JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 30, 43-64 (2014)

Efficient Error Analysis of a Real-Time Vision-Based Pointing Systems*

KUO-HUA LO[1], JEN-HUI CHUANG[1] AND HUA-TSUNG CHEN[2]
[1]*Department of Computer Science*
[2]*Information and Communications Technology Lab*
National Chiao Tung University
Hsinchu, 300 Taiwan

In this paper, an efficient error analysis of a real-time vision-based pointing system is proposed. We use two cameras to implement the pointing system according to a simplified 3D reconstruction scheme which is based on image feature extraction, homography, and 3D geometry. To that end, we study the relationship between image noises and the ultimate reconstruction errors, and develop efficient methods to find the error range of the latter given a range of the former. Experimental results show that the proposed approach can find the error range satisfactorily. Accordingly, users of similar pointing systems can get more robust pointing results by selecting a special pointer location, or possibly a special pair of cameras, that will result in minimal range of pointing error.

Keywords: error analysis, homography, 3D geometry, pointing systems, vision-based

1. INTRODUCTION

Nowadays, interactions between human and machines are no longer restricted by using a keyboard and a mouse since researchers have developed many ways to communicate with machines. Such new ways of communication are widely adopted in many applications such as an interactive game, the control of household robot, a presentation in a conference and so on. For effective support of these applications, automatic recognitions of hand gesture and pointing direction have been two important issues of human-computer interaction (HCI).

There are different types of HCI methods which include glove-based and vision-based approaches. Glove-based methods collect motion data of finger/hand through a glove which contains many sensors. These sensors can accurately measure motion data from hands for gesture recognition. However, the glove is expensive and heavy so its usability is greatly limited. In contrast, vision-based methods use cameras to obtain image data and have less burden on users. In recent years, conventional artificial intelligence tools such as HMM [1, 2] and Neural network [3-6] have been used widely for vision-based hand gesture recognition. In [6], an innovative neural classifier called Self Growing and Organized Neural Gas (SGONG) is applied to find morphology of hands as features. These features are used to classify the raised fingers into five classes by considering their probability distributions.

In [7], a template-based hand gesture recognition system that recognizes ten kinds of hand gestures is developed. A nearest neighbor classifier is used to identify the gesture

Received December 15, 2011; revised April 27, 2012; accepted September 11, 2012.
Communicated by Tyng-Luh Liu.
* This research is supported in part by NSC 100-2221-E-009-116-MY2, 101-2220-E-009-054, Ministry of Economic Affairs under grants 100-EC-17-A-02-S1-032, and "Aim for the Top University Plan" of the National Chiao Tung University and Ministry of Education, Taiwan.

for the hand region extracted by color and motion cues. In [8], three cameras are used to get multi-angle images of one hand. Accordingly, three trained support vector machines are used to classify hand gestures, respectively, and the classification results are integrated into a final result. All these methods provide natural ways to give instructions to machines through hand gestures.

For many HCI applications, pointing directions of a user can be transformed conveniently into instructions such as asking a robot to move to desired positions or controlling a computer by a virtual mouse. While real-time computations of the pointing direction (and its target) for a user are often needed, accuracy and stability of the computation are the most desirable attributes of such pointing systems. In the following, three types of methods for recognizing pointing direction will be reviewed, which include approaches based on human eye, laser pointer, and head and hand, respectively.

Some researchers find the gaze direction through eye tracking. Such information not only can point out where a user is looking at [9], but also useful for other applications. For example, in [10] eye moments are used to control an experimental multimedia system. In [11], the tracking results are used in an autostereoscopic display system, which can render the stereo video for the left and right eyes of a user according to their positions.

Some existing pointing systems have been developed by detecting the laser point on a projection plane [12-17]. The approaches are based on 2D plane projection to establish spatial relationships among image plane, projection plane, and display plane. While a laser point appears on the projection plane, the systems will first find the location of the laser point and then transform it into the display plane. By detecting the laser dot directly, these systems typically perform in high accuracy. However, such systems are not applicable to applications wherein the projection planes are not visible in all views, or do not exist at all. For example, it may not be used to operate electric appliances in a digital smart home environment even if the 3D positions of the electric appliances are known in advance.

In some other pointing systems, human hands are exploited to give instruction via associated direction vectors. For example, the connected line from the finger root to the fingertip is recognized as a pointing direction in [18], while the pointing direction is connected from head to hand in [19]. Similarly, one eye and one fingertip are consider to form a direction vector in [20], while similar vector is established by connecting a line from shoulder to arm in [21]. Instead of using skin color to detect pointing direction of a human hand, as in [19, 20], motion analysis of feature points of user's hand is adopted to estimate the shoulder point and the direction vector in [22]. In [1], a vision-based method is proposed to find the pointing directions which are extended from head to hand. In addition, artificial neural networks are used to find head orientation to improve the accuracy of pointing results. In general, to locate the pointing position in a 3D environment, some forms of 3D reconstruction need to be carried out to determine the direction vector. In [1, 20, 21, 23], 3D voxels of a pair feature points used in the pointing are calculated before such a vector is formed.

In order to study the accuracy and stability of pointing, a real-time, vision-based system similar to that presented in [19] but with pointing direction specified by a pointer is implemented (see Fig. 1) in this paper. By considering the intersection of planes in the 3D world, the system first calculates two planes each formed by two endpoints of the pointer and the center of one of the two cameras. The intersection of these two planes

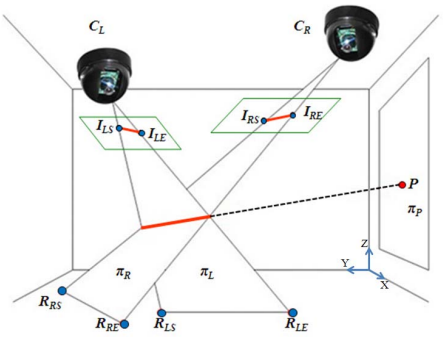


Fig. 1. Configuration of the pointing system and the reconstruction of a pointing point.

then forms the direction vector. Instead of explicitly deriving intrinsic and extrinsic camera parameters, the approach only needs the camera positions, and needs to calibrate the homographies, providing distortion in the camera from perspective projection is fixed.

For all pointing systems, different forms of measurement and computation errors can be generated during the reconstruction of the pointing line, which has five degrees of freedom, and a clear understanding of these errors may greatly improve the applicability of such systems. However, existing error analysis schemes are mainly concerned with planar localization, based on image data acquired by a single camera [24-28], as well as reconstruction of 3D point features using stereo cameras [29-32], which only have two/three degrees of freedom. In this paper, an efficient error analysis scheme is established for an experimental pointing system by evaluating the error range of pointing results on a projection plane, *e.g.*, a screen, with image data assumed to be corrupted by additive noises, as in some of the above approaches. Hopefully, with the help of such analysis, more robust pointing results can be achieved by selecting the most appropriate pointer positions, or pairs of cameras, that will result in minimal range of pointing error. While a pointer with bright color is used here to greatly reduce the influence of certain sources of error, *e.g.*, those due to errors in image feature extraction, the error analysis results will provide an upper bound of pointing accuracy for systems using different pointers, *e.g.*, those discussed in [1, 18-21, 23], if similar reconstruction process is adopted.

The rest of this paper is organized as follows. In Section 2, an experimental pointing system is then given, which is similar to [19] but adopts a simpler camera calibration process to reconstruct a hand-held pointer with bright color[1]. In Section 3, we proposed efficient error analysis methods[2] for such a pointing system so that the error range of the system output can be estimated in real-time. In Section 4, the experimental results show that the proposed approach can find the error range satisfactorily. Thus, users of similar pointing systems can achieve more robust pointing by selecting proper pointer positions, or possibly a special pair of cameras, that will result in minimal range of pointing error. Finally, some concluding remarks are given in Section 5.

---

[1] The calibration of intrinsic and extrinsic parameters can be achieved according to [33]. However, the implemented pointing system only needs the point correspondence between planes. In other words, only homographic transformation matrices are required.

[2] A preliminary approach of error analysis is presented in [34].

