Multimed Tools Appl (2011) 51:963–982
DOI 10.1007/s11042-009-0425-2

A multi-touch platform based on four corner cameras and methods for accurately locating contact points

De-xin Wang · Qing-bao Liu · Mao-jun Zhang

Published online: 3 December 2009

Abstract This paper presents a low-cost and scalable multi-touch platform which uses four cameras to reduce occlusion. Three methods are provided for locating contact points on this platform, including the lookup table, vanishing point, and 3D reconstruction. With each of these methods, respectively, the contact point is located using the projection center and the reference point, the projection center and the vanishing point, and the back-projected rays of epipolar geometry. If the four directing lines of a contact point intersect, the contact point is considered to be real; if the lines do not intersect, the point is rejected. Experimental results indicate that all three methods are capable of locating contact points even under conditions of occlusion. The lookup table and vanishing point methods are, respectively, best suited to small and large platforms, while the accuracy of 3D reconstruction method has been found to be sensitive to the physical setup. The approach proposed here can be directly installed on existing display platforms and thus should be of practical applicability in the near future.

Keywords Multi-touch · Surface computing · Human computer interface · Lookup table · Vanishing point · 3D reconstruction

1 Introduction

The multi-touch approach to user interfacing refers to a set of interaction techniques, including a touch screen or touchpad, as well as software that recognizes multiple

D.-x. Wang (✉) · M.-j. Zhang
Department of System Engineering, College of Information System and Management,
National University of Defense Technology, Changsha, China 410073
e-mail: nksky.wdx@gmail.com

M.-j. Zhang
e-mail: maojun.z@gmail.com

Q.-b. Liu
C4ISR Technology Key Laboratory of Science and Technology for National Defense,
National University of Defense Technology, Changsha, China 410073
e-mail: qbliu_peter@live.cn

simultaneous contact points. It allows users to interact with a system through the concurrent use of several fingers and permits multiple users to work together through collaborative hand gestures, making it especially useful for larger interaction scenarios such as interactive walls and tabletops [1]. It has gained much attention recently due to widely-disseminated research conducted by Han et al. [1, 2] and with the advent of the iPhone [1, 3] and Microsoft Surface [1, 4, 5]. Over the past several decades, significant advances have been made in multi-touch sensing, but many challenges remain in the development of systems suitable for mass applicability [6]. The outstanding issues include:

- Price/cost

  Most currently-available systems rely on projectors for the visual display. These systems are costly, particularly those with large displays which require high resolution and short-focus projectors [6].
- Occlusion/precision

  The majority of current approaches are capable of simultaneously locating a maximum of two contact points, and some enable only a single contact point. Even those that recognize two contact points provide an optimized rather than an actual result, degrading the locating precision of contact points and rendering them unsuitable for large displays aimed toward multiple users [7, 8].
- Packaging/setup

  Projector-based systems require long backspace distances or high ceilings for image projection, resulting in large, bulky, and enclosed packaging [6]. Some have special environmental requirements pertaining to ambient brightness, temperature, and humidity. Most cannot be installed directly on existing display platforms such as computer and LCD monitors or TV sets, limiting their potential for wide use.
- Scalability to large displays

  Some systems are not scalable to large displays or to DLP screens of any size without significant degradation of locating precision and exhorbitant cost.

This paper addresses these issues via a system that is low-cost, scaleable, and improves locating accuracy through the reduction of occlusion. We use four infrared cameras fixed at the corners of the frame to capture images synchronously. Our approach detects contact points in the view field of each camera, draws lines relating to the contact point to the camera position, and locates contact points by triangulation among these lines. Contact points are precisely located using three methods, including the lookup table, vanishing point, and 3D reconstruction, to draw the directing lines and compare the resulting locations. Experimental results indicate that the lookup table and vanishing point approaches are, respectively, best suited to small and large platforms. 3D reconstruction is the simplest method, but its accuracy can be degraded by its sensitivity to the physical setup. The solution proposed here should prove to be of significant utility to designers of interactive interfaces.

Section 2 of this paper presents a survey of research related to the topic, while Section 3 outlines the setup of our prototype system. In Section 4, we detail our method for the detection of contact points. Section 5 illustrates the use of the lookup table, vanishing point, and 3D reconstruction in locating contact points and provides an analysis of the accuracy of each approach. In Section 6, we compare the results of experiments performed with the three methods; a summary and directions for future work are offered in Section 7.

2 Related work

In this section, we will briefly survey the currently-available approaches to multi-touch interfacing and compare these methods to the prototype to be presented here.

2.1 Camera projector-based systems

- FTIR

Jeff Han [2] has developed touch surfaces based on FTIR (frustrated total internal reflection) involving a projector and an infrared camera. In his implementation, an acrylic surface lined with infrared LEDs acts as an optical waveguide whose property of total internal reflection is disturbed when a finger touches the acrylic surface. The infrared light scatters on the fingertips, creating white spots that are captured in images taken by an infrared camera behind the acrylic surface. These spots are analyzed by software to determine the fingertip positions [7].

An FTIR system can precisely locate multiple fingers without occlusion, and this has therefore become the mainstream approach at present. It is, however, burdened with the many special requirements of rear-projection systems, including a short-focus projector, projection screen, and the reconfiguration of the computer as a space-consuming table.

- Microsoft Surface

Microsoft Surface [4, 5] uses cameras positioned in an interactive table surface to track any object touching the table. While this system appears to be responsive and accurate, its cost is well beyond the reach of most consumers, a major obstacle to adoption, which our prototype alleviates.

- Multiple cameras

Itai, Katz et al. [7] have presented a multi-touch system using overhead and side cameras. The overhead camera calculates the distance of each finger from the touch surface and projects the corresponding points to locate the fingertip in three dimensions. Once the fingertip positions are acquired in overhead pixel coordinates, they must be converted into side-mounted pixel coordinates for contact detection. Ankur, Agarwal et al. [8] replace the overhead camera with a stereo camera to avoid this delay and detect the contact more precisely.

These systems use cameras to provide a high rate of input information and apply computer vision algorithms efficiently to minimize the interface latency. However, occlusion can be expected to occur with any side camera and is common even for only two contact points. The overhead- and side-camera setup is also space-consuming.

2.2 Capacitive sensor array projector-based systems

Other projector-based systems use capacitive sensors instead of a camera to track fingers. The Smart-Skin [9] and the DiamondTouch [10] are prominent systems in this product class. Since they are opaque, they require a front-projection display surface, which is space-consuming and places cumbersome restrictions on user movements; for example, hands must be exposed to the sensors without any occlusion.

In general, cost and space requirements are major obstacles to the wide adoption of projector-based systems. Such systems cannot be installed directly on existing display platforms, and they demand vigilance by users to ensure that occlusion does not occur.

