Eye-Gaze Interaction for Mobile Phones

Heiko Drewes
Media Informatics Group
Amalienstr. 17, 80333 Munich, Germany
heiko.drewes@ifi.lmu.de

Alexander De Luca
Media Informatics Group
Amalienstr. 17, 80333 Munich, Germany
alexander.de.luca@ifi.lmu.de

Albrecht Schmidt
Frauenhofer IAIS; University of Bonn
Schloss Birlinghoven, St. Augustin, Germany
albrecht.schmidt@acm.org

ABSTRACT
In this paper, we discuss the use of eye-gaze tracking technology for mobile phones. In particular we investigate how gaze interaction can be used to control applications on handheld devices. In contrast to eye-tracking systems for desktop computers, mobile devices imply several problems like the intensity of light for outdoor use and calibration issues. Therefore, we compared two different approaches for controlling mobile phones with the eyes: standard eye-gaze interaction based on the dwell-time method and gaze gestures. Gaze gestures are a new concept, which we think has the potential to overcome many of these problems. We conducted a user study to see whether people are able to interact with applications using these approaches. The results confirm that eye-gaze interaction for mobile phones is attractive for the users and that the gaze gestures are an alternative method for eye-gaze based interaction.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Input devices and strategies.*

General Terms
Algorithms, Design, Reliability, Human Factors.

Keywords
Eye-Tracking, Mobile Phones, Eye-Gaze Interaction, Eye-Gestures.

1. INTRODUCTION
Nowadays, eye-tracking technology works with a video camera and image processing algorithms to detect the user's pupil. Most mobile phones sold today have a camera already built-in. Even more, many modern phones have a secondary camera on the front side as well which is included for video conferencing. Looking retrospectively at the development of mobile devices, it is noticeable that their processing power as well as the quality of their components increases steadily. Thus, it is only a matter of time until eye-tracking technology can be offered within these devices in software for virtually no extra costs.

Eye-gaze interaction could be a convenient way of controlling mobile devices. It works without mechanical contacts, which means a dirt free interaction. The eyes can move quickly and normally without fatigue. As small mobile devices do not have elaborated input abilities like a big keyboard for two-handed typing and moreover, they do not have any pointing device like a mouse or a touchpad, eye-gaze could be an additional modality for input available in the near future.

Eye-trackers are used for more than twenty years now as outlined in [5]. They are used for computer input in the field of accessibility providing means of input for motor impaired people. A further use is in advertising, communication design and usability analysis. Since a couple of years commercially available desktop eye-trackers work reliable.

Nevertheless, eye-tracking technology is not free of problems. One of the problems is the inaccuracy of the eye-gaze position caused by the jitter of the eye. As a consequence all eye-gaze interaction objects need sizes of at least one degree visual angle. In the distance of an arm length, which is a typical distance for mobile phones, this size is a little bit less than the size of the thumb nail. This means that there can be only few interaction objects on the small-sized displays of mobile phones. Another problem is the freedom of move in front of an eye-tracker. Comfortable eye-trackers do not rely on tracking the eye only, but also on head-tracking to achieve this, but work on stationary systems. For handheld devices this problem becomes even more severe because they can change the direction of the camera very quickly due to movement of the hand. Thus dwell-time based eye-tracking interaction on mobile devices will require some form of head-tracking. A further problem lies in the use of an infrared LED to gain the glint (reflection spot on the eyeball), which is used as a reference point for the tracking. For outdoor use in bright sun the detection of the reflection spot becomes problematic. Finally the need of a calibration process is a problem for mobile devices, because it comprises too much effort for just a short interaction and the calibration is lost after putting the mobile phone to the ear or back into the pocket. Such attention shifts are very common for mobile devices.

All the problems mentioned above mainly result in accuracy problems of the tracked gaze position. For this reason we investigate in our research two different approaches. One approach is the eye-gaze interaction based on the dwell-time method, for which accuracy problems need to be solved. The other approach is an innovative concept based on gaze gestures. Gestures do not require an absolute position and hence we think

this approach has the potential to overcome the accuracy problems.

In the next section we will describe the basics of eye-tracking on mobile phones. It contains descriptions of common problems and finally explains the advantages of gaze gestures and how they work. After that, our prototype hardware and software as well as the whole experimental setup will be described in detail. This has been used for a two-part evaluation of which the procedure and results are explained. At the end, we will summarize our work and give an outlook on activities that have been planned as a follow up to this paper.

2. EYE-TRACKING ON MOBILE PHONES

2.1 Basics

Eye-tracking technology for interaction with mobile phones is not yet available. One reason is the lack of processing power to handle video streams on these devices in real-time. But it is foreseeable by extrapolating the technological trend, that the required processing power will be available over the next years.

Another issue typical for mobile phones is outdoor use. This implies varying light conditions and saturation effects by bright sun light, which makes it difficult to reliably detect the pupil and the glint within a camera picture. Approaches with differential pictures and infrared illumination synchronized with the frame rate of the video camera or the use of polarized light are promising [7]. Therewith, eye-tracking for at least most light conditions except the extremes should be achievable.

