

A Camera-Based Interface for Interaction with Mobile Handheld Computers

Martin Hachet, Joachim Pouderoux, Pascal Guitton

▶ **To cite this version:**

Martin Hachet, Joachim Pouderoux, Pascal Guitton. A Camera-Based Interface for Interaction with Mobile Handheld Computers. I3D05 - Symposium on Interactive 3D Graphics and Games, 2005, United States. pp.65 - 72. hal-00308240

HAL Id: hal-00308240
https://hal.science/hal-00308240

Submitted on 30 Jul 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Camera-Based Interface for Interaction with Mobile Handheld Computers

Martin Hachet* Joachim Pouderoux† Pascal Guitton‡
IPARLA Project, LaBRI - INRIA, University of Bordeaux

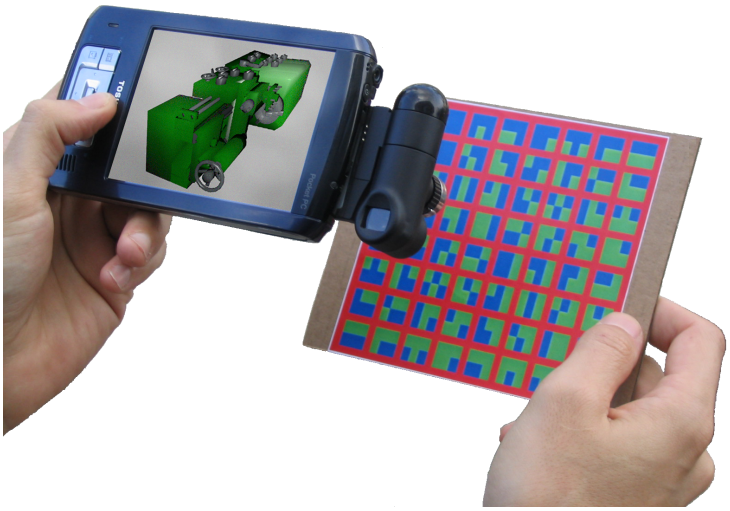


Figure 1: The camera-based interface.

Abstract

Recent advances in mobile computing allow the users to deal with 3D interactive graphics on handheld computers. Although the computing resources and screen resolutions grow steadily, user interfaces for handheld computers do not change significantly. Consequently, we designed a new 3-DOF interface adapted to the characteristics of handheld computers. This interface tracks the movement of a target that the user holds behind the screen by analyzing the video stream of the handheld computer camera. The position of the target is directly inferred from the color-codes that are printed on it using an efficient algorithm. The users can easily interact in real-time in a mobile setting. The visualization of the data is good as the target does not occlude the screen and the interaction techniques are not dependent on the orientation of the handheld computer. We used the interface in several test applications for the visualization of large images such as maps, the manipulation of 3D models, and the navigation in 3D scenes. This new interface favors the development of 2D and 3D interactive applications on handheld computers.

CR Categories: B.4.2 [Input/output and data communications]: Input/output devices—Channels and controllers; H.5.2 [Information interfaces and presentation]: User interfaces—Interaction styles

Keywords: handheld computer, user interface, interaction, visualization, video tracking

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

1 Introduction

Recent advances in mobile computing allow the users to deal with 3D interactive applications on handheld computers such as PDA and cell phones. These applications lead to new prospects for our everyday life. For example, a technician will be able to visualize and to manipulate on site the 3D model of the motor he has to repair, an archaeologist will compare the ruins of a building she saw with a computer-generated reconstruction displayed by her PDA, or a marketer will make a decision according to the strategic data he can analyze through a corresponding graph. Of course, video games have taken the benefits of mobile computing for a long time with products such as the famous GameBoy from Nintendo. They are now evolving from 2D to 3D.

Several challenges go with the development of handheld computers in terms of computing hardware, real-time visualization techniques and algorithms, and user interfaces. In this paper, we focus on this last challenge. Even if the direct pointers (stylus) seem to be widely accepted for interaction with PDA, we will see that they are not always the best solution, particularly when visualization is important. For such situations, we propose a new interface, based on the movements of a target held by the user behind the visualization interface, as illustrated in Figure 1. The target is tracked using the video stream of the camera of the handheld computer.

