A Formative Analysis of Mobile Devices and Gestures to Control a Multimedia Application from the Distance

Andreas Lorenz
RWTH Aachen University
Aachen, Germany
lorenz@dbis.rwth-aachen.de

Marc Jentsch
Fraunhofer Institute for Applied Information Technology
St. Augustin, Germany
marc.jentsch@fit.fraunhofer.de

Cyril Concolato
Telecom ParisTech
Paris, France
cyril.concolato@telecom-paristech.fr

Enrico Rukzio
Lancaster University
Lancaster, UK
rukzio@comp.lancs.ac.uk

Abstract— **The use of mobile and handheld devices is a desirable option for implementation of user interaction with remote services from a distance. Another prominent option to operate a remote application is the use of gestures performed in the air. This paper describes the design and realization of a system to enable mobile devices and gesture recognition tools to have control on a remote movie-player application. A small qualitative user study verified the use of mobile phones, switching between three input modalities, and the opportunity of another three methods of performing gestures in the air.**

I. INTRODUCTION

The design and implementation of interaction in pervasive computing environments cannot rely on traditional input devices like mouse and keyboard. For remote interaction from a distance, Lorenz et al. [1] revealed a dramatic increase of the error rate using wireless mouse and keyboard compared to a handheld device. Whether a control device is suitable for an intended interaction depends on the capabilities, personal preferences, situation and task of the user. If the physical shape of the equipment causes complaints or errors in operation, then the interaction could be improved either by revised design of the input hardware or by switching to another input device more aligned to the task and the personal attributes of the user.

The information flow for controlling remote services is directed from the user towards the system. In the scope of this research, the user requires the opportunity to provide input to control the behavior of the system. In recent research, two approaches have approved high potential for interacting with services in the environment of the user: The use of a mobile or handheld device, or performing gestures in the air, with the head, finger or other part of the body, or simply by moving around.

The opportunity to use mobile devices is a desirable option to enhance interaction with remote services, in particular if the user is experienced in its operation.

The system realized in this work addresses the use of mobile and gesture based input methods to wirelessly control multimedia applications in the current environment of the user. It demonstrates the use of different input devices crossing modalities.

II. RELATED WORK

Already in 1999, Eustice et al. [2] concluded that any wearable device with a minimum functionality could act as remote control for all appliances. Using for instance the Personal Universal Controller [3] a user can speak the name of a command through which this is executed by the system. Myers [4] illustrated how users are able to select from different interaction styles and devices. Rukzio [5] identified four main remote interaction styles to interact with objects in the real world: Touching, Pointing, Scanning, and User-mediated Object Selection.

A. Mobile and Handheld Interaction Devices

Many studies in smart environments have affirmed that users can easily interact with their context using handheld devices. Nichols and Myers [6], [7] presented positive results after performing an exhaustive study of the efficiency of users using handheld devices to remotely control a stereo and a telephone/digital answering machine. Some authors introduce the mobile phone as the user's favorite device for remote controlling (like [4],[8]). Others have already presented software solutions for Personal Digital Assistants (PDAs) that simulate a remote control, certifying that from the user's point of view the handheld interfaces are easier and clearer to use than remote controls or complex buttons panels [9].

This research was supported by the European Commission within the Network of Excellence "Interactive Media with Personal Networked Devices (InterMedia)", Project No. 038419.

Lumsden and Brewster [11] criticized that "*the interfaces and associated interaction techniques of most mobile and wearable computers are based on those of desktop GUIs*". They request a paradigm shift in interaction techniques beyond mouse and keyboard as mechanisms of interaction. Ballagas et al. [10] structured an analysis of mobile input techniques with five dimensions: graphical subtask (position, orient and select), dimensionality, relative vs. absolute movement, interaction style (direct vs. indirect) and feedback (continuous vs. discrete). The review of the relationships between input techniques gives insight to the key design factors of each technique. The design space helps designers of ubiquitous computing applications to select the most appropriate input mechanism to use in a given application.