2. AN EXPERIMENTAL POINTING SYSTEM

In this section we describe the configuration of a simple experimental pointing system used in this paper which is similar to [19] but using homographic transformations to derive the pointer direction. The system uses two cameras mounted on the ceiling, four reference points on the floor, and a projection plane perpendicular to the ground (see Fig. 1).[3] A two-fold simplification is associated with such a pointing system. First, unlike in [18], [20] which use color and brightness to find hand region, we use a pointer with bright color to reduce the complexity in feature extraction. Second, unlike in [19] and [21], the simple camera calibration similar to that used in [18] is adopted for 3D reconstruction based on homographic transformations. With such a simplified system configuration, the errors generated during the reconstruction process can be studied more easily and understood more clearly.

In the proposed approach, the left and right images are acquired simultaneously from the two cameras. For each of the stereo images, the image pixels of the pointer are obtained through a preprocessing step (see Appendix), and we calculate a best-fit line of these pixels via principal components analysis (PCA). The line intersects the bounding box of the above image pixels at two points, which are then regarded as (extended) endpoints of the pointer in the image. In this paper, the two sets of pointer endpoints are denoted as $\{I_{LS}, I_{LE}\}$ and $\{I_{RS}, I_{RE}\}$ for the left and right images, respectively.

Once the positions of the above endpoints are located in the left and right images, we use homographic transformation to find their projections, R_{LS}, R_{LE}, R_{RS}, and R_{RE} on the ground plane, as shown in Fig. 1.[4] Thus, plane π_L which contains R_{LS}, R_{LE}, and the center of the left camera C_L, and plane π_R which contains R_{RS}, R_{RE}, and C_R can be reconstructed. Planes π_L, π_R, and the projection plane π_P will then intersect at the pointing position P. Finally, we transform P into the 2D coordinate of the monitor screen through another homographic transformation, and display the reconstructed pointing position (RPP).

With the above simple reconstruction process (see Appendix), there is no need to find all camera parameters, as required in typical 3D reconstruction approaches, and the pointing system can operate efficiently in real-time. However, noises in the imaging process may result in reconstruction errors and thus unstable pointing position. To understand the influence of such undesirable effects, and hopefully to develop a scheme to reduce the influence accordingly, an efficient error analysis approach to the estimation of pointing errors is proposed and presented next.

3. ERROR ANALYSIS

For the real world implementation of the pointing system described above, the RPP

---

[3] The coordinates (in cm) of the two cameras C_L and C_R are (192, 365, 264) and (493, 122, 264), and the coordinate of the four corners of the projection plane are (115, 0, 243), (115, 0, 108), (295, 0, 108), and (295, 0, 243). In general, the pointing system can be used to identify a non-planar object at various locations in the 3D space. The projection plane is included here for demonstration purpose only.

[4] The corresponding transformations, H_L (for $I_{LX} \rightarrow R_{LX}$, $X = S, E$) and H_R, are found in advance by using positions of four reference points marked on the floor (not shown in Fig. 1), and their positions in the stereo images.

and actual pointing position are not always the same. Such discrepancies can be categorized into (i) static and (ii) dynamic errors. Static errors such as digitization, lens distortion and measurement errors are almost unavoidable. For example, when we determine the positions of four reference points on the ground and image planes, for calculating the transformation matrix between the two planes, computation or measurement errors may occur. Such errors can be corrected by an additional homographic transformation, and may even be unnoticeable to a user in reality because of the simultaneous self-adjusting ability resulted from the visual feedback during the pointing operation. However, dynamic errors may cause obvious jitters in RPP, which are usually unacceptable. Thus, the error analysis discussed in this section will focus on (ii).

There are several sources of the dynamic errors, and a major one is due to noises associated with image acquisition. For example, pixels of the pointer region are identified in each of the stereo images before the PCA is performed; however, size and shape of the region may change with time because of illumination changes and influences of noise from the camera sensors[5]. In the following, some error analysis methods will be developed to investigate the influence of dynamic errors on the RPPs of the proposed systems. The goal is to correctly and efficiently identify the range of error in the position of RPP.

For the pointing system shown in Fig. 1, π_P, C_L and C_R are fixed in position; therefore, RPP is decided by the reconstructed planes π_L and π_R, and in turn decided by pointer endpoints I_{LS}, I_{LE}, I_{RS} and I_{RE}. The process of the extraction of these points from stereo images is often influenced by the imaging noises mentioned above. As a result, the obtained pointer endpoints are not stable, so is the calculated RPP. Thus, the deviation of the RPPs due to the variations of I_{LS}, I_{LE}, I_{RS} and I_{RE} will be the main focus of this section.

For a preliminary examination of the above deviation, simulated noises of unit magnitude are added to these pointer endpoints. In particular, 24 simulated points placed evenly (with 15° spacing) along "noise" circles with radius of 1 pixel are generated for I_{LS} = (188,158), I_{LE} = (247,189), I_{RS} = (159,142), and I_{RE} = (226,155) in Fig. 1, as shown in Fig. 2 (see Fig. 3 for close-up views of these circles). In each run of the simulation, four points, each selected from one of the above four circles, are selected as endpoints of the pointer in the stereo images to reconstruct a RPP using aforementioned homographic transformations. Fig. 4 (a) shows all 24^4 RPPs (in red), with the convex hull of them (the range of reconstruction errors) shown in Fig. 4 (b), computed from the 24 × 4 simulated points shown in Fig. 3.

In general, it is desirable to have such a range calculated more efficiently, *e.g.*, with less simulated endpoints of the pointer. However, a direct reduction in the data size may underestimate range of reconstruction errors. For example, the blue region in Fig. 5 is obtained by using only 4 points (with 90º spacing) from each noise circle shown in Fig. 3.

From some close examinations of the relationship between the above reconstruct-

[5] Influences from more complex situations, *e.g.*, when the pointer's color is close to the background, are not considered in this paper since highly dynamic segmentation errors of the pointer due to pointer-background interaction may be so large that the error analysis of the RPPs will make no sense. (Similarly, extraction of reference points in the system calibration stage is also assumed to be free of such complex situations.) In general, more involved segmentation schemes will be needed to resolve such a problem, which is out of the scope of this paper. One way of resolving such a problem is to employ special hardware in the system setup, *e.g.*, attaching blinking LEDs [35] to the pointer.

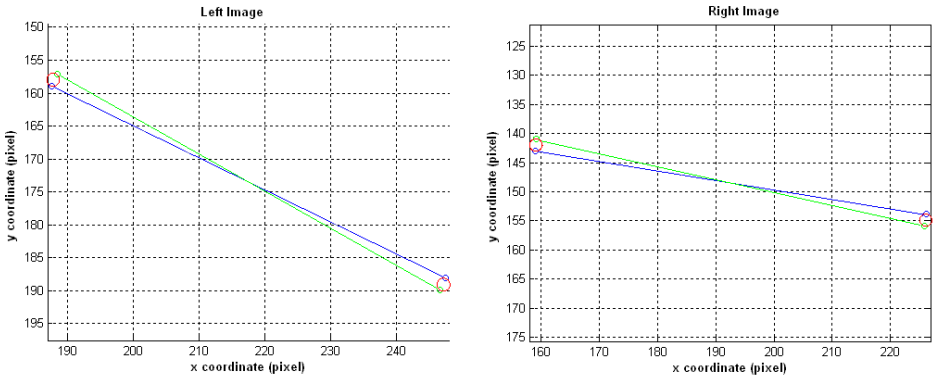


Fig. 2. Noise circles (simulated points) for the pointer endpoints located in stereo images shown in Fig. 1, and their CICTs (see text).

