MirrorTrack - Tracking with Reflection - Comparison with Top-Down Approach

Yannick Verdie Bing Fang Francis Quek
Center for Human Computer Interaction
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24060
{yannick.verdie, fangb, quek}@vt.edu

ABSTRACT

Tabletop hand tracking techniques have evolved much during the last few years from single to multiple cameras, offering users an improved interactive experience. MirrorTrack is one of such techniques. This paper demonstrates the comparison of accuracy between MirrorTrack and top-down approach, which is generally used for table top tasks. In this paper, we focus on the comparison of distance errors in finger trajectory, and clicking errors by manual monitoring.

Categories and Subject Descriptors

I.4 [**Image Processing and Computer Vision**]: Digitization and Image Capture, Applications; H.5.2 [**User Interfaces**]: Input devices and strategies, Interaction styles

General Terms

Design, Measurement, Performance

1. INTRODUCTION

Tabletop or surface interaction has garnered significant interest as a display and interaction configuration. The horizontal surface morphology of such systems are well-suited for touch and multi-touch interaction [12, 1, 5, 4]. Touch interaction requires detection and tracking of surface touch behavior of the human. The technologies for such detection may be generally divided into video-based and non-video-based.We consider video-based solutions more promising because their are less expensive to implement whereas non-camera solutions need more expensive technologies. This paper compares two video-based touch detection and tracking approaches differentiated by the placement of the camera(s), and the concomitant video processing approaches. Our approaches are motivated by the observation that the typical back-projected screen technologies with rear-positioned cameras are incompatible with the obvious display technologies involving flat panels like LCDs and plasma displays that can greatly reduce the cost and increase broader adoption of tabletop interaction systems.

The approaches we discuss in this paper are: 1. A top-down camera placement model where the camera is placed above the screen with the optical axis perpendicular to the display surface [8, 6, 9], and 2. A 'MirrorTrack' approach where multiple cameras are placed close to the display surface with their optical axes making sharply acute angles with the surface plane [3, 2].

ICMI-MLMI'09, November 2–4, 2009, Cambridge, MA, USA.

2. TOP-DOWN APPROACH

An obvious camera placement approach consists in having a camera above the display surface with its optical axis perpendicular to the display surface. In this configuration, the rectangular surface space maps directly into the camera image along with advantages of minimal perspective distortion and a maximization of the useful resolution of the camera. Top-down has advantage of near orthonormal projection with even pixel distribution similar to back projection camera system. Top-down approach was developed in accordance with previous contributions in this domain such as [8, 6, 9].

2.1 Hand segmentation

Hand segmentation aims at extracting the hands region from the frame. We employ a two-step process to realize this. One of the critical challenges to detection of the hand over the glossy display surface is to minimize the effect of the reflection and shadow of the hand in the image. Employing a HSV color space, we notice that H and S are very sensitive to the color content of the hand that is preserved in the reflection. To minimize this effect, at the initial stage of processing, we employ an image differencing approach in V space alone. This approach has advantages to using the commonly-used RGB space because the normalized (r,g) are precisely sensitive to the reflected color [7]. This processing step effectively segments the silhouette of the hand from the background. Once the silhouette is obtained, we reverse the color space process and employ H and S because this is now most sensitive to the skin color. This will remove any shadow boundary that were inadvertently included in the segmented silhouette. This also helps to remove other moving objects (e.g., screen objects) that are not close to the color of the hand [11].

2.2 Fingertips location

Various techniques such as *k-curvature* , *convex hull*, *circle Hough transform*, *shape filtering* or *template matching* [8, 6] have been used to find fingertips in computer vision. Our experiments with the tabletop shows that curvature based approaches can suffer from significant jitter because of the derivative effect in digital images. This effect is less pronounced with the latter two techniques cited. We choose to use a morphological template matching approach using a circle as a structuring element, because it is an efficient region processing algorithm. Moreover, it gives good results for scale invariant targets such a hand over a table top (we observe that in this

projection, finger projections may be approximated as cylinders, and the fingertip as a half-circle). Given the camera configuration, with the standoff-distance far exceeds the depth volume in which the finger resides. Under this condition, the size of the finger in the image is approximately constant.

2.3 Tracking and smoothing

The tracking of each fingertip is done by a polynomial curve fitting over five previous fingertips location. In this way, we also smooth the trajectory of each fingertip to continue to attenuate the jitter.

3. MIRRORTRACK

In the MirrorTrack approach, multiple cameras are placed at low azimuth angles with respect to the surface plane. The rationale for this is that given the specular nature of the display surface, it approximates a perfect mirror from the camera viewpoint in this configuration. Hence both the image of a hand or finger hovering close to the display surface, and its clean reflection can be detected in the video. This has two other key advantages. First, it eliminates much of the effect of the image in the video display. Second, it eliminates the specular reflections of typical overhead lighting common in most office, home, and school environments. Furthermore, given the perspective effects at the low azimuth, it is relatively easy to determine if a stereo ray intersection takes place over the surface or behind it [2, 3].
The MirrorTrack algorithm is composed of three main steps: Stereo Calibration, 2D Processing and 3D Processing.