B. *Gesture-Based Interaction*

Head-tracker solutions (like [12]) are designed to work with gestures for replacing traditional pointing devices. Using a Web-cam, it allows users to point and click by simply aiming their face. A combination of pointer position and keystroke input device is described in [13], using miniature video cameras that track finger position where the user can type or point in the air.

Sweep [14] lets users move a camera-phone along the desired direction of the cursor motion. By comparing consecutive frames of the camera, it offers indirect control of the cursor position. Direct Pointer [15] allows direct manipulation of the cursor with continuous visual feedback, closely resembling the laser pointer. It enables to use cameras equipped on handheld devices, such as mobile phones and PDAs. The primary advantage of this technique is that it only requires equipment that is readily available: an electronic display, a handheld digital camera, and a connection between the two. Comparable systems use a pre-calibrated, fixed camera to visually track the bright dot on the display (like [16],[17]). All these systems have the advantages of natural interaction and immediate feedback. Depending on the depth of objects in the camera images, short-distance motions may generate different distances for the cursor to move, making control difficult. Additional effort is required for the implementation of key strokes and text input.

III. SYSTEM DESIGN

A. *Infrastructure*

Figure 1 illustrates the technical components and their places in the overall image. The components of the architecture are distributed on two different hardware devices:

The **controlled device** is a computing device available in the current environment of the user. It is the host of the service, which is remotely controlled by the user. The controlled device usually is a Windows-PC, Mac, or Linux machine connected with a display for graphical output.

The other device denotes the **input device** employed by the user to control the remote service. It hosts an interactive application receiving the input from the user. The intended input device is a mobile phone or personal digital assistant (PDA), which is compared with the use of a gesture recognition tool.

The input device and the controlled device are connected with each other using a TCP/IP socket connection, preferably using wireless LAN (WLAN).

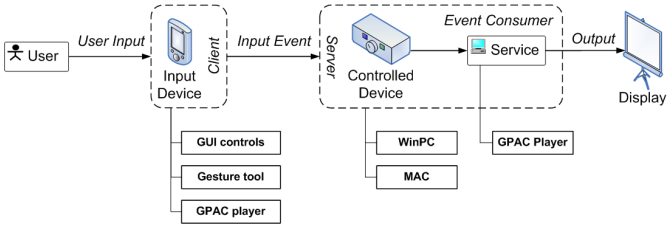

Figure 1 The technical infrastructure

The architecture is based on a client/server pattern. A component residing on the input device realizes the user interface to express input to the service, including the modality used with the input device. The physical device hosting the user interface components acts as the client device for the remote service.

The server component is the counterpart for the communication with the input device residing on the controlled device. This component receives the stream of input sent by the client, and distributes the information to the local interactive services. Examples for the server realization are networked applications, application-server, and web-server.

B. *Software Architecture*

The software architecture instantiates the framework introduced by Lorenz [18]. Figure 2 illustrates the architecture integrating the software modules. It supports three modes for the realization of the user interface on the mobile device, and three modes for using gestures observed by an infrared tracking camera.

1) Movie Control UI I Implementation: The user controls on the mobile device are separated from the controlled service. The user interface sends control commands to the service. It communicates using XML-RPC with a server listening on a specific port.

2) GPAC Client Player: In order to display a graphical user interface for the control of the remote service, the GPAC player is used on the mobile device. It detects, using UPnP, the presence of controllable services. These services are advertised by the GPAC Server Player. A widget is then downloaded and displayed on the mobile phone.

3) Widget: The widget is the visual part of the service. It is deported to the mobile phone using UPnP and HTTP. The widget is implemented using the Scalable Vector Graphics (SVG) rich media language and with JavaScript and the XMLHttpRequest API for communication with the XMLRPC_Adapter_Server.

4) XMLRPC-Adapter_Client: The XML-RPC adapter for delivery of input events. The adapter is responsible for connection establishment, data encoding, and implementation of the network protocol on behalf of the controls UI module. The event delivery is translated into a single method invocation of the adapter's responsible method.