2.3 Sensor array systems

a) Capacitive sensors

A multi-touch system using electrical capacitance [2, 9–12] involves a screen covered with electrodes. The touch of a user's finger on an electrode permits the flow of current through the contact point, allowing the system to locate the point of contact. However, inadvertent contact by any other part of the body will also be indiscriminately detected. Changes of temperature and humidity will cause screen drift, degrading accurate detection of contact points. Furthermore, the system is unable to handle occlusion caused by multiple contact points in the same portion of the array, and scaling to large displays will be costly.

b) Infrared array

ThinSight [13] employs an array of infrared emitters and detectors arranged behind an LCD panel to track fingers as they touch the surface. Each emitter emits infrared light that passes through the entire panel. Any reflective object in front of the display, such as a fingertip, will reflect back a fraction of the light, and this can be detected. Embedded electronics are used to reduce panel thickness, but this also substantially increases cost. The resolution of the system is currently quite low.

Lu et al. [14] have proposed another multi-touch screen using an infrared light array. The infrared LEDs are attached to the screen so that they emit in both horizontal and vertical directions. In each direction, there are two receivers corresponding to every transmitter, and all transmitters are set to different frequencies to avoid interference. To detect a touch, the system initially integrates the signals of two receivers in a single direction. If this proves insufficient, it will incorporate both the horizontal and vertical signals. Resolution is dependent on the number of LEDs, the scanning frequency, and the interpolation algorithm. This system is also unable to resolve occlusion caused by more than one contact point.

2.4 Multiple corner cameras

NextWindow [15] uses two corner cameras on top of the frame but is unable to resolve occlusion caused by more than one contact point. Another multiple-camera system, DViT [16], is closer to the prototype to be presented below but again can accommodate only a single contact point.

2.5 Others approaches

a) Acoustic sensing

The TViews [6, 17] prototype uses an HD LCD and tracks the location of objects using acoustic sensing. However, it does not support finger input, and its size is dependent on the size of the LCDs used.

b) High-definition LCD

Nima and Motamedi [6] have modified a high-definition LCD monitor as the display platform to reduce costs by eliminating the need for a projector. The camera setup and finger-and object-detecting algorithm used in this system are similar to those employed by the FTIR approach. However, the process of modifying the LCD monitors, which includes unfolding the circuit board, removing the white film sheet, and repackaging the monitor

into a table surface, is quite difficult. The resulting screen may be too large to be fully captured by a camera, and the display size is again dependent on the size of the LCDs used.

3 Term definition

- Contact point

A contact point refers to the touch or contact to the multi touch platform by a finger, hand or stylus, whose presence and location can be detected by the multi touch platform.

- Reference point

During system calibration, we manually place n contact points around the platform frame, and calculate their physical coordinates and image coordinates then save them as system parameters for later locating process. These contact points are called reference points.

- Vanishing point

For photographic lenses and the human eye, objects are projected along lines emerging from a single point, which also means that lines parallel in nature appear to intersect in the projected image. The single point or the intersection point is called vanishing point.

- Real/false point

The directing lines of contact points during the locating process will produce many intersection points, and the physically existing contact point of these intersection points are called real points, otherwise they are called false points.

4 Prototype setup

The system that we have developed is illustrated in Fig. 1. The platform consists of a rectangular frame, four infrared cameras with wide-angle lenses and visible light filters mounted at the corners of the frame, and an array of infrared LEDs positioned along the frame to supply background light. If there is no touch in the interaction area, each camera will capture an image with a white strip, while a touch by a finger or other object will be detected as a shadow on the strip. The object-detecting algorithm is applied to detect the

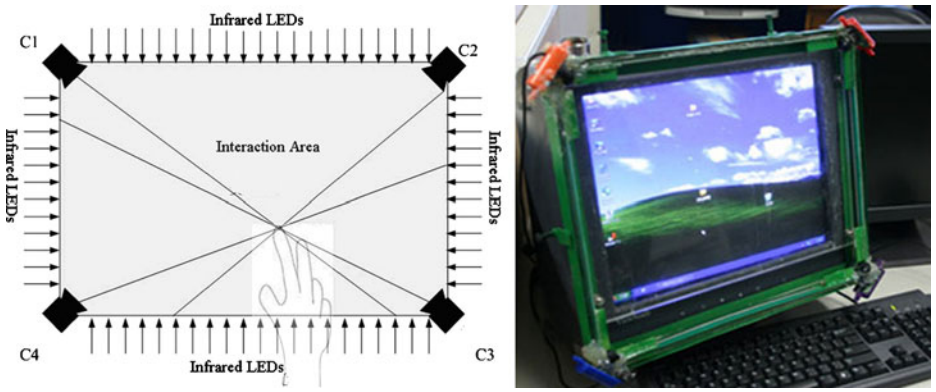

Fig. 1 Multi-touch platform using four cameras

contact point in each camera view, and the actual position of the contact point can be located by integrating the detected results from the four cameras.

The four cameras can be grouped into two diagonal camera pairs. Since the length of the frame is not equal to its width and each camera is located at the angular bisector of each corner, the principle of geometric optics indicates that any object will be captured by at least one member of each diagonal camera pair. As a result, a minimum of two lines can be drawn, with their point of intersection representing the actual position of the contact point.

This approach is much more effective in reducing occlusion than other existing methods. As illustrated in Fig. 2a, existing methods can be represented as scanning in two directions. If more than one point appears in a given direction, the point-location results will include many false positives. For example, in direction Y as illustrated below, points F1 and F2 will not be distinguished from each other, and neither will points F3 and F4. Similarly, in direction X, points F1 and F3 will be conflated, as will points F2 and F4. Points F1F4, F2F3, F1F3F2, F1F3F4, F2F4F1, F2F4F3, and F1F2F3F4 will produce the same detection results. It is evident that such a system cannot identify the number of contact points and locate their actual positions. Using our method, while points F1 and F4 will appear identical to camera C1, as will points F2 and F3, they can be distinguished by camera C3. Similarly, while points F3 and F4 will be indistinguishable to camera C2, as will points F1 and F2, they can be distinguished by camera C4. As indicated in Fig. 2b, the actual position of the contact point is identified by the intersection of the lines of sight emanating from each camera. We can thus determine the number of contact points and locate them precisely.

Situations may arise in which our method may also be confounded by occlusion. A point may be placed in such a way that no cameras or only one camera will have a view of the contact, so that its position cannot be cross-located. For example, placing a ring or cylinder on the surface will cause all points inside the ring to be invisible to the cameras. In practice, however, such a scenario is unlikely to arise, particularly during a dynamic interaction.

Setting aside such improbable situations, the differences between the proposed and existing systems are laid out in Table 1.

5 Detection of contact points by merging edges

As noted above, our proposed approach provides background illumination via infrared lighting and adds visible light filters to the corner cameras. These modifications allow the

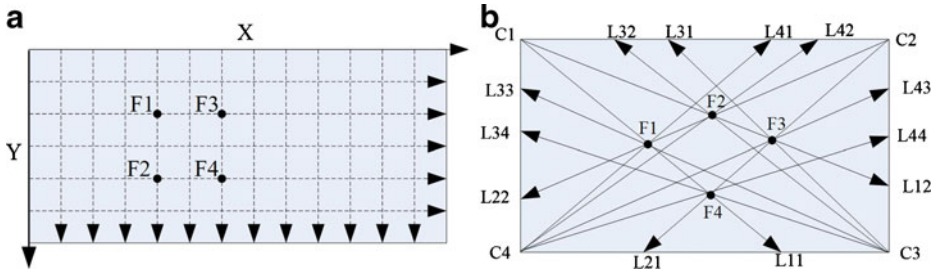

Fig. 2 Comparison of occlusion between existing and proposed methods. **a** Existing methods. **b** Proposed method

Table 1 Comparison of our prototype system with existing systems

Systems	Cost	Large size cost	Occlusion	Resolution	Special requirements	Setup
FTIR	high	high	none	high	brightness, space	modify
Surface	high	high	none	high	Space	modify
Multiple cameras with a projector	high	high	>=2	high	Space, user activity	modify
Multiple cameras without a projector	low	low	>=2	high	brightness, space	directly
Capacitive sensor array and a projector	high	high	>=2	low	temperature, humidity	directly
Capacitive sensor array	low	high	>=2	low	temperature, humidity	directly
IR sensor array	low	low	>=2	low	brightness	directly
Our	low	low	>=5	high	none	directly

platform to be used in any environment, without requiring any special controls on brightness, temperature, and humidity. Additionally, objects can be easily deleted by toggling from two-dimensional to one-dimensional object location.