3.1 Stereo calibration

We use Tsai's calibration model [10] with a set of 48 control points to calibrate the cameras [2]. The measured accuracy of our calibration algorithm on a 640×480 picture is around 0.2in which suffices for our purpose.

3.2 2D Processing - Pyramidal approach

We implement MirrorTrack as a sequential pyramidal process in three levels as shown in Figure 1.

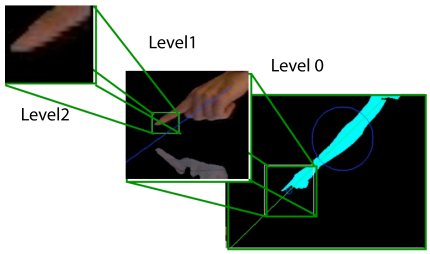

Figure 1: *Pyramidal approach of MirrorTrack algorithm. Level 0: Hand segmentation, Level 1: Hand segmentation and Template Matching, Level 2: Refinement of fingertips location*

3.2.1 Level 0

At this level, we resize our native image by an order of 2. A first rough estimate of our hand location can thus be found. We are using the hand segmentation introduced in section 2 that can segment the hand without its shadow and reflection. Then we approximate the blob as an ellipse to compute its spatial moment as described in [13]. The center of our *region of interest* (ROI) is finally defined as the intersection between the line generated from the major axis of the ellipse and the horizontal line passing through the lowest point of the blob.

3.2.2 Level 1

We take a higher resolution sub-image centered around the ROI computed in Level 0. We empirically resize the sub-image to 160×160 to include the whole hand.
In opposition to the hand segmentation algorithm discussed in section 2, we want to keep the hand and its reflection for MirrorTrack, so we process the background segmentation in *Hue* channel. Indeed, the hue value of the hand and its reflection are very similar and it is easy to classify the hand, its reflection, the background and hand shadows in *HSV* space.
Since we now have the hand and its reflection, we can apply template matching to locate the fingertips. We want to clearly detect when the hand touches its reflection to be able to classify this action correctly. That is why the template used here is an ellipse with horizontal major axis, because this specific ellipse matches the shape formed by the intersection of the hand and its reflection. We then threshold our probability map in order to reveal the fingertips location that are used for Level 2.

3.2.3 Level 2

This step uses the candidate coordinates found in the previous level to re-calculate a new ROI, and reset the coordinate values by looking at fingertips location with an edge operator.

3.2.4 Classification

Classification aims at determining the interactive action (*hover*, *click*, *move*) using the location of the fingertips with the image configuration we detected.

- Fingertip: The fingertip above the surface is detected in the upper half image.
- Reflection: The fingertip 'below' the surface is detected in the lower half image.
- Click: The fingertip and its reflection is detected to be merged together.

As we discuss in section 4, this classification increases the accuracy of the system because more information is available in defining whether or not the finger touches the surface.

3.3 3D Processing

3.3.1 Triangulation

We use the calibration information and the data from the 2D processing to compute 3D locations of the fingertips. Since we have three types of fingertip candidates, if we can combine them with the same classification (see section 3.2.4), we compute the 3D locations with the same type; otherwise, we just calculate all the possible 3D positions without considering the classification. This helps us to reduce the computational cost.

3.3.2 Tracking and smoothness

At this point, we have a set of all the possible 3D locations, and we prune it by looking at the closest candidate with the estimate position of our target. The estimation employs a polynomial curve fitting of 5 previous points to track the target in 3D space. Meanwhile, the estimation helps to smooth the trajectory of the target.

4. EXPERIMENTAL RESULTS

We will show that MirrorTracks accuracy is very close to the accuracy of a top-down tabletop single camera approach while providing hovering and touching information emending the main weakness of a top-down setup. We compare the top-down approach and MirrorTrack approach for distance and click accuracy. Moreover, we also compare the MirrorTrack approach with three setups: 1.

We set the low-azimuth to have best reflection. 2. We set the azimuth a little higher to have better depth resolution, but less reflection. 3. We set a high azimuth to have best depth resolution, but the worst reflection.

4.1 Description of the setup

We tested MirrorTrack with three different angles to study the influence of the angle over the accuracy and the capacity for this algorithm to take advantage of the reflection when available.