5) XMLRPC-Adapter_Server: The XML-RPC adapter for receiving and decoding the transmitted data. The adapter is responsible for connection management, data decoding and mapping of input events to shortcuts posted to the local event queue. The adapter acts like a web-server waiting at a specific network port for incoming events. The server component integrates the incoming events as mouse- and keyboard events into the event queue of the local operating system. It can therefore work together with any application on the server; if the GPAC player gained focus on the server, this application receives the shortcuts from the event queue.

6) GPAC Server and Player: The server part of this module is in charge of advertising the service and providing the widget to control the service. The player part of this module displays multimedia content and is interfaced with the system event queue. It can be controlled by XMLRPC messages translated by the XMLRPC_Adapter_Server.

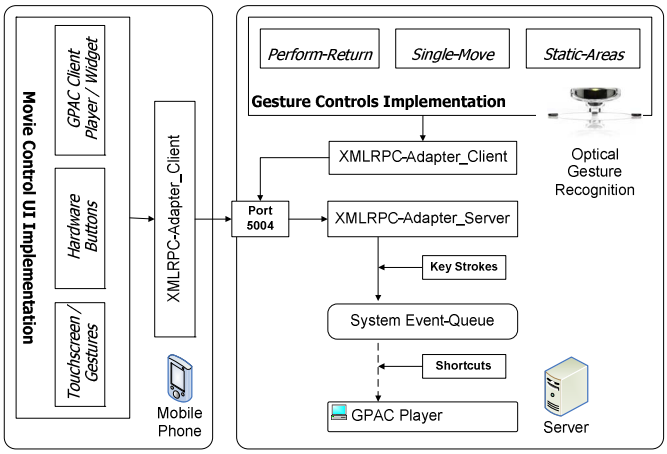

Figure 2 The software architecture

C. Control Flow

The user instantiates the process of event delivery by interacting with the system. The user interface on the input device recognizes the interaction, and triggers to create an input event. For triggering event consumption, the user interface on the input device remotely invokes a corresponding method of an instance of the XML-RPC adapter of the client. The adapter internally creates an input event, which is transformed into a corresponding representation for network transmission. The network adapters transmit the data on a network, potentially using handshake implementation for improving quality of service.

When using the widget approach, the user arrives with the mobile phone in the vicinity of the server. Since the GPAC Server advertises the controllable service, the mobile phone, running the GPAC Client Player, automatically receives the widget. The user selects the widget and the GPAC Client Player presents the control buttons to the user. When the user activates one of the buttons, messages are created and transmitted to the server.

For both cases, on the controlled device, the message is unpacked from the network representation, and handed out to consumers. The process ends, when the GPAC Server Player module received a copy of the input event.

IV. Realisation

The software has been realized using Java Mobile Edition (mobile device), C# (gestures in the air), Java Standard Edition (server), and C (GPAC tools).

This work implements the use of a remote graphical user interface on the mobile phone, the use of gestures performed on the touch sensitive display of the phone, and the transfer of a copy of the user interface from the service to the mobile device (widget-approach). The user can select from the three options to control the media player application:

1) Hardware Buttons: Use the hardware buttons of the mobile phone (see Figure 3, left).

2) Software Buttons (Widget): Activate software buttons. The software buttons mirror the controls of the application. The controls are downloaded from the remote application as a widget. For the user tests, the widgets were not downloaded on the fly but represented by software buttons on the mobile device (see Figure 3, central image).

3) Touchscreen/Gestures: Touching on the display, and perform gestures dragged on the screen of the mobile phone (see Figure 3, right).

Figure 3 The three interaction methods on the mobile phone: Hardware buttons, software buttons, and gestures on the touch-sensitive display

The user additionally has the opportunity of aiming gestures in the air. The movement of a pen in the air was tracked by an infrared camera. The changes of the position were bound to the movement of the mouse pointer on the screen to have visual feedback. Three techniques were implemented:

1) Perform-Return: Performing an effectuating gesture starting at the central area of the screen combined with a return movement to this central area (see Figure 4, left)

2) Single-Move: Performing an effectuating gesture from any place on the screen with the opportunity of slowly moving to any point on the screen (Figure 4, central image).

