Evaluation of Mobile Phones for Large Display Interaction

Jens Bauer, Sebastian Thelen, and Achim Ebert

Computer Graphics and HCI Group
University of Kaiserslautern, Germany
`{j_bauer,thelen,ebert}@cs.uni-kl.de`

Abstract

Large displays have become more and more common in the last few years. While interaction with these displays can be conducted using standard methods such as computer mouse and keyboard, this approach causes issues in multi-user environments, where the various conditions for providing multiple keyboards and mice, together with the facilities to employ them, cannot be met. To solve this problem, interaction using mobile phones was proposed by several authors. Previous solutions were specialized interaction metaphors only for certain applications. To gain more insight into general interaction patterns realizable with smart phones, we created a set of general test cases using a well-known taxonomy for interactions. These test cases were then evaluated in a user study, comparing smart phone usage against the traditional keyboard/mouse-combination. Results (time and user satisfaction) show strengths and weaknesses when using the new interaction with the smart phone. With further evaluations we draw conclusions on how to improve large display interaction using smart phones in general.

1998 ACM Subject Classification I.3.6 Interaction Techniques, H.5.2 Interaction styles, H.5.2 Input devices and strategies

Keywords and phrases User Study, Large Display Interaction

Digital Object Identifier 10.4230/OASIcs.VLUDS.2011.103

1 Introduction

The idea to use mobile devices, especially mobile (smart) phones, to control Large Displays is not new. As both Large Displays and mobile phones (and especially smart phones) have become more common in the last few years, this idea is feasible. Large Displays is an umbrella term for various setups. These include tiled display walls, projection screens (with one or multiple projectors), large Liquid Crystal Displays (LCD), Powerwalls, and more. All Large Displays share advantages and issues, as noted for example in [5]. The main benefit is the increased screen real estate, allowing to display more information at once and enabling users to employ their spatial memory for efficient navigation. The authors further identified the following drawbacks:

1. Keeping track of the mouse position
2. It becomes more time consuming to access distant items on the screen
3. Windows may appear in unexpected places, where the user is not focusing at the moment
4. The number of simultaneous tasks a user carries out is probable to increase. This in turn calls for better task management.
5. Problems with the configuration of the different screens, especially with their position relative to each other, may occur.
6. Failure to leverage the periphery of the combined display.

Proceedings of IRTG 1131 – Visualization of Large and Unstructured Data Sets Workshop 2011.
Editors: Christoph Garth, Ariane Middel, Hans Hagen; pp. 103–112
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
IRTG 1131

These problems can be tackled in various ways (as presented in the next section). Especially the first four problems may be solvable by moving away from 'Windows, Icons, Menus, Pointer' (WIMP) environments and introducing new interaction metaphors. Smart Phones are complex devices with the capabilities of serving not only as an input, but also as an output device. Exploiting these characteristics can help to solve these problems.

1.1 Related Work

1.1.1 Large Displays

Of course, the first three problems only arise in 'Windows, Icons, Menus, Pointer' (WIMP) environments. As these systems are the most common ones used today, they still deserve attention. Robertson et al. [11] came to a similar list of issues, adding the bezel-occlusion problem to the list. To solve the first problem they proposed the mouse-trail and the display of a short animation when the user presses a key. Both functions are implemented in current version of MS Windows. For the distant access problem the paper proposes a number of solutions, like scrolling a single whole screen or selecting a target window using a 2D-ray from the current mouse position. So far the solutions have not been integrated into any modern operating system. Another solution is to use a trackpad with both relative and absolute positioning of the cursor [10]. A solution for problem 3 is not directly proposed. Instead the authors of Robertson et al. [11] propose to work around this problem by using the techniques for problem 2 and the following methods suggested to solve the multi-tasking problem. To do that the authors want to allow grouping of windows and applications both on the desktop and the (Windows) task bar. Wallace et al. [15] show a method to automatically configure tiled display systems, addressing the fifth issue in the list above. By displaying special patterns and evaluating the result through a camera, the authors calculate display distortion an the relative positions of the displays. This information is then used to correct the actual image displayed on the individual displays. This leaves the last problem, which cannot be solved by system designers. Instead this is more a missed opportunity to utilize the capabilities of Large displays. Application designers have to keep this in mind when creating programs for large display environments. One possibility of using this periphery is the focus+context screen [2]. The bezels on tiled displays can be seen as a help for the user to organize his large desktop or as proposed in the Tiled++ approach [7] their effect can be compensated by projecting the missing content onto the bezels.

