A new context: Screen to face distance

Immanuel König, Philipp Beau, Klaus David
University Kassel
Wilhelmshöher Allee 73
Kassel, 34121, Germany
comtec@comtec.eecs.uni-kassel.de

Abstract—**Most would agree: human vision is the most important of the five senses. Tragically many elderly people lose their vision due to incurable diseases which could have been avoided if diagnosed early enough. Fortunately some of these diseases can be diagnosed or at least their symptoms detected with the use of simple tests. The use of smartphone or tablet applications have become common for these tests, so eye diseases can be detected early and even at home. However, none of the smartphone or tablet applications considers the screen to face distance of a person doing an eye test to be an important parameter for an eye test. In this paper we present an algorithm to derive a new context: the smartphone user's screen to face distance. Our algorithm utilizes the smartphone front camera and an eye detection algorithm. After initializing the algorithm with person specific values, the algorithm continuously measures the eye to eye distance to derive the user's actual screen to face distance. We also present an investigation on the algorithm accuracy and speed, which shows: a smartphone based screen to face distance measurement is possible in the distance range from 19cm to 94cm with a maximum deviation of 2.1cm and at a rate of three distance measurements per second.**

Keywords—face to screen distance; context; smartphone; tablet

I. Introduction

Vision is one of the most important human senses. Tragically many elderly persons contract eye diseases, their vision deteriorates and some even lose their sight. Age-related macular degeneration (AMD), for example, is an eye disease which can be treated if diagnosed early enough [1]. If diagnosed too late, it can cause irreparable eye damage. Fortunately AMD can be diagnosed with simple tests, even in the absence of medical staff [2]. These simple tests are offered by some smartphone and tablet applications, which can help to diagnose or at least indicate AMD and other eye disease even at home. But today's smartphone and tablet applications do not consider the face to screen distance of a person doing an eye test. By ignoring this parameter test results easily run the risk of being inaccurate, and an undetected eye disease can cause irreparable damage. An automatically derived screen to face distance context can help to make such measurements more reliable and therefore help to prevent irreparable eye damage.

A screen to face distance context is not only advantageous for medical applications but also for designing more user-friendly user interfaces for smartphone and tablet applications. The font size in an eBook reader on a smartphone for example, can be adjusted automatically using the screen to face distance context. This can enable a more comfortable reading experience. There are numerous further applications based on the screen face distance context, e.g. what to display at all, or whether text, of colored buttons, or switching to voice and many more.

Smartphones and tablets are used to collect multiple user contexts. The range of user contexts reaches from the mode of transportation [3] to the user activities like "sitting", "standing", or "walking" [4], or the user's location[5], just to name a few. The user contexts are collected utilizing a variety of smartphone and tablet built-in sensors, like the accelerometer, gyroscope, compass, GPS, barometer, light sensor, proximity sensor, microphones, and cameras. Nevertheless, in the last decade most user contexts concentrated on the user's position or movement, the cameras of smartphone or tablet were seldom used as a context source.

By all indications from early 2012 the built-in cameras of smartphones and tablets started being used more and more often as sensors. Google introduced face detection as part of their Android 4.0 API in 2012. This feature can be used to detect faces not only on pictures but also with the cameras of smartphones or tablets. Furthermore, Samsung introduced 2013 three new smartphone features utilizing the smartphone or tablet front camera: *Smart Stay*, *Smart Scroll,* and *Smart Pause*. *Smart Stay* prevents the screen from locking as long as a user looks to the phone. *Smart Scroll* scrolls the screen content when the smartphone or tablet user looks at a screen corner. *Smart Pause* pauses a video played on a smartphone or tablet when the user looks away from the screen. All three features continuously take pictures and examine them with eye detection and gaze tracking algorithms. However, the smartphone or tablet manufacturers do not use the front camera for a face to screen distance measurement.

In this paper, we present an algorithm for a new context: screen to face distance for smartphones and tablets. To the best of our knowledge, this has not been published before. We contribute an algorithm to calculate the smartphone user's screen to face distance. We also validate the accuracy and speed of the algorithm by an experiment with a typical smartphone.