Compared to desktop computers, handheld computers have several

characteristics that have to be taken into account for the development of an adapted user interface:

- **Limited visualization.** The first characteristic is the reduced size of the screen which is one of the main limitations of the handheld computers. Consequently, the visualization area should not be occluded by the hand of the user in order to favor a good visualization. Moreover, the display of the screen can only be perceived by a limited range of viewing angles, i.e. when the user is facing the handheld computer. Therefore, interaction should operate without having to modify the orientation of the handheld computer.
- **Mobile use.** Handheld computers aim at mobile use. Consequently, interfaces for interaction have to operate in real mobile conditions. In particular, they should operate independently, without being connected to an additional computer, and without assuming the user is sitting in front of a table.
- **Limited computing resources.** The PDA and cell phones have limited computer resources. Consequently, the user interfaces have to be as less computation demanding as possible in order to insure real time interaction and to free the maximum of available resources to the applications.
- **Low extensibility**. A last characteristic that has to be taken into account is the low extensibility of the handheld computers. The limited connection possibilities and the lack of standard libraries make the integration of new I/O components difficult. Hence, the challenge for new user interfaces for handheld computers is to work on different architectures, without any custom installations.

In our approach, we attempt to keep in mind these four primordial constraints of the handheld computers. Our aim is to provide an interface that enables the manipulation of data such as 3D scenes, while assuring a good visualization. We focus on a very light user interface, in terms of bulk as well as in terms of computing resources.

This paper is organized as follows. In the next section, we present and discuss some previous work. We describe our interface in detail in Section 3, and in Section 4, we show how it can be used for 2D and 3D interaction. Finally, in Section 5, we conclude and give directions to future work.

2 Previous work

Several techniques have been proposed to interact with handheld computers. Some are based on the existing I/O possibilities, and others use new hardware interfaces for interaction. In particular, some work has been based on position and orientation sensors, and recent work uses video streams for interaction.

Buttons and direct pointers

The standard input interfaces of handheld computers are the buttons defining numerical keyboards, function keys or cursor keys for discrete input, and direct pointers such as stylus associated to sensitive screens for continuous input. They fit well with many basic interaction tasks such as numerical entries or selection of items in small menus, as experienced by Kjedskov and Skov [2002]. From these input devices, Gong and Tarasewich [2004] propose to extend the general HCI recommendations of Shneiderman [1997] to be suited to handheld computers. For higher level interaction tasks, advanced techniques have to be used. In particular, for browsing information, the small screen size imposes the use of zooming user interfaces (ZUI) such as [Bederson and Hollan 1994] or [Igarashi and Hinckley 2000]. The main limitation of handheld computer standard devices is their limited degrees of freedom for interaction with 3D environments. Moreover, an important drawback of direct pointers is that they occlude the screens. As we recalled in the introduction, occlusion is a critical issue when dealing with very small screens.

New handheld computer interfaces

To overcome the limitation of handheld computer standard devices, several new approaches have been proposed over the last few years. Pierce and Mahaney [2003] presented an opportunistic annexing approach where the users benefit from the I/O resources available at a given moment (e.g. television and keyboard). Dusan et al. [2003] integrated speech input for multimodal interaction with handheld computers. Another example is the use of wearable chord keyboards or isometric joysticks [Krum et al. 2003].

Position and orientation sensors

Many interfaces are based on the movements of the handheld computers as input. The ScrollPad [Fallman et al. 2004] is a mouse-mounted PDA allowing the visualization of large documents while scrolling it on a table. The Peephole display [Yee 2003] is a comparable interface where the user moves the PDA in space. It is inspired from the pioneering investigations of Fitzmaurice [1993; 1993]. Recent implementations of this approach use tilt sensors [Rekimoto 1996; Hinckley et al. 2000], accelerometers and compasses [Rantakokko and Plomp 2003]. The problem of all these techniques is that the user's viewpoint to the screen changes permanently, losing the optimal visualization angle. Furthermore, the mobility is not always insured.

