Scroll, Tilt or Move It: Using Mobile Phones to Continuously Control Pointers on Large Public Displays

Sebastian Boring
University of Munich
Amalienstr. 17
80333 Munich, Germany
sebastian.boring@ifi.lmu.de

Marko Jurmu
MediaTeam Oulu, Dept. of
Electr. and Inf. Eng.
University of Oulu, Finland
marko.jurmu@ee.oulu.fi

Andreas Butz
University of Munich
Amalienstr. 17
80333 Munich, Germany
andreas.butz@ifi.lmu.de

ABSTRACT
Large and public displays mostly provide little interactivity due to technical constraints, making it difficult for people to capture interesting information or to influence the screen's content. Through the combination of large-scale visual output and the mobile phone as an input device, bidirectional interaction with large public displays can be enabled. In this paper, we propose and compare three different interaction techniques (*Scroll*, *Tilt* and *Move*) for continuous control of a pointer located on a remote display using a mobile phone. Since each of these techniques seemed to have arguments for and against them, we conducted a comparative evaluation and discovered their specific strengths and weaknesses. We report the implementation of the techniques, their design and results of our user study. The experiment revealed that while *Move* and *Tilt* can be faster, they also introduce higher error rates for selection tasks.

Author Keywords
Optical flow, accelerometers, input techniques/mappings, target acquisition, fatigue, cursor control.

ACM Classification Keywords
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Evaluation/methodology, Haptic I/O, Input devices and strategies, Interaction Styles, Prototyping*; D.2.2 [**Software Engineering**]: Design Tools and Techniques: *User Interfaces*

INTRODUCTION
Particularly large display technologies are starting to appear in our everyday lives. Not only can they be found in private environments (e.g., large-scale TVs, projectors or monitors), but more and more in public spaces such as airports, subway stations or shopping malls, where they mostly act as large ambient information displays. Instead of allowing users to capture interesting information or influencing the display's content, the usage models of these displays tend to be static, offering only unidirectional broadcast without possibilities for bidirectional interaction. In addition, home-installed large screens, e.g., projectors or flat panels, mostly lack sophisticated input capabilities beyond those of a TV remote.

OZCHI 2009, November 23-27, 2009, Melbourne, Australia.
Copyright the author(s) and CHISIG
Additional copies are available at the ACM Digital Library (http://portal.acm.org/dl.cfm) or ordered from the CHISIG secretary (secretary@chisig.org)
OZCHI 2009 Proceedings ISBN: x-xxxxx-xxx-x

On the other hand, small portable devices such as personal digital assistants (PDAs) and mobile phones are available at low prices and are by now usual companions in our everyday lives. Despite their limited visual output capabilities compared to large screens, they come with various built-in input options such as a joystick, a stylus, a keypad or touch-screens. Hence, combining the users' mobile devices and large public displays, for example, allow the control of a personal pointer on the public display, turning the phone into a ubiquitous input device for large screens (Ballagas et al., 2006). The interaction techniques need to be efficient, enjoyable and easy to learn. Unfortunately, each of them has its limitations in terms of accuracy, operation speed, efficiency and ergonomics.

In this paper we discuss three different techniques to control a pointer on a large public display by using a selection of mobile device input and sensing capabilities. We developed three distinct strategies for continuously controlling the user's pointer: first, the pointer can be moved constantly by pressing the phone's joystick in the respective direction (*Scrolling*). Second, the pointer is accelerated by tilting the mobile phone in the desired direction (*Tilting*), and third, the pointer's movement is linearly mapped to the phone's movement (*Moving*). We discuss the different input strategies followed by an extensive evaluation in which we analyze the feasibility of each input mapping in detail. Based on the findings in our evaluation, we provide insights on how to design pointer interaction on large screens using mobile devices.

RELATED WORK
The usage of mobile devices for interaction with large displays – static (paper-based) or dynamic (screen-based) – has been an active research focus in the past years (Ballagas et al., 2006). If users interact with static displays such as maps, the phone usually acts as a lens, digitally augmenting the current focus area (Rohs et al., 2007). Others investigate the remote control of pointers (Myers et al., 1998) and interaction (Want et al., 1995) on large screens using mobile stylus-based interfaces. Others investigated different issues related to large public display interaction. Ballagas et al. define three domains in which this interaction takes place: *personal*, *semi-public* and *public* (Ballagas, 2004). Based on these, they identify constraints that need to be taken into account for designing interaction techniques.

An analysis of existing pointing interaction techniques shows a trend of using the phone's camera in combination with visual markers. *Point & Shoot* (Ballagas et al., 2005) enables users to select objects on a large screen by pointing at them using the phone's camera. Users can select objects by pressing the phone's select button. This results in a grid of visual markers being temporarily superimposed on the large display's canvas. The phone takes a picture and uses the visual markers in order to detect the position the phone has been pointed at. *SpotCode* (Madhavapeddy et al., 2004) also uses the principle of tracking visual patterns. Another system called *Shoot & Copy* (Boring et al., 2007) allows users to take a picture of the desired content on a secondary screen. Using image processing techniques, the captured content gets recognized and is then sent back to the user. However, neither technique offers permanent visual feedback and hence makes continuous operations cumbersome. Furthermore, both systems keep the controls on the mobile device possibly leading to *macro attention shifts* (Holleis et al., 2007).

