Evaluation of Tracking Methods for Human-Computer Interaction *

Christopher Fagiani, [1] Margrit Betke, [2] and James Gips [1]
[1] Computer Science Department, Boston College, MA 02467, USA
[2] Computer Science Department, Boston University, MA 02215, USA

Abstract

Tracking methods are evaluated in a real-time feature tracking system used for human-computer interaction (HCI). The Camera Mouse, a HCI system for people with severe disabilities that interprets video input to manipulate the mouse pointer [1], was improved and used as the test platform for this study. Tracking methods tested are the Lucas-Kanade tracker [6] and a tracker based on normalized correlation [1]. Both methods are evaluated with and without multidimensional Kalman filters. Two-, four-, and six-dimensional filters are tested to model feature location, velocity, and acceleration. The various tracker and filter combinations are evaluated for accuracy, computational efficiency, and practicality. The normalized correlation coefficient tracker without Kalman filtering is found to be the tracker best suited for a variety of HCI tasks.

1 Introduction

Human-computer interaction is currently based on the traditional interfaces, keyboard and mouse. Since many personal computers now come with cameras as standard equipment, it is desirable to employ them for next-generation human-computer interaction devices. By relying on visual rather than tactile input, users are free to use their hands for other tasks while interacting with their computers, enabling software designers to develop new models of user interaction.

Camera-based interfaces are particularly important for people who cannot use the keyboard or mouse due to severe disabilities. The traditional human-computer interfaces require good manual dexterity and refined motor control. People with severe disabilities, such as cerebral palsy, may be quadriplegic, spastic, and nonverbal, and therefore unable to use traditional human-computer interfaces.

The goal of our research project is to refine tracking methods commonly used by camera-based interfaces and provide an accurate and robust system with small overhead, thereby expanding the range of uses to which it may be applied. The task is to increase tracking accuracy so that the amount of user intervention can decrease. Moving human-computer interface technology closer to the point when computers can be operated without the use of one's hands is particularly important for people with severe disabilities.

We improved the Camera Mouse [1], an interface that processes video input to control the mouse pointer, to serve as a testing platform for HCI experiments. Our system contains two components – a tracker and a filter. The tracker either uses Lucas and Kanade's (LK) method [6] or a template correlation method [1]. The tracking methods measure the location of a feature in the current frame based on its location in the previous frame. The trackers are tested with and without various multidimensional Kalman filters [5]. The Kalman filter predicts the location of the feature in the current frame based on the history of previous locations. It has been widely applied to visual tracking, e.g., [7]. Other probabilistic tracking techniques are promising [4, 8, 9], but it is beyond the scope of this paper to test them for practicality in human-computer interaction.

2 Methods

The **correlation tracking** method computes the normalized correlation coefficient for each point in a a 50×50 pixel search area that is centered around the position of the feature in the previous frame [1]. The size of the feature being tracked is 15×15 pixels. The point with the highest correlation is declared to be the new location of the feature. If no point in the search area has a correlation coefficient above a certain threshold, the feature is declared to be "lost" and the system must be re-initialized.

The **Lucas-Kanade tracker** [6] is based upon the principle of optical flow, the apparent motion of image brightness, and assumes a constant brightness of a feature during small movements. Lucas and Kanade solve for the unknown optical flow parameters of a feature by assuming constant brightness for the pixels in a window around the feature, weighing the squared error in this assumption by a Gaussian function for each pixel, and minimizing the sum of the weighted errors.

*Financial support by the National Science Foundation (IIS-0093367, EIA-0202067) and the Office of Naval Research is acknowledged. Email: betke@cs.bu.edu, http://www.cs.bu.edu.

0-7695-1858-3/02 $17.00 © 2002 IEEE

Kalman filters [5] of three different dimensions were evaluated in this system. The 2D filter estimates the feature's position, the 4D filter in addition considers the feature's velocity, and the 6D filter in addition considers acceleration. The filters' noise covariance matrices are approximated using training data [3]. The covariance of the true feature location and the location predicted by the tracking method is computed for a training sequence in which the true feature locations are manually marked. Combining the correlation tracker with a Kalman filter is advantageous, because with the filter estimate of the new feature position, the size of the area searched by the correlation tracker can be reduced.

