone: +41-1-632 3983, fax: +41-1-632 1173
www.fjeld.ch, morten@fjeld.ch

(2) Institute for Design and Construction Methods (IKB)
ETH, Tannenstr. 3, CH-8092 Zurich, Switzerland

(3) Center for Research on User-System Interaction (IPO)
TUE, Den Dolech 2, NL-5612 AZ Eindhoven, Netherlands

ABSTRACT

BUILD-IT is a planning tool based on computer vision technology, supporting complex planning and composition tasks. It allows a group of people, seated around a table, to interact with objects in a virtual scene using real bricks. A plan view of the scene is projected onto the table, where object manipulation takes place. Hence, manipulation and display are coincident. A perspective view is projected on the wall, controlled by a brick acting on a virtual camera. The camera requires control of *position*, *pan*, *tilt*, *zoom* and *roll* attributes. However, planar interaction with bricks provides only position and rotation information. The topic of this paper is how to bridge the gap between planar interaction and three dimensional (3D) camera control. This is done by introducing additional, *active* objects whose positional relation to the camera controls the above attributes.

KEYWORDS

Graspable interaction, camera control, augmented reality

INTRODUCTION

FIGURE 1: BUILD-IT offers a *plan view* for combined action and perception and a *side view* with a perspective of the situation.

BUILD-IT [1, 2] is an application that supports designers in the early design phase of floor-planning and configuration tasks. Originally, the application was designed to support providers of assembly lines and plants. However, potential uses range from visualisation and interior design to urban and city planning.

The system enables users, grouped around a table, to interact in a virtual scene, using real bricks to select and manipulate objects in the scene (Fig. 1). A *plan view* and a *storage space* are projected onto the table, a *side view* on the wall. The storage space allows users to create (delete) objects which are then placed in (removed from) the plan view. The side view offers a perspective of the same scene.

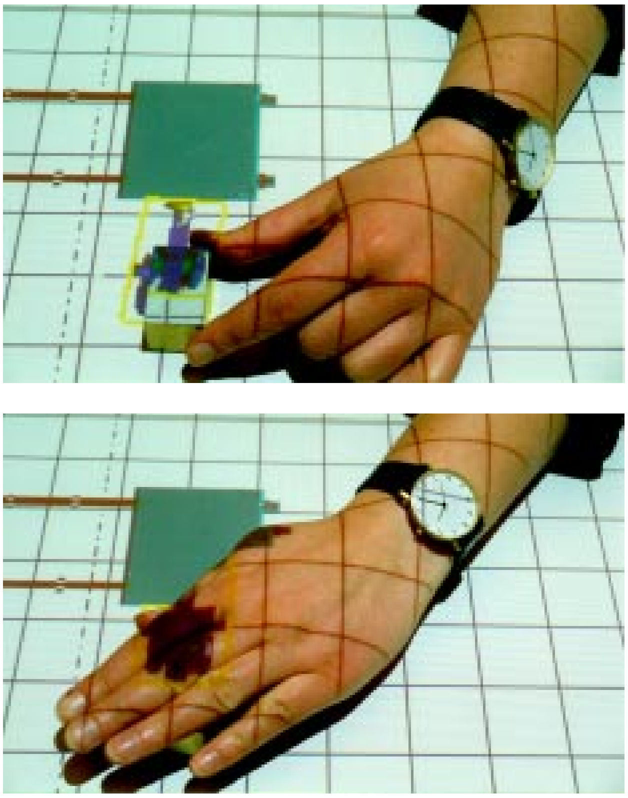

FIGURE 2: Positioning and rotation (top) and deselecting (bottom) of a projected object with a brick.

The basic characteristics of the systems are as follows (Fig. 2). Users select an object by putting a brick at the object position. The object can be positioned, rotated and fixed by simple brick manipulation. As soon as the brick is covered, the object is deselected and stays put. To allow for bi-manual interaction and for multiple user operation, the system supports multi-brick interaction. Graphical display is based on the class library MET++ [3]. 3D objects come from a Computer Aided Design system to BUILD-IT using Virtual Reality Modelling Language (VRML).

Some basic issues of two-dimensional, brick-based interaction were already explored [4]. The innovation of BUILD-IT, is that the objects are part of a 3D scene. Hence, the topic of this paper is how to bridge the gap between planar interaction and navigation in a 3D scene.