Cameras

The cameras of handheld computers have been used for augmented-reality applications. For example, Wagner and Schmalstieg [2003] use a PDA camera to recover the position of specific markers positioned in the real environment. Previously, Rekimoto and Nagao [1995] tracked their NaviCam by means of color-code IDs. Currently, Rohs [Rohs 2004] uses visual codes for several interaction tasks with camera-equipped cell phones. These three works are based on markers that have to be previously fixed in the real environment, limiting mobility.

Two-handed interaction

In our approach, we were inspired by previous work on two-handed interaction. Two-handed interaction demonstrated many benefits [Buxton and Myers 1986]. In particular with 3D user interfaces, two-handed interaction has been used to give the users a kinaesthetic reference frame [Balakrishnan and Hinckley 1999]. The principle is to perform actions with the dominant hand with respect to the non-dominant hand corresponding to the reference frame. For example, Mine [1998] proposed several interaction techniques for immersive virtual environments using two hands in order to benefit from the kinaesthetic sense (proprioception). Our interface exploits kinaesthesia as well.

3 A new camera-based interface

3.1 General description

Inspired by the forces and the weaknesses of the described previous work, and according to the recommendations we presented in the introduction, we developed a new interface for interaction with handheld computers.

To favor a **good visualization** we opted for a system where the screen is never occluded by the users' hand. Consequently, the users can concentrate on their data without being perturbed by any physical object. Moreover, we particularly wanted the users not to be constrained to move the handheld computer for interaction, letting them choose and keep the optimal viewing angle to the screen. In our system, the user interacts by doing movements behind the handheld computer.

Mobile use is the aim of handheld computers. Therefore, we developed a system that can be used everywhere at any time, without being connected to any desktop computer, even through a wireless network. To overcome the problem of **low extensibility**, we use the camera that is more and more often integrated in the handheld computer, or that can easily be plugged in the CF or SD ports. The inputs for interaction infer from the analysis of the images coming from the camera.

To deal with the **limited computing resources** of the handheld computers, we designed a target that will be described in the next section. By simply analyzing a few pixels of the video stream, we are able to quickly determine the 3D position of the target with respect to the handheld computer. Our algorithm requires very little CPU ticks, hence freeing the computing resources for the applications.

Summing up, the users interact with the data by moving a target behind the screen as illustrated in Figure 2. The target has approximately the size of a CD jacket. We detect the position of the target in 3D space, which makes our interface a 3-DOF interface. This approach takes benefit from the two-handed interaction and the kinaesthetic sense described above.



Figure 2: The camera of the handheld computer films the movements of a target behind the screen.

3.2 Implementation

Computer vision toolkits such as OpenCV[1] or ARtoolkit[2] track a target in real time on standard PC. For example, Woods et al. [2003] use the ARToolKit to emulate a 6-DOF flying mouse. Similarly, Hinckley [1999] developed a camera-based multi-DOF mouse for pose estimation from a 2D grid pattern. However, the use of such techniques on handheld computers is not reasonable as they are very CPU demanding. Furthermore, using a wireless connection to process the vision algorithms on a distant server as done by Wagner and Schmalstieg [2003] is critical since the response of the user's action depends on the QoS of the network. Moreover, such a technique is only possible when a wireless network is available. Therefore, we designed an efficient and fast algorithm that can be directly computed on the handheld computers.

The 3-DOF tracking of a single black pattern on a white target is simple to implement. However, the small field of view of the handheld computer cameras makes this technique inefficient since the target has to be far from the camera to be visible while moving. Moreover, the input images have to be entirely analyzed, which significantly increases computation time.

In order to favor wide movements and to reduce the number of operations for tracking, a second approach could be the use of color scales. Hence, a *x*-*y* location of the target could be directly inferred from the color of the pointed pixel. Several color spaces could be used, in particular the uniform CIE Luv color space. However, the low quality of the handheld computer cameras and the variation of the light conditions in mobile settings make this technique inappropriate.