Instead of using visual markers, experiments with optical flow analysis of the video stream captured by a camera have been conducted. Pears et al. (Pears et al., 2008) introduce a mechanism that allows pointing directly on the screen. However, their system requires the phone's camera to be pointed towards the screen at all times, which most likely increases fatigue in the user's arms (also known as the "*gorilla-arm-effect*" occurring when interacting with vertically mounted touch screen monitors). *Sweep* (Ballagas et al., 2005) mimics the behavior of a standard desktop mouse. When the user moves the phone into a certain direction, the pointer on the large screen moves in the same direction (with a linear factor). A further example is *Direct Pointer* (Jiang et al., 2006). Their system requires the camera to be pointed at the screen at all times. While this reduces distance-related problems of *Sweep*, users might experience fatigue in their arms, leading to short interaction cycles. As both systems allow a person to use the personal mobile phone as optical mouse, we based one of our interaction techniques on optical flow to allow a direct comparison.

While optical flow analysis relies on the phone's camera (*inside-out tracking*), other systems use fixed cameras to observe the phone's motion (*outside-in tracking*). Miyaoku et al. introduce *C-Blink* (Miyaoku et al., 2004) which represents such a system. A camera attached to a large screen identifies the motion of the phone's display and moves the pointer on the large screen accordingly. However, pointing the phone towards the screen most likely results in a rather unusual hand posture.

Recently, researchers are experimenting with using near field communication (NFC) in order to interact with static as well as dynamic displays. *Marked-up Maps* (Reilly et al., 2006) is a system that allows a mobile device to touch and select options on a static paper map. The map was augmented with RFID tags representing options touchable by an RFID reader connected to a mobile phone. In contrast to this approach, *Touch & Interact* (Hardy et al., 2008) uses a dynamic display by projecting a digital image on top of an RFID tag matrix. With this, users can touch the display at any position to make selections. However, all identified interaction techniques require the phone to be very close to the display.

Together with commercial applications such as the *iPhone Air Mouse*[1] and the *Nintendo Wii*[2], research has been done utilizing acceleration sensors built into mobile phones (Vajk et al., 2007). One example is *Toss-It* (Yatani et al., 2005) which allows users to "throw" images onto a screen in a multi-display environment. However, this technique only enables users to point at the screen they want to have the image on, but more accurate operations seems to be cumbersome. *MobiToss* (Scheible et al., 2008) utilizes the built-in sensors in two ways: First, users can "throw" a recently captured video clip onto a nearby screen. Second, users can then manipulate the clip by tilting the phone (e.g., applying different video effects). However, the usage of acceleration sensors for pointer control has not been examined.

INTERACTION TECHNIQUES

For controlling a pointer on the display, today's personal computers offer a standard mouse or equivalent devices, such as track points or touch pads. While suitable for single users in desktop environments, the single pointer paradigm needs to be extended for use in public spaces. It nevertheless provides an acceptable mental model as starting point for collaborative or concurrent use. Touch-based input on large public displays is not always an option due to protection reasons (e.g. against vandalism) as well as their relatively large dimensions. To interact with public screens instead of just passively viewing the information, we have developed three interaction techniques based on different input mappings, using the technologies built into a modern mobile phone.

Scrolling

The simplest control is to use a mapping from key presses to a constant motion: the cursor is either moving at constant speed in a certain direction or remains still. A common approach used in past research is to utilize the phone's joystick or arrow keys (Silfverberg et al., 2001). This technique gives both direction and movement at the same time with a control display (CD) ratio of 1. If the user presses an arrow (or tilts the joystick) in one direction, the cursor moves at constant speed in this direction until the user releases the key and the joystick respectively. The constant motion – causing high selection times for distant targets – could be sidestepped by using ballistic pointer movement (i.e. decreasing the CD ratio) similar to the mouse behavior in modern operating systems. This leads to an *overshooting effect* which will be discussed later. Moving in two dimensions is rather difficult as the user has to press two buttons simultaneously which in turn only allows moving the pointer diagonally. This limitation comes from the stand-alone design for mobile phones menus, which does not require the joystick to cover all eight directions.

Tilting

We created *Tilt* as a second solution, which maps the phone's movement to the pointer's movement in a differ-

[1] iPhone Air Mouse, http://www.mobileairmouse.com/

[2] Nintendo Wii, http://wii.com/

ent way. Similar to the input principle of an analogue joystick, we can use tilting in order to accelerate the pointer on the screen. The more the user tilts the phone in a certain direction, the faster the pointer.

From the built-in acceleration sensors we obtain different gravity measurements by moving or rotating the phone. We can then construct different mappings between the sensor readings and the two-dimensional mouse movement. We can use the rotation around the *x* and *y* axes to control the cursor's movement speed in each direction. This technique corresponds to the mapping used by analogue joysticks. The zero point is a small interval around the normal viewing orientation of the phone (see Figure 1). This allows users to keep the pointer at a certain position in a simpler fashion.