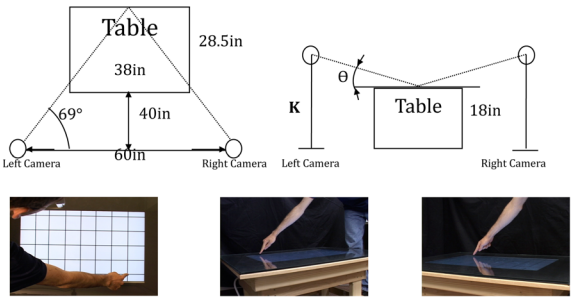

Figure 2: *Camera setup for MirrorTrack algorithm. Upper left: top down view of the setup, Upper right : frontal view of the setup with K varying from 25in to 35in. Lower from left to right: top-down camera, MirrorTrack left camera and MirrorTrack right camera.*

We ran three experiments by varying the height of the camera K to 25in, 30in and 35in, and the setting is shown in Fig. 2. The corresponding angles (θ) are 13°, 22° and 30°.

4.2 Results

In our experiment, we display a grid in the screen embedded in the table. We ask the user to move his finger across all the grid intersections on the display, and capture the motion for top-down and MirrorTrack cameras at the same time. Then, we use the tracking result from top-down camera as ground truth, and calculate the offset errors of the tracking results. Then, we label the real click by observing the original video clips, and use the labeled result as ground truth to compare the 'clicking' correctness for both top-down and MirrorTrack approaches.

4.2.1 Distance error

Before we can compare the distance error between top-down and MirrorTrack approaches, we need to map the both tracking trajectories into the same coordinates. As the resolution of each camera is 640×480, we use the same resolution for mapping. After mapping them into the same coordinate, we calculate the Euclidian distance between them. The following figures show the results of distance error by different settings (Fig. 3 shows the distance error when cameras are set as 22°, Fig. 4 shows the distance error when cameras are set as 13°).

4.2.2 Click error

In order to evaluate the capacity of MirrorTrack to differentiate hovering to clicking, we compare top-down and MirrorTrack to a ground truth manually selected from video. Fig. 5 shows the click detection for the whole captured video, and Fig. 6 shows one of the click for insight detail.

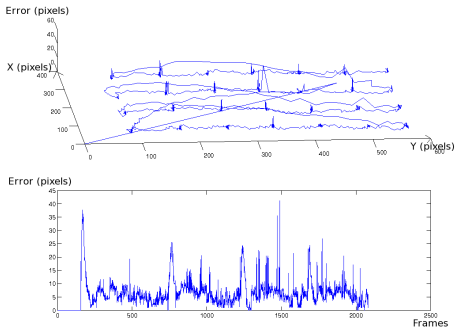

Figure 3: *Distance error when camera angle is equal to* 22°. *The upper figure shows the error by mapping coordinates. The lower one shows the error by frame.*

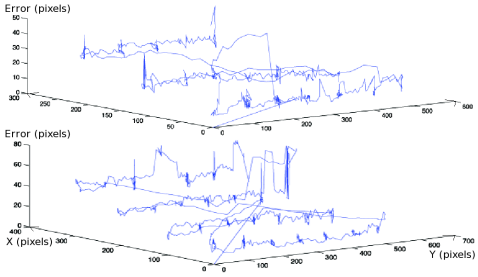

Figure 4: *Left and right figures: Distance error when camera angle is equal to respectively* 13° *and* 30°.

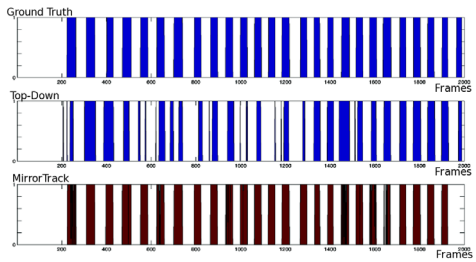

Figure 5: *Click error when camera angle is equal to* 22°. *The y coordinate is equal to 1 when clicking happens, and 0 otherwise.*

5. DISCUSSION

As shown in Fig. 3, the upper figure shows the error (z-axis) by mapping corresponding top-town and MirrorTracking tracking results into the same coordinates (640×480) when cameras' angles are set to be 22°. As we can observe, peaks appears around where clicking happens. Fig. 6 shows a randomly selected click event. From the top two figures, we can observe that the peaks actually happened before and after clicking action. This is due to the polynomial curve fitting we used to smooth the trajectory and estimation. Indeed, based on the history data, the trajectory before the 'click' is predicted as continuing to move down whereas the trajectory predicted when the 'click' is released is static. These artefacts generate those peaks, but do not affect general effectiveness of the system. Moreover, we can observe from Fig. 3 that, the peaks last around 6 frames, which also prove our hypothesis. We also notice that the error is not correlated with the finger location over the table.