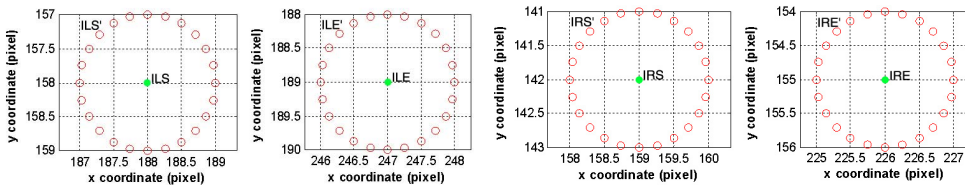
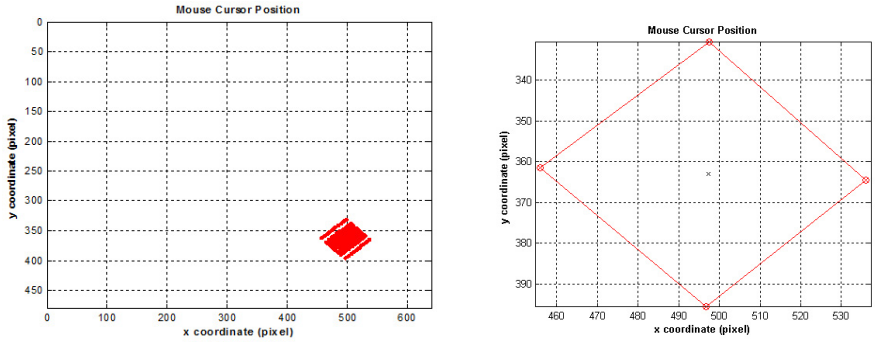


Fig. 3. Close-up views of the four groups (noise circles) of simulated points shown in Fig. 2.



(a) (b)

Fig. 4. (a) RPPs for simulated points shown in Fig. 3; (b) Range of reconstruction errors (with error-free reconstruction show by an "x").

tion errors and the locations of the four simulated endpoints of the pointer obtained from Fig. 3, it is found that the error range is mainly due to (two) extreme values in the slopes of $\overline{I_{LS}'I_{LE}'}$ (and $\overline{I_{RS}'I_{RE}'}$). Based on such an observation, we then try to use only the contacts of the internal common tangents (CICTs) of the two noise circles in each of the stereo images (see Fig. 2 for such tangents). The range of reconstruction error thus obtained is also shown in Fig. 5 (as four points connected by black line segments). One can see that such results almost coincide with that obtained using all (24) points from each

noise circle of simulated points shown in Fig. 3. A closer examination can be carried out by comparing the coordinates of the vertices shown in Fig. 5, as listed in Table 1. Thus, estimation of the error range from a larger number of the simulated points (24 × 4) can be replaced by using only the 8 (2 × 4) CICTs with negligible change in the estimation, and with the number of reconstructed RPPs reduced greatly (from 24^4 to 2^4).

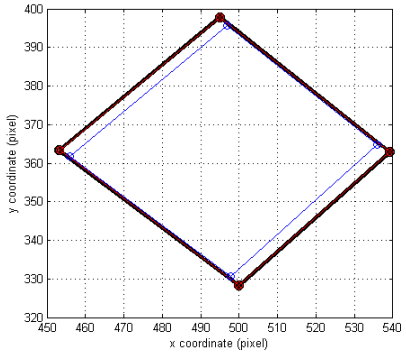


Fig. 5. Error range shown in Fig. 4 (b) (red), similar range but obtained by using only 4 points (with 90° spacing) from each noise circle in Fig. 3 (blue), and error range based on internal common tangents (black, see text).

Table 1. Coordinates of the vertices shown in Fig. 5.

	x_{max}	x_{min}	y_{max}	y_{min}
(blue)	535.9857	455.9600	395.6483	330.5393
(red)	539.2251	453.4022	397.8248	328.1907
(black)	539.2422	453.2395	397.8823	328.1027

The above observations regarding CICTs of two noise circles, *i.e.*, a RPP of a pointer from stereo images will be displaced much more when the pointer is rotated than if it's translated with comparable amount of movements of its endpoints, can be explained with a simple example, as discussed in the following. Consider a pointing system with geometric configuration similar to that shown in Fig. 1, and assume the pointer is initially perpendicular to the projection plane. When the pointer is translated by k in a direction parallel to the projection plane, the RPP will be translated by k too. However, if we fix the endpoint of the pointer which is far away from the projection plane as the center of rotation and rotate the pointer by θ degrees such that the other end of the pointer is displaced by $k = \theta r$, with r being the length of the pointer, the RPP will have a displacement of $k' > \theta d$ with d being the distance from the pointer to the projection plane. One can see that if $d >> r$, which is often the case in various pointing situations, the amount of movement of RPP with a rotated pointer is much larger than that due to a translated pointer, or $k' >> k$. Such an example reasonably explains why the estimated maximal error range (EMER) efficiently obtained using CICTs can represent the real error range with high accuracy, as the CICTs give the limits of the rotation angle of the pointer, with its end points confined to two noise circles in each of the stereo images.

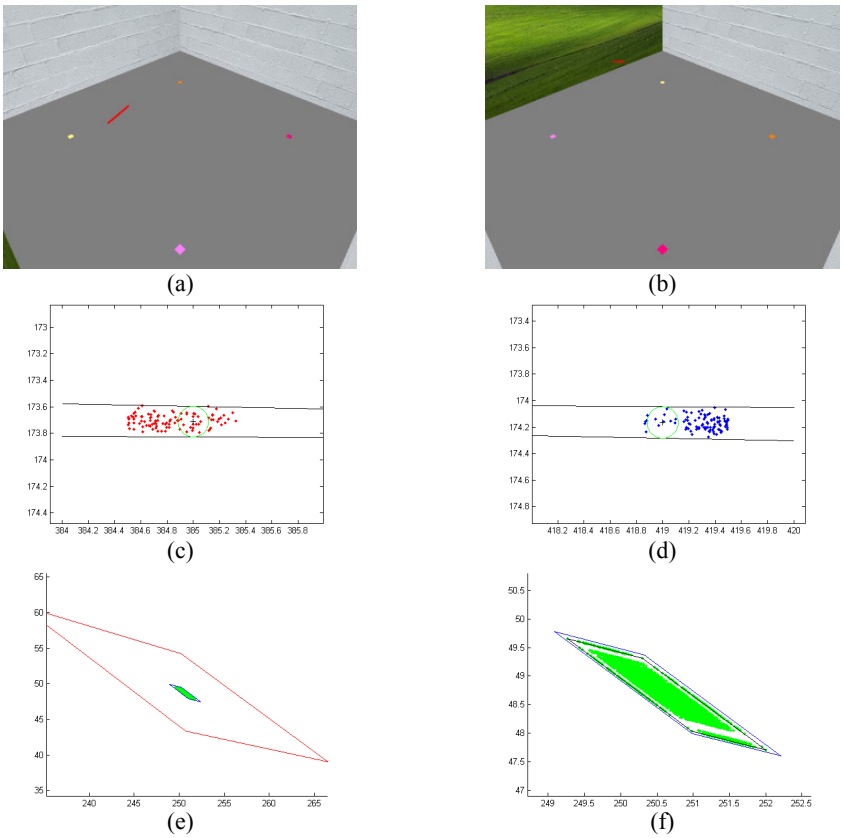


Fig. 6. (a) Image captured by C_3 when the pointer is pointing toward P_4; (b) Image captured by C_1 when the pointer is pointing toward P_4; (c) Simulated sample points of the left end point shown in (b); (d) Simulated sample points of the right end point shown in (b); (e) EMERs obtained with radii in (c) and (d) (in blue) and with unit radius (in red); (f) A tighter EMER (see text).

The use of unit circle for noise is only to provide a baseline for error estimation, which can in fact be adapted for specific applications. For pointing systems based on the estimation of two ends of an elongated pointer, the idea of CICTs can be generalized easily and applied to the spatial supports, regardless of their shapes[6], of the error distributions of the two points to estimate the EMER of the pointing position. Such supports can be obtained for a static pointer in each view by observing its two ends for some time.

In fact, the same EMER can also be obtained from CICTs of virtual noise circles centered at these end points by assigning suitable radii to the circles. For example, assume a synthesized imaging noise with a uniform distribution in [−0.5, 0.5] is added to each foreground pixel of the pointer shown in Figs. 6 (a) and (b). For Fig. 6 (b), the image locations of the left and right ends, estimated with the proposed approach for a pe-

[6] For example, error distributions can often be described by elliptical Gaussian blobs.

riod of 100 frames, are shown in Figs. 6 (c) and (d), respectively. By choosing radii of 0.1279 and 0.1079 for noise circles in Figs. 6 (c) and (d), respectively, it is easy to see that all estimated end points of the pointer are well bounded by the internal common tangents (two nearly parallel lines). Similar results can also be obtained for the (longer) pointer image shown in Fig. 6 (a) with (smaller) radii equal to 0.0796 and 0.0747, respectively.