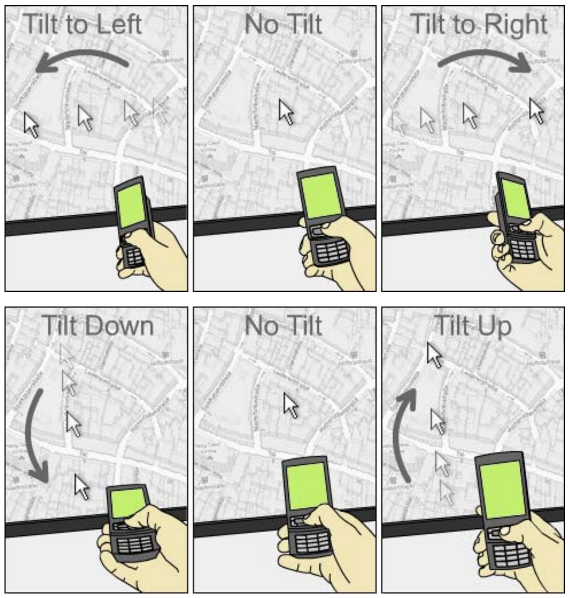

Figure 1. Tilting the phone left or right (top) increases (decreases) the pointer's horizontal speed. Tilting it up or down (bottom) increases (decreases) its vertical speed.

To simulate a mouse button being pressed, we utilize the mobile phone's joystick button. Pressing it is translated into a *down* event. Moving the cursor with the sensor input while having the button pressed results in *move* events. Finally, if the user releases the button, the cursor receives an *up* event. This allows us to perform the most common operations known from traditional cursor-based user interfaces (e.g., WIMP): *hover* (e.g., cursor moving above a certain element) and *drag* (cursor moving plus mouse button pressed). However, this only simulates the actions taken with the left mouse button. To enable right-clicks, systems could utilize a secondary button on the mobile phone or dwell times of a stopped cursor. However, holding the pointer still is a rather difficult task.

Moving

The third solution, called *Move*, is an interaction technique that maps the phone's movement linearly to the pointer's movement. As *Scroll* is very accurate but has high selection times for far distances (due to its CD ratio), we decided to use a technique that is inspired by both the standard computer mouse and *Sweep*. In order to change the speed, we chose a more direct mapping where start and stop of the operation is now determined by start and stop of the phone's motion. Similar to *Sweep*, we utilize the phone's camera in combination with optical flow image processing, allowing the phone to be pointed anywhere. With this, we get a direct mapping between the phone's and the pointer's motion (CD ratio is 1).

In contrast to *Scroll*, users can determine the speed of the pointer by simply moving the phone faster and slower respectively. This interaction technique is very similar to the computer mouse which is well-known to most users. The main difference is that no direct contact with a surface is needed. We have also found that rotating is possible with our implementation resulting in higher motion speed of the pointer (i.e., lower CD ratio). The combination of both allows the user to accurately position the pointer on the remote display (see Figure 2).

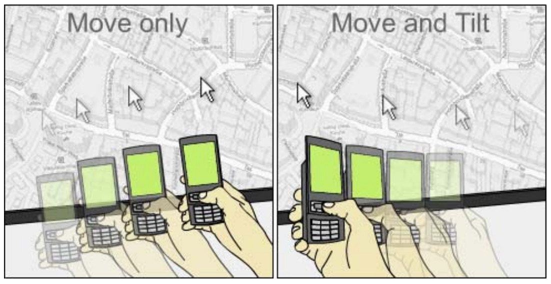

Figure 2. Left shows moving the phone without rotation. Right denotes movement plus rotation. By tilting the wrist, the pointer moves further with less arm movement.

In contrast to *Sweep*, the phone's motion always affects the pointer's position. If users want to suspend the pointer movement, they can place their finger on top of the camera's lens leading to a black image. This can be compared to lifting the mouse for repositioning it (*clutching*). To activate optical flow analysis again, users take their finger away. The simulation of a mouse button being pressed can be achieved as described in the section on *Tilt*.

Discussion and Prediction

After introducing our mappings, we give a prediction of their performance by analysing their input capabilities and dimensions. According to Card's input design space (Card et al., 1991), *Move* adds the most input dimensions to the phone as it allows continuous linear as well as rotary measurements, while *Tilt* only allows rotary measurements. However, *Move* and *Tilt* rely on the phone's select button for more complex interactions such as *selection* or *drag-and-drop*. In summary, adding the phone's camera and its acceleration sensors leads to a richer set of interaction capabilities to control a pointer's motion.

Scroll offers discrete input leading to a limited set of moving opportunities (linear or diagonal). Furthermore, complex operations, such as *selection* and *drag-and-drop* are not possible without using additional keys on the phone as moving and pressing the select button at the same time is difficult. On regular desktop computers, this can be achieved by combining the mouse's motion and one of its buttons being pressed. Hence, *Move* and *Tilt* add more input dimensions. For example, *Move* (*Tilt*) along the phone's *z*-axis (*y*-axis) could indicate scaling which is not possible without mode switch using *Scroll*.

Based on our understanding of the techniques and preliminary tests, we have created a basic selection model. This model tries to map the target's distance to the time needed to select it when using one of our input techniques. We acknowledge that the target's size also plays an important role in the selection process. However, we assume that the selection time increases similarly for all techniques when targets get smaller. It seems obvious that the performance of *Move* will decrease the further targets are away from the current pointer location. As seen in Figure 3, we expect an overshooting effect for targets that are very close or very far away for both *Move* and *Tilt*. This is based on the theory of having trouble when moving short or long distances due to sensitivity: for long distances, the pointer might already be too fast for slowing down towards a certain position. As *Move* mimics the computer mouse, we assume that overshooting will affect *Tilt* more than *Move*. We also predict that the overshooting effect will decrease for increased target sizes.