The remainder of this paper is organized as follows: Section II gives an overview on previous work and methods concerning the screen to face distance measurement. In Section III the proposed algorithm is explained in detail. The experiment to validate the algorithm is described in Section IV.

The results of the experiment are presented and discussed in Section V. In Section VI the conclusion is given.

II. State of the Art

Different approaches are used to measure the distance between a person and a camera. As shown below, one can distinguish between systems utilizing only one single camera and systems utilizing additional hardware like more cameras, including cameras producing three dimensional pictures, or special light sources. For our field of research it is also important to distinguish between PC and smartphone based systems, as we want to work with a smartphone or a tablet. In this section we present previous work using these approaches and their relation to our work.

Eastwood-Sutherland et al. presented a method of monitoring the screen to eye distance [1]. They utilized a stereoscopic camera and placed infrared markers on the head of a person in front of the camera. Their stereoscopic camera was a single camera with a special mirror attached in front to capture a stereoscopic image. Their goal was to measure the long term influences of computer monitors on the human eye and one property they investigated was the screen to eye distance. Similar to this approach Ponglangka et al. presented an approach where they attach two extra infrared light sources next to a computer screen and determine the distance by analyzing the light reflected by the eyes [7]. However, such systems are not suitable for our approach. We do not want the user to wear special markers on his head to keep the approach as simple and unobtrusive as possible.

Another PC based system to measure the camera to face distance was introduced by Rahman et al. Similar to our approach they measure the eye to eye distance of a person in front of the camera [9]. Afterwards they derive the camera to face distance from the eye to eye value. In contrast to our approach they do not initialize the system for each user. Instead they use a standard average value for the eye to eye distance. Doing this, the derived camera to face distance is more inaccurate than our approach. Their accuracy is subject to how much a certain person's eye to eye distance deviates from the standard value.

In the field of smartphone cameras for distance measurements Lee et al. presented a monitoring system utilizing the front camera of a smartphone [8]. The system monitors the posture of a smartphone user and advises him to prevent physical illness such as musculoskeletal disorders and eye problems. Therefore the distance derived from the front camera is used with a combination of the smartphone accelerometer and orientation sensor. Unfortunately they do not tell how their system detects the screen to camera distance and the resulting accuracy. Therefore we cannot draw any further conclusions as to whether their system would be applicable for us.

In this paper we present an algorithm useable on a current smartphone such as an LG Nexus 4 without additional lighting or additional cameras and best possible accuracy for a certain user. The algorithm is described in the next section.

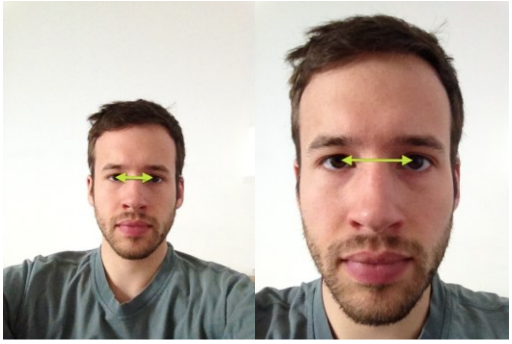

Fig. 1. Two photos of a person taken at a different distance (left picture: far, right picture: close). The eye to eye distance (yellow line) on a users picture relates to the camera to face distance.

III. Screen to Face Distance Algorithm

The screen to face distance algorithm has two phases: first an initialization phase and second a measurement phase. The initialization phase needs only to be done once for each user. The measurement phase is done continuously, as long as the face to screen distance has to be measured. Both phases utilize an eye detection algorithm. In this section we describe the algorithm phases in detail as well as the principal idea of measuring a distance with a camera. We also give our reasons for choosing our preferred eye detection algorithm.

A. Face to screen distance from eye to eye distance

The eye to eye distance on a user's picture relates to the camera to face distance. A human face will show a certain eye to eye distance on a photo. If the face is close to the camera, the eye to eye distance on the picture will appear large, if the same face is further away the eye to eye distance will appear smaller, shown in Fig. 1. Since the smartphone or a tablet front camera is usually mounted on the same plane with the screen, we consider the screen to face distance to be the same as the camera to face distance.