Fig. 6 (e) shows the EMER (in blue) obtained from the above CICTs, which actually provide a much tighter boundary of EMER than that obtained using a unit radius (in red). On the other hand, by considering the fact that samples of two ends of the pointer shown in Figs. 6 (c) and (d) are actually generated in pairs in each figure from PCA computation, instead of independently estimated, the resultant EMER (as shown in black in Fig. 6 (f) for a close-up view of 10000 RPPs generated for the presumably stationary pointer) can be reduced by 24%[7]. While a loose bound of EMER, which can be obtained easily with minimal computation cost, is useful for the identification of inappropriate camera pairs to be avoided, a tighter bound of EMER is crucial for the spatial arrangement of interactive content on the screen, *e.g.*, the minimum size of an icon to be pointed to.

4. EXPERIMENTAL RESULTS

In order to clearly verify the validity of the EMERs with respect to actual error distributions, we focus on the static pointing situation in the experiments, *i.e.*, we fix the pointer in space and measure the locus of RPPs. Thus, additional sources interferences, *e.g.*, due to multi-camera synchronization and/or motion blur of a moving pointer, can be avoided. The error analysis results obtained here can be applied in the future to situations involving highly dynamic pointing situations if these interferences can be well controlled or even eliminated, *e.g.*, via better imaging hardwares. We will first examine the proposed error estimation method by placing the pointer at different positions, and pointing to different positions on the projection plane, for both real and synthesized scenes. Then, pointing results obtained by selecting of a pair of cameras for each RPP according to the EMERs are compared with those obtained by using all cameras.

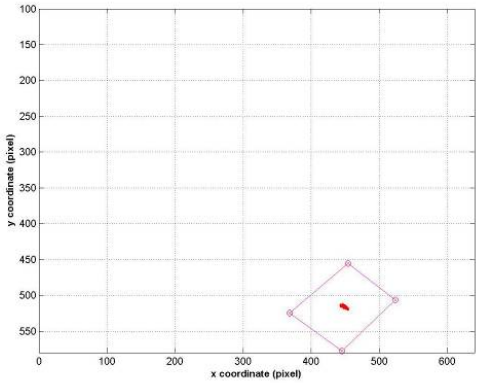
Figs. 7 (a) and (b) show an orange stick which is fixed in the workspace and is used in the experiment as a pointer. In Fig. 7 (c), the purple quadrilateral shows the EMER obtained for simulated 1-pixel error in point feature extraction, while the red dots are the actual positions of RPPs found by the pointing system during a period of 30 seconds. One can see that the latter is well bounded by the former. Figs. 8 (a) to (c) show results similar to those depicted in Fig. 7, but with a different pointer location. One can see that in this case, the actual errors and estimated errors have a nice match in their distributions which are spatially highly directional. In particular, the locations of the RPPs are now distributed in a fairly narrow region, with its elongated direction well predicted by the EMER.

To further investigate the relationship between EMERs and different pointing posi tions, and with respect to different camera pairs, a synthesized room of size 500cm by

[7] Instead of considering estimation error of the two ends of the pointer, it may be possible to use the error estimate of the line direction directly, which is currently under investigation.

(a)

(b)

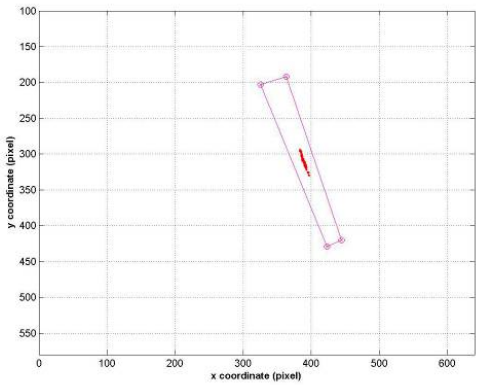


(c)

Fig. 7. (a) Left image; (b) Right image; (c) EMER and actual RPPs.

(a)

(b)



(c)

Fig. 8. (a) Left image; (b) Right image; (c) EMER and actual RPPs.

500cm is built. Fig. 9 (a) shows the top view of the layout of the room. Cameras C_1, C_2, C_3 and C_4, marked as crosses, are mounted on a ceiling of height 250cm while pointing toward the center (250, 0, 250) of the room. The red line represents the pointer and the green line corresponds to the projection plane, on which a user will point to nine fixed pointing positions P_1, P_2, …, P_9 as shown in Fig. 9 (b). The resultant EMERs computed for different camera pairs are shown in Fig. 10. Because of the left-right symmetry of camera configuration with respect to the pointer (which is located 100cm above ground level), highly symmetrical patterns of EMERs can be observed.

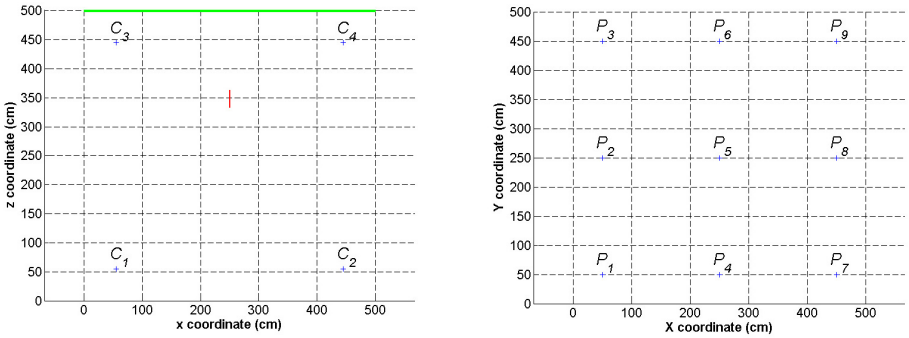
The above EMERs can serve as good references for a user to select camera pairs that will achieve the highest stability in the pointing process. Table 2 shows such suggestion of camera pairs for each of the nine pointing positions[8], which correspond to the minimum areas of EMER. On the other hand, huge EMERs in Fig. 10 also indicate inappropriate camera pairs *e.g.*, C_1&C_3 in Fig. 10 (c) and C_2&C_4 in Fig. 10 (d), that may result in highly unstable pointing and should be avoided. One of such EMERs of C_1&C_3 occurs while the pointer is pointed toward P_2. The problem is due to the very short pointer extracted in one of the pair of images (see Figs. 11 (a) and (b)), which is highly sensitive to image noise and may cause huge reconstruction errors. Similar problem occurs when the pointer is pointed toward P_8 (see Figs. 11 (c) and (d)). Note that some of EMERs shown in Fig. 10 are highly directional. Thus, suggestions other than those listed in Table 2 are possible if requirements of pointing accuracy for a particular application are not isotropic.

Fig. 12 shows similar experiment results obtained by moving the pointer left 150cm. The EMERs shown in Fig. 12 are not symmetrical due to the lack of symmetry in the geometry of system configuration. However, the huge EMER shown in Fig. 12 (a), which does not correspond to a very short pointer in the image, as shown in Fig. 13, it is due to the fact that the reconstructed planes π_R and π_L are almost parallel to each other.