3) Static-Areas: Moving the pen to a direction lets the mouse cursor move to the same direction on the screen. If the pen is hold still for a short time, an event associated to the area the cursor is currently pointing to, is executed (see Figure 4, right).

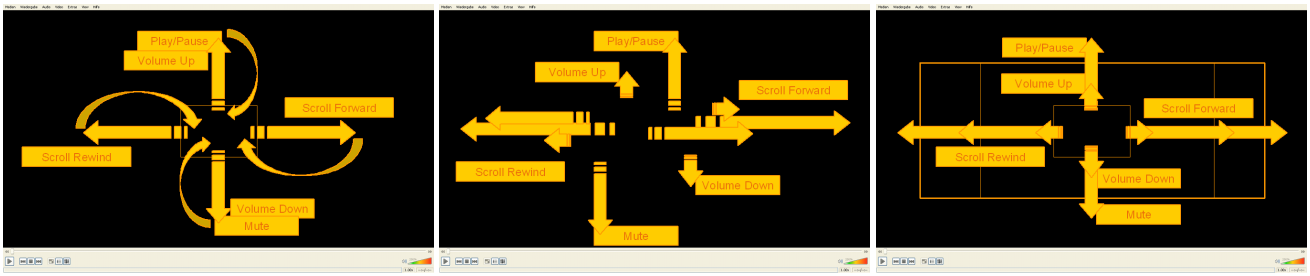

Figure 4 The three interaction methods using gestures in the air: Perform-Returen,Single-Move, and Static-Areas

For all three techniques it is taken into consideration how far the actions are performed. For example a smaller gesture to the right initiates a small forward event of the video while a larger gesture to the right initiates a larger forward event of the video. For the up movement, this difference indicates if a volume up or a play/pause event is triggered.

The implementation of the user interface maps the user input to control commands for the server, which are transmitted using XML-RPC. The server maps remote procedure calls to the creation of local shortcuts. The application that gained focus on the server receives the shortcuts from the local event queue as if typed by the user on the local keyboard.

When the widget approach is used to interact with the service, the widget is represented using the SVG language with additional JavaScript code which uses the XMLHttpRequest and MPEG-U API. The MPEG-U standard is currently being developed within the Moving Pictures Expert Group. This new standard enables to bridge networking technologies, such as UPnP, and presentation technologies, such as HTML or SVG, to enable widget mobility and widget communications.

According to the action item, the server implementation integrates the key strokes into the local event queue. It uses procedures that can be called remotely using any appropriate library of XML-RPC. The last column of the table defines the shortcut that is posted to the GPAC player by remotely invoking the method at the server.

V. QUALITATIVE USER STUDY

A qualitative evaluation was started to compare the three interaction techniques for interaction on the mobile device and the three interaction techniques of performing gestures in the air. The aim of the study was to find major anomalies in interaction design and implementation. A larger study will be conducted to evaluate the usefulness of a sub-set of the six interaction methods tested here.

A. Task

The users of the system had to remotely control a video on a large display from a distance of about 2 meters. The order of the interaction techniques to be used was randomized for each user. The users had the opportunity to try and learn each interaction technique in a 5 minutes training phase. The users were asked to perform a task of five steps of starting the video, increasing the volume to a certain level, scrolling forward to a given position, mute, and scrolling forward to another position. Each sequence was executed two times with each of the six interaction techniques.

B. Participants

A small group of four persons worked with the six interaction methods to operate the movie-player application. Three persons were male, one female. The persons were between 25 and 35 years old. All had experience in using mobile devices; one had previous experience in using gestures in the air to operate a computer system. All are right-handed.

C. Task Completion Time

The time needed to complete the task was measured for each of the six interaction methods. The average of the task completion time of both task executions was calculated. Figure 5 illustrates the average time needed by four test persons. The overall fastest time, and also the slowest performance, were both set using the software buttons on the mobile device. The best average was reached by the users using the hardware buttons of the mobile device. Noticeably, the fastest way of performing gesture in the air using static areas was faster than the averages of both the other techniques using the mobile device.