A reference is needed to derive a metric value for the screen to face distance: the eye to eye distance at a specific screen to face distance. We call this the reference picture and reference distance. By comparing the eye to eye distance of an actual camera picture with values from the reference picture, we can calculate the actual screen to face distance. We describe the relation between the eye to eye distance and the screen to face distance in respect to a reference by the following equation:

$$d_{sf} = p_{ref} / p_{sf} * d_{ref} \qquad (1)$$

d_{sf}[cm] is the screen to face distance and p_{sf}[pixel] is the eye to eye distance on the actual picture. d_{ref}[cm] is the screen to face distance of the reference picture and p_{ref}[pixel] is the eye to eye distance on the reference picture. As it can be seen in (1) we assume a linear relation between the screen to face distance and the eye to eye distance. The experiment outlined in Section IV and V reinforces our linearity assumption in a range from 19cm to 89cm screen to face distance.

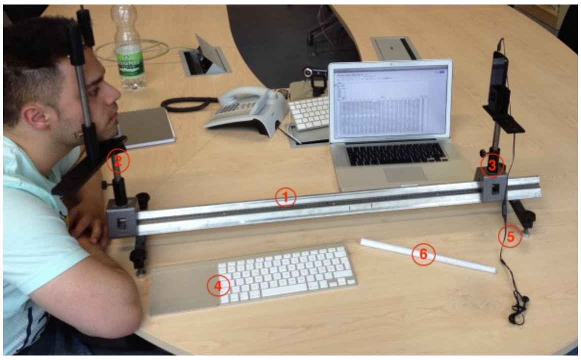

Fig. 2. The optical bench during the experiment. 1: optical bench, 2: test person with fixed head, 3: moveable Smartphone mount, 4: bluetooth keyboard to set the resolution, 5: headset button to trigger a measurement, 6: distance scale for the reference picture.

B. Choosing the eye detection algorithm

We had to choose an eye detection algorithm. In our investigation we focused on a comparison of OpenCV with the Android built-in face detector in two important aspects: firstly the detection confidence and secondly the detection speed. The confidence is important to enable the algorithm to work at all. If no face is detected then the eyes cannot be detected and thus distance cannot be calculated. The detection speed is important for a continuous screen to face distance measurement. Our comparison is based on an investigation presented in [1] and further described in the next two paragraphs.

Both algorithms perform with similar accuracy in detecting a face in a picture. The investigation presented in [1] was based on the task of finding 40 faces in 15 pictures. Both algorithms found 33 faces. OpenCV performed better on noisy pictures but also detected five non-existent faces.

The Android built-in face detection performed much faster than the OpenCV algorithm. The face detection, which is needed to detect the eyes afterwards, took seven seconds on average for OpenCV [1]. The Android built-in face detection needed 2.5 seconds on average to detect a face [1].

By considering both aspects, the confidence and speed of detection, we decided to use the Android built-in eye detection. Unfortunately the manufacturer, Google, does reveal which algorithm they use for the Android built-in eye detection. Nevertheless, the Android eye detection yields three eye related features for each recognized face: the mid-point of the eyes, posture of the face and distance of the eyes. The eye distance feature is directly used by our algorithm.

C. Taking the reference picture

The reference picture is taken while a scale is placed between the users face and the smartphone. To make the process as simple as possible we used a simple self-made scale. Therefore we rolled a DIN A4 paper at the long side and taped it several times. This resulted in a tubular distance scale of 29,7cm. Afterwards we placed the scale between the users face and the smartphone screen and took the reference picture.

To filter irritations our algorithm takes a series of ten pictures in about 300 millisecond and averages the eye to eye distances afterwards to calculate the reference picture value. The Android built-in eye detection is not as accurate as desirable which leads to a faulty eye to eye distance of the reference picture. All further screen to face measurements would be faulty when using faulty reference values.