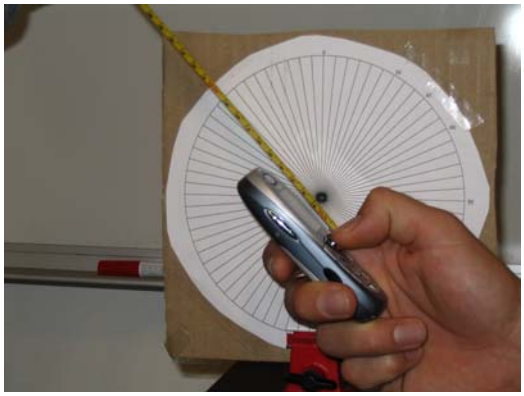


Figure 1: Measuring tilt and roll angle for standard interaction with mobile phones.

As mentioned in the introduction, eye-trackers, which allow free movement of the head, also depend on head-tracking. Commercially available eye-trackers for free head movements are desktop systems and the camera stays in a stable position. For a mobile phone doing eye-tracking the situation is different. The camera is built into the device, which is hold by the hand of the user and naturally, the hand moves constantly. Especially angular movement cause very quick changes in the camera picture.

To find the camera resolution required to achieve the same accuracy as the commercial eye-tracker, we did some experiments on the angle a mobile phone is held during standard operation. In Figure 1, a volunteer is asked to perform some standard tasks on his mobile phone in from of a setup allowing us to record the angles at which the mobile phone is held. The recorded data could be used to estimate the angles in which mobile phones are used. The recorded data could be used to establish vague estimations on the angles, mobile phones are used in. From our observation we estimated the tilt angle to be about 20° and a roll angle of about 10°.

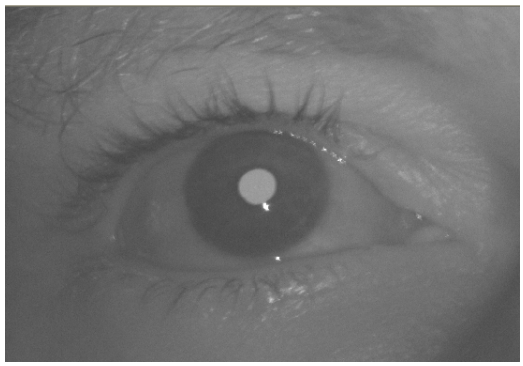
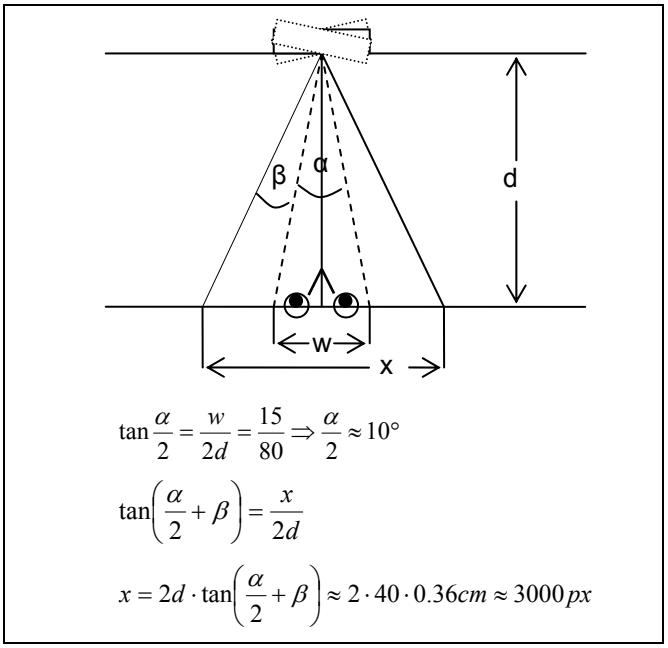


Figure 2: An image delivered by the camera of the eye-tracker.

The camera of our eye-tracker (ERICA [1]) has a video stream resolution of 640 x 480 pixels and projects an area of about 6 x 4.5 cm (see Figure 2). This means for 1cm we have approximately 100 pixels.



$$\tan\frac{\alpha}{2} = \frac{w}{2d} = \frac{15}{80} \Rightarrow \frac{\alpha}{2} \approx 10°$$

$$\tan\left(\frac{\alpha}{2} + \beta\right) = \frac{x}{2d}$$

$$x = 2d \cdot \tan\left(\frac{\alpha}{2} + \beta\right) \approx 2 \cdot 40 \cdot 0.36cm \approx 3000\,px$$

Figure 3: Estimation of the camera resolution necessary for a roll angle of 10°.

In the following example calculation we estimate the camera resolution on the handheld device to get the same resolution as the commercial tracker we used. With a face width w of 15 cm, a typical distance d of 40 cm and a roll angle β of 10° the necessary camera resolution in horizontal direction results to approximately 3000 pixels (see Figure 3). Already today, such camera resolutions are available in mobile phones, at least for still images. The supported video streams stay close under the required resolution.

[1] http://www.eyeresponse.com/

2.2 The Concept of Gaze Gestures

Gestures are a well-known concept for computer human interaction. Typical examples of gestures for manual input are Unistroke [2] and Cirrin [6]. Usually, a gesture consists of a sequence of strokes, which are performed in a sequential time order.