The system was designed to implement Kalman filters in two ways: with and without alternation. When the system runs with alternation, it uses the Kalman filter to estimate feature position in odd frames and a tracking algorithm on even frames.

Translation of Facial Motion into Mouse Movement. The movement of the feature within the image window is translated to screen coordinates through a function that takes into account the screen aspect ratio and resolution, the size of the image, and the apparent motion of the feature. The x-coordinate of the mouse pointer is $((0.5\,d_x - p_{x,i})m_x)/(0.5\,d_x) + 0.5\,m_x$ if $d_x/4 < p_{x,i} < 3d_x/4$, where d_x is the width of the image in pixels, m_x is the width of the monitor in pixels, and $p_{x,i}$ is the x-coordinate of the predicated feature location in frame i. Features in the exterior quarter of the image are mapped to the edge of the screen. The image is mirrored in the horizontal direction to make it easier for users to coordinate head movement with mouse pointer movement on the screen. A similar function translates vertical feature motion into vertical mouse pointer motion. Our method of translation makes it easy for the user to move the mouse to the edge of the screen while also reducing chances that the tracker will lose sight of the feature due to its movement out of the camera's field of view.

3 Hardware and Implementation

Input was gathered with a Sony EVI-D30 color video CCD camera and a Sony Handycam video camera using a Matrox Meteor II video capture board. Grayscale and color images were processed at a frame rate of 30 frames per second and a size of 320 × 240 pixels. The system used for development and testing was a dual 1GHz Pentium III machine with 256 MB of PC-133 SDRAM. The Intel OpenCV library was used for implementation of the LK tracker and Kalman filter.

All video data was captured under regular overhead fluorescent lighting; no special lights were used in our laboratory. The subjects were positioned approximately two feet from the camera and they remained at this distance for the duration of the experiments.

4 Experiments

The subjects were asked to use various HCI programs designed for people with severe disabilities. Such people often have difficulty moving their head from one point to another in a straight line with constant speed. Since the evaluation of tracking methods needs to account for potential spastic movements, the subjects were asked to make random or erratic movements in some experiments to simulate the types of movements encountered with users with severe disabilities.

Upon start-up, a facial feature is manually selected for tracking. Features are selected in a manner that maximizes the likelihood that the system can differentiate the feature from its surrounding area [2]. Image regions with significant brightness changes or contrasts in color or texture were used because of their large information content. Three features were tested: the corner of an eyebrow, a nostril, and the bridge of the subject's glasses.

4.1 HCI Experiments for Frame-by-frame Evaluation

Three subjects participated in experiments designed to evaluate the performance of the different trackers on a frame-by-frame basis. Three input sequences, each containing about 300 frames, were captured per person. In each sequence, the computed and true locations of the tracked feature in each frame were recorded. A subset of this data was also used to train the error covariance matrix for the Kalman filters. Approximately 10 seconds of input were classified manually.

In the first sequence, the subjects were asked to make random head movements, and in the other two sequences, the subjects used the system to control the Midas Touch spelling application shown in Fig. 1. The subjects were instructed to spell the word "Hello." Users spell words by moving the mouse pointer over a character and then dwelling within a small radius of pixels for a second to generate a mouse click.

