MVA2009 IAPR Conference on Machine Vision Applications, May 20-22, 2009, Yokohama, JAPAN

Building a multi-touch display based on computer vision techniques

D. Michel, A.A. Argyros, D. Grammenos, X. Zabulis, T. Sarmis
Institute of Computer Science, FORTH
Heraklion, Crete, Greece
{michel, argyros, gramenos, zabulis, sarmis}@ics.forth.gr

Abstract

We present the development of a multi-touch display based on computer vision techniques. The developed system is built upon low cost, off-the-shelf hardware components and a careful selection of computer vision techniques. The resulting system is capable of detecting and tracking several objects that may move freely on the surface of a wide projection screen. It also provides additional information regarding the detected and tracked objects, such as their orientation, their full contour, etc. All of the above are achieved robustly, in real time and regardless of the visual appearance of what may be independently projected on the projection screen. We also present indicative results from the exploitation of the developed system in three application scenarios and discuss directions for further research.

1 Introduction

Touch-enabled displays have already been around for more than a couple of decades. Until a few years ago, touch displays supported only a single point of contact, thus allowing only for the restricted type of human-computer interaction that a mouse click can offer, but with the advantage of direct and more intuitive target selection.

More recently, the concept of multi-touch displays has become quite popular and a number of related research and commercial efforts emerged. Currently, probably the most widely known and publicized multi-touch displays are the Apple iPhone [1], which is already a highly successful commercial product, and the Microsoft Surface [10] that has just been released to the market.

Multi-touch displays enable their users to concurrently use more than the tip of one finger at a time, allowing for whole-hand, bimanual, as well as true multi-user interaction. Furthermore, depending on the sensing approach employed, some multi-touch displays may also be able to recognize shape and object outlines, understand gestures, or read coded tags placed underneath objects.

In this context, this paper presents our efforts towards the creation of a large-scale, though low cost, multi-touch display using computer vision techniques.

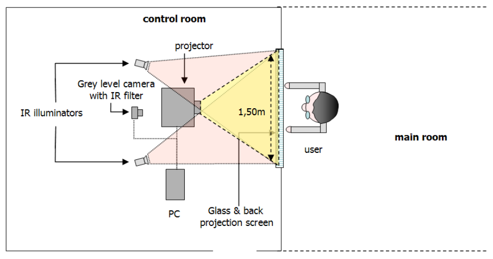

control room
projector
Grey level camera with IR filter
IR illuminators
1,50m
user
main room
PC
Glass & back projection screen

Figure. 1. Layout of the developed multi-touch display.

2 Related work

Depending on the sensing technologies used, multi-touch displays can broadly be classified in two categories, the physical signal measuring displays and the computer-vision based displays.

The physical signal measuring displays use a surface that can detect physical signals (e.g., pressure, electromagnetic induction, sound, etc.) created when the users' hands or other objects come in contact with it. Some representative examples include Apple's iPhone [1], Smart-Skin [11] and Mitsubishi's DiamondTouch [3] which are all based on capacitive sensing. iPhone has a 3.5-inch LCD display, while Smart-Skin and DiamondTouch are large solid surfaces that use front projection for displaying information. DiamondTouch has also the ability to differentiate among up to 4 users by coupling signals emitted by antennas embedded in the table, through receivers on which the users are standing. The main advantage of such systems is that they are not affected by lighting conditions and are very easy to move around, install and configure. On the other hand, their key disadvantages include their fixed size, the great difficulty and cost of scaling them up, the fact that they do not function when users wear gloves or interact with non-capacitive objects and their inability to detect visual information, such as tags, text or colors.

Computer-vision based displays use one or more cameras to detect users and interpret their interaction with objects. Due to their nature, they can capture more than just touch data. Typically, in order to easily overcome the dynamic visual clutter created by the displayed image, infrared cameras are used in conjunction with infrared light emitting sources, which are either projecting light towards the screen (e.g., [9], [10], [13], [8]), or inside it (e.g., [4]). Various system configurations have been proposed based on whether the cameras and the user are in the same side of the screen or on opposite sides. An interesting setup is proposed by Leibe et al [8] who used cameras in both sides of the display in order to combine multi-touch with 3D gesture recognition and object reconstruction.

Putting the camera behind the display, opposite to the side of the user requires a semi-transparent screen coupled with front, or back projection. The advantage of using back projection is that the display is not occluded by the user's hand or body. Furthermore, object identification can be supported by adding coded tags underneath them [7]. The main disadvantage is that of space, since, in order to create a large image, the projector should be placed at a considerable distance from the projection surface. As an example, in the case of tabletop displays this might not be possible at all. Front projection overcomes space limitations and can practically be applied on any type of surface (including walls, floors or furniture) of any size. On the other hand, the major drawback of this approach is the fact that the user occludes parts of the projected information.