RGB color-codes

In order to allow wide movements close or far to the camera, and to limit the issues due to the variable light conditions, our technique is based on pure RGB color-codes. We use the target shown in Figure 3 composed of several cells separated by red borders. The target is a 12×12 cm wide square divided into 8×8 = 64 cells. Each cell is composed of two horizontal lines. The upper line relates to the *x* coordinate while the lower one relates to the *y* coordinate of the target. Each of these two lines is composed of a triplet of blue and green colors. By assigning 0 to blue and 1 to green, each triplet corresponds to a binary code. Consequently, in our target, the first line of any cell codes the column number in the target while the second line codes the line number.

A pixel from the video input is defined by its three RGB components. Each of the pixels analyzed by our algorithm is assigned to one of the three clusters RED, GREEN or BLUE according to the maximum of its components.

In the general case, it is very simple (and fast) to determine which location on the target the camera is pointing to. We describe the technique in the following. The *seed* is a reference point that is initially set to the center of the input image. The snapshots on the right are parts of the input images.

[1] OpenCV: http://www.intel.com/research/mrl/research/opencv
[2] ARToolKit: http://www.hitl.washington.edu/artoolkit

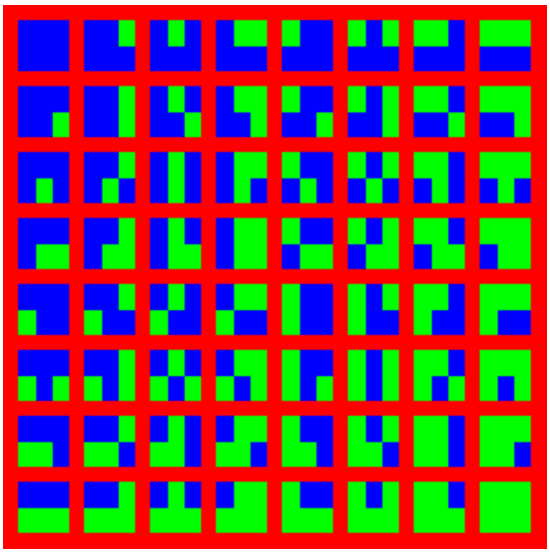
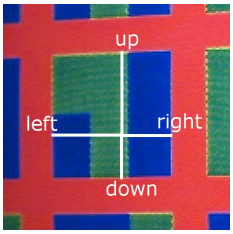
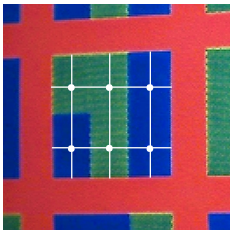


Figure 3: The RGB target composed of binary code cells.

From the seed, we search for the borders of the cell to the right, to the left, to the top and to the bottom by following the simple rule: *while the pixel is not red, consider the next pixel.*



Once the borders of the cell have been identified, it is very easy to recover the color-codes by looking at the six pixels highlighted on the right.



The color-codes determine which cell is pointed to. In our example, the first line of the cell has as the binary code 110, which corresponds to the sixth column. The second line is 010, corresponding to the third line.

The location pointed by the center of the camera has the following x coordinate on the target:

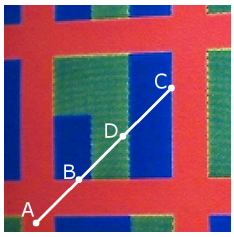
$$x_{target} = N * W + ls * \frac{W}{lr} \quad (1)$$

where N is the column number, W is the width of the column on the target, ls corresponds to the number of pixels between *left* and *seed*, and lr is the number of pixels between *left* and *right*. The y_{target} coordinate is computed identically.

The x-y position of the target with respect to the camera is directly given by x_{target} and y_{target}. The z coordinate is inferred from lr, the number of pixels between *left* and *right*. Indeed, the farther the target is from the camera, the smaller is the distance *left to right*.

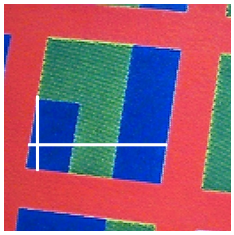
We have described a technique to determine the position of the target when the seed belongs to a cell. When the seed does not, it has to be shifted to the next cell as follows.

A is the initial seed. *While the current pixel is red, go to the next up-right pixel.* Set B. *While the current pixel is not red, go to the next up-right pixel.* Set C. The new seed D is defined by the middle of BC.