As for a more rigorous and quantified evaluation, a large number of pointer positions are considered in a virtual room of size 1000cm by 500cm, as shown in Fig. 14. The camera positions are from the top view as triangles with screen on the upper side and a ceiling height of 250cm. The pointer, which is located 100cm above ground level, is placed at 200 different positions. Figs. 14 (a) and (b) show best camera pairs[9] selected for different pointer positions such that the EMERs for pointing toward P_5 obtained by using (i) unit radius and (ii) separately estimated radii as in Fig. 6, respectively, are minimized. One can see the two sets of the selections are quite similar with the latter based on tighter bounds of EMER than the former. On the other hand, Fig. 14 (c) shows a set of camera pairs which will result in maximum EMERs, based on estimations using (ii), for different pointer positions and should be avoided. It is readily observable that none of such pairs are recommended in Fig. 14 (a) by using (i). Similar results are pre-

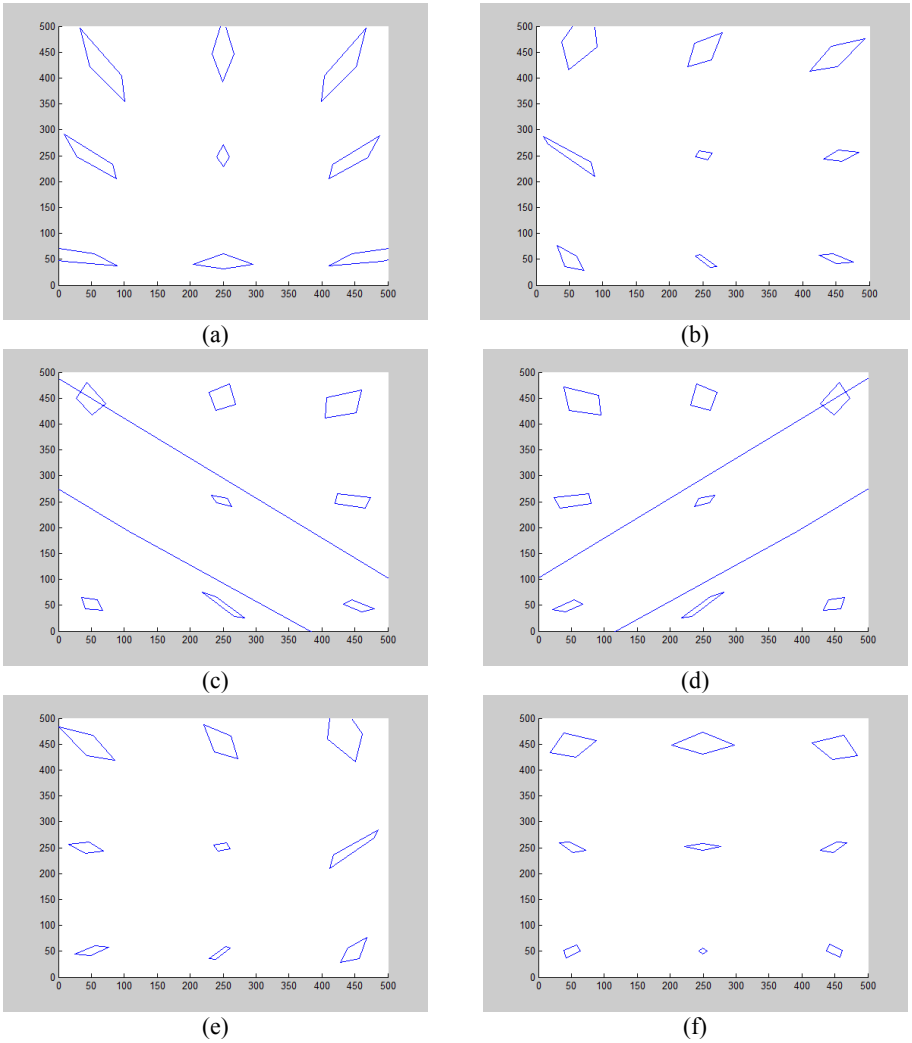
[8] These positions are mainly used to show that if arbitrary camera pairs are adopted for different locations on the projection plane, the resultant RPPs may be too unstable to be useful. For other locations, the trend of RPP stability may be estimated via interpolation, which is omitted for brevity. On the other hand, since the proposed CICT-based error analysis is extremely efficient, the EMER, as well as the preferred camera pair, may be estimated on the run, as the pointer ends are extracted, for arbitrary RPP and user (and pointer) locations. The above arguments can also be applied to the next set of experiments which use selected (200) pointer locations to show that using unit circles is as good as using more precise (often smaller) circles to simulate noises in terms of helping the user to avoid camera pair(s) of worst stability performance.

[9] Camera pairs C_1&C_2, C_1&C_3, C_1&C_4, C_2&C_3, C_2&C_4, and C_3&C_4 are represented by red, green, blue, cyan, magenta, and yellow colors, respectively.



(a) (b)

Fig. 9. (a) Layout of the synthesized room; (b) Pointing positions on the projection plane.

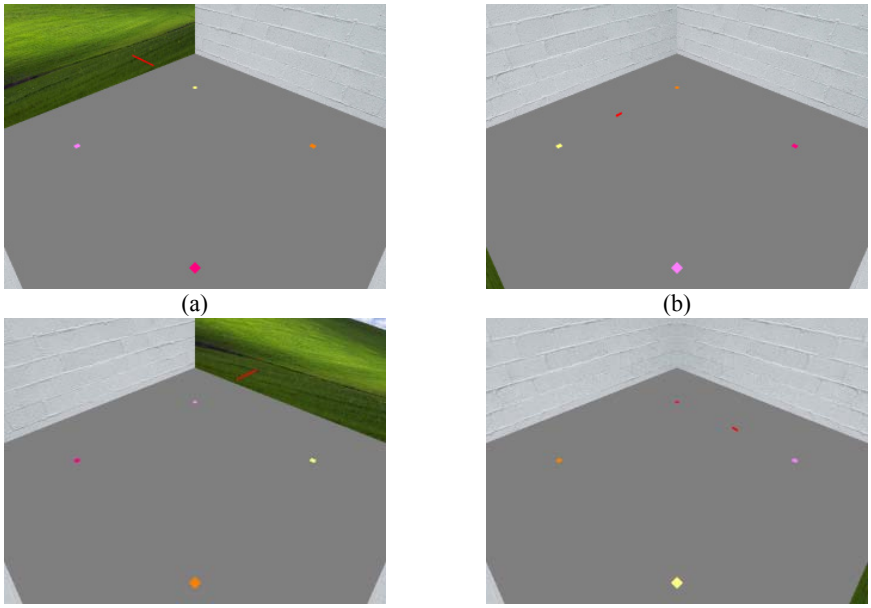


(a) (b)

(c) (d)

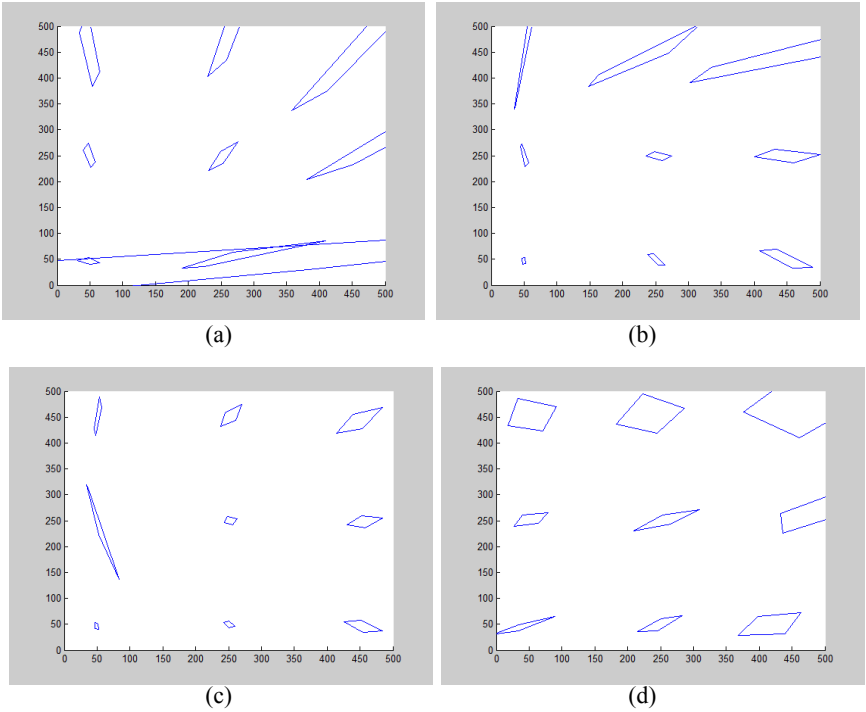
(e) (f)

Fig. 10. Estimated maximal error ranges for different camera pairs: (a) C_1&C_2; (b) C_2&C_3; (c) C_1&C_3; (d) C_2&C_4; (e) C_1&C_4; (f) C_3&C_4.



