Smartphone Input Using Its Integrated Projector and Built-In Camera

Sergiu Dotenco, Timo Götzelmann, and Florian Gallwitz

Nuremberg Institute of Technology, Germany
{sergiu.dotenco,timo.goetzelmann,florian.gallwitz}@ohm-university.eu

Abstract. Touch input on modern smartphones can be tedious, especially if the touchscreen is small. Smartphones with integrated projectors can be used to overcome this limitation by projecting the screen contents onto a surface, allowing the user to interact with the projection by means of simple hand gestures. In this work, we propose a novel approach for projector smartphones that allows the user to remotely interact with the smartphone screen via its projection. We detect user's interaction using the built-in camera, and forward detected hand gestures as touch input events to the operating system. In order to avoid costly computations, we additionally use built-in motion sensors. We verify the proposed method using an implementation for the consumer smartphone Samsung Galaxy Beam equipped with a deflection mirror.

Keywords: Mobile computing, user interfaces, small screen, fat finger problem, touch input, fingertip detection, projector smartphone, DLP projector.

1 Introduction

In the last few years, the size of touchscreens increased rapidly. Today, mobile devices are available in various sizes and form factors. There is a smooth transition between smartphones and tablet computers with a touch sensitive user interface. Modern devices often feature screen sizes up to 5 inches. These so called *phablets* have even larger screens. However, as the smartphones are typically carried in bags or pockets, their size is subject to physical constraints. Often, the size of the smartphone and its handiness are inversely proportional.

As the size of smartphones generally faces physical limitations, the size of the touch input area is affected by this limitation as well. A user study in ergonomics by Colle and Hiszem [3] has shown that a correlation between the size of the touch input area and the user input performance exists. A typical example is the input using a virtual keyboard which can be very tedious, since the space reserved for individual keys is limited.

Despite the use of high resolution displays combined with precise capacitive touchscreens and sophisticated algorithms that can filter out unwanted input events, the problem of limited screen space remains unsolved. The input using a virtual keyboard is even more problematic for small devices.

M. Kurosu (Ed.): Human-Computer Interaction, Part III, HCII 2014, LNCS 8512, pp. 124–133, 2014.

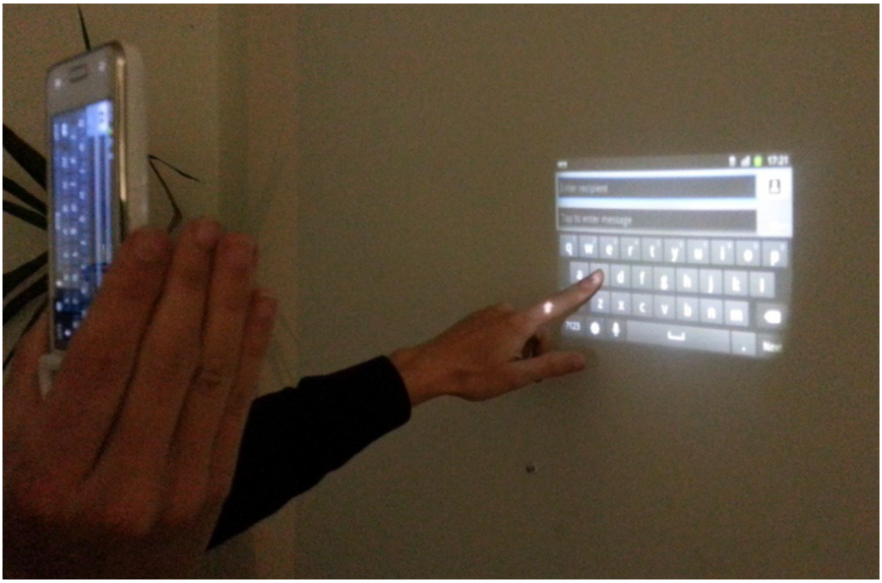

Fig. 1. In our approach, we use a smartphone with an integrated projector and a built-in RGB camera. The user holds the smartphone in one hand and projects the screen content onto a planar surface, while the camera is used to detect the projected screen and the user's interaction with it. The detected hand motion and gestures are forwarded to the smartphone's operating system as touch input events.

To increase the input area of any smartphone, a portable, small-sized projector attached to the smartphone could be used to project the screen contents onto a planar surface. Smartphones with an integrated projector, however, are clearly more flexible. Although, there are only few smartphones on the market that have an integrated projector, such as the Samsung Galaxy Beam, Apple Inc. as one of the major smartphone vendors has also shown interest in integrating a projector into their iPhone product line [5].