FROM SIMPLE CAMERA CONTROL TO THE EYECATCHER OBJECT

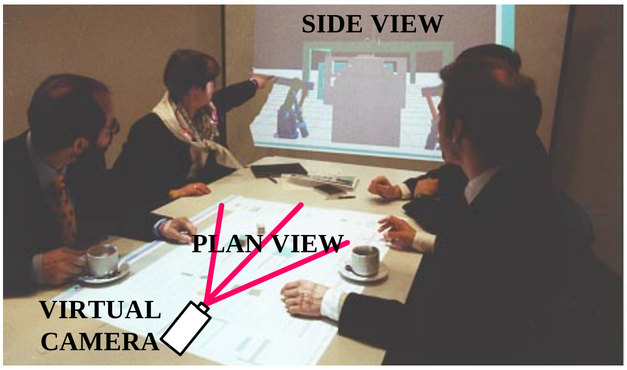

FIGURE 3: The camera object sets the side view. *Travelling* and *pan* are set by brick positioning and rotation.

A virtual camera, situated in the plan view, sets the side view (Fig. 3). Simple camera control can be performed using one brick only. Hence, *travelling* (lateral placement) is set by brick positioning, *pan* (orientation) by rotation of the brick. There can be one or many cameras in the plan view, but only one is effective at a time. The *effective* camera is the one that was manipulated most recently.

To offer users complete camera control, including *tilt* (slope of line of sight), *zoom* (field of view) and *roll* (rotation around line of sight) [5], it is necessary to overcome the limitations of planar interaction. These limitations are that a brick only provides its position and rotation in the x-y plane.

One strategy which was considered is the use of a specialised brick modelling a real object with added properties. This would require extending the properties sensed by the computer vision input. Since we want to explore software solutions, this approach was not pursued.

Instead the strategy used was to introduce *active*, virtual objects. Active objects feature *intelligent* behaviour and support *complex operations*. They do not have real-world analogues, but are *synthetic*. They only exist during the operation they support, so they are *transient*. In the following, we show an example of how the specific need for camera control led to an active object.

An active object, the EyeCatcher, was realised. With this object, camera attributes depend on the positional relation between camera and object. The EyeCatcher is offered in the storage space (Fig. 4). Using one, or two, instances of this object, it is possible to control the following attributes of the effective camera: *pan*, *tilt*, *zoom* and *roll* (Fig. 5). The pan angle controlled by an EyeCatcher overrides the pan angle set by simple camera control. EyeCatcher orientation has not been given any function so far.

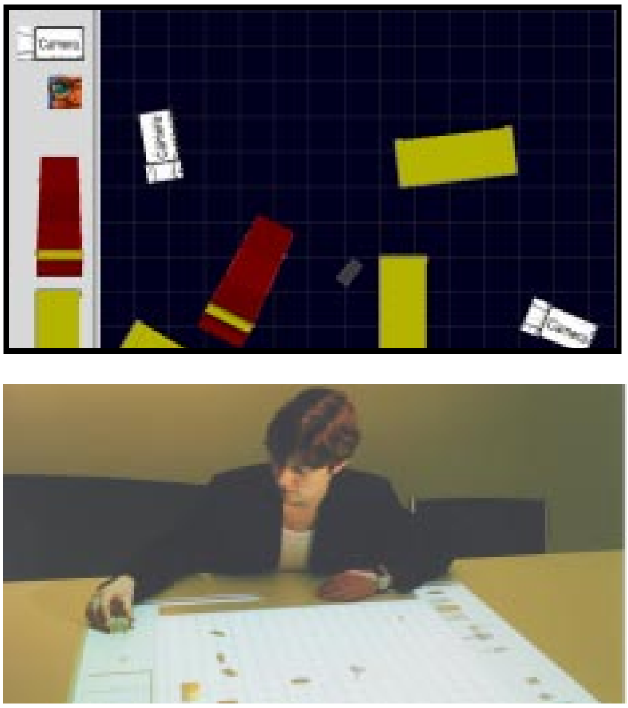

FIGURE 4: Camera, EyeCatcher and other objects are situated in the storage space, left, whereas the plan view with placed cameras and objects is to the right (top). New instances are created by picking an object and placing it into the plan view (bottom).

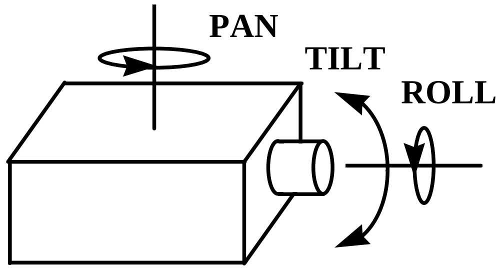

FIGURE 5: Using one or two instances of the EyeCatcher object makes it possible to set *pan*, *tilt*, *zoom* (not shown) and *roll*.