The tracking accuracy of the algorithms was evaluated by comparing the distance the mouse pointer should have moved, assuming perfect tracking, and the distance the mouse pointer was actually moved. The mean of the Euclidean distance between these two points over all frames in the input sequence was taken as a single value to quantify tracker performance relative to one another. The mean error E is defined as

$$E = \frac{1}{n}\sum_{i=0}^{n-1} \|P_{d,i} - P_{t,i}\|^2, \qquad (1)$$

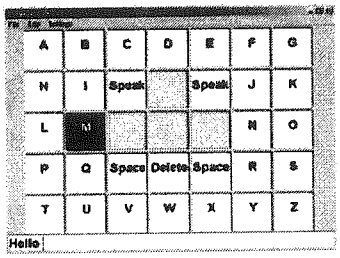

Figure 1: The Midas Touch spelling application.

where n is the number of frames in the image sequence, $P_{d,i}$ is the desired number of pixels the pointer should have moved in the ith frame, and $P_{t,i}$ is the true number of pixels the mouse pointer moved in the ith frame. This method of evaluation, as opposed to measuring the location of the feature, does not penalize the system for errors made at the beginning of the trial more than those made near the end. The error calculation takes only the change in mouse coordinates from the previous frame to the current frame into account, not the absolute error encountered since the start of the tracking. In this sense, the error is non-cumulative. Similarly, as long as the motion being tracked is in the same direction and magnitude as the actual motion, the tracker is evaluated favorably.

Other evaluations of feature trackers have employed an error metric that measures the drift in feature position from the ground-truth feature position, e.g. [10, 8]. Such an evaluation method, while well-suited to systems solely concerned with feature tracking and providing valuable insights about tracker performance, is less relevant to the Camera Mouse application. The Camera Mouse system does not need to track the same feature to operate properly. The movement of the mouse pointer will remain true to the user's wishes as long as the direction and magnitude of feature motion can still be measured. This distinction makes the effects of drift less significant and our evaluation method therefore focuses on the practicality of the system.

4.2 Long Human-Computer Interaction

Four HCI applications were tested in longer experiments – Midas Touch, Eagle Aliens, Speech Staggered, and Eagle Paint (see Figs. 1 and 2). The programs were selected because they differ in the types of movement each requires. Mouse movement while using Eagle Aliens is characterized by sudden changes in direction, as the user is trying to hit aliens that appear in random locations. Erratic, seemingly random movements characterize mouse movement associated with Eagle Paint since the user is able to "paint" by moving the mouse pointer anywhere in the application window. The movement during this application may be slow and deliberate or fast and wild. The two spelling programs, Midas Touch and Speech Staggered, require the user to move the mouse over a desired character or group, respectively, and hold the pointer in the same region for a period of time to generate a mouse click.

Nine minutes of video input was captured in this manner. Evaluation of each tracker on the videotaped data was carried out by hand. Due to the length of the input sequences (over 16,200 frames) the true feature location could not be established manually for every frame. Instead, drift was measured in terms of the degree to which the tracked region deviated from the position to which it was initialized. In addition, the number of times the operator needed to manually reset the tracker for each tracking method on each input sequence was recorded.

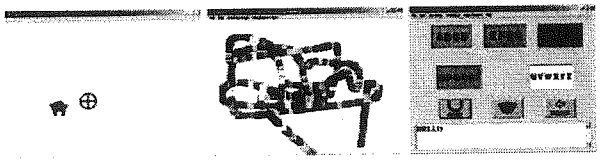

Figure 2: HCI software: Eagle Aliens, Eagle Paint, and Speech Staggered.

5 Results and Discussion

The LK tracker without Kalman filtering and the normalized correlation coefficient tracker performed with similar levels of reliability. Both were accurate enough to allow the user to manipulate the mouse pointer as desired. While drifting occurred, mouse pointer motion was conserved and therefore the drifting had no apparent effect on the overall performance. When the user moved, the region tracked by the system moved approximately the same distance and in the same direction as the true feature (see Fig. 3).

Features were deemed suitable for tracking if they were easily differentiable from their immediate surroundings. This quality may be observed by examining the autocorrelation of the features (see Fig. 4), which should have a pronounced peak [2].

Each of the three facial features exhibits a strong peak in the autocorrelation surface. Though, of the features used, the eyebrow was most easily differentiated from its surroundings, whereas the glasses template was slightly more difficult to differentiate. As shown in Ref. [2], the width of the autocorrelation peak corresponds to the coherence scale of the feature and inversely relates to its information content. Since all features used in this study exhibit similar coherence

Figure 3: Performance of the LK tracker. The lighter (green) square denotes the ideal feature location whereas the darker (red) square denotes the location returned by the tracking algorithm.

scales, the choice of the feature itself was deemed to have a negligible impact on the results. Had the features been more difficult to differentiate from their surroundings, it is likely that the error rates for each of the tracking methods tested would have been greater.