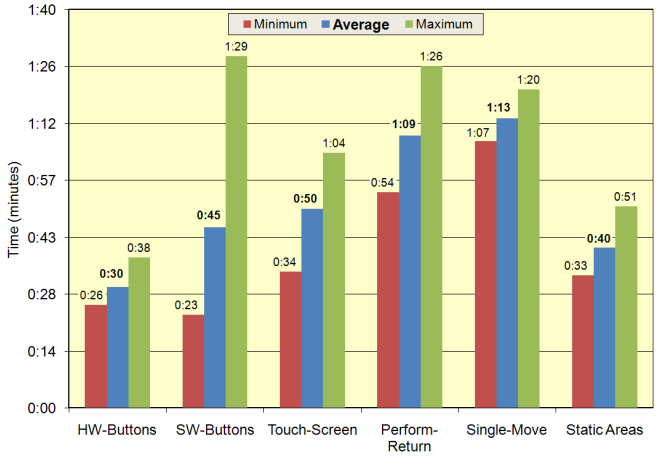

Figure 5 Time needed to perform a tasks of 5 single steps

D. Results

After the user finished the task, the user was asked to fill in the 19 items of the Post-Study System Usability Questionnaire (PSSUQ [19]). Figure 6 illustrates the average ratings for all 19 items (OVERALL, left bar), and the average regarding system usefulness (SYSUSE, right bar). Lower numbers indicate a higher satisfaction of the user. The hardware buttons

of the mobile phone reached a remarkable high level of satisfaction, for OVERALL and in particular for SYSUSE. The software buttons were on a similar high level of satisfaction, though the SYSUSE-average was higher (indicating lower satisfaction). Though the technique of having static areas for the gesture was fast, the user's satisfaction dropped down in comparison to the hardware- or software buttons of the mobile device.

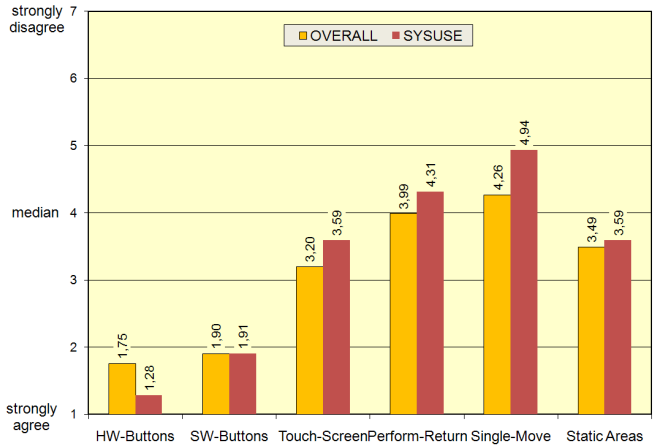

Figure 6 Average ratings to the questionnaire

For time and satisfaction, the technique of performing gestures on the touch screen of the mobile device was on similar level with performing gestures in the air.

E. Observations and Comments

The observations revealed that the hardware buttons were in favor because all persons were familiar with this approach. The users did not change foci between the control device and the application to control: The users operated the hardware buttons blind without looking at the mobile device. Therefore, the task was done faster than with all other techniques. The users did not have problems to distinguish between pressing a button and holding a button to trigger different commands.

The software buttons had the disadvantage that users had to change the focus between the large display and the display of the mobile device to activate the command: The users looked at the application on the display and, if asked to activate a specific command, the users moved the focus to the mobile phone to precisely activate the right software button. Some users increased the speed for a series of the same commands by first moving the hand to the right position on the screen of the mobile phone, fixing the position of the hand, and then looking at the large display and activating the same software button several times, e.g. to adjust the volume stepwise.

Performing the gestures on the display of the mobile device needed more training than the other approaches using the mobile phone. People who tried the hardware buttons before the gestures on the screen were faster in learning because of the similar directions. People who started with gestures in the air also quickly adopted to use the gestures on the mobile device, but had serious problems in activating the play-action. They got confused of the different implementation of the gesture in the air (large move to the top) and on the screen of the mobile phone (tab on the screen).