It is possible to detect "blob" objects by morphological operations and to determine their boundary rectangles using a connected component-finding method. With this approach, the center of the boundary rectangle is set at the pixel coordinate of the contact point. When the contact points are closely proximate, however, they will appear to merge into a single blob, generating inaccurate results with centers slightly offset from their actual positions.

To address this inaccuracy and increase detection speed, we further simplify the background images by shrinking the white strip into a white line and then detecting points on the line instead of blobs on the strip. This process is accomplished through the following steps:

1 Use a binary mask image (height m, width n) to generate a white strip of captured images. A pixel in the mask image that is located within the strip will have a value of 1; pixels outside the strip have a value of 0.

$$Mask(x_0, y_0) = \begin{cases} 1, (x_0, y_0) \in whitestrip \\ 0, else \end{cases} \tag{1}$$

2 Identify the centerline of the white strip, and $[i]$ returns the closest integer value for i.

$$Line(x) = \left[\sum_{i=1\cdots m} Mask(x, i) \times i \Big/ \sum_{i=1\cdots m} Mask(x, i) \right] \tag{2}$$

3 Calculate mean values for pixels on the line. If $f(x_0, y_0)$ is the gray value of pixel (x_0, y_0) for captured images, replace the value of the detected line by the mean value of its 3×3 neighbor, generating a mean value $Meanvalue(x_0, y_0)$ by

$$Meanvalue(x, L(x)) = \sum_{i=-1,0,1} Mask(x+i, Line(x+i) * f(x+i, Line(x+i))) \Big/ \sum_{i=-1,0,1} Mask(x+i, Line(x+i)) \tag{3}$$

We captured images of the system without any contact points over a 10-minute period and analyzed the maximum and minimum mean values of every pixel on the line for all four cameras, as illustrated in Fig. 3.

We have observed that substantial variability exists in the gray values both of different regions within a given image and of a single pixel over a period of time. For this reason, time-based detecting methods are typically unable to establish a threshold sufficiently robust to detect all contact points. Instead, we have elected to detect contact points within a single frame, regardless of time information, through the following process:

(1) Acquire mean values for pixels on the line of currently captured image, following Eq. 3

(2) Acquire the gray gradient value for pixels on the line

$$Gradient(x_0, Line(x_0)) = 2*Meanvalue(x_0 - 1, Line(x_0)) - 2*Meanvalue(x_0 + 1, Line(x_0)) \quad (4)$$

(3) Find local extrema from gray gradient values, where maximum extrema are the left edges of contact points *LeftEdge*, minimum extrema are the right edges of contact points *RightEdge*, and both extrema meet the conditions $|LeftEdge(i)| > T_1$, $|RightEdge(i)| > T_1|$. Only the gray gradient value larger than T_1 can be accepted as an edge. T_1

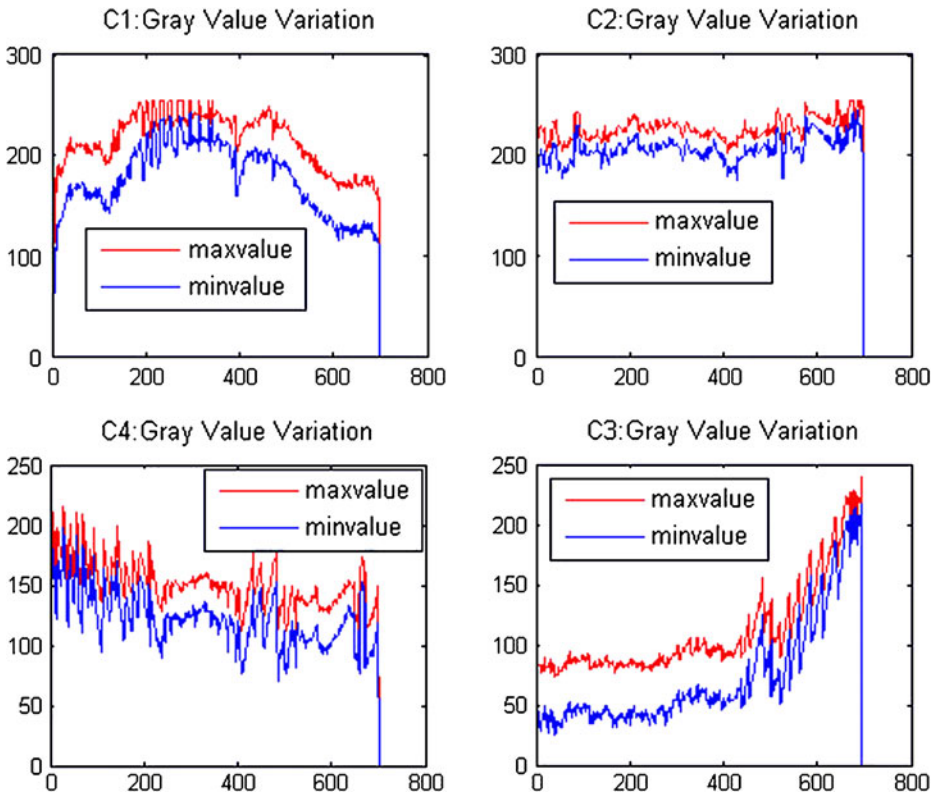

Fig. 3 3×3 neighbor mean value variation for over a 10-minute period

can be determined by experimental statistics and can be a value of quite a wide range rather than a single value.

(4) Merge the edges meeting the condition $T_2 < RightEdge(j) - LeftEdge(i)) < T_3$ to determine the detected contact points. *Detect* by $Detect(k) = (LeftEdge(i) + RightEdge(j))/2$ will be the pixel coordinate for the contact point. T_2 and T_3 are thresholds to eliminate isolated *LeftEdge* or *RightEdge*. Only those objects with width between T_2 and T_3 can be accepted. They can be set, for example, the minimal desired object width for T_2 and the maximal desired object width for T_3.

The same process is followed to detect contact points and determine their pixel coordinates for all four cameras. Figure 4 shows the captured images, Fig. 5 the 3×3 neighbor mean values for captured images and detected results, and Fig. 6 the gradient value for captured images and detected edges; all results were obtained using the proposed method.