In the following, we present our approach to building a vision-based multi-touch display that addresses these issues. The rest of the paper is organized as follows. Section 3 gives an overview of the system's hardware setup. In section 4, the computer vision techniques employed for offline system calibration and online system performance are presented. Section 5 presents sample applications built on top of the developed system. Finally, section 6 summarizes the paper and discusses subjects of related ongoing and future research.

3 Hardware layout

The currently implemented system setup (see Figure. 1) consists of (a) a grey level camera equipped with an infrared filter (low pass filter at 850 nm), (b) a short-throw video projector (resolution 1024x768), (c) two infrared (IR) illuminators (LED spot lights), (d) a rear-projection acrylic rigid screen (size 1,50x1,10 m^2) backed up by a thick layer of glass and, (e) a typical Pentium PC equipped with a firewire video acquisition card. All the electronic equipment is hidden inside a control room while the glass and screen are housed in a "window" between the control room and an adjacent room. Thus, users experience the screen as a large wall-mounted display.

4 Computer vision modules

The basic principle of operation of the developed system is that the IR illuminators emit their light towards the screen which passes through it and is reflected by the objects. Images of the projection screen are then acquired by the camera. The IR filter that is mounted on the camera blocks the visible spectrum and is only sensitive to the infrared light reflected by the objects behind the screen, if any. Thus, the acquired image is not affected by the video displayed on the screen through the projector (observe, for example, the difference between images in Fig. 2(a) and Fig. 2(c)). Geometric correction needs to be applied to acquired images. One type of geometric correction is responsible for canceling out distortions introduced because of the camera lens (barrel distortion). Another type of geometric correction rectifies perspective effects introduced because of the relative placement of the camera with respect to the projection screen. Moreover, the response of the camera to the reflected infrared light needs to be tuned through a photometric calibration process that assigns different threshold values to different screen portions. The binary image resulting after thresholding is fed to an algorithm that identifies blobs and tracks them over time. The final output of this computer vision system includes morphological properties of the detected and tracked blobs, together with their trajectories in time. The rest of this section gives more details on all of the previously identified processing steps.

4.1 Geometric image correction

To reduce the cost and the complexity of the developed system, a single camera is used to acquire an image of the whole projection screen. To also reduce the distance between the camera and the screen, a wide field of view camera is used. Thus, lens distortions are inevitably introduced and need to be corrected. As an example, Figs. 2(a) and 2(c) show that the straight lines of the projection screen appear distorted (curved). The correction of this distortion is achieved through a standard camera calibration procedure [14], [6]. A checkerboard pattern of a size analogous to the size of the screen is placed in front of the screen. Identified corners are then fed into the algorithm implemented in [6] which recovers the image transformation required to cancel out the introduced lens distortion. During this operation, the IR filter is not mounted on the camera.

Due to the placement of the camera in front of the projection screen and the perspective distortions introduced, the rectangular effective area of the screen view is mapped onto a quadrilateral. Since the ultimate goal is to be able to map coordinates of detected blobs onto the computer's desktop coordinate system, another geometric correction needs to be applied to the input images. To achieve this, we exploit the fact that the screen is a planar surface. It is known [5] that two arbitrary views of a planar surface are linked together through a homography transformation, an invertible 3x3 matrix H. More specifically, assume a 3D point P of a planar surface that projects onto points p and p' in the views of two cameras. If H is the homography linking these two views of the plane, it holds that $p \simeq Hp'$. The homography H can be computed by employing the coordinates of at least four point correspondences. In our case, these four correspondences are provided manually. More specifically, an image of the projection screen is acquired and its four corners are mapped to the four coordinates *(0, 0), (0, h), (w, h) and (w, 0)* of the computer's desktop, where *w* and *h* stand for desktop's width and height, respectively. The IR filter is not mounted on the camera during this operation. Figures 2(a)-2(b) and 2(c)-2(d), show examples of the geometric correction process.

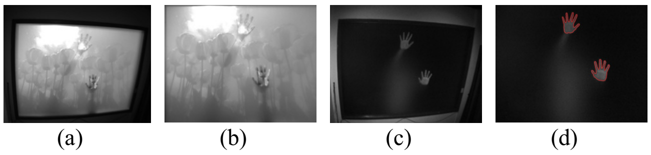


Figure 2. (a), (c): Images acquired by the camera without and with the IR filter; (b), (d): Geometrically corrected images corresponding to (a) and (c). In (d) the results of object contour detection after thresholding are also superimposed.