As a possible solution to the addressed problems, we propose a novel approach that allows users to interact with a projection of a smartphone screen, and explore the feasibility of using standard hardware in consumer smartphones to overcome the size limitations of typical smartphone touchscreens. We assume that integrated projectors will be a standard feature of smartphones in the future, as cameras are today. This would also allow to develop devices with very small screens or even devices without actual screens. Holding the device in one hand, the user will then project smartphone's user interface onto a surface. The other hand can be used to interact with the projection. Simultaneously, the camera captures the user interaction and forwards detected gestures to the system. This process is illustrated in Fig. 1.

The remainder of this paper is organized as follows. Fig. 1 presents the related work. In Fig. 1 we introduce our approach. Fig. 3 provides implementation details. In Fig. 3 we evaluate the approach. Finally, the conclusions are given in Fig. 4.

2 Related Work

Today's consumer market offers already a few smartphones with an integrated projector, such as Samsung Galaxy Beam, Sharp SH-06C, and Intex V.Show IN 8810. But more manufacturers are likely to follow suit, most notably Apple Inc. showing interest in such devices by filing a related patent application [5].

The challenge of interacting with a projection has been tackled previously in several ways. Mistry et al. [12] introduced a wearable gesture interface. Using a small wearable projector and a mobile camera, they proposed a method for interacting with an abstract ten digit keyboard. Their system recognizes finger input using a marker based technique. Harrison et al. [9] showed impressive results using a wearable interface combined with stereo cameras and depth sensors. Winkler et al. [16] conducted a study in which they point out the potential of interaction using projector smartphones. However, they use a mid-air pointing technique, and rely on an external tracking system and infrared markers attached to user's finger. Specifically for the virtual keyboard input, the concept study [7] investigated the freehand interaction with a projected keyboard.

3 Our Approach

Our approach is intended to work on smartphones with a built-in RGB camera and an integrated projector. Using the smartphone, the user projects the screen contents onto a planar surface while holding the smartphone in one hand, and simultaneously interacting with the projected screen using the other one, as seen in Fig. 1). In the first step of our method, we detect the screen projection using the built-in RGB camera. Then, we determine the perspective transformation matrix using the quadrilateral corners of the projected screen. Later, this allows to map the coordinates from the image domain to those of the smartphone display. Detecting the screen projection is performed only if the attitude of the smartphone changes, e.g., due to hand jitter, as opposed to performing the detection in every video frame. This allows to avoid costly and unnecessary computations. Finally, we use the skin color segmentation method by Rahman et al. [14] to detect the hand of the user and the tap gesture, which we defined as a closed hand.

3.1 Detecting Screen Projection

Since the built-in RGB camera will generally capture an area much bigger than that of the projected screen, the projection has to be located in the captured video frames. After that, projection screen coordinates can be mapped to those