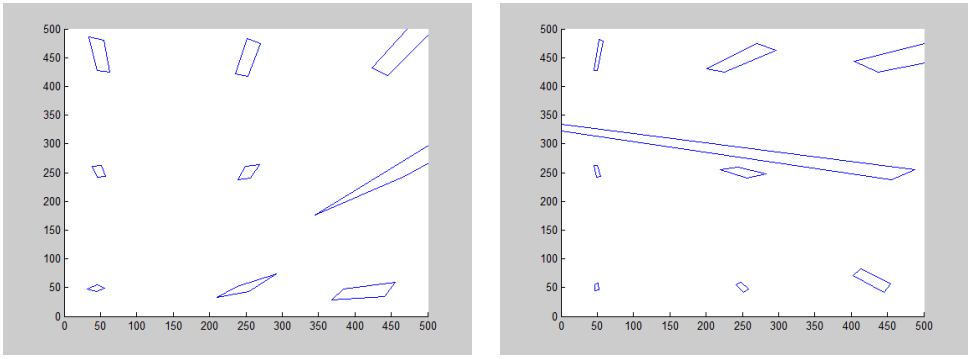
(a) (b) (c) (d)

Fig. 11. (a) Image captured by C_1 when the pointer is pointing toward P_2; (b) Image captured by C_3 when the pointer is pointing toward P_2; (c) Image captured by C_2 when the pointer is pointing toward P_8; (d) Image captured by C_4 when the pointer is pointing toward P_8.



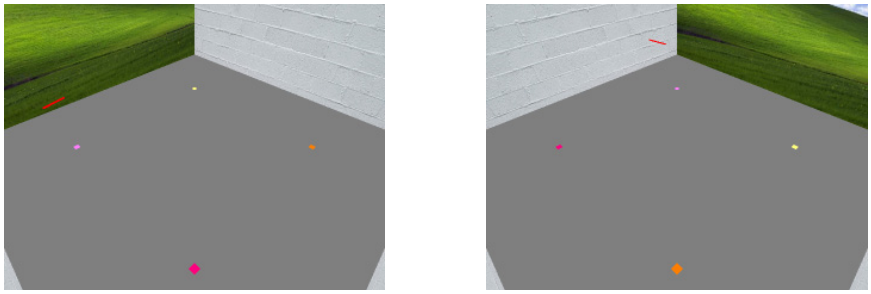
(a) (b) (c) (d)

Fig. 12. Estimated maximal error ranges for the pointer moved left 150cm for different camera pairs: (a) C_1&C_2; (b) C_2&C_3; (c) C_1&C_3; (d) C_2&C_4; (e) C_1&C_4; (f) C_3&C_4.



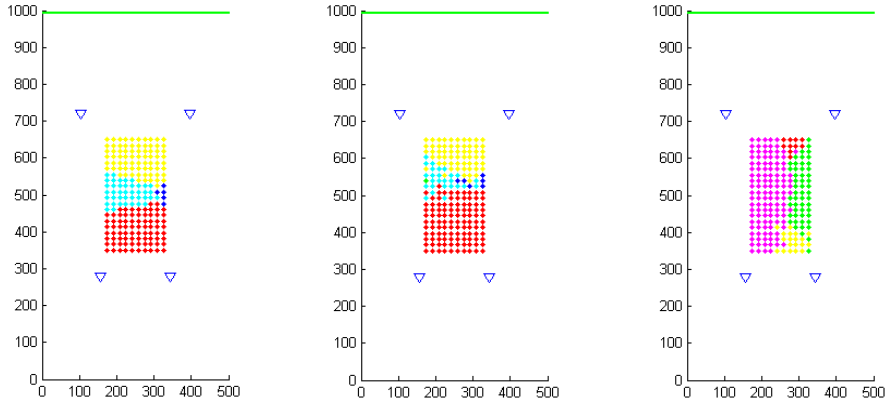
(e) (f)

Fig. 12. (Cont'd) Estimated maximal error ranges for the pointer moved left 150cm for different camera pairs: (a) C_1&C_2; (b) C_2&C_3; (c) C_1&C_3; (d) C_2&C_4; (e) C_1&C_4; (f) C_3&C_4.



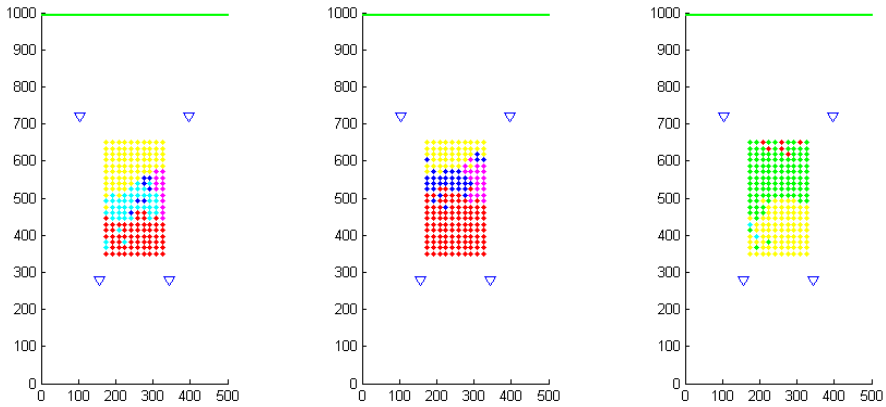
(a) (b)

Fig. 13. (a) Image captured by C_1 when the pointer is pointing toward P_7; (b) Image captured by C_2 when the pointer is pointing toward P_7.



(a) (b) (c)

Fig. 14. Effects of using (i) simple (unit radius noise circle) and (ii) more accurately estimated error distribution on camera pair selection at 200 pointer locations, for pointing toward P_5; (a) Best pairs suggested by using (i); (b) best pairs suggested by using (ii); (c) worst pairs identified by using (ii).



(a) (b) (c)

Fig. 15. Results similar to Fig. 14 but with pointing target changed to P_3.

sented in Fig. 15 for pointing to P_3. These results suggest that simpler estimation obtained by using (i) may perform reasonably well in terms of selecting camera pairs to work with so that unstable pointing results can be avoided.

As for the pointing accuracy, the mean values (and standard deviations) of the pointing errors of the 200 selected camera pairs shown in Figs. 14 (a) and (b) are 1.5450cm (1.0163cm) and 1.5130cm (1.0100cm), respectively. Similar quantities for camera pairs shown in Figs. 15 (a) and (b) are 2.7534cm (2.2294cm) and 2.2892cm (1.5744cm), respectively. Such results suggest that the estimation of using (i) can achieve comparable accuracy as that with (ii).

A Comparison to Using All Cameras

While the goal of the proposed error analysis is to identify one of camera pairs that will result in best pointing performance in terms of pointing stability, the underlying assumption is that when highly unstable RPPs are reconstructed with data obtained from using all cameras, the problem can be alleviated by not using inappropriate cameras (or camera pairs) if possible. For example, if more than two cameras are used for the pointing system shown in Fig. 1, the proposed approach will choose two cameras to find the RPP, while a least square solution of RPP can be found by using all cameras, as in [19].[10] To verify the above assumption, additional experiments are conducted for the simulation environment described in Fig. 9, with additive noises similar to that described in Section 3 for Fig. 6, for a period of 100 frames.

Table 3 shows pointing errors generated by (i) the proposed method which selects a camera pair for each pointing position (from P_1 to P_9) according to Table 2[11] and (ii) the least square approach discussed in [19] which uses all cameras.[12] One can see that simi-

[10] For [19], the pointing direction is defined by the hand-head line, and the RPP is obtained as the least square solution of the intersection of projections of this line on the projection plane from all cameras. If there are only two cameras, as shown in Fig. 1, the two approaches will generate identical RPP.

[11] For P_5 (P_6), C_2&C_3 (C_2&C_4) are selected.

[12] To ensure a fair comparison, the two end points of the pointer adopted in our system for error analysis are used to define the pointing line in each camera view for both (i) and (ii).

Table 2. Suggestion of camera pairs for different pointing positions.