1.1.2 Mobile Phones

Large displays are often used in collaborative scenarios, where naturally multiple users will want to interact with the application(s). While this can be facilitated by the traditional approach of keyboard and mouse, this does not scale well to the number of users involved. Also mouse and keyboard need to be mounted on some surface to be used without difficulties which imposes additional constraints on the environment of the large display and also the users (who is not able to move around freely). Mobile phones as ubiquitous input devices [1] have the ability to control a large display and may be able to replace mouse and keyboard altogether. Since modern (smart) phones feature multi-touch displays, cameras, Global Positioning System (GPS), accelerometers, compass, connectivity via 3G, WiFi and bluetooth and other capabilities, smart phones are subject of much work done already. This includes research for very special applications, like the 3D Human Brain Atlas [14]. Other work is focusing on data exchange [6][12] or the control of a traditional pointer using the phone [9]. On a more abstract level the design space of mobile phones were researched by

Ballagas et al. [1]. They used the taxonomy introduced by Foley et. al [8] which can still be used today.

1.2 Taxonomy of Foley et al.

This taxonomy uses six categories of input subtasks which are composited to generate actual input tasks. It should be noted, that the different categories are named not using the continuous verb form, but instead the infinitive form (e.g., Position instead of Positioning).

1. *Position* All subtasks asking the user to set the position of an object (including the position of the user) or to move an object around to a new position
2. *Orient* A task similar to position, the user is supposed to set the orientation or rotation of an object
3. *Select* Lets the user pick an object (e.g., an option from a list, select an object in 3D-space, etc)
4. *Ink* (also called *Path* in [1]) This is in effect a combination of multiple Position and/or Orient tasks and is used to define the path of an object and its orientation on this path. As *Ink* has some slightly different requirements it is a task for itself
5. *Quantify* setting a numerical value or some option derived of a numerical value (e.g., setting a volume out of the options "quiet", "normal", "loud" with each option corresponds to a db-value)
6. *Text* entering plain text, text markups are covered by other tasks

Text is already evaluated in detail in the work of Butt and Cockburn [4] and also in the work of Silfverberg et al. [13]. The subtasks *Ink* and *Quantify* can be expressed using the remaining three subtasks (when accepting some limitations), *Position*, *Orient* and *Select* are the most interesting subtasks.

These subtasks are used to further distinguish the tasks given in the test scenarios presented in the next section.

2 Evaluation Basis

This paper focuses on the work published in [3]. For that paper a user study was conducted with four different case studies on a 3x3 tiled high-resolution display. 17 test candidates of various ages and different levels of user experience were asked to complete all test scenarios using a smart phone (HTC Touch Diamond 2) and also using the traditional keyboard or mouse or a combination of both.

2.1 Description of Test Cases

A short summary of the test scenarios follows. For a more detailed description, see [3].

2.1.1 Stacking Cubes

In this three-dimensional test scenario users were asked to stack three cubes one on another in a simple physics driven environment. This involves 3 degrees-of-freedom (DOF) movement in space, without regard to rotation. This could be accomplished by either using a keyboard (using two keys per DOF), a mouse (normal position tracking + mouse wheel) or a smart phone (using the touch screen to perform movement in two dimensions and tilting the phone itself to perform movement in another two dimensions, resulting in 2 ways to control x-direction) as input device. It should be noted that the phone is the only device providing