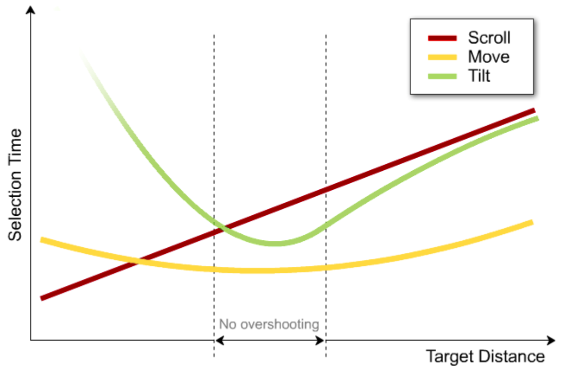

Figure 3. Model of expected selection times. In the central region, the overshooting effect does not occur.

EXPERIMENTAL EVALUATION

In our experiment we set out to analyze in detail the suitability of theoretically chosen techniques and their predicted behavior when controlling a pointer on a large screen. Participants acquired targets of different sizes and positions using each of our input techniques.

Task and Stimuli

We modeled our task setup based on the design used in (Vogel et al., 2007) and other target selection studies. Each participant was presented with a series of target selection trials in a multidirectional tapping test based on ISO 9241-9 standard (Douglas, 1999) by utilizing each of the three pointing techniques. Three different target sizes were each placed in two distances at eight angles.

We asked our participants to acquire the targets as fast as possible while being accurate during selection. Participants were allowed to hold the mobile phone in their dominant hand in the same way they usually hold their phone. At the beginning of each trial, the pointer stayed fixed on top of a solid red square (size: 48 pixels) at the center of the screen indicating the start button. At the same time, the target was rendered as a white square with a red border (see Figure 4a). Distance, angle and size were parameterized in such a way that targets never reached the screen border, which thus allows overshooting the target. A street map was displayed in the background of the stimuli to increase ecological validity.

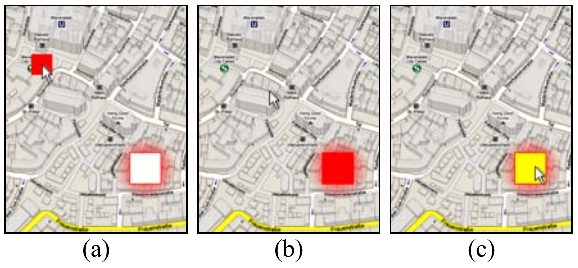

Figure 4. (a) Start of the trial. (b) After the start, the target turns red. (c) Target turns yellow when hovering over it.

Once the user activated the trial by pressing the phone's select button, the start button disappeared and the target turned into a solid red square as shown in Figure 4b. The user was then able to move the pointer freely on the screen using the currently active input technique. When the pointer was moving within the target, the target provided visual feedback to the user by turning into a yellow square (see Figure 4c). A trial was completed when the user hit the phone's select button while the pointer was inside the target. If the user clicked outside of the target, the trial continued increasing the error count. Users advanced to the next trial when they successfully selected a target. During the test we recorded target acquisition time and errors as the number of unsuccessful target selections. Furthermore, we recorded the pointer's trace for each trial to detect possible target overshooting events.

Study Design and Hypotheses

A repeated measures within-subject factorial design was used. The independent variables were *Technique* (*Scroll*, *Move* and *Tilt*), target *Size* (*24*, *48* and *72 pixels*), target *Distance* (*96* and *336 pixels*) and target *Direction* (*N*, *NE*, *E*, *SE*, *S*, *SW*, *W* and *NW*). By evaluating typical screen dimensions and distances between user and screen, even targets sized 24 pixels seemed to be too small, but we kept it to detect limitations of the techniques. During the study we counterbalanced the usage of *Technique* among all our participants. For each user, we combined the 3 target *Sizes* with each of the 2 target *Distances* as well as each of the 8 target *Directions*. These triplets were then presented in random order in each block. For each *Technique*, participants had one practice block and three timed blocks. In summary we collect the following number of data points per participant (excluding the practice blocks):

3 *Techniques* (*Scroll*, *Tilt* and *Move*) ×
3 *Blocks* ×
3 *Sizes* (*24*, *48* and *72 pixels*) ×
2 *Distances* (*96* and *336 pixels*) ×
8 *Directions* (*N*, *NE*, *E*, *SE*, *S*, *SW*, *W* and *NW*)

= 432 selections per participant

In contrast to other Fitt's law experiments, participants had to press and release the phone's select button when the pointer was inside the target. Leaving the target while the button is pressed was possible (Forlines et al., 2005). Hence, a selection was counted as error if either the button press or the button release occurred outside the target.

Based on our understanding of the techniques regarding their predicted selection performance, we had three hypotheses: First, we expected *Move* to outperform *Tilt* for small targets in any distance due to less overshooting effects (H1). Second, we expected *Move* to outperform *Scroll* for medium sized and large targets with higher distance in terms of selection time, but at the expense of increased error rates due to the pointer's high movement speed (H2). And third, we assumed *Move* and *Tilt* to have noticeable higher error rates compared to *Scroll* for small targets regardless of their distance (H3).