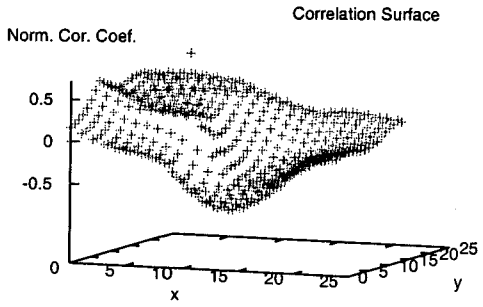

Figure 4: Values of normalized correlation coefficient obtained when trying to match a template consisting of the corner of the subject's eyebrow with itself. The peak of the autocorrelation surface represents the location at which an exact match is obtained.

5.1 Frame-by-frame Evaluation

Tracking Accuracy. The normalized correlation coefficient tracker in combination with 2D-, 4D-, or 6D-Kalman filtering without alternation resulted in the most accurate tracking performance as long as the feature was moving slowly. Sudden movements, in which the feature moved out of the search area, caused significant drift but did not result in a loss of the facial region containing the feature. This is important since a complete loss of the feature would require user intervention to re-initialize the tracker.

Processing Time. The least time-consuming tracker was based on the normalized correlation coefficient with Kalman filtering. The normalized correlation coefficient tracker without Kalman filtering was the most demanding on processing resources, since it must *search* for the feature instead of mostly just validating the feature location, but was still able to operate at approximately 30 frames per second. Since the increase in computing time still allowed for the processing of real-time data, the cost was deemed acceptable for the human-computer interface system. Figure 5 summarizes the processing times of each system variant.

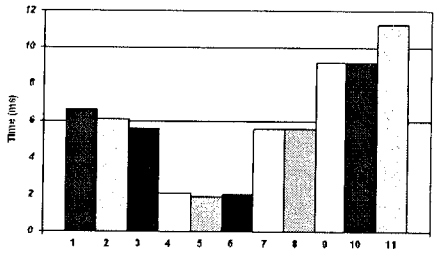

Figure 5: The mean time, in milliseconds, to process a frame of data using a combination of a Kalman filter and tracking algorithm: Kalman filter with correlation and alternation in 2D, 4D, and 6D (1–3), with correlation without alternation in 2D, 4D, and 6D (4–6), with LK-tracking with alternation in 2D, 4D, and 6D (7–9), and the LK (10) and correlation (11) trackers without Kalman filters.

Speed versus Accuracy. In addition to the differing levels of demand placed upon the CPU, each algorithm performed with a differing degree of accuracy. In some cases, as expected, a decrease in computation time was accompanied by a decrease in accuracy, as when applying Kalman filtering with alternation to the normalized correlation coefficient tracker. This, however, was not always the case. Although the normalized correlation coefficient tracker used more time to locate the feature than the LK tracker, it was still deemed to be the best suited tracker for our HCI applications since it produced a smaller mean error. This being said, the tracker with the best balance between time and accuracy was the normalized correlation coefficient tracker without Kalman filtering (see Fig. 6).

Smooth versus Erratic Movements. Mean feature displacement was 110.2 pixels per frame in the

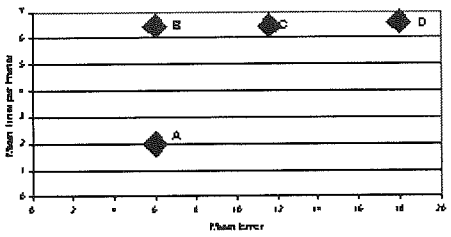

Figure 6: Mean tracking error versus computation time. Point A represents the group of correlation trackers with Kalman filters without alternation. Point B represents the normalized correlation coefficient and LK trackers without Kalman filtering. Point C represents the correlation trackers with Kalman filtering and alternation. Point D represents the LK tracker with Kalman filtering with alternation.

random movement sequences and 55.8 pixels per frame in the spelling application sequences. Although the graph in Fig. 6 shows that the correlation tracker with a 2D Kalman filter was faster with approximately the same mean error, it was not considered to be the best method because of its propensity to drift when the user makes sudden movements. When examined on data in which the subject exhibited random movement, the correlation tracker, when combined with Kalman filtering, performed with a much higher error than the version without Kalman filtering (see Fig. 7).