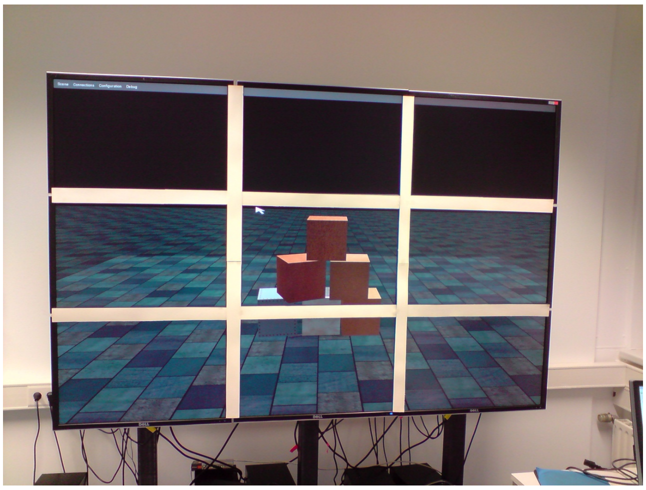

Figure 1 Stacking Cubes, the user has to stack up three cubes in a physics-enabled environment.

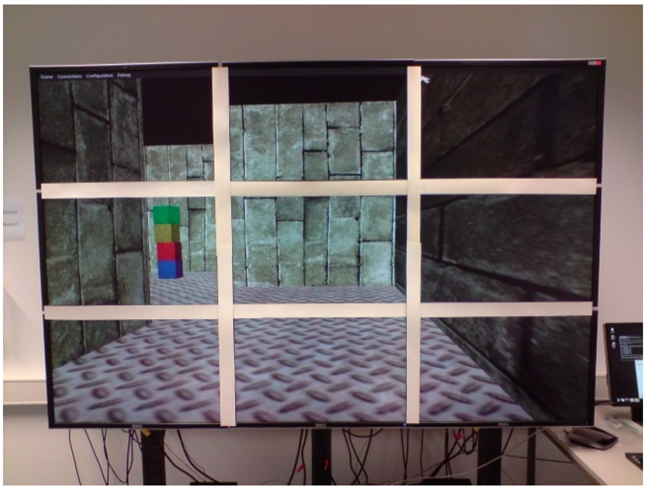

Figure 2 Maze, colored cubes can be found at one spot in the maze.

continuous movement in all dimensions. This scenario uses the subtasks of *Position* and *Select*. Figure 1 shows how the scene looks like in the study.

2.1.2 Maze

The second scenario features the first-person view of a simple maze (as can be seen in Figure 2). The test candidates have to navigate through the maze (with the help of a map) to a certain spot where they can pick up a colored cube by just touching it. Afterwards, the goal is to backtrack the way and to drop the cube into a bin outside the maze. The most commonly used input method for this case is probably the combination of keyboard and mouse, known to a wide audience of first-person computer games. Furthermore, control is possible using only keyboard (arrow-keys) or only the mouse (mouse position orients the view and holding the mouse button accelerates). Using the smart phone as it was a joystick (tilting the phone in the desired direction) was the last input method implemented. This scenario consists of the subtasks of *Position* and *Orient*.

Figure 3 City Map Annotation, the small flags from the upper left corner can be dragged-and-dropped onto the map.

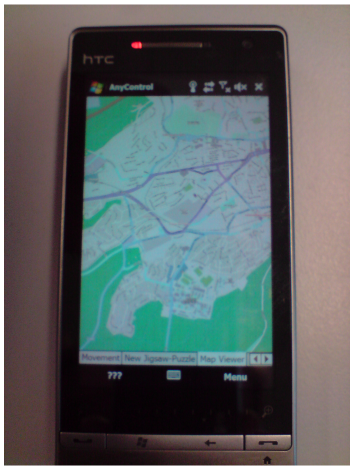

Figure 4 The city map on the phone.