Apparatus

We conducted this study using a large, vertically mounted 50" plasma TV from Panasonic (model: TH-50PV71FA) acting as a public display. The plasma panel has a resolution of 1366 × 768 pixels and has – with an aspect ratio of 16:9 – a physical screen size of 1106 × 622 mm. These measurements give an effective resolution of 1.235 pixels per millimeter. Hence, the physical distances were 77.7 mm (96 pixels) and 272.1 mm (336 pixels) respectively. The physical target sizes were 19.4 mm (24 pixels), 38.9 mm (48 pixels) and 58.3 mm (72 pixels). The resulting indices of difficulty ranged from 1.22 bits to 3.91 bits. For the client we used a Nokia N95 8GB which has three built-in accelerometers for each of its local axes.

The Participants of our study were located about 1.5 meters away from the display. Considering public screen diagonals between 100 and 200 inches (common sizes in subway stations), our prototype gives the same dimensional impression when a person is standing between 3 and 6 meters away from such a display.

Participants

We recruited 12 volunteers (3 female), ranging in age from 22 to 31 (average: 25.1). Three participants were left-handed. All of the subjects use a computer (70% for work purposes) and rated their computer expertise as 4.5 on a five point scale where 1 equals "no expertise" and 5 equals "expert". All of them own a mobile phone and rated their expertise with 3.58 on the same scale. In addition, all participants have previous knowledge about sensor-based or optical input devices.

RESULTS AND DISCUSSION

Our experimental evaluation revealed new insights regarding performance and error rate of our interaction techniques. In this section, we will show our results regarding on selection time, error rate, subjective preference and observations followed by a detailed discussion.

Results

Repeated measures analysis of variance revealed that there was no significant effect on selection time and error rate when presenting the three *Techniques* in different order. This indicates that a within-subject design was appropriate for our study. A subsequent 3 × 3 (*Technique* × *Block*) within subject analysis of variance on selection time found a significant main effect for *Block* ($F_{2,22}$ = 16.314, p < .001) and revealed the presence of a learning effect. Post-hoc analysis showed that *Block 1* was significantly slower than *Block 2* and *Block 3* (all p ≤ .003) respectively. No significant speed difference has been found from *Block 2* to *Block 3*. Thus, in subsequent analysis, *Block 1* was not included and the selection time was aggregated for the remaining two blocks.

Selection Time

We measured the selection from the moment the user pressed the phone's select button until the target was successfully selected. We performed a 3 × 2 × 8 (*Technique* × *Distance* × *Direction*) within subjects analysis of variance and found significant main effects as well as interactions for both *Distance* ($F_{1,11}$ = 217.788, p < .001) and *Direction* ($F_{3.9,43.2}$ = 3.877, p = .009, Greenhouse-Geisser) indicating that aggregation across both is not suitable. However, we grouped the 8 *Directions* into linearly (*N*, *E*, *S* and *W*) and diagonally (*NE*, *SE*, *SW* and *NW*) placed targets and aggregated repetitions for each participant. We performed a 3 × 2 × 4 (*Technique* × *Distance* × *Direction*) within subjects analysis of variance for both linearly and diagonally placed targets and found no significant main effects or interactions for *Direction*.

We performed a 3 × 2 × 2 × 3 (*Technique* × *Distance* × *Direction* × *Size*) within subjects analysis of variance on median selection times aggregated across *Blocks* and *Direction* (linearly versus diagonally). Significant main effects were found for all independent variables: *Technique* ($F_{2,22}$ = 14.536, p <.001), *Distance* ($F_{1,11}$ = 217.788, p < .001), *Direction* ($F_{1,11}$ = 17.917, p = .001) and *Size* ($F_{2, 22}$ = 117.305, p < .001). However, most interesting are the significant interaction effects of *Technique* × *Direction* ($F_{2,22}$ = 4.632, p = .021) and *Technique* × *Distance* × *Size* ($F_{4,44}$ = 6.984, p < .001). Post hoc multiple means comparison tests showed that the difference in selection time for linearly versus diagonally targets stayed significantly lower for *Move* compared to *Tilt*, but there's no significance compared to *Scroll*. These results are a bit unexpected as *Move* and *Tilt* are both capable of moving diagonally without any constraints. Further analysis revealed that there is a significant difference between linearly and diagonally placed targets for *Tilt* (p < .001). Moving diagonally with *Tilt* is significantly slower than moving linearly. Further, we show that *Scroll* is slightly faster for diagonally placed targets, suggesting that tilting the phone around two axes is harder than around one axis.

For all *Distances* and *Sizes*, *Move* is faster than the other two *Techniques*. For the short *Distance* (i.e., 96 pixels), *Move* is significantly faster than *Tilt* for small target *Sizes* (p = .022). *Scroll* is also faster, but does not provide a significant difference. However, this is a strong indication of the overshooting effect. It suggests that users have problems in moving the pointer for a short *Distance* by tilting the phone due to high sensitivity. For medium sized targets (i.e., 48 pixels), all *Techniques* performed similarly, whereas *Move* outperforms the other *Techniques* significantly for large targets (all p < .02). With this we can show that *Move* performs better for short distances without showing any overshooting effects.

For the long *Distance* (i.e., 336 pixels), post hoc multiple mean comparison tests reveal that *Move* is significantly faster than *Tilt* for all target *Sizes* (all $p < .016$). Together with the results for the short *Distance*, this supports our hypothesis H1. In addition, *Move* outperforms *Scroll* for target *Sizes* greater than 24 pixels (all $p < .001$). There is no significant difference between *Move* and *Scroll* for small targets. An explanation of this effect is that participants rushed to the target, but then needed some time to finalize the pointer's position accurately on the target. This supports our hypothesis H2. The analysis shows that *Tilt* only performs significantly better than *Scroll* for large targets ($p < .002$). This is a further indication that *Tilt* suffers from overshooting effects for more distant targets when users accelerate the pointer too fast, pass the target and have to go back. The fact that *Tilt* is significantly faster for large targets can be explained by users "flying" over the target but successfully selecting it.