One advantage of gestures is that the number of available commands for one screen can be increased indefinitely by simply increasing the set of usable gestures (there is obviously a practical limit set by the complexity of the gesture for the user). If in contrast, commands are selected by looking at a specific area on the screen, for example buttons, as used in dwell-time based gaze interaction, the number of available commands is limited by the size of the screen and the resolution that can be achieved by eye-gaze, i.e. the number of distinct areas that can be placed on it.

That gestures do not require screen real estate is one of central reasons, why gestures are an appropriate interaction technique for small devices. This is very important because dwell-time based eye-gaze interfaces result in big sizes for interaction objects, i.e. large buttons and menu items. Especially on small displays of current mobile phones this is a major obstacle. Additionally using gestures would free the screen so it can be used for outputs of the applications. These observations are the reason why gaze gestures are worth to be examined in more detail..

2.2.1 The Gesture Algorithm

When looking for gesture recognition algorithms we assessed the popular mouse gesture plug-in[2] for the Firefox browser. The algorithm has been developed for mouse interaction and takes the current *x-y* position, calculates the difference to a starting point and does an integer division for both coordinates by the grid size *s*. If the result is different from zero for at least one coordinate, the algorithm puts out a character representing the direction of the movement and the current position becomes the starting point for the next movement. Otherwise the algorithm waits for the next coordinates.

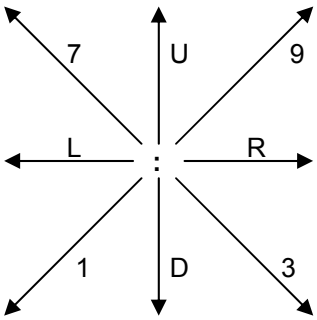


Figure 4: The characters representing the directions.

Figure 5 depicts an example for three different strokes and how coordinates are converted to a stroke, regarding the chosen grid size. Horizontal and vertical directions are represented by the letters U, D, L, and R for up, down, left and right respectively. Diagonal directions are given by 1, 3, 7, and 9 corresponding to the familiar layout of the number pad on a standard keyboard. The whole direction representation set is depicted in Figure 4. In a second step, the algorithm recognizes the actual gestures by comparing the produced string that represents the movements with a given gesture pattern. If the gesture pattern matches, an action can be triggered.

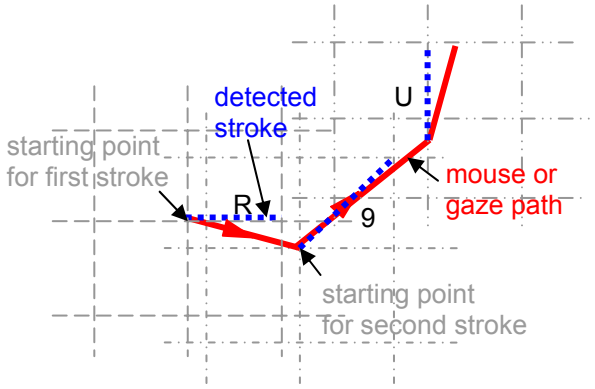


Figure 5: Stroke detection.

Another gesture recognition algorithm that inspired our work is EdgeWrite [8]. This algorithm uses the order in which four points, the corners of a square, are reached by the input device. The concept is depicted in Figure 6. It is easy to see, that all EdgeWrite gestures can be expressed with the tokens from the mouse gesture algorithm and consequently are a subset of the mouse gestures. This provides capability to recognize a complex alphabet.

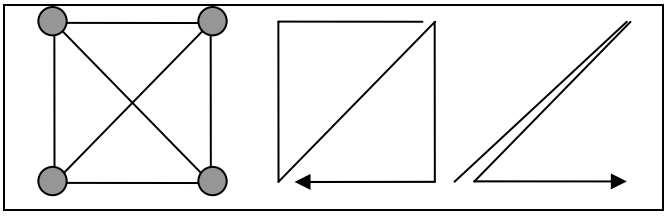


Figure 6: The four corners and the six connecting lines used for the EdgeWrite gestures and two examples for EdgeWritegestures (digits 0 and 2).

2.2.2 Implementation of Gaze Gesture Recognition

For our purpose, the basic approach used when working with mouse gestures is not feasible at all. The mouse gesture recognition algorithm is only used when a gesture key is pressed. This signals the algorithm that the user intends to do a gesture. In the basic setup, this key is the right mouse key. Of course, this has to be performed by the user. When looking at pure eye-gesture interaction, using a gesture key is not possible. Therefore, for our application of gaze gestures, we changed the algorithm to continuous recognition. That is, gestures are recognized in the moment they are performed. However for this to work sensible it means that we have to discriminate natural eye movement from gaze gestures. To achieve this, we enhanced the algorithm with a timeout value *t*. When the algorithm did not output a stroke within the time *t*, it outputs a colon as a timeout token as depicted in the middle of Figure 4. The timeout resets the gesture recognition on every fixation that lasts longer than this threshold.

2.2.3 The Robustness of Gaze Gestures

The gesture algorithm described above uses relative movements only. As a consequence it is greatly immune against any calibration shift caused by moving the mobile phone relative to the eyes. By choosing the size *s* of the grid, the size of the

[2] http://optimoz.mozdev.org/gestures/

gestures and the robustness on accuracy can be influenced. With a large grid size the gaze gestures are insensitive to inaccuracies up to size *s*.