2.1.3 City Map Annotation

On a local city map (Figure 3), the candidates are able to place flag markers. These markers contain some more information: A descriptive text and a picture. The flags can be placed by selecting the flag in the upper left corner in the map view and dragging it to the desired position. Then a small popup-window will appear (the size can be seen in Figure 3 in the lower middle screen), where the user can enter the description and select a picture to be displayed. The smart phone displays a smaller version of the same map, that can be panned and zoomed individually without affecting the main view on the large display (Figure 4). Unfortunately, the smart phone did not support multi-touch interaction, so zooming had to be done by using a menu. With the same menu a selection mode can be activated. Then the next tap on the map will place a flag marker and open a new view on the phone where the user can enter description text and select or even take a photo. As none of these actions will directly affect the main view (besides adding the marker on the map at some point), multiple users can perform this task at once without interfering with each other). In this scenario *Position*, *Select* and *Text* was used.

2.1.4 Jigsaw-Puzzle

One of three simple 5x5 tile jigsaw-puzzle had to be solved in this test case. The puzzle was shown on the tiled wall (Figure 5) and simultaneously a live copy of it on the smart phone (Figure 6). On the phone the tiles can be dragged around with a simple touch and drag. If necessary the user also can zoom in or out of the puzzle. Additionally, each test candidate had to solve another puzzle (for a total of three) with the keyboard and the mouse. *Postion* and *Select* were the subtasks needed for the jigsaw-puzzle.

2.2 Evaluation Setup

The evaluation was done with the already mentioned 17 participants of various ages and levels of computer experience. They were asked to carry out the tasks described above in random order using all available input methods, again in random order. In each case the

Figure 5 Jigsaw-Puzzle, the tiles have to be brought in the correct positions.

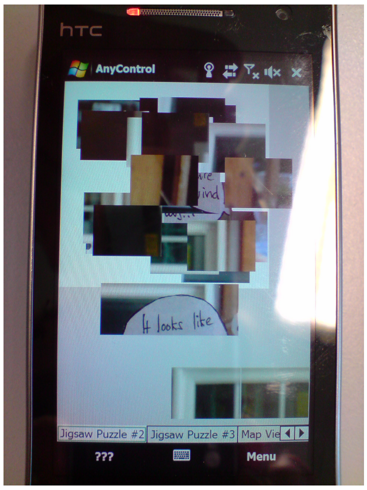

Figure 6 Jigsaw-Puzzle on Phone.

candidate had a short opportunity to get used to the input method, to a point where he or she was able to perform the task. Unfortunately not enough time was available for the candidates get a higher level of proficiency. This probably had a negative effect on the score of the smart phone especially, as no user has used a phone for large display interaction before, but each one had at least a bit of training using a computer mouse and a keyboard. For a quantitative analysis the time to complete each task with each input method was measured. To get comparable times of all candidates, the times were normalized by dividing each time by the accumulated time of all input methods of the observed task. This yields times on a scale from 0 to 1. Another quantitative measure was a grade given by every participant for each input method per test case. Possible grades range from 1 (best) to 6 (worst). To get some qualitative results, each candidate was also asked for his/her comments on the smart phone control and also for improvement proposals. The rest of the paper will now describe the results formally and draw conclusions.

3 Formal Evaluation

The measured times (total and normalized) are on a ratio scale, grades are ordinal. A set of popular descriptive statistics about the normalized times can be found in Table 1. Using a one-way analysis of variance (ANOVA) for each test scenario, the mean times can be shown to be statistically significant different on a confidence level of 5%. The basic requirements of the ANOVA, the mean times being normal distributed and all mean times having the same variance, are assumed to be met. For timed tasks normal distribution can safely be assumed and to be sure about the variances a Levene-Test has been performed for each test scenario. Unfortunately the Levene-Test did not confirm (on a 1% level) that the variances in scene 2 are equal, but as the ANOVA is known to be a very robust test, it was done anyway, but this fact has to be kept in mind. The ANOVA itself showed that the mean-time differences in the input methods are statistically significant on a 5% level (with F-values of 7.109, 21.733, 43.898 and 67.096 for scenes one to four, respectively). For scene 2, the F-value can show significant differences in means for confidence intervals below even 0.1%. This fact and a Welch-Test (also testing for differences in mean values, but also valid for non-equal variances)

Table 1 Descriptive Statistics of Normalized Times. 1st Q and 3rd Q stand for 1st and 3rd quartile, CV is the coefficient of variation and IQR is the interquartile range.