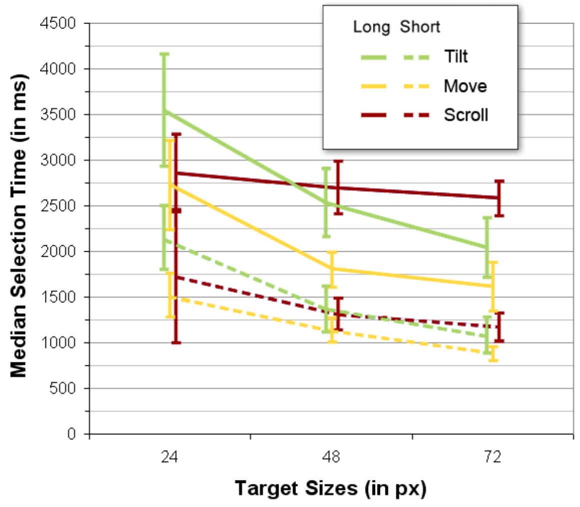

Figure 5. Median Selection Time for *Technique* and *Distance* aggregated across *Direction* with 95% confidence intervals.

Figure 5 reveals an interesting tendency, namely that target distance is affecting the selection times of all *Techniques* in a similar way. It also shows that for larger targets, *Move* is clearly faster and *Tilt* is slightly faster than *Scroll*, which is consistent with the statistical results. Furthermore it shows that with the non-constant *Move* (and *Tilt*), participants had a mean selection time for target sizes 48 and 72 pixels of 1401 ms (and 1868 ms) compared to *Scroll* with a mean trial time of 2023 ms. This results in 31% improvement for *Move* (8% for *Tilt*). The improvement of *Move* for small targets was considerably smaller (13% compared to *Scroll*) whereas *Tilt* was even slower than *Scroll* with a loss of 15% which can be explained with the presence of the overshooting effect.

Error Rate

We counted errors for each trial in terms of wrong target selections and aggregated them to perform a $3 \times 2 \times 3$ (*Technique* × *Distance* × *Size*) within subjects analysis of variance. As expected, there were significant main effects for *Technique*, *Distance* and *Size* (all $p \leq .001$). We further found a *Technique* × *Distance* × *Size* interaction ($F_{4,44} = 3.143$, $p = .023$). Post hoc multiple means comparison tests revealed that *Scroll* is significantly better than *Tilt* for all *Distances* and *Sizes* (all $p < .03$). For the long *Distance*, it was also significantly better than *Move* (all $p < .02$). This supports our Hypothesis H3 as *Scroll* is significantly less error prone (see Figure 6) but also significantly slower compared to *Move* and *Tilt*.

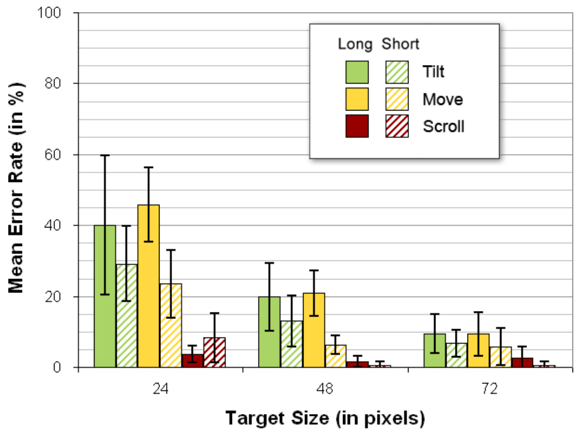

Figure 6. Mean Error Rates for *Technique* and *Distance*. The error bars represent 95% confidence intervals.

No significant differences in error rate were found between *Move* and *Tilt*. However, both techniques showed considerably high error rates. For the long *Distance*, *Move* had error rates of 46% (21%) for target *Sizes* of 24 pixels (48 pixels). Only for large targets the error rate of 9% was fairly low. An explanation for the high error rate of smaller targets is due to slight phone movement while pressing and releasing the phone's select button. *Tilt* showed nearly the same error rates for the long *Distance* and is explained by the overshooting effect when participants pressed the select button inside the target and released it when they had already left it (due to high movement speed of the pointer). However, this effect mostly occurs for small targets, which explains the fairly low error rate for large ones. For the short *Distance*, the error rates for *Move* (23% for 24 px, 6% for 48 and 72 px) were smaller than the ones for *Tilt* (29% for 24 px, 13% for 48 px and 7% for 72 px) but not significantly. Considering the high selection times and noticeably high error rates of *Tilt* compared to the other *Techniques* for small targets, the overshooting effect as predicted in our selection model is of great importance. Overcoming this effect could lead to better results for *Tilt*.

Subjective Ratings

In post-study questionnaires, participants ranked the *Techniques* on five-point Likert scales by fatigue effects, comfort, operation speed and accuracy. Unsurprisingly, participants did not perceive any fatigue for shoulder, arm and wrist when using *Scroll*. However, the perceived fatigue for fingers was higher. We argue that finger fatigue is strongly device-dependent as some phone models require more key pressure than others. Hence, this statement is valid only for our device.