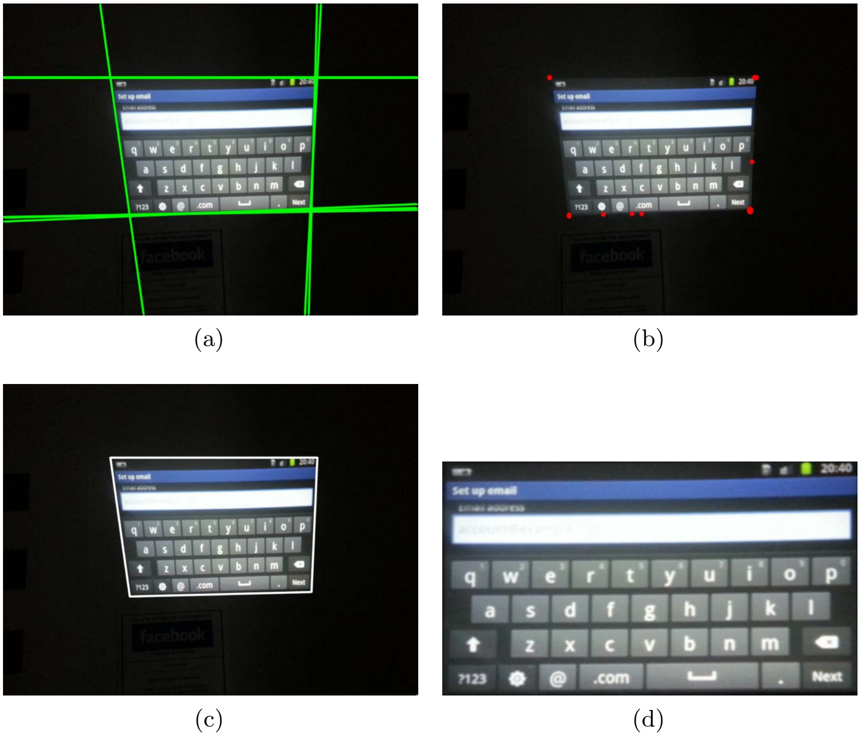

Fig. 2. Steps involved in detecting the projection screen and its perspective rectification: (a) line detection using probabilistic Hough transform, (b) intersections (shown as circles) between the lines, (c) quadrilateral that resulted from selecting four intersection points nearest to the image edges, (d) rectified perspective

of the touchscreen. As soon as the projected screen has been detected, the perspective of the projection screen can be rectified and the fingertip position (in display coordinates) can be determined.

In order to detect the projected screen, we assume that the area surrounding the projection is much darker than the actual projection. Here, we threshold captured video frames according to their luminance using a suitable threshold value. Ideally, this operation will result in a quadrilateral shaped blob that corresponds to the projected screen. To remove small defects in the blob, we apply morphological erosion and dilations. We then proceed with the extraction of the edge map using the Canny edge detector [2], and finally apply the probabilistic linear Hough transform to detect lines and to obtain the quadrilateral from line intersections points. The four line intersection points that build the quadrilateral are chosen such that they are close to the corresponding edges of the video frame. This ensures that the quadrilateral with the largest area will be obtained.

3.2 Perspective Rectification

In general, the user will not be able to hold the smartphone exactly orthogonal to the projection surface. We therefore determine the perspective transformation matrix M from the four corners of the quadrilateral. This allows us to rectify the perspective distortion, and to locate the fingertip position in touchscreen coordinates. Fig. 2 illustrates the steps involved in the detection of the projected screen.

3.3 Avoiding Costly Computations

Detection of the projected screen as well as touch input is sensitive to marginal changes of the smartphone attitude. Natural jitter of user's hand holding the smartphone, however, is generally unavoidable. Still, to omit unnecessary detection of the projected screen in every video frame, we perform the detection only if the attitude of the smartphone changes significantly. Alternatively, gravity and geomagnetic sensors could be used to detect motion.

In cases where the attitude changes significantly (e.g., by $1°$), the projection screen is detected again. Otherwise, we reuse the previously determined perspective transformation matrix M to rectify the video frames.

3.4 Detecting User Input

To detect the hand, we use skin color segmentation. For gesture recognition, we additionally assume that just one finger is used for interaction with the projected screen (e.g., the index finger). For a tap, the user has to close the hand. The tap will be performed at the fingertip position detected previously.

Fingertip Detection. As the computational resources on a typical consumer smartphone are limited, we use a computationally inexpensive method for fingertip detection. We start with the segmented skin areas, followed by the extraction of their contours. After that, the contour with the largest area is selected. We then reduce the number of points in the selected contour using the Douglas-Peucker algorithm [4]. As long as the resulting contour consists of only few points, we extract the corresponding convex hull, and determine the combination of convex points with the largest Euclidean distance. We used a threshold of 20 points which suffice to approximate the contour of a hand or a finger. Contours consisting of more points are unlikely to represent the contour of a hand. The candidate fingertip position is the point with the largest distance to the video frame border (see Figs. 2 and 2). Finally, we combine the candidate point with its nearest neighbors, and compute their arithmetic mean as the fingertip position.

This approach does not require the whole hand to be visible, since no shape information is used to match the hand or the fingertip.

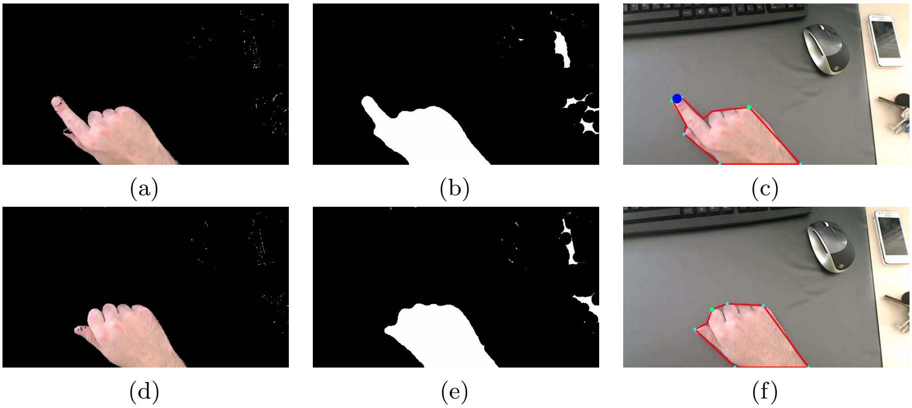

Fig. 3. Illustration of single steps performed in order to detect the hand gesture in a toy video sequence. (a) (d) Skin color segmentation, (b) (e) thresholding followed by morphological dilation and erosion, (c) (f) the selected contour. The blue circle in (c) indicates the position of the detected fingertip.