		Min	1st Q	Median	3rd Q	Max	Mean	Std Dev	CV	Range	IQR	Curtosis	Skewnes
Scene 1	Keyboard	0.104	0.172	0.261	0.307	0.570	0.265	0.125	0.471	0.467	0.135	0.346	0.797
	Mouse	0.079	0.190	0.316	0.383	0.576	0.301	0.141	0.469	0.498	0.192	-0.840	0.327
	Phone	0.230	0.348	0.401	0.512	0.740	0.435	0.136	0.314	0.510	0.164	-0.170	0.548
Scene 2	Keyboard	0.145	0.169	0.208	0.228	0.264	0.204	0.035	0.174	0.120	0.059	-1.087	-0.038
	KB + M	0.119	0.177	0.196	0.216	0.328	0.204	0.051	0.248	0.209	0.039	1.591	0.951
	Mouse	0.157	0.226	0.245	0.275	0.372	0.250	0.048	0.194	0.215	0.049	1.531	0.478
	Phone	0.211	0.281	0.335	0.412	0.483	0.343	0.076	0.221	0.271	0.131	-1.112	0.124
Scn 3	Keyboard	0.235	0.295	0.385	0.476	0.543	0.380	0.102	0.269	0.308	0.181	-1.423	-0.062
	Phone	0.457	0.524	0.615	0.705	0.765	0.620	0.102	0.165	0.308	0.181	-1.423	0.062
Scene 4	Keyboard	0.563	0.650	0.697	0.761	0.829	0.701	0.073	0.104	0.267	0.111	-0.654	-0.178
	Mouse	0.171	0.239	0.303	0.350	0.437	0.299	0.073	0.245	0.267	0.111	-0.654	0.178
	Phone	0.263	0.405	0.476	0.558	0.771	0.483	0.128	0.265	0.508	0.153	0.295	0.253

also showing differences in the mean normalized times leads us to the conclusion, that the ANOVA yield correct results even for scene 2. To get deeper insight into which mean times actually differ, a Tukey-HSD test was conducted. This test shows the pair-wise (in-)difference between the normalized mean times. The results are shown in tables 2-4.

The most important fact from this results is that the mean times of the smart phone users always differ significantly from all others. Knowing this, we can safely interpret the normalized times to get an overview of the test results. Since the grades are on an ordinal scale, no ANOVA was conducted for them.

A (Pearson) correlation test shows significant (again on a 5% level) correlation in the different times taken to solve each scenario using the smart phone. An exception for this is scene 3, the *Map Annotation*. A possible explanation for this may be the fact, that in this test case the smart phone's built-in menu had to be used a lot. This was difficult for most users, as nobody had any experience with a Windows 6 smart phone. The correlation for the other test cases show, that there is some kind of taste or distaste for the phone control. This fact is hardly surprising, but still notable. It also hints, that the usage of the native menu of the phone is a very unintuitive way of providing interaction possibilities. This was also mentioned by a few of the test candidates.

Table 2 Descriptive Statistics of Grades. CV is the coefficient of variation.

		Mode	Median	Mean	Std Dev	CV
Scene 1	Keyboard	2	2	2.1	0.96	0.45
	Mouse	3	3	2.5	1.19	0.48
	Phone	4	3	3.3	1.07	0.33
Scene 2	Keyboard	2	2	2.35	0.836	0.36
	KB + M	2	2	2.12	0.963	0.45
	Mouse	3	3	2.88	1.1315	0.39
	Phone	3	3	3.19	0.9656	0.3
Scn.3	Keyboard	1	1	1.4	0.48	0.35
	Phone	2	2	2.3	0.89	0.39
Scene 4	Keyboard	4	4	4	1.6202	0.41
	Mouse	1	1.5	1.69	0.8455	0.5
	Phone	2	2	2.5	1	0.4

Table 3 Tukey-HSD Results for scenario 1.

	KB	CM	SP
KB			✓
CM			✓
SP	✓	✓	

Table 4 Tukey-HSD Results for scenario 2.