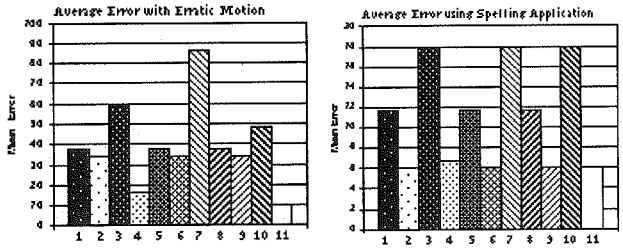

Figure 7: Average error for input with random movement (left) and Midas Touch application (right) for 300 frames. The bars represent the tracking methods as listed in Fig. 5. The correlation tracker without Kalman filtering (11) is strong in both experiments.

Results for Spelling Application. In the Midas Touch application, the correlation tracker performed with the highest level of accuracy and the LK tracker was the second best (see Fig. 8). Although some drift was observed, the motion recorded was consistent with the movement of the feature and, therefore, the system performed as desired.

Results for Erratic Movements. The normal-

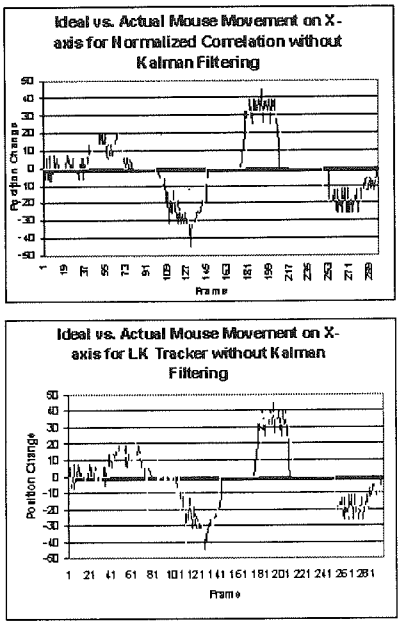

Figure 8: The correlation tracker performed the best among all tested tracker/filter combinations (top). The LK tracker was second best (bottom). The dark (blue) line is the number of pixels the mouse should have been moved in the horizontal direction and the lighter (yellow) line is the amount the mouse pointer was moved. The LK tracker had a more frequent occurrence of large errors, e.g., at frame 110.

ized correlation coefficient tracker performed slightly better than the LK tracker and much better than all the other combinations of tracking and filters (see Fig. 9). The degree to which the correlation tracker outperformed the LK tracker was more pronounced on the random data.

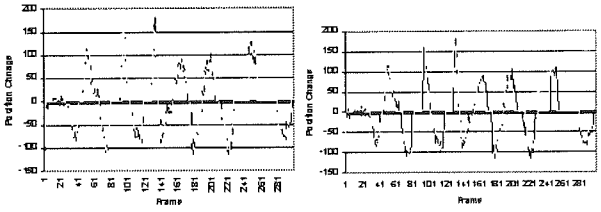

Figure 9: Erratic movement experiments: The performance of the normalized correlation coefficient (top) was slightly better than that of the LK tracker (bottom), both without Kalman filtering.

Kalman Filtering with Alternation. Tracking methods employing Kalman filtering with alternation exhibited a higher level of error on frames in which the

Kalman filter was used to estimate feature position.