After following this detection process for all four cameras, we arrive at the pixel coordinate set of all contact points $G = \{Gj\}$, $G_j = \left\{g_{jt_j}\right\}$, where g_{jt_j} is pixel coordinate of the contact point t_j in camera C_j and $t_j = 1 \cdots\cdots n_j$.

6 Locating contact points by directing lines and triangulation

Simplification of the camera images from two dimensions to one dimension, as described above, produces detecting results that represent directions of contact points rather than their actual positions. While this step prohibits direct location of contact points, we can draw direction lines for all contact points in all cameras and then locate contact points by triangulation. Three methods can be employed for fixing these direction lines: a lookup table, vanishing points, and 3D reconstruction.

6.1 Build directing lines for objects

The physical coordinates of the contact points are defined using five coordinates, as shown in Fig. 7. Respectively, C_1XYZ, C_2XYZ, C_3XYZ, and C_4XYZ represent the physical

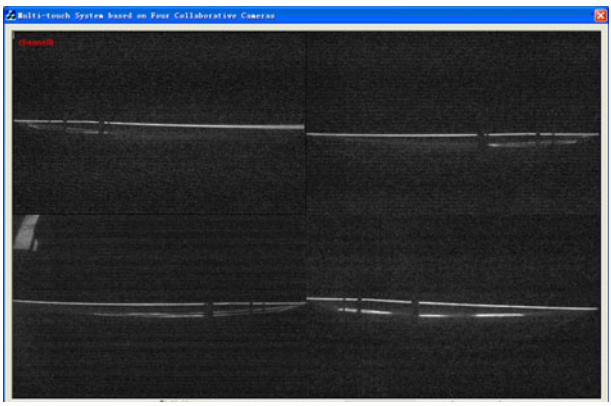

Fig. 4 Captured images

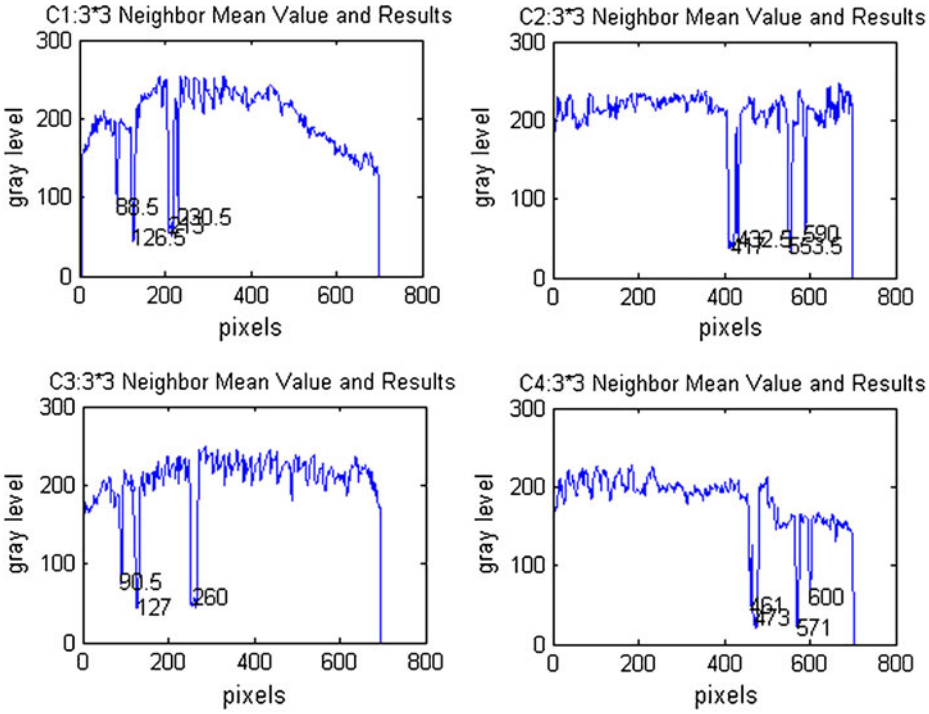

Fig. 5 3×3 neighbor mean value and detected results

coordinates corresponding to C_1, C_2, C_3, and C_4, and IXYZ represents the interaction area with an origin at the top-left corner of the physical frame.

If we assume the interaction area coordinate of a contact point to be $X = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ and the pixel coordinate for one camera to be $m = \begin{bmatrix} u \\ 1 \end{bmatrix}$, then, according to the pin-hole projection model, we get $sm = PX$, where P is the projection matrix of the camera with a 2×3 shape and 5° of freedom, and s is the scale parameter.

Let us set $P_{2*3} = \begin{bmatrix} \mathrm{P}_{1T} \\ \mathrm{P}_{2T} \end{bmatrix}$, where P_{1T} is the first row vector of P, and P_{2T} is the second row vector of P; then $\begin{cases} su = \mathrm{P}_{1T}X \\ s = \mathrm{P}_{2T}X \end{cases}$, and $\mathrm{P}_{1T}X - uP_{2T}X = 0$. Given n points, we can write this in matrix form as $AP = 0$, where $A = \begin{bmatrix} x_1 - u_1x_1 \\ x_2 - u_2x_2 \\ \cdots \\ x_n - u_nx_n \end{bmatrix}$ and solve P by the singular value decomposition of A.

If the projection center of a camera is $C = [C' \quad 1]^T$ and $P = [H \quad p_3]$, where H is a 2×2 matrix of the first two column vectors of P and p_3 is the third column vector of P, we get a value for C, when $PC = 0$, $HC' = -p_3$, and $C' = H^{-1}p_3$, of $C = \begin{bmatrix} -H^{-1}p_3 \\ 1 \end{bmatrix}$ [18, 19].

Once the projection center of a camera has been determined, only one additional point is required to locate the contact point in the camera view; this may be either a reference point or the vanishing point.

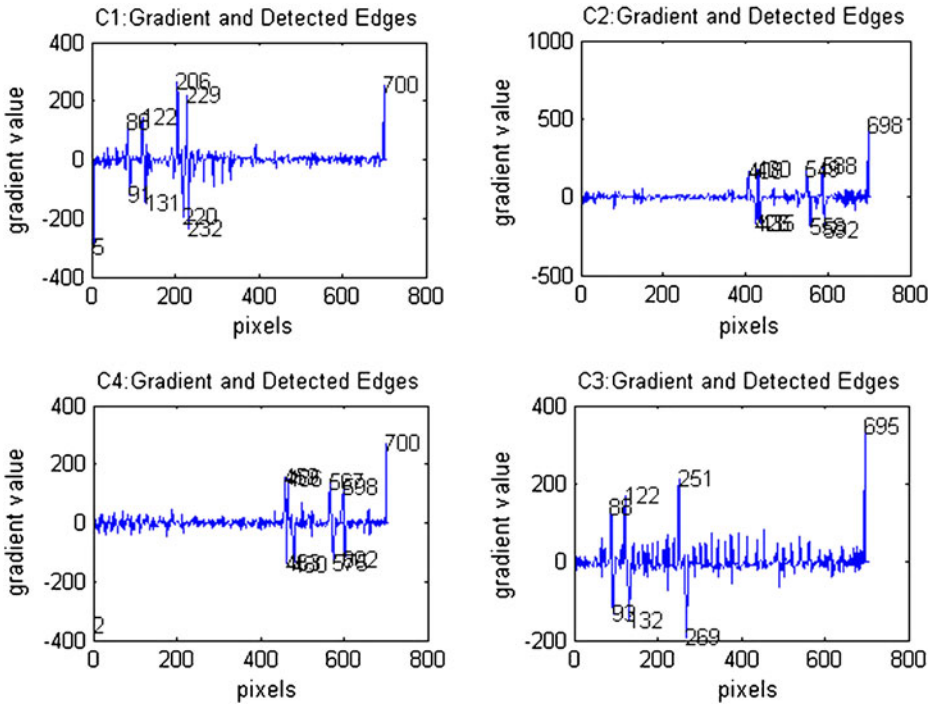

Fig. 6 Gradient value and detected edges

6.1.1 Lookup table

According to the principle of geometric optics, if two contact points share the same pixel value, they must be located on a single line originating from the projection center of the camera; this allows us to locate the contact point by its reference point.