The use of different distances in horizontal movements on the screen of the mobile phone was problematic because of the limited space in portrait mode. Six commands for different speed of scrolling the video were placed on only 45mm of horizontal space of the display. In addition, the tab on the screen to activate the play-action was often not recognized well by the recognition engine on the mobile phone. Users had problems in finding the right time-span of pressing on the screen without moving the hand.

For all gestures, no matter if performed on the screen of the mobile phone or in the air, users invested time in training the operation. All users went through the video performing each command several times. The users did not invest time for training the hardware buttons and software buttons of the mobile phone. In particular, users who tried another method beforehand used the hardware buttons straightaway.

Performing the gestures in the air, all users tended to unintentionally perform returning gestures to the starting point of the movement; nevertheless, the Perform-Return gesture was slow and had low satisfaction. The resting area for Perform-Return and Static-Areas should be perceivable by the users. For Static-Areas, the areas, their exact borders, and assigned commands should be visible to the user. Opposed to this, users accepted that this would disturb the quality of the application, which is very much dependent on the visual appearance.

For all gestures performed in the air, users asked for an inactive mode, in which the tracking of the pen is disabled in order to rest the hand. In particular, the approach of Static-Areas would benefit if the user can activate a command and then rest the hand anywhere else. Some users proposed to apply the method of a laser pointer to have a small button on the pen to active the light explicitly. The technical setting used in this work did not support this approach, because the pen is always inactive only reflecting the light from the camera. The user would need to turn the camera on or of which is only possible at the computer the camera is connected with.

The users required more feedback for all gestures performed in the air. For the Perform-Return gesture and the Static-Areas gesture they required visibility of the central area. They also wanted to know the last command recognized from the gesture engine. This would support error prevention, recovering from errors, and learning to perform the intended gesture. The users also would like to know the predicted which command before it is consumed by the system. One user asked for a kind of eraser, to delete a misinterpretation by the recognition engine before the wrong action is activated. In particular for Perform-Return, it would be possible to illustrate the recognized command after the Perform-move, and to erase it by simply postponing the Return-move for a second.

One user got extremely frustrated of the gestures in the air because the engine did not understand the intention of the user, who in turn tried harder and less precise.

VI. Conclusions and Future Work

This work compared six interaction methods to control a multimedia application from a distance:

1) Hardware Buttons of the mobile phone: Because people were familiar with the use of the buttons, they quickly adopted its usage to the task. The use of the hardware buttons is narrowed to simple controls, low number of commands, which have a comprehensible mapping to the four directions.

2) Software buttons / Widgets: The users quickly anticipated the behavior of the application, enabling intuitive use of this approach. In general, the software buttons can mirror all interaction components of the application to control, leading to the same functionality of the user interface. The speed of using it is limited because of the changes in foci of the user if operating a remote application.

3) Gestures on the touch screen: Because the use of rather simple gestures, it was intuitively used; but limited in the provided functionality. If it is possible to map the controls to hardware or software buttons, then the use of the other approaches is faster and seems to be more convincing.

4) Perform-Return in the air: Users agreed that they return to the starting point of the move for resting the hand. The method can be used for any closed gesture, i.e. if the gesture logically ends close to the starting point. If divided into two phases, discarding a movement before activating a command is possible.

5) Single-Move in the air: Generally, this approach allows recognizing any gesture performed in the air. For the small set of simple commands used in this application, it was the slowest of all approaches and assigned with lowest level of satisfaction. The threshold speed to move the hand unobserved by the recognition engine needs precise adjustments.

6) Static-Areas: This approach was the fastest of all gestures performed in the air. Because it only depends on the size of the movement and not of the speed and acceleration, it can be applied quickly. The set of possible commands is narrowed by the available space on the display. It requires visual feedback of the current position, in this case implemented by the mouse pointer controlled by the pen movements.

The future work will fine-tune the interaction methods and perform a qualitative user study to select two or three interaction methods from the current six methods. A larger user study will compare the selected interaction methods with the current available approach to use a wireless mouse and keyboard in a home environment.