USING ONE EYECATCHER

The *pan* and the *tilt* angles (Fig. 6) are specified so that the EyeCatcher is kept in the centre of the side view (Fig. 7). Since the EyeCatcher normally is situated at *ground-level* in the virtual scene and the camera at *eye-level* (1.6 m above the ground), the camera will tilt downwards. What happens when an EyeCatcher is put onto other objects is discussed below.

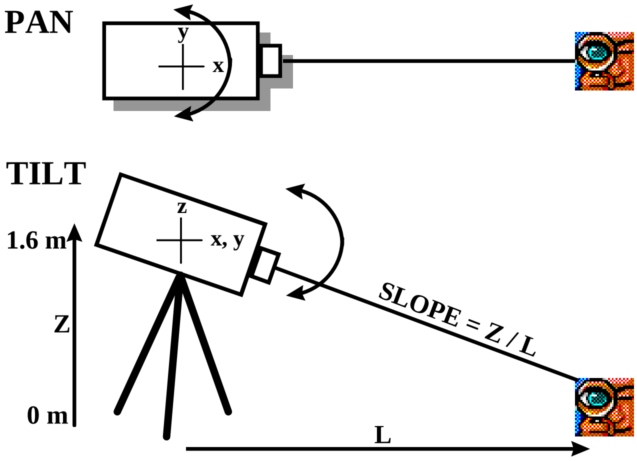

FIGURE 6: Schematic drawing of pan and tilt.

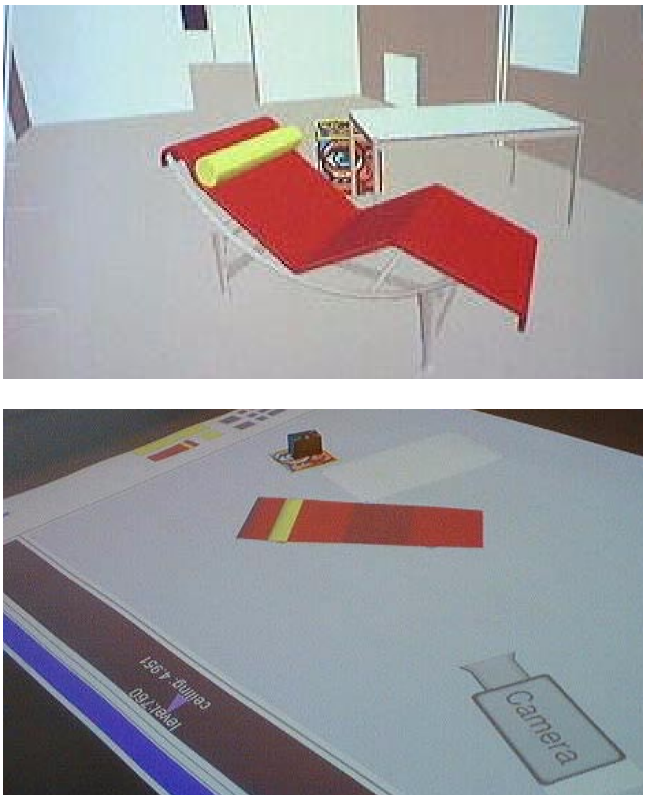

FIGURE 7: Using one EyeCatcher and one brick.
Side view (top) and plan view (bottom).

USING TWO EYECATCHERS

The *pan* and the *tilt* angles are specified so that the mid-point between the EyeCatchers is kept in the centre of the side view (Fig. 9). The *zoom* angle is specified so that the EyeCatchers are situated at the side view edges (Figs. 8 and 9).The *roll* angle is specified so that the EyeCatchers, as seen in the side view, are connected by a horizontal line.

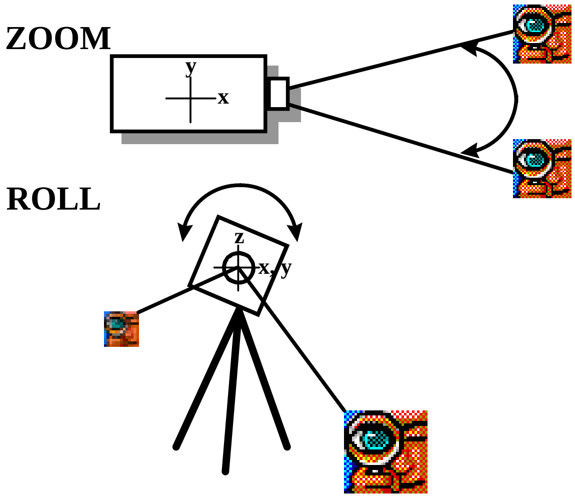

FIGURE 8: Schematic drawing of zoom and roll.