First, we designate n reference points at equal intervals along the frame and determine their physical coordinates $W = \{w_i\}$, $w_i = \left(w_i^x, w_i^y\right)$, $i = 1 \cdots n$, according to the frame size. By placing an object at each reference point and using four cameras to capture and detect each of these objects, we can compile the pixel coordinate set for all reference points $V = \{V_j\}$, $Vj = \{V_{ji}\}$, $v_{ji} = \left(v_{ji}^x, v_{ji}^y\right)$, where v_{ji} denotes the pixel coordinate of reference point i in camera Cj, and $j = 1,2,3,4$, $i = 1 \cdots n$, as shown in Fig. 8.

Referring to the lookup table of calibration results, we can determine a corresponding reference point for the pixel coordinate of each detected contact point. If the pixel coordinates for the correlated touch and reference points happen to differ, the correct reference point can be established through interpolation. We can then draw a line passing from projection center of the camera to the corresponding reference point and thus fix a location for the contact point.

For example, given a contact point in C_1 with a pixel coordinate g_{1t_1}, we can find a reference point v_{ij_1} located at the minimal distance from g_{1t_1}, where $j_1 = 1 \cdots\cdots n$. If $g_{1t_1} = v_{j_1}$, then the corresponding reference point is w_{j_1}, and $q_{1t_1} = w_{j_1}$ where q_{1t_1} is the

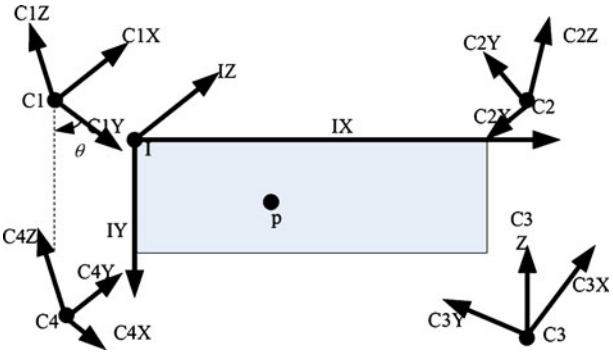

Fig. 7 Custom coordinates

target physical coordinate for contact point g_{1t_1}. If $g_{1t_1} \neq v_{1j_1}$, we can use linear interpolation to determine q_{1t_1}, as follows:

$$\begin{cases} v'_{1j_1} = v_{1j_1+1}, w'_{j_1} = w_{j_1+1}, \text{if } g_{1t_1} > v_{1j_1} \\ v'_{1j_1} = v_{1j_1-1}, w'_{j_1} = w_{j_1-1}, \text{if } g_{1t_1} < v_{1j_1} \end{cases} \tag{5}$$

$$R_1 = \frac{d\left(g_{1t_1}, v_{1j_1}\right)}{d\left(v_{1j_1}, v'_{1j_1}\right)}, R_2 = \frac{d\left(g_{1t_1}, v'_{1j_1}\right)}{d\left(v_{1j_1}, v'_{1j_1}\right)} \tag{6}$$

$$q_{1t_1} = R_1 * w'_{j_1} + R_2 * w_{j_1} \tag{7}$$

Reference Points	1	2	3	$i-1$	i	$i+1$	$i+2$	$\cdots$	n
Physical Coordinates of Reference Points	dw_1	dw_2	dw_3	dw_{i-1}	dw_i	dw_{i+1}	dw_{i+2}	$\cdots$	dw_n
Pixel Coordinates of Reference Points In C1	ca_{11}	ca_{12}	ca_{13}	ca_{1i-1}	ca_{1i}	ca_{1i+1}	ca_{1i+2}	$\cdots$	ca_{1n}
Pixel Coordinates of Reference Points In C2	ca_{21}	ca_{22}	ca_{23}	ca_{2i-1}	ca_{2i}	ca_{2i+1}	ca_{2i+2}	$\cdots$	ca_{2n}
Pixel Coordinates of Reference Points In C3	ca_{31}	ca_{32}	ca_{33}	ca_{3i-1}	ca_{3i}	ca_{3i+1}	ca_{3i+2}	$\cdots$	ca_{3n}
Pixel Coordinates of Reference Points In C4	ca_{41}	ca_{42}	ca_{43}	ca_{4i-}	ca_{4i}	ca_{4i+1}	ca_{4i+2}	$\cdots$	ca_{4n}

Fig. 8 Lookup table

When w_{j_1} is the physical coordinate of v_{1j_1}, w_{j_1+1} is the physical coordinate of the reference point following w_{j_1}, w_{j_1-1} is the physical coordinate of the reference point prior to w_{j_1}, v'_{1j_1} is the reference pixel coordinate used to interpolate, w'_{j_1} is its physical coordinate, and $d(a, b)$ is the distance between a and b.

We can draw a line passing through C_1 and q_{1t_1} to locate contact point g_{1t_1} in C_1, which can be denoted by $y_{1t_1} = k_{1t_1}x_{1t_1} + b_{1t_1}$, $k_{1t_1} = \frac{y_{t_1} - C_1^y}{x_{t_1} - C_1^x}$, $b_{1t_1} = C_1^y - k_{1t_1} * C_1^x$.

The same method can be used to establish lines for the other cameras, resulting in the line set $L = \{L_1, L_2, L_3, L_4\}$ where $L_j = \{l_{j1}, l_{jt_1}, \cdots\cdots, l_{jn_1}\}$, represents all the lines in camera C_j, and l_{jt_j} is the directing line for contact point t_j in camera Cj.

6.1.2 Vanishing point

Since the vanishing point in the direction of a contact point for a given camera can be represented by $q = -H^{-1}m$, we can draw the directing line using to the projection center and the vanishing point.

Given a contact point in camera C_1 with pixel coordinate g_{1t} and vanishing point $q_{1t} = -H_1^{-1}g_{1t}$ [18, 19], then the line locating the contact point can be represented by $l_{1t} = [C_1]_\times q_{1t}$, where $[\mathbf{x}]_\times = \begin{bmatrix} 0 & -t & y \\ t & 0 & -x \\ -y & x & 0 \end{bmatrix}$ and $\mathbf{x} = (x, y, t)^T$ [18, 19]. In the same manner, given a contact point in camera C_2 with pixel coordinate g_{2j} and vanishing point $q_{2j} = -H_2^{-1}g_{2j}$, a contact point in camera C_3 with pixel coordinate g_{3k} and vanishing point $q_{3k} = -H_3^{-1}g_{3k}$, and a contact point in camera C_4 with pixel coordinate g_{4s} and vanishing point $q_{4s} = -H_4^{-1}g_{4s}$, we can determine that the other three directing lines will be $l_{2j} = [C_2]_\times q_{2j}$, $l_{3k} = [C_3]_\times q_{3k}$, and $l_{4s} = [C_4]_\times q_{4s}$.