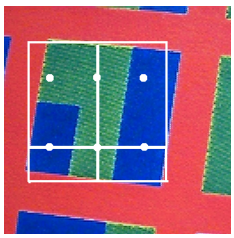


To make our technique more robust, a last test is performed before looking for the borders of the cell. This last test detects the error that could come from a tilt of the target.

Since the cells are squares, the lengths *right-left* and *up-down* are supposed to be equal. If not, we are in the situation illustrated on the right.



In this particular case, we shift the seed to the middle of the longest segment between *right-left* and *up-down*. Applying this heuristic allows to deal with small tilts, which is enough from our experience.



The technique described here operates well. The users are able to perform wide and fine displacements of the target, far or close to the camera. These displacements are processed by our algorithm and provide x, y, and z coordinates for each frame. These coordinates are absolute since they do not depend on the previous records. When a record is too far from the average of the 3 previous records, it is considered as an error and not taken into account.

3.3 Technical considerations

For the first tests, we use a PocketPC Toshiba e800 PDA with a 400MHz XScale ARM CPU. With such a PDA the x, y, and z coordinates can be estimated in less than one quarter of millisecond. Consequently, our approach allows a real-time interaction and can be used without penalizing the application. Thanks to its efficiency, our interface could be used with smaller CPU such as cell phones.

We use the Compact Flash FlyCAM camera with a resolution of 160×120 pixels. As noticed in [Wagner and Schmalstieg 2003], this camera is not able to deliver more than 7-8 images per seconds, which constitutes the main bottleneck today. However, we assume that more and more efficient integrated cameras will equip the PDA and the cell phones, and that the video streams will be easily exploitable.

Once folded, the target can be put in the cover of the PDA.

For the 3D applications we are going to describe in next section, we use klimt[3], an OpenGL—ES implementation for PocketPC.

[3]klimt: http://studierstube.org/klimt

4 Applications

Many applications can benefit from our interface, for the visualization of large documents (e.g. maps, pictures and web pages) or for interaction with 3D environments. In the following, we present some test applications we have developed.

4.1 Pan and zoom

The visualization of large documents such as maps is a difficult task when dealing handheld computers. Our 3-DOF interface is particularly well adapted to such a task since the users are able to directly control pan and zoom, without occluding the screen. They just have to bring the target closer in order to focus on a specific location. They bring it farther for general views. The two-handed interaction and the kinaesthetic reference frame allow an efficient interaction as the users benefit from a spatial feedback in addition to the visual feedback. Figure 4 illustrates the use of our interface for the visualization of a large map.



Figure 4: Large map visualization.

4.2 3D interaction

The following interaction techniques show how our interface can contribute to the development of 3D applications on handheld computers. These interaction techniques relate to manipulation, navigation, selection, and system control.

Manipulation

Our interface provides 3-DOF that can easily be attached either to the translations or to the rotations of a 3D object according to the pressed button. The mapping between the translations of both the held target and the manipulated object is direct. It gives the users the feeling to directly hold the object. Consequently, users are able to translate the objects in 3D space without any difficulties. Rotating an object is less direct as the user's translations are mapped to the rotations of the object. However, all the users who tested the system immediately understood the technique without any explanations, because the semantic link between the users' actions and the resulting actions on the screen operates well. The mapping between the movement of the target and the rotation of the object is illustrated in Figure 5.

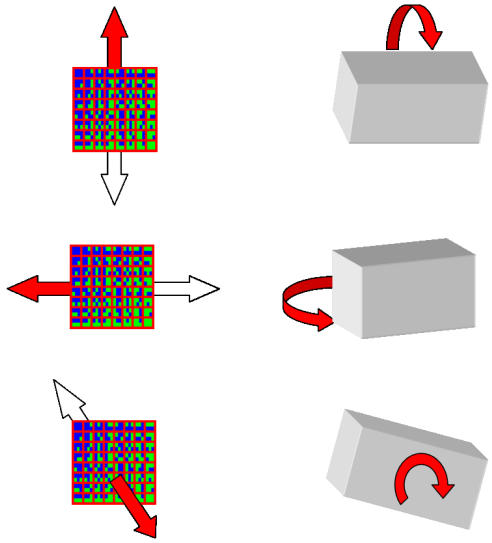


Figure 5: Mapping between the target movements and the object rotations.