These properties make it possible to use an eye-tracker without any calibration, because the normal calibration process estimates the scaling and the translation to map the x-y position of the pupil to screen coordinates. As the gestures only use relative movements, absolute coordinates and thus the translation part is not needed at all. The scaling converts the sizes of movements of the pupil into sizes on the display and depends only on the distance of the eye from the display. The distance of the eye to the mobile phone does not vary much during operation and consequently it is possible to work with a fixed scaling factor throughout the whole interaction.

3. PROTOTYPE

Dealing with the new interaction technique of gaze gestures, there are neither ready-to-use hardware components available nor are there applications that make use of it. Hence to evaluate such a new approach we had to create a prototypical setup. For our prototype, we were combining existing hardware to create a prototypical device for our purpose. Naturally, we had to create the software ourselves as well. In this section, we will describe both, the experimental setup that provides the basis for acquiring the interaction and the design of the application.

3.1 Experimental Setup

When designing the experimental setup we considered mainly two different options. One is based on a mobile phone emulator on the PC and the second one uses a real phone. During the development and testing of the applications we realized that there is a significant difference in the user's perception between these options. Having a real phone (even if stationary in front of the screen) conveys a much more realistic interaction experience than the emulator. Additionally having a real phone provided us with the possibility to explore this new interaction technique on different physical devices not looking like the emulator skin.

To create this realistic setting we decided to use a commercial eye-tracker in combination with a mobile phone (Nokia N80) that supports wireless LAN for the experimental setup. Additionally we had a wooden apparatus with a hook-and-loop fastener attached to a screen in front of the participants to provide a fixed position for the phone. The camera of the eye-tracker was located underneath it. Since the eye-tracker we are using does not provide head tracking, we were using a chin rest for stabilizing the participants' heads. The whole experimental setup with a participant is shown in Figure 7.

It is important to note that the setting has been used since it exploits available hardware. Furthermore, it is appropriate for verifying the interaction techniques.

During several sessions, the users held the phone in their hands and we could see the impact of movement on the interaction performance of both techniques. As the participants were not able to hold the hand still, which lead to a broken calibration on the phone, we used the apparatus for fixing the mobile phone to the screen during the study. As mentioned before, for the gesture algorithm small hand movements did not matter at all and thus, it still worked. But the classical dwell-time approach did not work anymore if the movement had been in the size of the interaction object, i.e. 2 cm, which is the height of the scroll button shown in Figure 9. Therefore, we decided to use the fixation of the mobile device for the final user study.

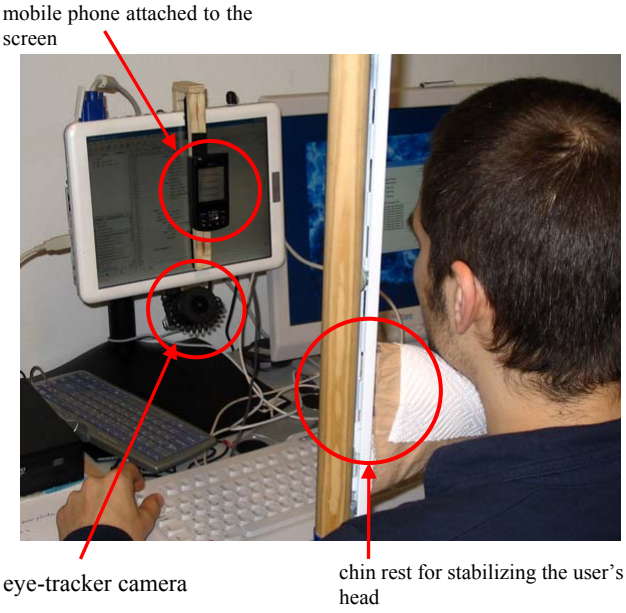


Figure 7: Experimental Setup with user.

To react on gaze-input, the mobile phone requires the coordinates generated by the eye-tracker. Thus, for transmitting the gaze coordinates to the mobile phone, we developed software, written in C++, which acts as a simple socket server and sends out the coordinates to every connected device. The mobile phone was connected to this server using its built-in WLAN functionality. The eye-tracking application on the mobile phone was written in Java ME, which is Java for small devices with limited resources.

Due to our setup, the calibration becomes a little more complicated. Therefore, the prototype uses a two-step calibration mechanism. At first, the users have to be calibrated to the eye-tracker respectively its screen using the standard calibration mechanism of ERICA. After that they need to be calibrated to the mobile phone screen. For the second step, we had to implement our own calibration mechanism which will be explained in the next section.

3.2 Mobile Phone Application

The design rational for the prototype application has been to provide means for evaluating two different types of gaze interaction with mobile phones. On the one hand the classical eye-gaze interaction that uses dwell-time for command invocation is considered. On the other hand is the translation from eye movements into commands, which we call gaze gesture interaction, is assessed. This section describes the application on the mobile phone that implements these interactions.

3.2.1 Calibration

In the prototype setup, users are calibrated to the screen of the commercial eye-tracking system. This means, the users' gaze is translated to a pixel coordinate relative to this screen. Since the mobile phone has a different resolution and potentially a different absolute position, the scaling for both, the x and y-axis as well as the offset for both axes have to be calculated. This is required to

translate each screen-coordinate to its corresponding equivalent on the mobile device.