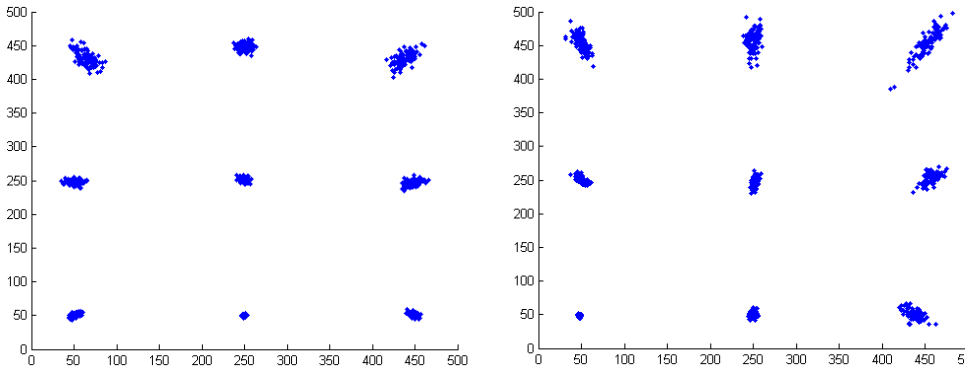
Pointing positions	Camera pairs for smallest error range
P_1	C_3&C_4
P_2	C_3&C_4
P_3	C_1&C_3
P_4	C_3&C_4
P_5	C_1&C_4 or C_2&C_3
P_6	C_1&C_3 or C_2&C_4
P_7	C_3&C_4
P_8	C_3&C_4
P_9	C_2&C_4

Table 3. Pointing errors of the two methods for the pointer placed at (250, 100, 350).

Pointing position	Mean error (standard deviation) cm	
	Our method	[19]
P_1	4.04(2.04)	4.16(2.32)
P_2	5.96(3.78)	6.35(3.66)
P_3	13.35(8.14)	24.48(11.73)
P_4	1.67(0.92)	1.58(0.87)
P_5	5.09(2.91)	4.33(2.21)
P_6	7.64(4.02)	7.10(3.50)
P_7	4.14(2.36)	3.79(2.31)
P_8	6.70(4.23)	7.40(4.09)
P_9	12.90(8.44)	24.74(10.09)

lar pointing accuracy (within 0.76cm) can be achieved by both (i) and (ii) for all pointing positions, except for P_3 and P_9 which correspond to the largest pointing error on the average for both methods. Intuitively, one would expect that most unstable pointing results will be generated for these two points, as shown in Fig. 16 (a), since they correspond to the smallest angle between the pointer and the projection plane. Note that up to 47% reduction (from 24.74cm to 12.90cm) in mean pointing error can be achieved with the proposed camera selection scheme for these worst case scenarios.

Additional observations can be made for more general system configurations wherein the pointer is moved left by 150cm from that specified above, as shown in Table 4. Unlike the nearly symmetric pattern shown in Fig. 16 (a), the corresponding distributions of the RPPs shown in Fig. 16 (b) are not symmetric since the camera locations are no longer symmetric with respect to the pointer position. Again, more than 40% reduction (from 19.01cm to 10.87cm) in mean pointing error can be achieved for the worst case situation with the proposed approach compared with the least square one. The above results suggest that the camera selection scheme based on the efficient error analysis proposed in this paper can indeed help the pointing accuracy and stability.



(a) (b)

Fig. 16. Distribution of RPPs of the nine pointing positions for the pointer placed at (a) (250, 100, 350) and (b) (100, 100, 350).

(a) (b)

Fig. 17. (a) An input image; (b) The detected pointer and its bounding box.

5. CONCLUSION

In this paper, a simple and real-time pointing system is implemented so that the pointing error can be examined closely. A pointer with bright color is used in the pointing process to reduce the complexity in extracting its direction in an image, and error ranges in the pointing position are estimated by synthetic image noises. To greatly increase the efficiency of the estimation, a fast analysis method is developed which only utilizes an extremely limited subset of noise data. With the help of such analysis, suitable operation positions may be suggested to a user of similar pointing systems if the pointer can be used in different locations in a 3D workspace. Moreover, in a multi-camera environment, the overall pointing operation can achieve smallest error ranges, and most stable pointing results, by automatically selecting a pair of cameras based on the proposed error analysis scheme. While experiments are conducted and studied in this paper for static pointing situations, the proposed approach is applicable to more dynamic situations, *e.g.*, in applications wherein instructions are given via various trajectories of pointing positions.

6. APPENDIX

6.1 The Preprocessing Step

The objective of this step is to extract the region of the pointer, analyze its orientation, and locate its two endpoints in an image. The pixels belonging to the region can be found by measuring similarities of the specified color distributions[13] which are obtained in advance. The measurement is achieved by thresholding in HSI color space to find out the pointer while avoiding the interference of the light changes. The pointer detection result of Fig. 17 (a) is shown in Fig. 17 (b). One can see that the pixels of the pointer do connect to each other and occupy a sufficient and elongated area. According to such observations, the connected component labeling is used to identify connected regions, and the region which has largest elongated area is selected as the region of the pointer. After that, principal components analysis is used to find its two axes. Assume a connected region which has n points is represented as $X = [x_1, x_2, \ldots, x_n]^T$. The mean value of the connected region is represented as $m = (\sum_{i=1}^{n} x_i)/n$. The covariance matrix S can be calculated by $\sum_{i=1}^{n}(X-m)(X-m)^T$. Next, the eigenvalues and eigenvectors can be found by eigen decomposition. The eigenvector corresponding to the largest eigenvalue can then be used to calculate a best fit line passing through the pointer. Finally, the two intersection points of (i) this line and (ii) the bounding box of the connected region will be defined as the two endpoints of the pointer.

6.2 Reconstruction of Pointing Points by Homographic Transformations

In order to find the pointing positions, 3D coordinates of R_{LS}, R_{LE}, R_{RS}, and R_{RE} are needed. These coordinates can be calculated from the above endpoints in the stereo images by using 3 × 3 homographic matrices, namely H_L and H_R, which can provide transformations of homogeneous coordinates between the image planes and the ground plane shown in Fig. 1. For example, given $I_{LS} = [u, v]^T$ and $I_{LE} = [u', v']^T$, we can obtain the 2D coordinates of $R_{LS} = [x, y]^T$ and $R_{LE} = [x', y']^T$ on the ground plane as

$$[x, y, 1]^T = H_L [u, v, 1]^T \tag{1}$$

and

$$[x', y', 1]^T = H_L [u', v', 1]^T, \tag{2}$$

respectively. Similarly, R_{RS} and R_{RE} can be found by H_R.

Next, we need to find the 3D plane equations of π_L and π_R, with the former being determined by C_L, R_{LS}, and R_{LE}, and the latter being determined by C_R, R_{RS}, and R_{RE}. Let π_L, π_R and π_P, be represented by equations

[13] The color distributions of a pointer are measured under several light sources. In our experiments, the measured color distributions are H: 340º~20º, S: 0.5~0.9 and I: 0.35~0.7. In addition, in order to obtain a complete pointer region without many holes, we release the threshold as H: 300º~40º, S: 0.2~1.0 and I: 0.3~1.0. In general, if the color is not changed suddenly and can be correctly detected by the assigned color distributions at an initial stage, the color distributions can be updated and utilized continuously.

$$\alpha_L X + \beta_L Y + \gamma_L Z = \eta_L, \tag{3}$$

$$\alpha_R X + \beta_R Y + \gamma_R Z = \eta_R, \tag{4}$$

and

$$\alpha_P X + \beta_P Y + \gamma_P Z = \eta_P, \tag{5}$$

respectively, the pointing point P can then be calculated as

$$P = \begin{bmatrix} \alpha_L & \beta_L & \gamma_L \\ \alpha_R & \beta_R & \gamma_R \\ \alpha_P & \beta_P & \gamma_P \end{bmatrix}^{-1} \begin{bmatrix} \eta_L \\ \eta_R \\ \eta_P \end{bmatrix}. \tag{6}$$

Table 4. Pointing errors of the two methods for the pointer placed at (100, 100, 350).