Navigation

The 3-DOF provided by the interface could be used to first control the translations of the camera, and then its rotations. However, we preferred to implement a technique to directly fly into the 3D scene by using the *forward-backward* DOF to control the *forward-backward* velocity in the scene and the 2 other DOF to modify the *yaw* and *pitch* angles. Hence, by controlling 3 DOF at the same time, users are able to easily navigate in large 3D scenes. To go to the left, they move the target to the left. To go up, they move it up. Finally, to go slower, they just have to bring the target closer. Once again, the visualization is not altered as the user permanently sees the whole screen over the best orientation. Figure 6 shows an example of such a 3D navigation in a height field created using AutoMNT [Pouderoux et al. 2004].

Selection and system control

Several interaction techniques can be used to select 3D objects. For example, a technique could be to use the target as a butterfly net, or to extend the classical 2D box selection to 3D space. Therefore, users would be able to select groups of objects by locking them up into 3D selection areas. This approach is used by Encarnação et al. [1999] with their translucent sketchpad.

The control of the application is a real problem when dealing with handheld computers. Effectively, the small size of the screen forbids the use of large popup menus, as classically done on standard screens. Therefore, new user interfaces have to be found. We propose to use treemaps that hierarchically refine to select the menu items. The level of hierarchy is controlled by the z DOF while the x and y DOF enable to navigate over the items for a given level. For example, we can imagine that at the first level (when the target is far from the camera) five items divide the treemap: "file", "edit"... By bringing closer the target in the "file" section, eight new items appear, and so on. After a small moment, users know where the



Figure 6: Height field navigation.

items are on the target allowing them to quickly control the system. For example, the item ”save as” will be on the top left of the target.

5 Conclusions and future work

Numerical data have been visualized on computer monitors for a long time. Today, handheld computers also allow mobile visualization which leads to many prospects in our everyday life. However, the intrinsic characteristics of the handheld computers make us think about new user interfaces.

The classical input devices of the handheld computer (buttons and direct pointers) can be very efficient for some basic interaction tasks. However, they become obsolete for the visualization and/or 3D applications. That's why we developed a new 3-DOF interface for interaction with handheld computers. This interface aims at favoring a good visualization at any time, in real mobile settings, with real low computing resources. Our interface based on a simple video stream analysis works well and the users who tested it were really enthusiastic. They particularly like to permanently see the whole screen when navigating in 2D maps or 3D environments. Moreover, they really enjoyed the smoothness of the 3D trajectories they were able to control.

The next step of our research project will consist in evaluating the interface. In particular, we will ask subjects to perform a given task with our interface and with a 2D direct pointer. For example, the task could be a 3D docking task or 3D labyrinth navigation task. Our interface has also been implemented on a standard PC using a web-cam. It can be a low-cost alternative to classical 3D interfaces. We have to evaluate the benefit of this approach for such environments.

In this paper, we presented a new interface for 2D and 3D mobile applications. Farther than these applications, we think that this interface has a great potential for information visualization (InfoVis) applications such as the visualization of large graphs. Similar to the application for the visualization of pictures that we have presented, InfoVis applications would benefit from two-handed interaction and kinaesthetic reference frame for an efficient interaction. Many InfoVis techniques, such as Fisheye, could be used. We are currently focusing on these techniques.

Acknowledgements

The work described in this paper was supported by a grant from the Conseil Régional d'Aquitaine, France.

References

BALAKRISHNAN, R., AND HINCKLEY, K. 1999. The role of kinesthetic reference frames in two-handed input performance. In *Proceedings of the 12th annual ACM symposium on User interface software and technology*, ACM Press, 171–178.

BEDERSON, B. B., AND HOLLAN, J. D. 1994. Pad++: A zooming graphical interface for exploring alternate interface physics. In *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology (UIST'94)*, 17–26.

BUXTON, W., AND MYERS, B. 1986. A study in two-handed input. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, 321–326.

DUSAN, S., GADBOIS, G., AND FLANAGAN, J. 2003. Multimodal interaction on pda's integrating speech and pen inputs. In *Proceedings of EUROSPEECH*, 2225–2228.