	KB	KM	CM	SP
KB				✓
KM				✓
CM				✓
SP	✓	✓	✓	

The ARC-Pad cursor control method [10] was also implemented. This control turns the phone into a trackpad, where a tap causes the mouse cursor to jump to the position on the screen relative to the position the tap happened on the phone, e.g., A one finger tap in the center of the screen lets the mouse cursor jump to the center of the screen. A swipe on the touchscreen moves the mouse cursor normally. The goal of this control is to have fast cursor positioning together with the accuracy of a touchpad control. While the idea sounded very feasible, early tests showed very bad results. Therefore, a formal evaluation of this interaction pattern was not conducted. The main cause of the ARC-Pads issues was the inaccuracy when selecting a cursor position by tap. The resulting corrections of the position took too much time to be comfortable for the users.

The test candidates were also asked for an informal description of their experience using the new input mechanisms. Many of them stated that they had little to no experience with the control of the 3D scenarios. They also pointed out, that the control was a big lagging (Probably caused by the slow CPU on the smart phone). Virtually all users liked the 2D scenarios, where direct interaction was enabled. This was a very intuitive way of solving the tasks at hand. The issues identified here were almost all about the absence of multi-touch and the need to use the clunky system menu to activate selection mode in the Map Annotation Scenario.

4 Conclusion

The smart phone did not get the best grades or the best times. What still makes it a viable input option is the fact that it solves the problem of scaling the interaction against the number of users. Using the improvement suggestions made by the test candidates will further improve the interaction metaphors used in this first study. Together with newer and more capable hardware the smart phone seems to get on par with the other input methods. Unfortunately, there is no formal study at this time to show this, as this is still work in progress. Informal evaluation including the comments made by the test candidates shows, that all candidates liked the interaction metaphors and were also able to understand them. The main complaint was about insufficient hardware capabilities, especially sketchy

Table 5 Tukey-HSD Results for scenario 4.

	KB	CM	SP
KB		✓	✓
CM	✓		✓
SP	✓	✓	

KB = Keyboard, CM = Computer Mouse, KM = Keyboard + Mouse SP = Smart Phone; A checkmark denotes a significant difference in mean times.

accelerometers and missing multi-touch capabilities. This is something to include in further approaches. Informal studies made using the most recent Apple iPhone show much better results. Unfortunately at the time of writing no formal evaluation is was available to be included here. The better accelerometers, combined with filtering of the acceleration sample data provided by the sensors yield very stable results and greatly improve user experience. If this translates into faster solutions of the given tasks is to show in a formal study similar to the one presented here. For the 3D scenarios, *Stacking Cubes* and *Maze* users were mainly burdened with the special hand posture needed to activate the accelerometer on the HTC Touch Diamond 2. Using the touch screen to do the activation causes better results. In *Map Annotation* and *Jigsaw-Puzzle* using the touch screen of the smart phone for direct touch interaction was preferred by most users. What made the tasks in both cases a bit more problematic, was the inclusion of the system menu, as already stated in the last section. For those 2D tasks the multi-touch capabilities of modern smart phones will greatly improve performance of these tasks. Usage of multiple finger to move multiple jigsaw-puzzle tiles and using pinching gestures for map navigation allows for a more intuitive interface and greater user satisfaction.

When grouping the scenarios using the Foley-Taxonomy, *Select* was a task that can be done with the smart phone most easily. The smart phone provides direct touch interaction for selection tasks, while the mouse only provides indirect methods. Using the keyboard either means finding the correct key for the regarded object or instead cycling through all available objects until the object to be selected appears. *Orient* was not performed so well with the smart phone. The main cause may be the use of the accelerometers for this. Maybe better accelerometers, more experience with this kind of interaction or a new metaphor will solve this, but further research is needed. For *Position* the results seem to be mixed at first. But when looking at the different techniques used, one can see, that position with the accelerometer did not perform well, for the reasons already stated. *Position* with the touch screen worked very well and got a high level of user satisfaction.