Using this method, the set of all directing lines L can be established for all detected points in all camera views.

6.1.3 3D reconstruction

Stereovision reconstruction commonly involves the identification of a 3D point from its measured pixel positions in two or more views. The 3D point is located at the cross-point of back-projected rays of epipolar geometry and can be obtained by triangulation. The same method can be used to direct the contact point in each camera.

With our four-camera setup, we set the C_1XYZ coordinate as the reference coordinate and group the other cameras with C_1. Since C_1 and C_3 are located at diagonal positions on the frame, it is difficult to calibrate their motion parameters. This can be addressed by selecting another camera as a transition, and establishing three pairs; for example C_1C_2, C_1C_4, and C_2C_3 or C_4C_3.

Each pair of cameras is then calibrated using by SFM (structure from motion) [18] to determine the intrinsic parameters and motion parameters between the cameras of each pair, including projection matrices P_1,P_2,P_3,P_4, transition matrices T_{21},T_{41},T_{32},T_{34}, and rotation matrices R_{21},R_{41},R_{32},R_{34}, where P_j represents the projection matrix of C_j, T_{ij} represents the translation matrix for C_iXYZ to C_jXYZ, and R_{ij} represents the rotation matrix for C_iXYZ to C_jXYZ.

We can select at least two points to calibrate the motion parameters from C_1XYZ to IXYZ, including the rotation matrix R_{C_1I} and the transition matrix T_{C_1I}.

6.2 Location of contact points by triangulation of directing lines

Through the techniques described above, we can determine the set of all directing lines originating at all four cameras which pertain to a given contact point. We can then locate the contact point by triangulation with the four directing lines using the lookup table, vanishing point, or 3D reconstruction.

6.2.1 Lookup table and vanishing point

The directing lines determined for all contact points can be grouped by selecting a line l_{1t_1} from L_1, a line l_{2t_2} from L_2, a line l_{3t_3} from L_3, and a line l_{4t_4} from L_4. This group of four lines can be denoted by an equation, as follows:

$$\begin{bmatrix} k_{1t_1} & -1 & b_{1t_1} \\ k_{2t_2} & -1 & b_{2t_2} \\ k_{3t_3} & -1 & b_{3t_3} \\ k_{4t_4} & -1 & b_{4t_4} \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \tag{8}$$

$$\begin{cases} \left[[C_1]_\times H_1^{-1} g_{1t_1}\right]^T f = 0 \\ \left[[C_2]_\times H_2^{-1} g_{2t_2}\right]^T f = 0 \\ \left[[C_3]_\times H_3^{-1} g_{3t_3}\right]^T f = 0 \\ \left[[C_4]_\times H_4^{-1} g_{4t_4}\right]^T f = 0 \end{cases} \tag{9}$$

where ($t_1 = 1\cdots n_1$; $t_2 = 1\cdots n_2$; $t_3 = 1\cdots n_3$; $t_4 = 1\cdots n_4$). These equations can be solved for $\hat{f} = (x, y)$ using standard linear least squares. Using these equations, we can generate $C_{n_1}^1 * C_{n_2}^1 * C_{n_3}^1 * C_{n_4}^1$, and by fitting and removing the redundant solutions, we can arrive at the set of contact point physical coordinates $F = \{f_e\}$.

It is an iterative process to identify F, and e is the number of contact points at each iteration step. For each iteration, if $\hat{f}$ meets the following conditions, we will accept $\hat{f}$ as a real point $f_{e+1} = \hat{f}$, and $F = F \cup \{f_{e+1}\}$:

$$\min\left(D\left(\hat{f}, l_{ij}\right)\right) \le Th_1 \tag{10}$$

$$\min_{i=1\cdots e}\left(D\left(\hat{f}, F_2\right)\right) \le Th_2 \tag{11}$$

Here, $D(p, l)$ represents the distance between a point and a line. Equation 10 indicates that the distances from $\hat{f} = (x, y)$ to all lines fall below a threshold Th_1, which can be the mean size of a point in all cameras. Equation 11 specifies that there are no redundant answers in F, where Th_2 may be a small value such as 0.1.If these conditions are not satisfied, the contact point indicated by these four lines must be a false positive and should be removed.

6.2.2 3D reconstruction

Alternatively, contact points can be determined using 3D reconstruction. We begin by randomly selecting an object from the detection results of each camera and reconstructing

the contact point according to calibration results for each camera pair. If all the reconstructed points converge, the contact point must be real; otherwise, it will be eliminated as a false result.

For example, select an object from C1 with its pixel coordinate g_{1t_1} and another object from the detection results of C2 with its pixel coordinate g_{2t_2}. Given the intrinsic and motion parameters of the camera pair C2 and C1, we can determine their projection matrices $P_1 = K_1[I|0]$, $P_2 = K_2[R_{21}|T_{21}]$ and then reconstruct the object by the triangulation rule $r = \arg\min \sum_{i=1,2} \left\| g_{it_i} - \hat{g}_{it_i}(P_i, r) \right\|^2$ to get the 3D reconstructed point r_1. For the other two camera pairs, we can determine two additional 3D reconstructed points r_2 and r_3, transform them into interaction coordinates, and arrive at $p_1 = R_{C_1I} * r_1 + T_{C_1I}$, $p_2 = R_{C_2I} * r_2 + T_{C_2I}$, and $p_3 = R_{C_3I} * r_3 + T_{C_3I}$. However, due to measurement noise, these points would not converge. We can set a threshold to filter an acceptable level of convergence. If $\arg\min_{i=1,2,3;j=1,2,3} \left(D(p_i, p_j)\right) \leq Th_1$, we will accept these three points as the reconstructed results of a real point $\hat{f} = (p_1 + p_2 + p_3)/3$. Otherwise, the contact point identified by these four lines must be considered to be a false positive that should be removed.

Using similar 3D reconstructions, we can generate $C^1_{n_1} * C^1_{n_2} * C^1_{n_3} * C^1_{n_4}$r, and, by removing the redundant solutions according to Eq. 11, we can arrive at the set of contact point physical coordinates *F*.

6.3 Accuracy analysis

The camera resolution determines the resolution of captured images. If the captured image size is *iw***ih*, where *iw* is the width and *ih* is the height, and the size of the interaction area is *pw***ph*, where *pw* is the width and *ph* is the height, when two neighboring lines of the frame are a line in the captured image, then the pixel resolution of our method is $PD = iw/pl$, where $pl = pw + ph$.

Accuracy using the lookup table method is also dependent on the distance between two reference points. If this distance is *S* and the maximum pixel distance between two reference points is *MD*, then the physical resolution of our platform is *PD***S*/*MD*.