Therefore, we use a simple two-point calibration approach as shown in Figure 8. Users have to look at a point in the upper left corner for at least two seconds and afterwards in the lower right corner of the screen for the same amount of time. The results are two coordinates in the resolution of the eye-tracker screen. These are used to calculate the scaling and offset for both axes.

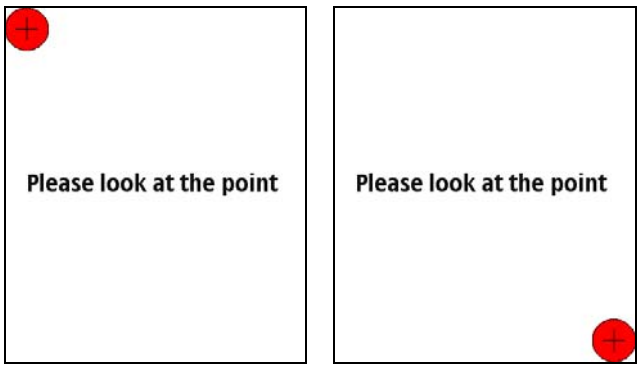


Figure 8: Calibration screen. Step 1(left) and step 2(right).

3.2.2 *Classical Eye-Gaze*

For the classical eye-gaze part of the mobile application, we decided to use an application people are used to: a phone book, which can be used to search for people and call them. The implementation consists of a list of names, ordered by first name, which can be scrolled up and down and of which names can be chosen to start a call.

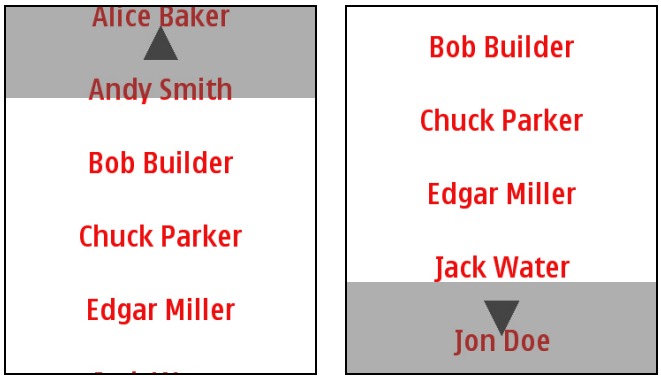


Figure 9: Phone book application.

The interaction with this list is based on dwell-time, that is, commands are executed by looking at a specific area for longer than a predefined threshold. Figure 9 shows the screen and its three interaction areas. The upper and lower fourth of the screen are used for scrolling. If a user looks at one of those areas for more than two seconds the list scrolls up respectively down. The longer the look remains in the area, the faster the list scrolls until the maximum speed has been reached. To activate a name, the inner 1/2 part of the screen has to be used. If a name is inside of this area it can be chosen by looking at it for the same threshold time as for activating the scrolling.

Visual feedback is provided to the users to support them. If the gaze is pointed at an area, it is highlighted more intense. Therewith, it indicates, that an interaction is about to happen.

The application includes a safety mechanism, for the case a wrong part is invoked from the list. Whenever a name is chosen, either with or without purpose, the users are asked to decide whether or not to call the person using a yes-or-no dialogue with two activation areas as shown in Figure 10. For this dialogue, dwell-time interaction is used as well. That is, looking at yes or no for a specific amount of time will activate the respective command.

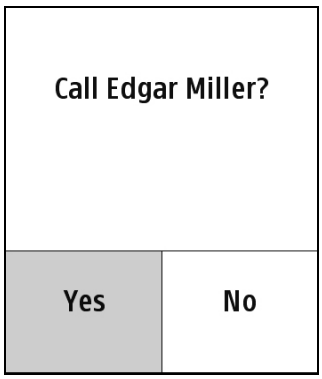


Figure 10: Call screen with two choices.

3.2.3 *Gaze-Gestures*

The concept of gaze-gesture interaction differs fundamentally from the classical approach using dwell-time. Instead of fixed absolute coordinates it only depends on relative coordinates, because not the position of a pattern but only the pattern itself is of importance for the invocation of a specific command. Implicitly, this means that for eye gesture recognition no exact calibration is required and movements of the mobile phone do not matter. As mentioned before, regarding the factor mobility, this is a huge advantage compared to the classical approach.

Since for gestures the length of one eye-stroke is required, a minimum value, the grid size s, must be defined. For our prototype we decided to use 60% of the smaller value of the mobile phone display extent, the screen width or the screen height. That is, the formula looks like this:

$$s = 0.6 * Min(width, height)$$

It is reasonable to assume, that the distances covered by eye movements are smaller than the display, at least in horizontal direction. In vertical direction the gaze position can move from the display to the keys. Choosing a grid size close to the dimension of the display increases the chances to distinguish reliably between intentional gestures and natural eye movements. This also means that the corners of the mobile phone display are perfect fixation points for the gestures. The grid size chosen for our prototype (Nokia N80) was 235 pixels on the mobile phone display, which translates to approximately 80 pixels on the eye-tracker or a visual angle of 2.2°. Additionally to this value, a timeout has to be specified which interrupts a sequence of strokes. That is, if the user fixates a position for longer than the timeout, the gesture recognition produces the timeout token. For our prototype we set this value to 1.5 seconds.