4.2 Photometric calibration

Photometric calibration is required because the two employed IR illuminators do not emit their light uniformly in space. As a consequence, the infrared light measured by the camera is not uniform over the projection screen. Practically, the infrared light measured closer to the borders of the screen is weaker than that in the center. Therefore, the selection of a single threshold on the camera response is not possible. To cope with this problem, an online photometric calibration process has been devised. A 5x5 grid of locations is spread over the entire projection screen. At each location *(i, j)*, a small rectangular area is displayed. The response t^L_{ij} of the camera is measured in the case where no object appears in front of each tile. t^L_{ij} is the average image value in the tile at position *(i, j)*. Then, the user is prompted to put his hand on each of these tiles. The size of each tile is selected so that it can be completely covered by a human hand. The new camera response t^H_{ij} is measured again. A threshold t_{ij} is then determined as:

$$t_{ij}=s\, t^L_{ij}+(1-s)\, t^H_{ij} \tag{1}$$

In Eq. (1), *s* is a sensitivity factor taking values in the range [0..1], controlling the degree of conservativeness on local threshold selection. The photometric correction is finalized by performing a 2D interpolation of the computed thresholds t_{ij}, which assigns a threshold to each individual pixel of the input image. Figure 3 shows 3D plots of the threshold values determined through this process. Thresholds are higher in the center of the display surface because the measured IR reflectance is higher in the center compared to the periphery.

The described photometric calibration process is simple, fast, effective and can easily be performed by non-computer specialists. This largely facilitates the deployment of the system in a wide range of applications. Besides taking care of the non-uniformity of IR reflectance and camera response, this process simultaneously adjusts the system to possibly changing lighting conditions, either in front of behind the projection screen.

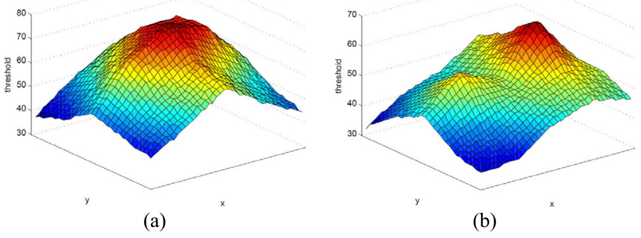


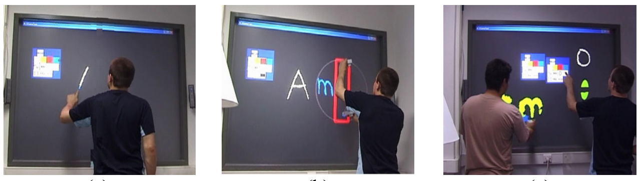
Figure 3. The thresholds identified by the photometric calibration process (a) for one IR spot (b) for two IR spots.

4.3 Blob detection and tracking

During online operation, the input image acquired by the camera is geometrically corrected (cf. section 4.1) and then thresholded based on the pixel-wise defined thresholds (cf. section 4.2). At this stage of computation, the intermediate representation is an image with dimensions *W*x*H*, depicting the effective screen surface. This is a binary image, with a value of 1 corresponding to detected objects, and a value of 0 for background. To group individual pixels into objects and track them in time, we rely on an existing blob tracker [2]. While in the original work blobs correspond to skin-colored objects detected with a Bayesian classifier, in the context of this work blobs are formed based on the binary image described earlier. The employed tracker can handle multiple objects that may move in complex trajectories and occlude each other. The number of tracked objects may vary in time. As a byproduct, the tracker provides the contour for each tracked object and its 2D orientation. Tracking is performed in real time (30fps) in 320x240 images.

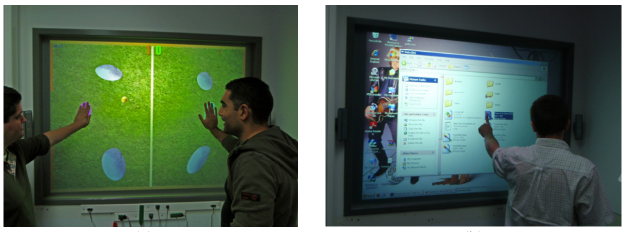
4.4 Performance considerations

Aiming at improved performance, the order in which processes run in the actual implementation of the system is modified. The matrix of thresholds t_{ij} is mapped back to the original, distorted view and thresholding is performed on the original, raw image acquired by the camera. Once blobs are detected, they are mapped to the corrected view where tracking is performed. Thus, geometric correction is applied to a small fraction of data (i.e., blob centroids, blob contour points, etc), saving a lot of computations on unnecessary homography transformations. Additionally, image resolution can be properly tailored to match accuracy / performance requirements.



(a) (b) (c)

Figure 4. Use of the developed system in a multi-touch drawing application: (a) single user drawing, (b) single-user, bimanual manipulation of a geometric primitive, (c) multi-user drawing.