D. Measuring with the screen to face distance algorithm

After taking the reference picture the algorithm measures the screen to face distance continuously. Therefore the algorithm performs the five following steps: 1. Take a picture with the front camera. 2. Detect the face and eyes. 3. Calculate the eye to eye distance. 4. Use (1) and the reference picture values to calculate the screen to face distance. 5. Average the screen to face distance over the last five measurements. The last step is added to filter irritations of the Android built-in eye detection.

IV. EXPERIMENT

In this section we describe our accuracy and speed validation of the presented screen to face algorithm. The methodical approach we used to validate our proposed algorithm is an experiment. The experiment consisted of a smartphone, a test application using the above described algorithm, an optical bench and two persons for a series of screen to face distance measurements. Each element is described in one of the following paragraphs.

An optical bench is a device from the field of optical engineering. Several objects can be placed, either stationary or at different positions among the optical bench. For the experiment the smartphone was placed at different positions along the bench while a person's head was held stationary at the other end. The smartphone was moved towards or away from the person's head and the actual distance was read from the scale on the bench. The arrangement can be seen in Fig. 2.

The smartphone used for the experiment was an LG Nexus 4 (E960). The Nexus 4 operating system is Android 4.2 Jelly Bean. It uses a quad-core ARM Cortex A9 CPU where each core is clocked with 1500MHz. The smartphones RAM capacity is 2048MB and it has a front camera with a maximum resolution of 1280x1024 pixels. During the development and for testing, but not as part of the experiment, we also used a Samsung Galaxy II and an Asus MeMo Pad HD 7 tablet. The algorithm worked on all tested smartphones and tablets.

We implemented the described screen to face distance algorithm in a test application. The test application is able to measure the face to screen distance as well as the CPU time for a measurement. Also the front camera resolution can be set manually. The application guides through the measurement process by displaying the next distance for the smartphone on the optical bench. To trigger a measurement we connected a headset and utilized the headset button so we did not have to touch the smartphone once the measurement had started. The test application also saved the results in a table after a series of measurements was completed and transferred the data to a computer for analysis.

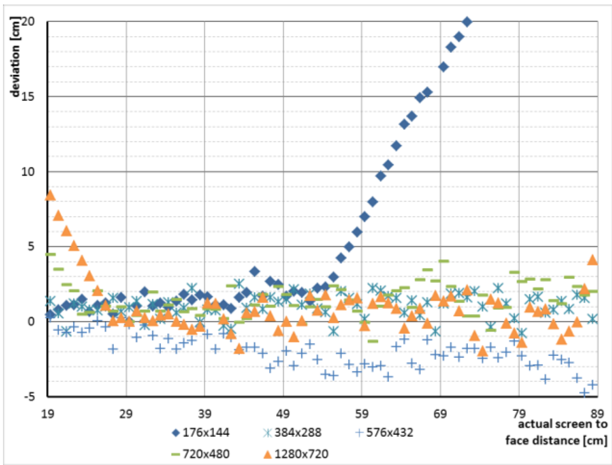

Fig. 3. Measured screen to face distance deviation over actual distance using different resolutions

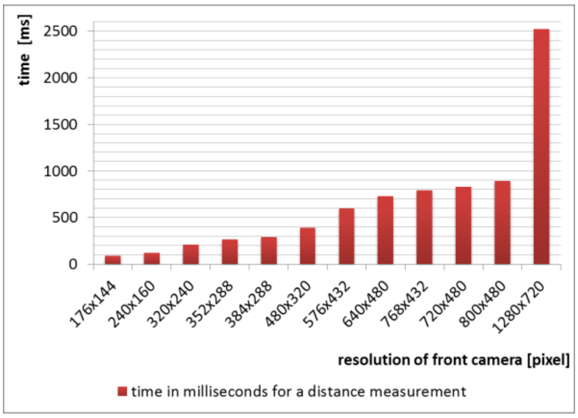

Fig. 4. Time needed for a single screen to face distaance measurement using different resolutions

We decided to measure the accuracy over the range of 19cm to 89cm screen to face distance. The lower boundary of 19cm is the smallest screen to face distance where the algorithm is still working. However, a smartphone or a tablet is usually not used this close to the face. The upper boundary of 89cm is further away than most people can hold a smartphone with their stretched arm. We started with the lower boundary and increased the distance in 1cm steps after each measurement.