Tap Gesture Detection. To distinguish between an open hand with a pointing finger and closed one, we extract convexity defects of the convex hull (see Figs. 2 and 2). If the maximum depth of all convexity defects is above a certain threshold, we detect the hand as open. Otherwise, the hand is considered to be closed, and a tap at the previous fingertip position is executed.

4 Implementation

We implemented the proposed approach for the Android operating system, which according to a market survey by Gartner [6] at present dominates the global smartphone market. The approach, however, could easily be implemented for other systems such as Apple's iOS, Symbian, Blackberry OS, Windows Phone 7, etc., as long as the smartphone is equipped with a camera and a projector.

In our implementation, we used the OpenCV library [1]. For performance reasons, we implemented the image processing functionality in native code. An Android service was used to capture the video frames, to execute projection screen and fingertip detection, and to react to attitude changes. We used an image resolution of 640×480 pixels for captured video frames. Detected tap gestures were forwarded to a test application by our Android service. Once a finger tap has been detected, tactile feedback is given by the vibration actuator.

5 Qualitative Evaluation

We performed a qualitative evaluation of our prototype implementation. As prototype system we used Samsung Galaxy Beam (Model GT-I 8530), a consumer smartphone with an integrated projector.

(a) Back view

(b) Front view

Fig. 4. Mirror mechanism for Samsung Galaxy Beam which reflects the projection in the direction of the back-facing camera

One major issue that concerns most consumer projector smartphones is that the projector is often mounted perpendicularly to smartphone's back-facing camera. To overcome this problem, we built a mirror mechanism using rapid prototyping. This mirror can be clipped at the top of the smartphone, which allows to reflect the projection into the direction of the camera. The mirror mechanism is shown in Fig. 4. A stereolithography file (STL) that can be used to 3D print the mirror mechanism can be obtained from our website [8].

As our approach is sensitive to illumination conditions, the test setup required setting up an environment with properly balanced illumination. On the one hand, the environment has to be rather dark in order to achieve a high contrast between the projection surface and the area surrounding the projection. On the other hand, the environment should not be too dark, because in this case the hand will not be illuminated properly anymore, resulting in incorrect hand skin segmentation.

The illumination conditions not only concern the environment itself, but also the smartphone screen contents. Specifically, dark colored user interface elements, whose projection covers the hand or even the pointing finger, can tamper the skin color, producing incorrect detections. This problem also applies in case of extremely bright portions of the user interface.

Despite implementing the algorithms in native code, our projection screen detection achieved only a rate of 0.9 frames per second (FPS) ($\approx$ 1110 ms per frame). In combination with the fingertip detection, the frame rate decreased to around 0.5 frames per second ($\approx$ 1000 ms). This is generally too slow for fast and smooth input.

6 Conclusion and Future Work

We presented a novel approach for joint use of a built-in smartphone camera and an integrated projector. We discussed the requirements arising from the freehand use of a smartphone camera and a projector. Motion sensors were integrated into our framework to avoid unnecessary detections. Actuators were used to give feedback to the user once a finger tap has been detected.

Overall, our implementation showed the feasibility of the proposed method. However, the implementation needs to be optimized in order to work reliably under varying illumination conditions and different projection surfaces. Determining the optimal parameters for our implementation is still ongoing work.

The algorithms for projected screen detection, perspective rectification, and the recognition of the user input are sensitive to illumination conditions. Additionally, mobile projectors such as the built-in projector of the smartphone Samsung Galaxy Beam achieve only 15 lumens, requiring the smartphone to be used in a rather dark environment. In our implementation, parameters had thus to be adjusted manually to suit illumination conditions. Since smartphones are often equipped with light sensors, they could be used to automatically adjust the parameters.