For the vanishing point method, the final accuracy depends on the number of equations and their solutions and will be similar to the accuracy of the look-up table method.

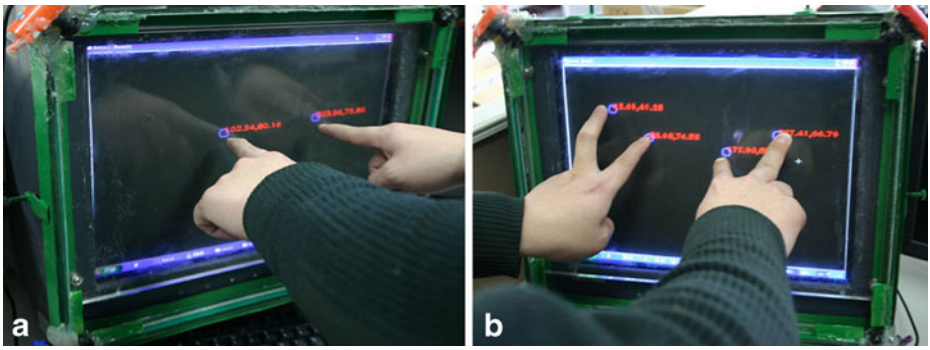

Fig. 9 Location results generated by our prototype system with **a** two contact points **b** four contact points

When 3D reconstruction is employed, accuracy is influenced by the estimate of projection matrices and motion parameters. If the reprojection error of each camera is $[T_X^1 \quad T_Y^1]$, $[T_X^2 \quad T_Y^2]$, $[T_X^3 \quad T_Y^3]$, $[T_X^4 \quad T_Y^4]$, where T_X^i is the horizontal error and T_Y^i is the vertical error for camera I, then the reconstructed results for all camera pairs would be located within a rectangle of width of $2*TH_X$ and height $2*TH_Y$, where $TH_X = \max(T_X^1, T_X^2, T_X^3, T_X^4)$ and $TH_Y = \max(T_Y^1, T_Y^2, T_Y^3, T_Y^4)$. The physical resolution of our method would therefore be $PD*TH_X$ in the horizontal dimension and $PD*TH_Y$ in the vertical dimension.

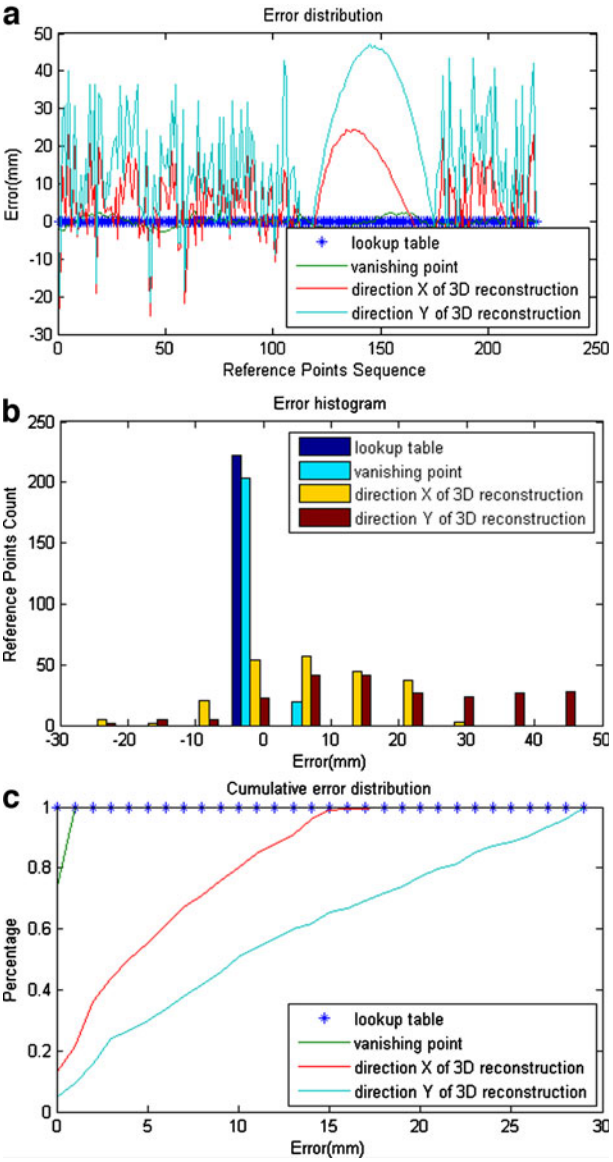

Fig. 10 Location results generated via the reference point method as compared to the lookup table approach, comparing **a** Error distribution **b** Error histogram **c** Cumulative error distribution

7 Experimental results

We have constructed a prototype multi-touch system with an interaction area equal to the display region of a common 17-inch monitor (350 mm×400 mm) to permit its applicability to any monitor of this size.

The detection and location results are provided in Fig. 9.

We positioned 225 reference points spaced at 5 mm intervals along the perimeter of the frame. Taking the lookup table as the default method, we then located these reference points using the vanishing point and 3D reconstruction methods and compared their accuracy.

Both the lookup table and vanishing point methods take one dimension of a reference point as fixed by the frame width or height. We calculated the error in locating the other dimension, with results provided in Fig. 10.

It is evident that the accuracy of the vanishing point method is close to that of the lookup table method. Nearly 80% of errors are within 2 mm and all are within 5 mm, while the errors produced by 3D reconstruction expanded from 0 to 25 mm in a uniform distribution.

Since we used a checkerboard pattern [19] for the stereo calibration of the four cameras under infrared light, it is difficult to extract corner points of the pattern and match them. As a result, the estimated projection and motion matrices appear to have large errors. If a pattern could be found which could be clearly photographed in infrared light, we believe that the accuracy of the 3D reconstruction method could be improved to a point where it could be of practical utility.

The lookup table method requires that reference points be established along the perimeter of the frame and their pixel coordinates be established for all cameras. This accurate but tedious process renders this method inapplicable to large platforms. Since the vanishing point method does not require the establishment of a dense grid of reference points but produces results nearly as accurate as those of the lookup table, this method is most applicable to larger platforms.

Consequently, we have applied the vanishing point method to a large platform built on a two-channel 2.4 m×1.35 m DLP screen, as illustrated in Fig. 11:

The DLP screen is protected with an acrylic covering that reflects lights and produces shadows on the screen.

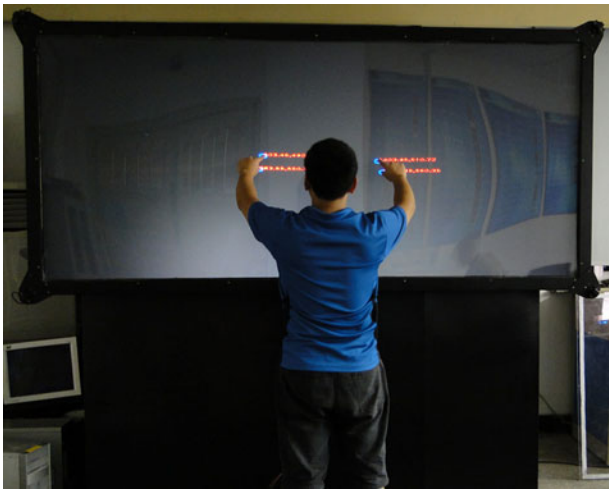

Fig. 11 Large platform to be used with the vanishing-point method

8 Conclusion and future work

In this paper, we have described a first prototype toward the implementation of a low-cost, scalable multi-touch system using four corner cameras. We have discussed the range of methods available for locating points of contact with the screen, including the detection of objects by merging edges, drawing directing lines, and locating contact points by triangulation with four directing lines. We have proposed three methods to direct and locate contacts. A comparison of experimental location results reveals that all these methods are capable of locating contact points even under conditions of occlusion. The lookup table method is best suited to small platforms and the vanishing point method has advantages for larger platforms, while the accuracy of 3D reconstruction has been shown to be sensitive to aspects of the physical setup and needs further refinement.