To test the gaze gestures we implemented a simple open and close interaction in our prototype. One gaze gesture opens the integrated web browser of the mobile phone, while another gesture closes the browser. The gaze gestures used are depicted in Figure 11.

We designed the gestures so that the edges and corners of the mobile phone screen can be used as help points and help lines. The layout and complexity of the gestures used are a tradeoff. On the one hand they should be short, simple and easy to remember. On the other hand the gesture should not occur in the eye movements during normal interaction. To check the occurrences of gestures while interacting with a mobile phone, we evaluated eye movements for standard mobile phone usage, such as writing SMS. The results are presented in the section on evaluation.

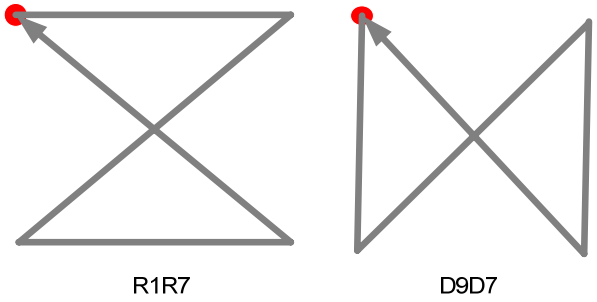


Figure 11: Open browser (left) and close browser (right) gestures. The red dot marks the start-point.

In contrast to the dwell-time interaction, there is no intermediate feedback for the gestures. The only feedback is not on the gesture but on the action (e.g. the browser is started respectively closed).

4. EVALUATION

In our work, we want to find out whether eye-gaze interaction is an appropriate way of interacting with a mobile phone and how it compares to dwell-time based eye gaze interaction. Our research questions include: Are people able to perform eye-gaze interaction on mobile phones? How do people rate the different interaction types? Which interaction type is preferred by the users?

To find the answers to these questions, we performed a two-phased evaluation using the prototype described earlier. At first, we conducted a user study, for which we let the users perform the two different tasks of our mobile application and asked them to answer a questionnaire. In the second phase, a big dataset of user gazes for normal mobile phone interaction has been analyzed to find gesture appearances.

This section describes the conduction of the user study and the evaluation. At the end the various results of both evaluations will be listed and explained in detail.

4.1 Procedure

4.1.1 User Study

To evaluate the interaction types, we conducted a user study with eight participants in the age between 23 and 50. The average age was 30 years. Two participants were female, six were male. All of them owned a mobile phone, thus, they were used to dealing with mobile phones. Only one of them had ever tried eye-gaze interaction before.

In the first phase, the general idea of eye-gaze interaction was explained to each participant, whereupon, they were calibrated to the commercial eye-tracker. After that, the mobile phone was attached to the screen and the second calibration was performed.

For the study, each user had to perform two tasks. At first, the participants were told to select a predefined name from the phone book with dwell-time interaction for scrolling and selecting. The name was chosen by the tester and given to each participant after the functionality of the application was explained to them. The condition was that the name may not be located at the first two positions nor at the last two the list because they are the easiest to choose since scrolling is only possible in one direction for them. Without telling them, the time has been logged for each user. In particular we recorded how long it took them to choose the name.

The second task was for testing gaze gesture interaction. After the idea and functionality of the gaze gesture application had been explained to the participants, they were told to open the browser with the gesture. Therefore, each participant could try to execute the gesture until they finally managed to do so. When they managed to open the browser, they were asked to perform the gaze gesture for closing it again.

At the end of the practical part, the participants were asked to freely answer questions prepared in a questionnaire. Questions included demographic data and preferences regarding the interaction types.

4.1.2 Gesture Occurrence Evaluation

The main goal of the gaze gesture occurrence evaluation was to find out, whether the gaze gestures used in the study appear in standard mobile phone interactions. Therefore, we were able to access a dataset that was created during the work for [3], in which Holleis et al. were creating a keystroke level model for advanced mobile phone interaction. For their evaluation, they recorded the gaze-coordinates of 11 persons for three different standard tasks: writing a text message, changing the ring tone of the mobile phone and setting the alarm clock. Each participant performed these tasks with his or her own mobile phone. Fortunately, they used the same commercial eye-tracker and thus the same data format. On this dataset, we applied the same algorithm that we used for the gesture recognition.

4.2 Results

4.2.1 Execution

Each participant was able to perform both tasks without greater problems. For the phone book selection, the fastest participant took 12 seconds to call the person while the slowest one needed 37 seconds. Two persons accidentally chose the wrong person first but had no problem recovering from their error. To perform a single gesture, the fastest participant needed approximately 300ms per stroke and the slowest one needed 860 ms per stroke. This means that setting the upper time limit to 1000 ms would not decrease the gesture recognition efficiency but would decrease the number of unintended gestures. Thus, we think that for future experiments, a threshold of 1 second for each stroke would be the most appropriate.

4.2.2 User Preferences

After the practical part, one of the first questions for the participants was about the interaction type they preferred and why. As depicted in Figure 12, two participants preferred gestures while the majority of five people tended to the dwell-time interaction. Only one participant was undecided. Amongst the two participants that favored the gestures, one noted that he uses

mouse gestures with an internet browser and thus, knows how helpful they can become if one gets used to them.