FIGURE 9: Using two EyeCatchers and two bricks.
Side view (top) and plan view (bottom).

PUTTING THE EYECATCHER ONTO OTHER OBJECTS

When putting the EyeCatcher onto another object (Fig. 10), it puts itself on top of this object. This effects the tilt and roll angles. Pan and zoom angles, however, are not influenced, as they are independent of EyeCatcher height.

FIGURE 10: Putting EyeCatcher on other objects.

DEACTIVATING THE EYECATCHER

When an EyeCatcher is deactivated it also disappears (Fig. 11); it is *transient*. However, all the attributes of the camera remain the same. If an other EyeCatcher still remains and is used any further, the situation will adjust to the case of one EyeCatcher. If no EyeCatcher remains, and a brick sits on the camera, then this brick resumes its original control of the camera.

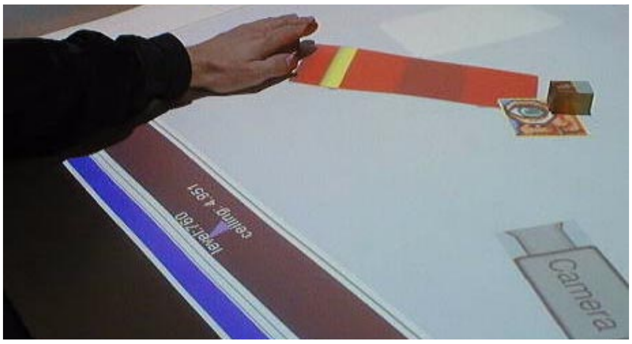

FIGURE 11: Deactivating one EyeCatcher, a second remains.

DISCUSSION

Of particular interest is the fact that the *virtual camera attributes* have some analogues in real-world camera handling and human perception. Travelling, pan, tilt and roll have direct analogues in human perception, zoom has little. Being more specific, tilt and roll correspond to different head movements. However, when performed, tilt and roll are compensated in reference to some world frame or horizon. Hence, there are issues to resolve about the sense of controlling such attributes which are, at the same time, compensated by humans. In implementation, the same issues arose.

These observations indicate that design of the camera control should largely depend on basic characteristics of human perception. The choice of putting the camera at a *human* eye-level may be well justified. There seems to be a need for usability studies where alternative control strategies of camera attributes are explored in order to support efficient task-solving behaviour.

CONCLUSION

A concept for 3D camera control based on active objects was realised. Active objects prove an advantage over specialised bricks, since they require no extended input sensing. However, the implementation raises questions about the sense of controlling camera attributes that are compensated in human perception. Finally, it may be of interest to generalise the suggested concept for 3D control to other domains than camera control.

REFERENCES

[1] M. Fjeld, M. Bichsel and M. Rauterberg, BUILD-IT: An Intuitive Design Tool Based on Direct Object Manipulation. *In I. Wachsmut & M. Frölich (eds.) Gesture and Sign Language in Human-Computer Interaction*, Lecture Notes in Artificial Intelligence, Vol. 1371 (Berlin: Springer-Verlag, 1998), 297-308.

[2] M. Fjeld, K. Lauche, S. Dierssen, M. Bichsel and M. Rauterberg, BUILD-IT, A Brick-based integral Solution Supporting Multidisciplinary Design Tasks. *In A. Sutcliffe, J. Ziegler & P. Johnson (eds.) Designing Effective and Usable Multimedia Systems* (Boston: Kluwer Academic Publishers, 1998), 131-142.

[3] P. Ackermann, *Developing Object-Oriented Multimedia Software Based on the MET++ Application Framework* (Heidelberg: dpunkt Verlag für digitale Technologie, 1996).

[4] G. W. Fitzmaurice, *Graspable User Interfaces*. PhD at the University of Toronto (Toronto: 1996), Ch. 7. www.dgp.utoronto.ca/people/GeorgeFitzmaurice/thesis/Thesis.gf.html

[5] D. Arijon, *Grammar of the film language*.
(Los Angeles: Silman-James Press, 1991).