Since our prototype utilizes four corner cameras, it has following advantages:

- The platform offers significant reduction in problems resulting from occlusion and permits multiple points of contact to be located precisely.
- The system is low-cost and scalable, requiring only infrared LEDs and four cameras rather than expensive components such as a projector and an acrylic layer. It can be implemented in sizes comparable to those of traditional monitors but can also be extended to larger platforms, such as DLP screens.
- It imposes no special environmental requirements regarding brightness, space, humidity, or temperature, and can be directly installed on existing display platforms such as computer monitors, TV sets, and LCDs.

The solution proposed here should prove to be of significant utility to designers of interactive interfaces and should readily lead to the development of platforms that can be widely disseminated.

Building upon the methods set forth here for the location of individual contact points, we hope to establish a relationship structure among multiple points to delineate motion trajectories and permit the recognition of dynamic gestures [20]. Other potential areas of exploration include 3D object manipulation [21] and the collaboration of multiple users on a large-scale interface [22].

Acknowledgements This work is supported by National Natural Science Foundation of P.R. China under grant NO. 60705013 and National 863 plan of P.R. China under grant NO. 2009AA01Z328. Thanks to all reviewers for their thoughtful comments.

References

1. Pennock J, Tabrizi MHN (2008) A survey of input sensing and processing techniques for multi-touch systems. The 2008 International Conference on Computer Design(CDES'08), Las Vegas, Nevada, USA, Jul 2008, pp 10–16
2. Han JY (2005), Low-cost multi-touch sensing through frustrated total internal reflection. Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology, Seattle, WA, USA., Oct 2005, pp 315–319
3. iPhone (2009) http://www.apple.com/iphone. 2 Sep 2009
4. Microsoft Surface (2009) http://www.microsoft.com/surface/index.html. 2 Sep 2009
5. Buxton B (2008) Surface and tangible computing, and the "small" matter of people and design. Solid-State Circuits Conference, (ISSCC 2008), Digest of Technical Papers, IEEE International, San Francisco, CA, USA, Feb 2008, pp 24–29
6. Motamedi N (2008) HD Touch: multi-touch and object sensing on a high definition LCD TV. ACM Conference on Human Factors in Computing Systems, CHI '08, Florence, Italy, Apr, 2008. Extended Abstracts

7. Katz I, Gabayan K, Aghajan H (2007) A multi-touch surface using multiple cameras. Advanced concepts for intelligent vision systems (ACIVS). Delft University, Delft, pp 97–108
8. Agarwal A, Izadi S, Chandraker M, Blake A (2007) High precision multi-touch sensing on surfaces using overhead cameras. Second Annual IEEE International Workshop on Horizontal Interactive Human–Computer Systems (TABLETOP'07), Newport, Rhode Island, Oct 2007, pp 197–200
9. Rekimoto J (2002) SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing Our World, Changing Ourselves, Minneapolis, Minnesota, USA, Apr., 2002, pp 113–120
10. Dietz P, Leigh D (2001) DiamondTouch: a multi-user touch technology, Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST), Orlando Florida, Nov 2001, pp 219–226
11. N-trig (2009) www.n-trig.com. 2 Sep 2009
12. Elo TouchSystems (2009) www.elotouch.com. 2 Sep 2009
13. Hodges S, Izadi S, Butler A, Rrustemi A, Buxton B (2007) ThinSight: versatile multi-touch sensing for thin form-factor displays. In: Proceedings of the 20th Annual ACM Conference on User Interface Software and Technology, Rhode Island, USA, Oct 2007, pp 259–268
14. R-X Lui, J-C Zhou, J-M Li (2007) An infrared touch screen and its multi-touch detecting method. Chinese patent application NO. 200710031082.6, Oct 2007
15. Nextwindow (2009) http://www.nextwindow.com. 2 Sep. 2009
16. DViT whitepaper (2009) http://smarttech.com/DViT/DViT_white_paper.pdf. 2 Sep 2009
17. Mazalek A, Reynolds M, Glorianna Davenport (2007) The TViews table in the home. Second Annual IEEE International Workshop on Horizontal Interactive Human–Computer Systems (TABLETOP'07). Newport, Rhode Island, Oct 2007, pp 52–59
18. Torr PHS (2002) A structure and motion toolkit in matlab. Technical Report MSR-TR-2002-56, Microsoft Research, 7 JJ Thomson Avenue, Cambridge, CB3 0FB, UK, http://research.microsoft.com/~philtorr/
19. Camera Calibration Toolbox for Matlab (2009) http://www.vision.caltech.edu/bouguetj/calib_doc/index.html. 2 Sep 2009
20. Wobbrock JO, Morris MR, Wilson AD (2009) User-defined gestures for surface computing. ACM CHI 2009 ~ Tabletop Gestures, Boston, MA, USA, Apr 2009, pp 1083–1092
21. Steinicke F, Hinrichs K, Schöning J et al (2008) Multi-touching 3D data: towards direct interaction in stereoscopic display environments coupled with mobile devices, PPD 2008: workshop on designing multi-touch interaction techniques for coupled public and private displays, as part of AVI 2008, Naples, Italy, May 2008, pp 46–49
22. Morrison A, Jacucci G, Peltonen P (2008) CityWall: limitations of a multi-touch environment. Workshop on designing multi-touch interaction techniques for coupled public and private displays, as part of AVI 2008, Naples, Italy, May 2008, pp 31–35

De-xin Wang was born in Fujian, China, in 1983. He received the B.S. degree from NanKai University, TianJin, China in 2004, and received the M.S. degree from National University of Defense Technology (NUDT), Changsha, China, in 2006. He is currently a Ph. D. student of NUDT.

His research interests include human computer interaction, multimedia information system, entertainment and edutainment.

Qing-bao Liu received the mathematics B.S. degree from Jiangxi Normal University in 1990, the M.Sc. and Ph.D. degrees in management science and engineering from National University of Defense Technology in1995 and 2006, respectively. He is currently an Associate Professor of National University of Defense Technology, Changsha, China.

His research interests include machine learning, data mining, data warehouse, and pattern recognition.

Mao-jun Zhang was born in HuBei, China, in 1971. He received the B.S. and Ph.D. degrees in system engineering from National University of Defense Technology, Changsha, China, in 1992 and 1997, respectively. He is currently a professor in the department of system engineering, National University of Defense Technology.

His research interests include image/video processing, information system engineering, system imulation and virtual reality technology.