Horizontal versus Vertical Motion. All of the trackers were more accurate when measuring horizontal movement than vertical movement. This may be due to lighting changes that occur more rapidly with vertical than with horizontal movement. In addition, moving the head up and down in a nodding movement may cause the tracked feature to become deformed or occluded. In all of the cases examined, the range of movement was much greater in the horizontal direction. The average range for horizontal movement was 247 pixels, which spans 100% of the range of allowable horizontal motion. The average range for vertical movement, however, was 88 pixels, which only corresponds to 48% of the possible movement in the vertical direction.

5.2 Long Human-Computer Interaction

Tracking without Kalman Filtering. Tracker performance for longer HCI sequences was comparable to the performance for the shorter sequences discussed above. Both the LK and the normalized correlation coefficient trackers without Kalman filters performed well. The features were never lost during the trials. The LK tracker exhibited minimal drift. The tracked template drifted from one eyebrow to the other in one of the sequences in which the correlation tracker was employed.

Correlation Tracking with Kalman Filtering. The correlation trackers with Kalman filtering did not lose the feature, but drift was more apparent than for the correlation tracker without Kalman filtering. In the 4D and 6D cases, drift was observed while the subject was using Eagle Aliens. The feature, for example, drifted from one eyebrow to the other and then to the eyelid. This behavior is to be expected since the Eagle Aliens application is characterized by much more rapid movements than the spelling applications. Since the subject may be changing direction of movement suddenly, it is not surprising that the Kalman filter yields poor estimates of feature location, since a Gaussian noise model is assumed.

LK Tracking with Kalman Filtering. When the LK tracker was used in conjunction with Kalman filtering, the tracker lost the feature multiple times. In the 2D and 4D cases, the feature was lost twice per sequence tested and, in the 6D case, the feature was lost an average of nine times on each sequence tested. The tracked region drifted from the eyebrow to the forehead, then off the face and onto the background. In the 4D and 6D cases, tracking was jittery thereby making it difficult for the subject to generate mouse clicks by holding the pointer steady for one second. Drift was encountered while the subject was using both Eagle Paint and Eagle Aliens.

6 Conclusions

The Kalman filters significantly improved computational efficiency of the tracking methods and thereby allowing for third-party applications to have a greater share of CPU time. However, the filters were not suited for some of the human-computer interaction tasks that required erratic motion.

The normalized correlation coefficient tracker without Kalman filtering proved to be the most accurate algorithm for a range of HCI tasks. It was slightly more demanding on the CPU than the second best method, the Lucas-Kanade (LK) tracker. Since processor speeds have been doubling every eighteen months, computational efficiency is here not deemed to be as important as tracking accuracy. With this in mind, the normalized correlation coefficient is recommended as the tracking method best suited for interfaces designed for people with severe disabilities.

References

[1] M. Betke, J. Gips, and P. Fleming. The Camera Mouse: Visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Trans Neural Syst Rehabil Eng*, 10(1):1–10, March 2002.

[2] M. Betke and N. C. Makris. Recognition, resolution and complexity of objects subject to affine transformation. *Int J Comput Vis*, 44(1):5–40, August 2001.

[3] J. Hoffbeck and D. Landgrebe. Covariance matrix estimation and classification with limited training data. *PAMI*, 18(7):763–767, 1996.

[4] M. Isard and A. Blake. CONDENSATION-conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.

[5] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering, Trans. of ASME, Ser. D*, 82(1):35–45, March 1960.

[6] B. Lucas and T. Kanade. An iterative registration technique with an application to stereo vision. In *IJCAI*, pages 674–679, 1981.

[7] N. Oliver, A. P. Pentland, and F. Berard. LAFTER: lips and face real time tracker. In *CVPR*, pages 123–129, 1997.

[8] A. Rahimi, L.-P. Morency, and T. Darrell. Reducing drift in parametric motion tracking. In *CVPR*, pages 315–322, 2001.

[9] C. Rasmussen and G. D. Hager. Probabilistic data association methods for tracking complex visual objects. *PAMI*, 23(6):560–576, 2001.

[10] J. Shi and C. Tomasi. Good features to track. In *CVPR*, pages 593–600, 1994.