A series of measurements consisted of four initial steps, executed once:

- 1st Set the front camera resolution,
- 2nd Make sure the person's head is stationary on the head rest on the optical bench,
- 3rd Place the smartphone in the optical bench and
- 4th Initialize the algorithm by taking the reference picture.

The next two steps had to be repeated until the whole range of distances had been measured: first: setup the distance displayed by the test application and second: press headset button to measure the screen to face distance. After the test application had measured the distance, the new distance for the optical bench was displayed (old distance plus 1 cm) and the process was repeated.

We tested the accuracy and speed of the proposed algorithm in twelve different smartphone front camera resolutions: 176x144, 240x160, 320x240, 352x288, 384x288, 480x320, 576x432, 640x480, 720x480, 768x432, 800x480 and 1280x720 pixels. This enables us to conclude which resolution yields the best accuracy and how much time the screen to face measurement takes for a certain resolution.

We used two different persons for the experiment. Person one can be seen in Fig. 1, person two in Fig. 2. One series of measurements consisted of 71 measurements (19cm, 20cm, 21cm … 89cm distance) for a given resolution. Each measurement series was repeated three times. This was done for all twelve resolutions and with each person.

V. Results

In this section the experiment results are presented. First we present the accuracy over the distance separated by the camera resolution. In the second subsection we present the utilized CPU time for a screen to face distance measurement, depending on the chosen camera resolution. In the third subsection we discuss the results.

A. *Accuracy over the screen to face distance and camera resolution*

Our metric for accuracy is the deviation of the measured distance from the actual distance. To calculate the deviation we subtracted the measured screen to face distance from the actual screen to face distance. One could also calculate a relation between the measured value and the actual value. However, for a technical usage such as an automated eye exam, it is more important to know the maximum distance deviation. This also enables us to compare the accuracy over different distances and detect trends.

Fig. 3 shows the screen to face distance algorithm accuracy by displaying the deviation over the actual distance. We show only five of the twelve investigated resolutions to keep the graph simple. The not shown resolutions had the same trend: low resolutions were performing better in the close distance, high resolution were performing better more far away. The impacts of this trend are discussed in the next subsection. The resolution of 384x288 pixels performed best in our measurement over the whole range of 19cm till 89cm with a maximum deviation of 2.1cm. Also three distance measurements per second were possible at this resolution with our test device.

The accuracy depends on the actual screen to face distance and the chosen camera resolution. One pixel displacement in the eye detection has more influence if the face is far away. In (1) p_{fs} (the measured eye to eye pixel distance) is in the denominator. Faces further away will result in small p_{fs} values, closer faces will result in bigger p_{fs} values. One pixel displacement added to an already small number has a much larger influence than one pixel displacement added to a large number. This is why the screen to face distance itself influences the accuracy. Also if the resolution is too low to detect the eyes properly, the screen to face distance algorithm will lose accuracy.

B. Utilized CPU time

The CPU time for a screen to face distance measurement was taken with a timer in our test application. We started the timer before the picture was taken and stopped it after the distance was calculated. Thus the CPU time includes the picture taking, the face and eye detection and the distance calculation time. Further investigations showed that nearly 99% of the CPU time is needed for the face and eye detection.

The CPU time relays mainly on the chosen camera resolution. Each pixel of a picture taken with the camera has to be processed from the face detection algorithm. Therefore the CPU time correlates very well with the number of pixels in a certain resolution, as can be seen in Fig. 4.

C. Discussion

The experiment proves the feasibility of a screen to face distance measurement with a current smartphone such as an LG Nexus 4 at an interval of three measurements per second. However, the following aspects still need to be taken into consideration when choosing the best resolution for a specific application.

For small distances the algorithm performs better at a low camera resolution. For a high resolution and a low distance the eyes are not detected anymore by the android built-in face detection. As the algorithm design has not been disclosed, we can only guess the reason why. In our opinion there may be a maximum threshold to enable face detection for the algorithm. A face close to the camera captured with high resolution may be above the detection threshold, i.e. the face remains undetected.