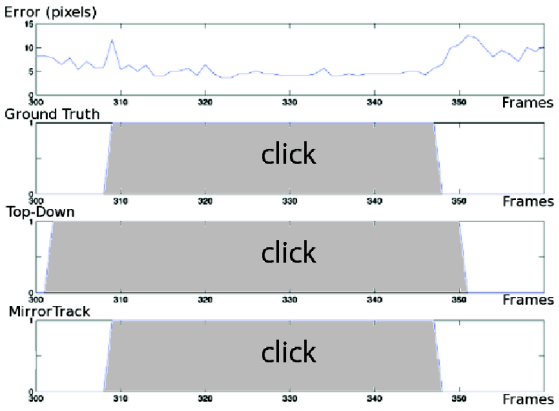

Figure 6: *Click error when camera angle is equal to* 22°. *The y coordinate (except the 1st one) is equal to 1 when clicking happens, and 0 otherwise.*

From Fig. 4, we demonstrate that the peaks happen no matter what angle of camera setting is used. However, we notice that the setting of camera affect the overall distance error. When cameras are set to 13°, we increase the accuracy of detecting finger reflection, but lose the resolution for *x*-axis, thus, we lose accuracy on that axis on the whole. When cameras are set to 30°, we increase the resolution for *x*-axis, but loss the accuracy of detecting finger reflection. Therefore, the general distance error is worse.

We use Fig. 5 to show the 'click error' when cameras are set to 22°. The top figure shows the 'real click' which is selected manually. Fig. 6 shows one of these clicks. From these figures, we can observe that MirrorTrack has a more accurate result for click than top-down approach, and it is close to the 'real' clicking. Even though top-down results may be improved by tunning the predetermined rules to detect a 'click', this approach will still be limited per se because it does not detect the 'real' click. In opposition, MirrorTrack detects 'clicking' by detecting the finger and its reflection to produce more accurate results, which are independent from any predetermined rules.

Finally, the accuracy of our prototype can still be improved for some reasons: we use template matching in 3D space by approximating the movement of the hand to be in a plane. While this assumption is appropriate for the top-down approach, it introduces errors in MirrorTrack implementation because of perspective effects. It could be relevant to use the convex-hull of the hand and its abduction angle of the fingers to detect fingertips because this method is scale invariant. Another solution could be using shape filtering and update the criteria depending on the size of the blob for scale invariance. Moreover, the step described in section 3.2.3 was not fully implemented during the experiment and could significantly improve the fingertips location to allow us estimating better results.

6. CONCLUSION

In this paper, we presented a novel technique for hand devices interaction that allows new action by using hand's reflection. We have shown that the accuracy of our results with MirrorTrack is similar to a top-down setup while the touch action is detected and not modeled by predetermined rules. We also studied the influence of the angle on the system to find a trade-off between capacity to detect reflection and accuracy during the triangulation.

Using MirrorTrack on flat LCD screen for day-to-day usage seems to be a very promising and attractive solution to enhance interaction with devices. It supports hover mode that has been identified as desirable. Future work will be conducted on the usability of the system with various users.

7. REFERENCES

[1] H. Benko, A. D. Wilson, and P. Baudisch. Precise selection techniques for multi-touch screens. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1263–1272, New York, NY, USA, 2006. ACM.

[2] P.-K. Chung, B. Fang, R. W. Ehrich, and F. Quek. Mirrortrack. volume 0, pages 1–5, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[3] P.-K. Chung, B. Fang, and F. Quek. Mirrortrack - a vision based multi-touch system for glossy display surfaces. pages 571–576, 29 2008-Aug. 1 2008.

[4] P. Dietz and D. Leigh. Diamondtouch: a multi-user touch technology. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 219–226, New York, NY, USA, 2001. ACM.

[5] J. Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 115–118, New York, NY, USA, 2005. ACM.

[6] J. Letessier and F. Bérard. Visual tracking of bare fingers for interactive surfaces. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 119–122, New York, NY, USA, 2004. ACM.

[7] J. Letessier and F. Bérard. Visual tracking of bare fingers for interactive surfaces. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 119–122, New York, NY USA, 2004. ACM.

[8] S. Malik and J. Laszlo. Visual touchpad: A two-handed gestural input device. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 289–296, New York, NY, USA, 2004. ACM.

[9] A. Sanghi, H. Arora, K. Gupta, and V. B. Vats. A fingertip detection and tracking system as a virtual mouse, a signature input device and an application selector. In *Devices, Circuits and Systems, 2008. ICCDCS 2008. 7th International Caribbean Conference on*, pages 1–4, 2008.

[10] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. pages 221–244, 1992.

[11] V. V. Vassili, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques. In *in Proc. Graphicon-2003*, pages 85–92, 2003.

[12] A. D. Wilson. Touchlight: an imaging touch screen and display for gesture-based interaction. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 69–76, New York, NY, USA, 2004. ACM.

[13] E. Yoruk, E. Konukoglu, B. Sankur, and J. Darbon. Shape-based hand recognition. *Image Processing, IEEE Transactions on*, 15(7):1803–1815, July 2006.