Pointing position	Mean error (standard deviation) cm	
	Our method	[19]
P_1	2.12(1.11)	3.11(1.14)
P_2	4.28(2.80)	4.84(3.04)
P_3	9.43(6.82)	11.39(7.28)
P_4	2.67(1.82)	4.11(2.05)
P_5	3.40(1.71)	6.63(3.38)
P_6	5.99(3.04)	14.40(8.64)
P_7	7.02(5.10)	14.86(6.97)
P_8	7.23(4.75)	11.51(5.46)
P_9	10.87(5.07)	19.01(13.72)

REFERENCES

1. K. Nickel and R. Stiefelhagen, "Visual recognition of pointing gestures for human-robot interaction," *Image and Vision Computing*, Vol. 25, 2007, pp. 1875-1884.
2. M. A. Moni and A. B. M. S. Ali, "HMM based hand gesture recognition: a review on techniques and approaches," in *Proceedings of IEEE International Conference on Computer Science and Information Technology*, 2009, pp. 433-437.
3. Y. Sato, M. Saito, and H. Koike, "Real-time input of 3D pose and gestures of a user's hand and its applications for HCI," in *Proceedings of IEEE Virtual Reality*, 2001, pp. 79-86.
4. E. Stergiopoulou and N. Papamarkos, "Hand gesture recognition using a neural network shape fitting technique," *Engineering Applications of Artificial Intelligence*, Vol. 22, 2009, pp. 1141-1158.
5. W. Jinwen, Q. Hequn, G. Junjie, and Y. Chen, "The hand shape recognition of human computer interaction with artificial neural network," in *Proceedings of IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurements Systems*, 2009, pp. 350-354.

6. E. Stergiopoulou and N. Papamarkos, "A new technique for hand gesture recognition," in *Proceedings of IEEE International Conference on Image Processing*, 2006, pp. 2657-2660.
7. B. Stenger, "Template-based hand pose recognition using multiple cues," in *Proceedings of the 7th Asian Conference on Computer Vision*, 2006, pp. 551-560.
8. Y.-T. Chen and K.-T. Tseng, "Multiple-angle hand gesture recognition by fusing SVM classifiers," in *Proceedings of IEEE International Conference on Automation Science and Engineering*, 2007, pp. 527-530.
9. Y. Nakanishi, T. Fujii, K. Kiatjima, Y. Sato, and H. Koike, "Vision-based face tracking system for large displays," in *Proceedings of the 4th International Conference on Ubiquitous Computing*, 2002, pp. 152-159.
10. S. Pastoor, J. Liu, and S. Renault, "An experimental multimedia system allowing 3D visualization and eye-controlled interaction without user-worn devices," *IEEE Transactions on Multimedia*, Vol. 1, 1999, pp. 41-52.
11. K. Talmi and J. Liu, "Eye and gaze tracking for visually controlled interactive stereoscopic displays," *Signal Processing-Image Communication*, Vol. 14, 1999, pp. 799-810.
12. R. Sukthankar, R. Stockton, and M. Mullin, "Smarter presentations: exploiting homography in camera-projector systems," in *Proceedings of the International Conference on Computer Vision*, 2001, pp. 247-253.
13. J.-F. Lapointe and G. Godin, "On-screen laser spot detection for large display interaction," in *Proceedings of IEEE International Workshop on Haptic Audio Visual Environments and their Applications*, 2005, pp. 72-76.
14. D. Laberge and J.-F. Lapointe, "An auto-calibrated laser-pointing interface for collaborative environments," in *Proceedings of International Conference on Virtual Systems and Multimedia*, 2003, pp. 501-508.
15. D. R. O. Jr. and T. Nielsen, "Laser pointer interaction," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 2001, pp. 17-22.
16. F. Vogt, J. Wong, S. S. Fels, and D. Cavens, "Tracking multiple laser pointers for large screen interaction," *Extended Abstracts of ACM UIST*, 2003, pp. 95-96.
17. S.-P. Li and T. Kvan, "Enhancing interaction in architectural presentations with laser pointers," *International Journal of Architectural Computing*, Vol. 3, 2005, pp. 503-517.
18. S. Sato and S. Sakane, "Interactive hand pointer that projects a mark in the real work space," *Transactions of the Institute of Electrical Engineers of Japan*, Vol. 121-C, 2001, pp. 1464-1470.
19. C. Colombo, A. D. Bimbo, and A. Valli, "Visual capture and understanding of hand pointing actions in a 3-D environment," *IEEE Transactions on Systems*, *Man*, *and Cybernetics*, Part B, Vol. 33, 2003, pp. 677-686.
20. Y.-P. Hung, Y.-S. Yang, Y.-S. Chen, I.-B. Hsieh, and C.-S. Fuh, "Free-hand pointer by use of an active stereo vision system," in *Proceedings of International Conference on Pattern Recognition*, 1998, pp. 1244-1246.
21. E. Hosoya, H. Sato, M. Kitabata, H. Nojima, and A. Onozawa, "Arm-pointer: 3D pointing interface for real-world interaction," in *Proceedings of the European Conference on Computer Vision Workshop on Human Computer Interaction*, 2004, pp. 72-82.

22. P. Matikainen, P. Pillai, L. Mummert, R. Sukthankar, and M. Hebert, "Prop-free pointing detection in dynamic cluttered environments," in *Proceedings of International Conference on Automatic Face and Gesture Recognition*, 2011, pp. 374-381.
23. Y. Guan and M. Zheng, "Real-time 3D pointing gesture recognition for natural HCI," in *Proceedings of World Congress on Intelligent Control and Automation*, 2008, pp. 2433-2436.
24. J.-H Chuang, J.-H. Kao, H.-H. Lin, and Y.-Ting Chiu, "Practical error analysis of cross-ratio-based planar localization," in *Proceedings of Pacific Rim Symposium on Image Video and Technology*, 2007, pp. 727-736.
25. N. X. Dao, B. J. You, and S. R. Oh, "Visual navigation for indoor mobile robots using a single camera," in *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, 2005, pp. 1992-1997.
26. J.-S. Liu and J.-H. Chuang, "A geometry-based error estimation for cross-ratios," *Pattern Recognition*, Vol. 35, 2002, pp. 155-167.
27. E. Krotkov, "Mobile robot localization using a single image," in *Proceedings of International Conference on Robotics and Automation*, 1989, pp. 978-983.
28. S. Se, D. Lowe, and J. Little, "Local and global localization for mobile robots using visual landmarks," in *Proceedings of International Conference on Intelligent Robots and Systems*, 2001, pp. 414-420.
29. S. D. Blostein and T. S. Huang, "Error analysis in stereo determination of 3-D point positions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9, 1987, pp. 752-765.
30. L. Matthies and S. Shafer, "Error modeling in stereo navigation," *IEEE Journal of Robotic and Automation*, Vol. 3, 1987, pp. 239-248.
31. J. N. Sanders-Reed, "Error propagation in two-sensor 3D position estimation," *Optical Engineering*, Vol. 40, 2001, pp. 627-636.
32. N. Georis, M. Petrou, and J. Kittler, "Error guided design of a 3D vision system," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, 1998, pp. 366-379.
33. Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, 1997, pp. 1330-1334.
34. Y.-T. Tsai, K.-H. Lo, H.-L. Huang, and J.-H. Chuang, "Error analysis of a real-time vision-based pointing system," in *Proceedings of International Computer Symposium*, Vol. 2, 2008, pp. 202-206.
35. R. Eshel and Y. Moses, "Tracking in a dense crowd using multiple cameras," *International Journal of Computer Vision*, Vol. 88, 2010, pp. 129-143.

Kuo-Hua Lo (羅國華) is currently a Ph.D. student in Department of Computer Science of National Chiao Tung University. He received the B.S. degree in computer science and engineering from Tatung University in 2004, and the M.S. degree in computer science and information engineering from National Dong Hwa University in 2006. His research interests include image processing, pattern recognition, computer vision and computer graphics.

Jen-Hui Chuang (莊仁輝) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1980, the M.S. degree in Electrical and Computer Engineering from the University of California, Santa Barbara, in 1983, and the Ph.D. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign, Urbana, in 1991.

Since 1991, he has been on the faculty of the Department of Computer Science at National Chiao Tung University, Hsinchu, Taiwan, where he is currently a Professor. His research interests include robotics, computer vision, 3-D modeling, and image processing.

Hua-Tsung Chen (陳華總) received the B.S. M.S. and Ph.D. degrees in Computer Science from National Chiao Tung University, Hsinchu, Taiwan in 2001, 2003 and 2009, respectively.

He is currently an Assistant Research Fellow with Information and Communications Technology Lab, National Chiao Tung University, Hsinchu, Taiwan. His research interests include computer vision, video signal processing, content-based video indexing and retrieval, multimedia information system and music signal processing.