ENCARNAO, L., BIMBER, O., SCHMALSTIEG, D., AND CHANDLER, S. 1999. A translucent sketchpad for the virtual table exploring motion-based gesture recognition. In *Proceedings of Eurograhics '99*, 179–190.

FALLMAN, D., LUND, A., AND WIBERG, M. 2004. Scrollpad: Tangible scrolling with mobile devices. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 9 (To appear)*.

FITZMAURICE, G. W., ZHAI, S., AND CHIGNELL, M. H. 1993. Virtual reality for palmtop computers. *ACM Trans. Inf. Syst. 11*, 3, 197–218.

FITZMAURICE, G. W. 1993. Situated information spaces and spatially aware palmtop computers. *Communications of the ACM*.

GONG, J., AND TARASEWICH, P. 2004. Guidelines for handheld device interface design. In *Proceedings of DSI 2004 (To appear)*.

HINCKLEY, K., SINCLAIR, M., HANSON, E., SZELISKI, R., AND CONWAY, M. 1999. The VideoMouse: a camera-based multi-degree-of-freedom input device. In *UIST '99: Proceedings of the 12th annual ACM symposium on User interface software and technology*, ACM Press, 103–112.

HINCKLEY, K., PIERCE, J. S., SINCLAIR, M., AND HORVITZ, E. 2000. Sensing techniques for mobile interaction. In *Proceedings of the 13th annual ACM symposium on User Interface Software and Technology (UIST 2000)*, 91–100.

IGARASHI, T., AND HINCKLEY, K. 2000. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the 13th annual ACM symposium on User Interface Software and Technology (UIST 2000)*, ACM Press, 139–148.

KJEDSKOV, J., AND SKOV, M. B. 2002. Interaction design for handheld computers. In *Proceedings of the 5th Asian Pacific Conference on Human-Computer Interaction, APCHI 2002*, Science Press.

KRUM, D. M., MELBY, R., RIBARSKY, W., AND HODGES, L. 2003. Isometric pointer interfaces for wearable 3D visualization. In *CHI '03 extended abstracts on Human factors in computing systems*, ACM Press, 774–775.

MINE, M. 1998. *Exploiting proprioception in virtual-environment interaction*. PhD thesis, University of North Carolina at Chapel Hill.

PIERCE, J., MAHANEY, H., AND ABOWD, G. 2003. Opportunistic annexing for handheld devices: Opportunities and challenges. Tech. rep., Georgia Institute of Technology GIT-GVU-03-31.

POUDEROUX, J., GONZATO, J.-C., GUITTON, P., AND GRANIER, X., 2004. A software for reconstructing 3d-terrains from scanned maps. ACM SIGGRAPH 2004 Sketches and Applications, http://iparla.labri.fr/softwares/automnt/.

RANTAKOKKO, T., AND PLOMP, J. 2003. An adaptive map-based interface for situated services. In *Proceedings of Smart Object Conference*.

REKIMOTO, J., AND NAGAO, K. 1995. The world through the computer: Computer augmented interaction with real world environments. In *ACM Symposium on User Interface Software and Technology (UIST 1995)*, 29–36.

REKIMOTO, J. 1996. Tilting operations for small screen interfaces. In *ACM Symposium on User Interface Software and Technology (UIST 96)*, 167–168.

ROHS, M. 2004. Real-world interaction with camera-phones. In *2nd International Symposium on Ubiquitous Computing Systems (UCS 2004)*.

SHNEIDERMAN, B. 1997. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc.

WAGNER, D., AND SCHMALSTIEG, D. 2003. First steps towards handheld augmented reality. In *Proceedings of Seventh IEEE International Symposium on Wearable Computers*.

WOODS, E., MASON, P., AND BILLINGHURST, M. 2003. Magicmouse: an inexpensive 6-degree-of-freedom mouse. In *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Austalasia and South East Asia*, ACM Press, 285–286.

YEE, K.-P. 2003. Peephole displays: pen interaction on spatially aware handheld computers. In *Proceedings of the conference on Human factors in computing systems*, ACM Press, 1–8.