(a) (b)

Figure 5. Use of the developed system (a) in the context of a pong game, (b) for controlling the Windows environment through hand gestures.

5 Sample applications

Three simple applications were developed for testing the performance and capabilities of the developed system. The first application is a basic drawing environment (see Fig. 4). The projection screen is considered as a drawing canvas and one or more palettes are projected on it. Each palette can be freely moved around and has a few buttons allowing the user to (a) select a color for drawing, (b) get in erase mode, (c) freely draw on the board using fingers, hands, or any kind of object (e.g., real paintbrushes) (d) draw predefined shapes such as a rectangle, an ellipse and a line and, (e) manipulate (e.g., translate, rotate, resize, modify the thickness) these shapes using single- or multi-handed interaction. Several users may simultaneously operate on this canvas, each with his own palette. When several palettes are concurrently present, each part of the canvas is affected by the settings of the closest palette.

The second application developed is a version of the classical "pong" game (see Fig. 5(a)). Note that this application needs to access the orientation of the hands of the players, to make possible the correct estimation of the direction of the ball's motion. Additionally, the speed of the ball is set proportional to the speed of the hand bouncing it, which is computed through its effective tracking in time.

A third application was developed aiming at using the proposed system to control the Windows environment through hand gestures (see Fig. 5(b)). Simple hands activity in front of the projection screen is mapped onto typical mouse events (click, double-click, right click and drag) allowing a user to operate the Windows environment. The collected experiences from a series of user trials reveal that the developed system is a robust and effective means for supporting such types of applications. Videos demonstrating the use of the developed multi-touch display are available at http://www.ics.forth.gr/ ~argyros/research/smartboard.html.

6 Conclusions & future work

In this paper, the development of a multi-touch screen has been described. We showed that such a multi-touch screen can be developed based on cheap, commodity hardware and an appropriate selection of computer vision techniques. We also demonstrated three instances of the use of such a system in example application scenarios. Currently, we are in the process of implementing a variant of this display following the *frustrated total internal reflection* (FTIR) approach [4]. Furthermore, we aim at investigating the interaction techniques and methods that can be supported by using cameras both in front of and behind the display.

Acknowledgments: This work has been partially supported by the FORTH-ICS internal RTD Programme "AmI: Ambient Intelligence Environments".

References

[1] Apple Inc. 2008. http://www.apple.com/iphone/

[2] Argyros, AA, and Lourakis, M.I.A., Real time Tracking of Multiple Skin-Colored Objects with a Possibly Moving Camera, in Proc. of ECCV'04, Springer-Verlag, vol. 3, pp. 368-379, Prague, Czech Republic, May 11-14, 2004.

[3] Dietz, P. and Leigh, D., DiamondTouch: a Multi-user Touch Technology, In Proc. of UIST'01, Orlando, Florida, Nov. 11-14, 2001.

[4] Han, J. Y., Low-cost Multi-touch Sensing Through Frustrated Total Internal Reflection, In Proc. of the UIST'05, Seattle, USA, Oct. 23-26, 2005.

[5] Hartley, R. and Zisserman, A. Multiple View Geometry in Computer Vision. Cambridge University Press, 2000.

[6] Intel OpenCV Computer Vision Library (C++), http://www.intel.com/research/mrl/research/opencv/.

[7] Kaltenbrunner, M. and Bencina, R., reacTIVision: A Computer Vision Framework for Table-based Tangible Interaction, In Proc. of TEI'07, Baton Rouge, Louisiana, Feb. 15 - 17, 2007.

[8] Leibe, B. et al, Toward Spontaneous Interaction with the Perceptive Workbench, IEEE Comp. Graphics and Apps, 20 (6) 54-65, 2000.

[9] Matsushita, N. and Rekimoto, J., HoloWall: Designing a Finger, Hand, Body, and Object Sensitive Wall, In Proc. of UIST'97, Banff, Alberta, Canada, Oct. 14-17, 1997.

[10] Microsoft Co., 2008. Microsoft Surface, http://www.surface.com

[11] Rekimoto, J., SmartSkin: an Infrastructure for Freehand Manipulation on Interactive Surfaces, In Proc. of the SIGCHI Conference on Human Factors in Computing Systems, Minneapolis, USA, Apr. 20-25, 2002.

[12] Smith, D., Building a Multi-Touch Sensitive Table, http://dundee.cs.queensu.ca/wiki/index.php/Building_a_Multi-Touch_Sensitive_Table, 2007.

[13] Wilson, A.D., PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System. In Proc. of the UIST'05, Seattle, USA, Oct. 23-26, 2005.

[14] Zhang, Z., A Flexible New Technique for Camera Calibration, IEEE Transactions on PAMI, 22(11):1330-1334, 2000.