Somewhat surprising, the general comfort did not show any trend towards a *Technique* indicating that all of them more or less perform in similar ways. We expected *Scroll* to outperform *Move* and *Tilt*, but considering the high finger fatigue and slow selection times, participants probably rated *Scroll* less convenient. However, *Scroll* was perceived as easier-to-use compared to the other *Techniques* (significant compared to *Tilt*). Rating *Tilt* as a hard-to-use technique can be explained by the participants' frustration during the user study. *Tilt* was nearly as fast as *Scroll*, but pointer movement was unpredictable.

Expected ratings were given for operation speed: both *Move* and *Tilt* were perceived much better, but only *Move* achieved a significant difference. Based on the error rates of *Move* and *Tilt* the subjective rating for accuracy is also not surprising, as *Scroll* received much higher scores (even significantly compared to *Move*). According to selection times and error rates, the fact that *Tilt* was rated slightly better than *Move* is surprising. One possible explanation could be that *Tilt* adds a "*skill*" component to the interaction, turning it into a candidate for game input.

Observations

During the study we observed the overshooting effect when using *Tilt* as described above. Participants constantly moved the pointer too far and had to go back to select the target. In general, this led to higher selection times as well as increased error rates. For the long *Distance*, this mainly occurred due to high cursor movement (steep tilting angle). For the short *Distance*, participants were sometimes not able to slowly accelerate the pointer leading to the same effect. *Scroll* and *Move* did not suffer from this effect. Figure 7 shows the overshooting effect for distant targets placed diagonally. For small targets, there is slight overshooting for each *Technique*, but *Tilt* seems to be the worst. For medium sized and large targets, overshooting decreases clearly for *Move* and *Scroll*, but is only slightly reduced for *Tilt*.

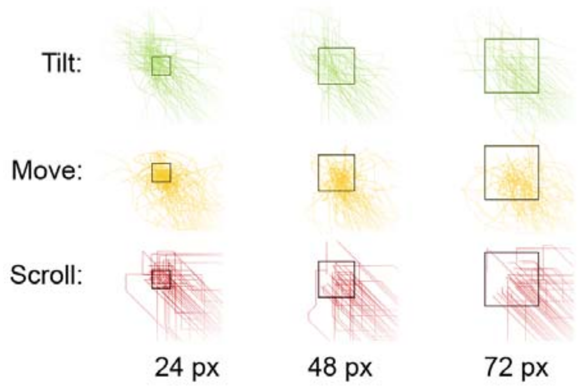

Figure 7. Overshooting Effect demonstrated for far, diagonal target placement (distance: 336 pixels, direction: NW).

For *Move* and *Tilt* we observed various postures for the phone leading to different fatigue ratings. Participants not moving their whole arm while using *Move* resulted in less perceived fatigue. When using hand and wrist movements only, less wrist fatigue was experienced. We observed that sooner or later, participants switched to wrist movement to decrease arm fatigue. Hence, the gorilla-arm-effect can be reduced using *Move* by turning the wrist only. Accuracy was increased when participants braced their arm on the body. We also observed how clutching (i.e., moving the phone without moving the pointer) was done when using *Move*. Most of the participants were able to use it without any problems. However, when moving the finger too slow on top of the lens or vice versa, our implementation did not recognize this action leading to a very high cursor movement. This only affected two participants. We are confident that this can be solved by using a more intelligent optical flow analysis algorithm.

Discussion

Our experimental results support all of our hypotheses. While *Move* and *Tilt* perform better in terms of selection time, they both suffer from high error rates. In addition, participants' subjective ratings are much better for *Scroll* regarding accuracy and overall ease. However, the ratings for general comfort are comparable for all *Techniques*. Given the fact that we could only test novice users, it is reasonable to assume that error rates might further drop after an extended period of usage. As stated by several participants, the "*skill*" component makes the non-constant techniques interesting candidates for game input. The reason for the relatively high error rates of *Tilt* mainly is the overshooting effect. This could be compensated by decreasing the maximum speed for the pointer at the expense of increased selection time. When using *Move*, the error rates were high due to slight phone movement when pressing and releasing the select button. One solution to this is turning off optical flow analysis once the button has been pressed and switching it on after the button has been released, which would limit the ability of dragging objects on the screen. A further solution could be to suspend motion analysis by skipping a single frame for both pressing and releasing the button.

As expected, the score for operation speed (*Move* and *Tilt*) was consistently high compared to *Scroll* in the subjective ratings. The speed of *Scroll* could be improved either by increasing the pointer's speed or by using a ballistic pointer behavior (at the expense of overshooting effects). For fatigue effects when using *Move* we have seen an increased rating if participants only turned their wrist to control the pointer. Participants would not need to move their whole arm reducing the gorilla-arm-effect known from interacting with vertically mounted displays.

CONCLUSIONS AND FUTURE WORK

In this paper, we have presented three different interaction techniques for continuously controlling a pointer on a remote large screen. These techniques rely on different CD ratio adjustments. In addition, we created a prediction model for selection times and comparatively evaluated the techniques with twelve participants to support this. Each technique revealed its respective strengths and weaknesses regarding selection time and error rates.