The finger detection in our experimental implementation relies on a simple methodology. The detection rate and accuracy could be improved by placing markers on fingertips. However, markerless systems are clearly be more flexible. Alternatively, the fingertip could be located using Hough circle detection. Each time the recognized circle of the fingertip vanishes, it could be recognized as a tap. Several sophisticated approaches were proposed that can be integrated to solve this challenging problem (e.g., [10, 11, 13]).

As opposed to detecting and rectifying the perspective of the projected screen after its detection, the projection itself could be perspectively corrected in advance using the technique suggested by Raskar and Beardsley [15]. This would allow to pre-warp the projection, ultimately eliminating the need in rectifying the perspective. Alternatively, motion and geomagnetic sensors available in many smartphones could be used to determine the perspective transformation. This could be done by initially placing the smartphone on the projection surface, allowing to measure its inclination and distance between the surface and future smartphone position.

To improve the detection of the fingertip without relying on skin color models, the display contents could be subtracted from the projection surface. This approach, however, would have to take different lightning conditions into consideration. Also, the accuracy of the detected fingertip position could be increased by removing radial distortions of the smartphone camera.

Compared to desktop systems, there are special necessities for testing the usability for mobile devices [17]. For the concept of the joint use of smartphone's camera and projector, it will neither be sufficient to apply desktop usability measures nor to solely apply usability measures for mobile systems. For this novel type of interaction, distinct usability measures have to be found. Our current aim is to carry out user studies to optimize the proposed input method.

References

1. Bradski, G.: Programmer's toolchest – the OpenCV library. Dr Dobb's Journal-Software Tools for the Professional Programmer 25(11), 120–126 (2000)
2. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 8(6), 679–698 (1986), `http://dx.doi.org/10.1109/TPAMI.1986.4767851`
3. Colle, H.A., Hiszem, K.J.: Standing at a kiosk: Effects of key size and spacing on touch screen numeric keypad performance and user preference. Ergonomics 47(13), 1406–1423 (2004)
4. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: The International Journal for Geographic Information and Geovisualization 10(2), 112–122 (1973)
5. Fai, A.Y.S.: Projected display shared workspaces. Patent US 20110197147 A1 (2011)
6. Gartner: Gartner says worldwide sales of mobile phones declined 3 percent in third quarter of 2012; smartphone sales increased 47 percent (2012), `http://www.gartner.com`
7. Götzelmann, T.: Concept of the joint use of smartphone camera and projector for keyboard inputs. In: Proceedings of the 3rd International Symposium on Computing in Science & Engineering, ISCSE 2013, pp. 52–57. Gediz University Press, Gediz (2013a)
8. Götzelmann, T.: Deflection mirror for projector smartphones (stl file for 3d printers) (2013), `http://www.in.th-nuernberg.de/professors/goetzelmann/BeamerPhoneDeflectionMirror.stl`
9. Harrison, C., Benko, H., Wilson, A.D.: Omnitouch: Wearable multitouch interaction everywhere. In: Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, UIST 2011, pp. 441–450. ACM, New York (2011), `http://doi.acm.org/10.1145/2047196.2047255`
10. Hasan, M.M., Mishra, P.K.: Real time fingers and palm locating using dynamic circle templates. International Journal of Computer Applications 41(6), 33–43 (2012)
11. Jiang, X.-H., Li, J.-W., Wang, K.-Q., Pang, Y.-W.: A robust method of fingertip detection in complex background. In: International Conference on Machine Learning and Cybernetics (ICMLC), vol. 4, pp. 1468–1473 (2012)
12. Mistry, P., Maes, P., Chang, L.: Wuw - wear ur world: a wearable gestural interface. In: CHI 2009 Extended Abstracts on Human Factors in Computing Systems, CHI EA 2009, pp. 4111–4116. ACM, New York (2009)
13. Oka, K., Sato, Y., Koike, H.: Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems. In: Fifth IEEE International Conference on Automatic Face and Gesture Recognition, pp. 429–434. IEEE (2002)

14. Rahman, N.A.B.A., Wei, K.C., See, J.: RGB-H-CbCr Skin Colour Model for Human Face Detection. In: Proceedings of the MMU International Symposium on Information & Communication Technologies (2006)
15. Raskar, R., Beardsley, P.: A self-correcting projector. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. II-504–II-508 (2001)
16. Winkler, C., Pfeuffer, K., Rukzio, E.: Investigating mid-air pointing interaction for projector phones. In: Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces, ITS 2012, pp. 85–94. ACM Press, New York (2012)
17. Zhang, D., Adipat, B.: Challenges, methodologies, and issues in the usability testing of mobile applications. International Journal of Human-Computer Interaction 18(3), 293–308 (2005)