For longer screen to face distances the higher camera resolutions perform better. The resolution of 176x144pixels for example, does not work for distances over 51cm. At this distance and resolution a human face on a picture is no longer clear, thus the face detection algorithm cannot detect the eyes anymore.

To save battery life and CPU time, the camera resolution should be chosen as low as possible regarding the targeted application. A higher camera resolution captures more pixels than a low resolution. Each pixel has to be processed by the smartphone CPU and this blocks the CPU from other tasks, uses more phone battery energy and increases the distance measurement interval. On the other hand, the camera resolution should be high enough to cover the whole targeted range of screen to face distances.

VI. CONCLUSION

In this paper we presented an algorithm for a new context: the user's smartphone screen to face distance. We also presented an experiment proving: a screen to face distance is possible to calculate with a current smartphone such as an LG Nexus 4. The experiment also revealed that the best camera resolution to measure distances in the range of 19cm till 89cm is 384x288 pixels, where our presented screen to face distance algorithm had a maximum distance deviation of 2.1cm. Also three distance measurements per second were possible with this resolution and our test device.

This algorithm can be used in different applications, like more reliable eye testing applications or more user-friendly smartphone interfaces. In eye testing applications the screen to face distance context can be used to adapt the test or at least to blank out the screen when the distance is changed above a certain threshold. This could help to detect eye diseases in their early stages, thereby helping protect elderly people from irreparable eye damage. In the field of smartphone user-interfaces, automatically adapting text sizes can be realized by utilizing the screen to face distance context. This can enable more comfortable text reading on a smartphone or tablet.

REFERENCES

[1] Geeta A. Lalwani, Philip J. Rosenfeld, Anne E. Fung, Sander R. Dubovy, Stephen Michels, William Feuer, Janet L. Davis, Harry W. Flynn Jr, Maria Esquiabro, "A Variable-dosing Regimen with Intravitreal Ranibizumab for Neovascular Age-related Macular Degeneration: Year 2 of the PrONTO Study," American Journal of Ophthalmology, Volume 148, Issue 1, July 2009, pp. 43,58

[2] Meyer C, H, Lapolice D, J, "Computer-Based Visual Evaluation as a Screening Tool after Intravitreal Injections of Vascular Endothelial Growth Factor Inhibitors." Ophthalmologica 2008

[3] Stenneth, L.; Thompson, K.; Stone, W.; Alowibdi, J., "Automated transportation transfer detection using GPS enabled smartphones," Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on , Sept. 2012

[4] S. L. Lau and K. David, "Movement recognition using the accelerometer in Smartphones," in Future Network & MobileSummit 2010, Florence, Italy, June 16-18 2010

[5] Yohan Chon; Hojung Cha, "LifeMap: A Smartphone-Based Context Provider for Location-Based Services," Pervasive Computing, IEEE , vol.10, no.2, pp.58,67, April-June 2011

[6] Eastwood-Sutherland, C.; Gale, T.J., "Eye-screen distance monitoring for computer use," Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE, pp.2164,2167

[7] Ponglangka, W.; Theera-Umpon, N.; Auephanwiriyakul, S., "Eye-gaze distance estimation based on gray-level intensity of image patch," Intelligent Signal Processing and Communications Systems (ISPACS), 2011 International Symposium on, pp.1,5

[8] Lee, H.; Lee, S.; Choi, Y. S.; Youngwan Seo; Eunsoo Shim, "A new posture monitoring system for preventing physical illness of smartphone users," Consumer Communications and Networking Conference (CCNC), 2013 IEEE , pp.821,825

[9] Rahman, K.A.; Hossain, M.S.; Bhuiyan, M.A.; Tao Zhang; Hasanuzzaman, M.; Ueno, H., "Person to Camera Distance Measurement Based on Eye-Distance," Multimedia and Ubiquitous Engineering, 2009. MUE '09. Third International Conference on, pp.137,141

[10] Online Article, Small Screen Design, „About Face Detection on Android," Internet: http://www.smallscreendesign.com/2011/01/25/about-face-detection-on-android-part-1/, 2011, last checked 14.10.2013