Acknowledgments

We thank all partners of the InterMedia project for intensively discussing the demonstrator “Controlling Remote Displays” that has been a primary source of feedback. We thank all participants in the user studies for their attendance and valuable feedback.

References

[1] Lorenz, A., Fernandez de Castro, C., and Rukzio, E. (2009). Using handheld devices for mobile interaction with displays in home environments. In Proceedings of MobileHCI 2009, pages 133-142.

[2] Eustice, K., Lehman, T., Morales, A., Munson, M., Edlund, S., and Guillen, M. (1999). A universal information appliance. IBM Systems Journal, 38(4):575-601.

[3] Nichols, J., Myers, B. A., Higgins, M., Hughes, J., Harris, T. K., Rosenfeld, R., and Pignol, M. (2002). Generating remote control interfaces for complex appliances. In 15th annual ACM symposium on User interface software and technology, Paris, France. ACM Press.

[4] Myers, B. A. (2002). Mobile devices for control. In Mobile HCI '02: Proceedings of the 4th International Symposium on Mobile Human-Computer Interaction, pages 1-8, London, UK. Springer-Verlag.

[5] Rukzio, E. (2006). Physical Mobile Interactions: Mobile Devices as Pervasive Mediators for Interactions with the Real World. PhD thesis, University of Munich.

[6] Nichols, J. (2001). Using handhelds as controls for everyday appliances: A paper prototype study. In ACM CHI'2001 Extended Abstracts, pages 443-444.

[7] Nichols, J. and Myers, B. A. (2003). Studying the use of handhelds to control smart appliances. In ICDCSW '03: Proceedings of the 23rd International Conference on Distributed Computing Systems, page 274, Washington, DC, USA. IEEE Computer Society.

[8] Koskela, T. and Vänänen-Vainio-Mattila, K. (2004). Evolution towards smart home environments: empirical evaluation of three user interfaces. Personal Ubiquitous Computing, 8(3-4):234-240.

[9] Roduner, C., Langheinrich, M., Floerkemeier, C., and Schwarzentrub, B. (2007). Operating appliances with mobile phones - strengths and limits of a universal interaction device. In 5th International Conference on Pervasive Computing, pages 198-215, Toronto, Canada. LNCS, Springer, Berlin-Heidelberg.

[10] Ballagas, R., Rohs, M., Sheridan, J., and Borchers, J. (2008). The design space of ubiquitous mobile input. In Lumsden, J., editor, Handbook of Research on User Interface Design and Evaluation for Mobile Technologies. IGI Global, Hershey, PA, USA.

[11] Lumsden, J. and Brewster, S. (2003). A paradigm shift: alternative interaction techniques for use with mobile & wearable devices. In CASCON '03: Proceedings of the 2003 conference of the Centre for Advanced Studies on Collaborative research, pages 197-210. IBM Press.

[12] Kjeldsen, R. (2006). Improvements in vision-based pointer control. In Assets '06: Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility, pages 189-196, New York, NY, USA. ACM.

[13] Ahmad, F. and Musilek, P. (2006). A keystroke and pointer control input interface for wearable computers. In Fourth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2006), pages 2-11.

[14] Ballagas, R., Rohs, M., and Sheridan, J. (2005). Sweep and point & shoot: Phonecam-based interactions for large public displays. In Proceedings of CHI05.

[15] Jiang, H., Ofek, E., Moraveji, N., and Shi, Y. (2006). Direct pointer: direct manipulation for large-display interaction using handheld cameras. In Proceedings CHI'06, pages 1107-1110. ACM Press.

[16] Olsen, D. R. and Nielsen, T. (2001). Laser pointer interaction. In Proceedings CHI'01.

[17] Oh, J.-Y. and Stuerzlinger, W. (2002). Laser pointers as collaborative pointing devices. In Proceedings of Graphics Interface'02.

[18] Lorenz, A. (2009). Separation of user interfaces from services of ambient computing environments: A conceptual framework. In Proceedings of the 4th workshop on Mobile Interaction with the Real World, pages 113-125.

[19] Lewis, J. R. (1995): IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use. In: International Journal of Human-Computer Interaction, 7(1):57-7