Consequent to this question, the participants were asked about advantages and disadvantages of the respective interaction types. For dwell-time, it was noted by five people (four of them favoring dwell time) that for them it is a more natural interaction. One reason for this was what one called it: similar to the interaction one would have done with the hands. Hence, they rated the gestures to be more complex and stated that it could be hard to remember the gestures, even more, if they get too complex.

Two participants, who stated that they liked the gesture approach, said that they found it hard to imagine whether this concept has the potential to support manifold applications like dwell-time. They named this limitation as a disadvantage of gaze-gestures.

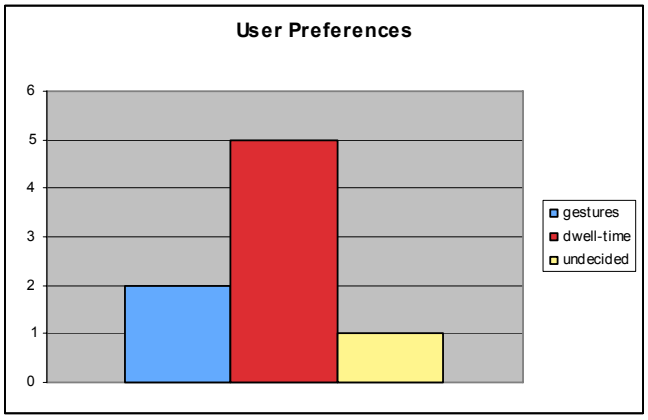


Figure 12: Users favoring a specific interaction type.

The biggest disadvantage of dwell-time lies in the way, in which people normally interact with computer devices. They read and then they interact. The problem is that a look that is longer than the threshold invokes an action. Thus, people get stressed. Five participants explicitly mentioned that they found it hard to "not perform" an action when they just wanted to read the names in the list for example. This effect is well-known and first mentioned by Jacob [4], who named this the "Midas Touch Problem". Wherever you look, something will happen. Two of the participant described themselves "in panic" when they were forced to look away from a specific point to avoid interaction. It seems important to mention, that the two people favoring the gestures were amongst the five mentioning this problem. This weakness shows the power of gestures, since for them, this problem simply does not exist. Gestures and thus commands are only invoked if the user intends to do so.

Our results show, that the classical gaze-interaction approach using dwell-time is by far the more intuitive one. This is promoted by its characteristic of being a direct interaction, which enables feedback mechanisms. Nevertheless, the gesture approach has proven to be an appropriate alternative for gaze interaction. Two participants even favored it mostly due to its robustness compared to the dwell-time method. Moreover, it is a two-sided robustness. On the one hand, it is robust to input errors, since it is unlikely that unintended gestures appear, as explained in the next chapter. On the other hand, it does not depend on an exact calibration. For mobile phones, this is an undeniable advantage, since it lies in the nature of mobile devices that they are constantly moved and cannot be positioned fix in front of the users' eyes.

4.2.3 Gestures Occurrence

Overall, the dataset available to us, contained 37 minutes of eye-gaze interaction with mobile phones. On this set, we applied our gesture algorithm with the same values that we used for our application. The maximum time threshold (t) was set to 1.5 seconds and the grid size s was set to 80 pixels (in the pixel units of the commercial eye tracker ERICA). During the 37 minutes of recorded data, there was only one appearance of one gesture, the one for opening the browser (see Figure 11 right).

Table 1: Gesture string for the same dataset of 95 seconds eye-movement but with different parameters.

Parameters	Gesture String
s=60, t=1500	7LRDU37RLRD7::::RL::RDUL3U::LRLRL:UD37::::::RD7::RL::UD::UDRL:::RL37DU::::::::::RLRDL::DU37:3L37:U:DU
s=80, t=1500	7LDU3D7::3U:::::::DU:DU:::RL:37::::::RL:R:::L:::R:L::RLRLDU::::::::::RLRDL::::D::::UDU
s=100 t=1000	7LDU3::::DU::::::::::::::::::::RL::DU::::::::::RL:::R:::L::::::::::::::D::::U::D::::::::::::UL:::::::::37::::R

In our algorithm the value used for the minimum stroke depends on the screen size of the mobile phone: the smaller the mobile phone screen, the smaller the value. Thus, we decided to run the test a second time, with a smaller value as well, to see if the number of unintended gestures increases. Table 1 shows the gesture strings for the same dataset of coordinates but for different grid sizes. The table explicitly states why a second test row is necessary, since for the smaller grid size s, more strokes and thus, more possible unintended gestures occur As expected, setting the minimum stroke value to 60 pixels three unintentional occurrences of gaze gestures, this time of the closing gesture, were found in the 37 minutes of interaction, which still is a very small amount. Table 2 presents these results.

Table 2: Occurrences of unintended gestures during 37 minutes for three different interaction tasks.