As expected, *Move* and *Tilt* allow fast selection times but at the expense of higher error rates. The error rates, however, can be reduced by adding a "snapping" behavior. This also would decrease the overshooting effects observed during the study. We also made interesting observations regarding the users' posture and the fatigue effects involved when using the techniques. This comparison gives a solid base for designing distributed applications for multi-device interaction that vary in controlling detail and overall interaction times. We hope that this study informs other people's work regarding the input modalities as well as large display interaction.

In the future, we plan to use these statistically solid results as well as the more casual observations to guide the design of further applications for multi-device interaction. One interesting direction is the combination of private and public information spaces represented by personal mobile phones and public displays. By allowing remote control of a public screen by still keeping private information on the mobile phone, we hope to gain new insights in how to combine the high visual output and individual input. This further raises interesting questions about multi-user interactions on large public screens by having one input control per user.

ACKNOWLEDGMENTS

This work has been funded by "Deutsche Forschungsgemeinschaft" (DFG). It has been conducted within the UbiLife project framework. The authors would like to thank TEKES and the participating companies for financial support, and the participants of our study for their time and patience. We further thank the reviewers for their valuable feedback. The second author would further like to thank the GETA graduate school and TES foundation for financing this research.

REFERENCES

Ballagas, R. BYOD: Bring Your Own Device. Workshop Ubiquitous Display Environments, UbiComp '04, 2004

Ballagas, R., Rohs, M., and Sheridan, J.G. Sweep and Point & Shoot: Phonecam-Based Interactions for Large Public Displays. In Proc. Of CHI 2005, 1200-1203

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

Boring, S., Altendorfer, M., Broll, G., Hilliges, O., and Butz, A. Shoot & Copy: Phonecam-Based Information Transfer from Public Displays onto Mobile Phones. Proc. Mobility '07, ACM Press (2007), 24-31

Card, S.K., MacKinlay, J.D. and Robertson, G.G. A Morphological Analysis of the Design Space of Input Devices. ACM Transactions on Information Systems, 9, 2 (1991), 99-122

Douglas, S.A., Kirkpatrick, A.E., and MacKenzie, I.S. Testing Pointing Device Performance and User Assessment with the ISO 9241, Part 9 Standard, Proc. CHI '99, ACM Press (1999), 215-222

Forlines, C., Balakrishnan, R., Beardsley, P., van Baar, J., and Raskar, R. Zoom-and-Pick: Facilitating Visual Zooming and Precision Pointing with Interactive Handheld Projectors, UIST '05, ACM (2005), 73-82

Hardy, R., and Rukzio, E. Touch & Interact: Touch-Based Interaction of Mobile Phones with Displays. Proc. of MobileHCI '08, ACM Press (2008), 245-254

Holleis, P., et al. Keystroke-Level Model for Advanced Mobile Phone Interaction. Proc. CHI '07, 1505-1514

Jiang, H., Ofek, E., Moraveji, N., and Shi, Y. Direct Pointer: Direct Manipulation Technique for Large-Display Interaction using Handheld Cameras. Proc. CHI '06, ACM (2006), 1107-1110

Madhavapeddy, A., Scott, D., Sharp, R., and Upton, E. Using Camera-Phones to Enhance Human Computer Interaction. Adj. Proc. UbiComp '04, 2004

Miyaoku, K., Higashino, S., and Tonomura, Y. C-Blink: A Hue-Difference-Based Light Signal Marker for Large Screen Interaction via any Mobile Terminal. Proc. UIST '04, ACM (2004)

Myers, B.A., Stiel, H., and Gargiulo, R. Collaboration using Multiple PDAs Connected to a PC. Proc. CSCW '98, ACM (1998), 285-294

Pears, N., Olivier, P., and Jackson, D. Display Registration for Device Interaction – a Proof of Principle Prototype. Proc. VisApp '08, 2008, 446-451

Reilly, D., Rodgers, M., Argue, R., Nunes, M., and Inkpen, K. Marked-up Maps: Combining Paper Maps and Electronic Information Resources. Personal and Ubiquitous Computing 10, 4 (2006), 215-226

Rohs, M., Schöning, J., Raubal, M., Essl, G., and Krüger, A. Map Navigation with Mobile Devices: Virtual versus Physical Movement with and without Context. Proc. ICMI 2007, ACM Press (2007), 146-153

Scheible, J., Ojala, T., and Coulton, P. MobiToss: A Novel Gesture Based Interface for Creating and Sharing Mobile Multimedia Art on Large Public Displays. Proc. ACM Multimedia, ACM Press (2008), 957-960

Silfverberg, M., MacKenzie, I.S., and Kauppinen, T. An Isometric Joystick as a Pointing Device for Handheld Information Terminals. Proc. GI '01, 2001, 119-126

Vajk, T., Bamford, W., Coulton, P., and Edwards, R. Using a Mobile Phone as a 'Wii like' Controller. Computer Games Technology, 2 (2008)

Vogel, D., and Baudisch, P. Shift: A Technique for Operating Pen-based Interfaces Using Touch. Proc. CHI '07, ACM Press (2007), 657-666

Want, R., Schilit, B.N., Adams, N.I., Gold, R., Petersen, K., Goldberg, D., Ellis, J.R., and Weiser, M. The ParcTab Ubiquitous Experiment. Technical Report CSL-95-1, PARC (1995)

Yatani, K., Tamura, K., Hiroki, K., Sugimoto, M., and Hashizume, H. Toss-It: Intuitive Information Transfer Techniques for Mobile Devices. Proc. CHI '05, ACM Press (2005), 1881-1884