Future improvements are, as already described, to use better hardware. But besides that, small improvements on the software side can also be done. As a general note, it is very advisable to use a communication protocol with a low memory footprint with smart phones. This can help to reduce lag in the connection between Large Display and phone. While this increases development time, lag decreases the user experience by a large amount. When using the accelerometers, it is also recommended to allow user configurable settings for home positions, dead zones and sensitivity. As seen in the Map Annotation test scenario, menus should be avoided if possible, as tends to break the intuitively of the interface. If really needed menus should only contain the most necessary items and be large enough to be selected with ease.

From the view of an interaction designer, the results show that touch screen input (and output) can be very well used for large display interaction. Even older phones have a touch screen good enough for this task. Employing multi-touch gestures enhances the user experience, but is not needed for basic functionalities. Of course it is necessary to keep an eye on the smaller screen real estate on the phone, but especially for 2D tasks this is the preferred way to go. Using the accelerometer for 3D interaction is a good approach per se, but in reality requires some practice on the user's side. This is therefore only feasible if the same interaction pattern can be reused many times.

References

1 R. Ballagas, J. Borchers, M. Rohs, and J.G. Sheridan. The Smart Phone: A Ubiquitous Input Device. *IEEE Pervasive Computing*, 5(1):70–77, January 2006.

2 Patrick Baudisch and Nathaniel Good. Focus Plus Context Screens: Combining Display Technology With Visualization Techniques. *interface software and technology*, 3(2), 2001.

3 Jens Bauer, Sebastian Thelen, and Achim Ebert. Using Smart Phones for Large-Display Interaction. In *2nd International Conference on User Science and Engineering (I-USEr)*, 2011.

4 Lee Butts and Andy Cockburn. An Evaluation of Mobile Phone Text Input Methods. *Australian Computer Science Communications*, 24(4):55–59, 2002.

5 Mary Czerwinski, George Robertson, and Brian Meyers. Large Display Research Overview. *CHI 06 extended*, page 69, 2006.

6 Raimund Dachselt and Robert Buchholz. Throw and Tilt – Seamless Interaction Across Devices Using Mobile Phone Gestures. *Proc. MEIS 2008*, pages 272–278, 2008.

7 Achim Ebert, Sebastian Thelen, P.S. Olech, Joerg Meyer, and Hans Hagen. Tiled++: An Enhanced Tiled Hi-res Display Wall. *Visualization and Computer Graphics, IEEE Transactions on*, 16(1):120–132, 2010.

8 James Foley, Victor Wallace, and Peggy Chan. Human factors of Computer Graphics Interaction Techniques. *IEEE Computer Graphics and Applications*, 4(11):13–48, 1984.

9 Antonio Haro, Koichi Mori, Tolga Capin, and Stephen Wilkinson. Mobile Camera-Based User Interaction. *Computer Vision in Human-Computer Interaction*, pages 79–89, 2005.

10 D.C. McCallum and P. Irani. Arc-pad: Absolute + Relative Cursor Positioning for Large Displays with a Mobile Touchscreen. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 153–156. ACM, 2009.

11 George Robertson, Mary Czerwinski, Patrick Baudisch, Brian Meyers, Daniel Robbins, Greg Smith, and Desney Tan. Large Display User Experience. *Computer Graphics and Applications, IEEE*, 25(4):44–51, 2005.

12 Khoovirajsingh Seewoonauth, Enrico Rukzio, Robert Hardy, and Paul Holleis. Touch & Connect and Touch & Select: Interacting with a Computer by Touching it with a Mobile Phone. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, page 36. ACM, 2009.

13 Miika Silfverberg, I. Scott MacKenzie, and Panu Korhonen. Predicting Text Entry Speed on Mobile Phones. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '00*, 2(1):9–16, 2000.

14 Sebastian Thelen, Joerg Meyer, Achim Ebert, and Hans Hagen. A 3D Human Brain Atlas. *Modelling the Physiological Human*, pages 173–186, 2009.

15 Grant Wallace, Han Chen, and Kai Li. Automatic Alignment of Tiled Displays for a Desktop Environment. *Journal of Software*, 15(12):1786, 2005.