	R1R7	D9D7
s=60 t=1500	0	3
s=80 t=1500	1	0
s=100 t=1000	0	0

The calculations with different parameter settings show that the chance of unintended invocation by gaze gestures decreases if the value s is increased or t is decreased. This means that adjusting these values minimizes the risk of unwillingly executed

commands by gaze gestures. We showed that in our study, no stroke was performed that took the participant longer than 860ms. Thus, our threshold of 1500ms was set way too high. To be on the safe side, we propose a new value *t* of 1000ms, i.e. one second. We set *s* to 80 pixels because we estimated 60% of the smaller extent of the mobile phone's screen to be a good value. However, all participants used the corners of the screen to perform their gesture. From the analysis with different parameters we suggest to use values of *s*=100px, which is 80% of the width of our prototype mobile phone and *t*=1000ms. With these values no unintended occurrences of the gaze gestures used in the study were found as see in Table 2.

Even though our results show that the gestures are not likely to appear in normal eye-gaze interaction, they can. Consequently, commands may be invoked unwillingly. Thus, we propose an improvement to our algorithm. Not every gesture is needed in every context. For example a gesture for opening an application makes no sense when the application is currently open. The idea is to use context-sensitive gestures instead of the current approach. That is, a gesture is only active in a specific context and will be inactive in any other case. Therewith, the risk of performing a gesture without purpose is reduced to a minimum.

5. DISCUSSION AND FUTURE WORK

In this paper, we described our work conducted to evaluate eye gaze interaction as a new input method for mobile phones. We analyzed and compared a dwell time based gaze interaction approach to a completely new concept that we refer to as gaze-gestures. Gaze gestures have several advantages compared to the interaction based on dwell-time.

Most notably:

- The gestures are robust to movements or the absolute position of the user's gaze, since only relative movements are considered. This makes it possible to use this interaction technique without exact calibration.
- Gaze gestures are unlikely to be executed unwillingly and the number of gestures that can be practically used is very large.
- No screen real estate is required for eye-gesture interaction.

In a user study, we were able to show that users liked the idea of gaze interaction with mobile phones and none of the participants had grave problems performing the tasks given to them. For further evaluation of the gaze gestures concept, we used an existing dataset of 37 minutes of standard interaction with mobile phones to see, whether the chosen gestures could be performed unintended. The appearance rate was very low with a number of one occurrence for our gestures.

From the evaluation, we learned that the values for stroke size and the time threshold highly influence the occurrence rate. Thus, combined with the time measurements from the user study we determined the optimal values to be s=100 pixel and t=1000ms. For further studies, these values should be used. Although the majority of participants preferred the classical gaze interaction, our evaluation shows, that gestures have a great potential for the use with mobile phones.

Even though, the concept has proven useful, there are still technical challenges to solve. For dwell-time based interaction, a high accuracy of the tracking has to be reached combined with auto calibration. Since for gestures these limitations do not exist, they could become faster available because only a low accuracy and low cost tracker is required. In [1] Dickie et al. are using a low cost solution that could most probably be extended to be used for facilitating gaze gesture interaction. Even though a movable prototype was not needed for our evaluation, since it concerned the interaction techniques and their applicability to small screen devices, for future studies, we are planning to build a hand-held prototype to validate our concept in outdoor studies.

In the evaluation, some participants mentioned that gestures may not be appropriate for a big range of applications, but we think they are. Therefore, one of the next logical steps will be to develop and, more importantly, evaluate multiple domains of eye-gesture interaction. Ideally, they should include 1:1 realizations of dwell-time controllable applications, which would allow a much more intense comparison of the two approaches.

6. ACKNOWLEDGMENTS

We would like to thank Nokia, Finland that provided the hardware used for this prototype.

7. REFERENCES

[1] Dickie, C. et al. 2005. eyeLook: using attention to facilitate mobile media consumption. In Proceedings of the 18th Annual ACM Symposium on User interface Software and Technology (Seattle, WA, USA, October 23 - 26, 2005). UIST '05. ACM Press, New York, NY, 103-106.

[2] Goldberg, D.; Richardson, C. (1993): Touch-typing with a stylus. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems CHI '93. S. 80 – 87.

[3] Holleis, P. et al. Keystroke-Level Model for Advanced Mobile Phone Interaction. To appear in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems 2007 (CHI '07), San Jose, CA, USA, April/May 2007.

[4] Jacob, R. J. 1990. What you look at is what you get: eye movement-based interaction techniques. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Empowering People (Seattle, Washington, United States, April 01 - 05, 1990). J. C. Chew and J. Whiteside, Eds. CHI '90. ACM Press, New York, NY, 11-18.

[5] Majaranta, P. and Räihä, J. Twenty Years of Eye Typing: Systems and Design Issues, Proceedings of Eye Tracking Research and Applications [ETRA2002], pages 15-22, New Orleans LA, ACM, 2002.

[6] Mankoff, J. et al. Cirrin: a word-level unistroke keyboard for pen input. In Proceedings of the 11th Annual ACM Symposium on User interface Software and Technology (San Francisco, California, United States, November 01 - 04, 1998). UIST '98. ACM Press, New York, NY, 213-214.

[7] Morimoto CH, Koons D, Amir A, Flickner M (1999) Frame-rate pupil detector and gaze tracker. In: Proceedings of the IEEE ICCV'99 frame-rate workshop.

[8] Wobbrock, J. O., Myers, B. A., and Kembel, J. A.: EdgeWrite: a stylus-based text entry method designed for high accuracy and stability of motion. UIST '03. (2003), 61-70.