Visual Input for Pen-Based Computers

Mario E. Munich *Pietro Perona*
California Institute of Technology 136-93
Pasadena, CA 91125, USA
e-mail: {mariomu,perona}@vision.caltech.edu

Abstract

Handwriting may be captured using a video camera, rather than the customary pressure-sensitive tablet. This paper presents a simple system based on correlation and recursive prediction methods that can track the tip of the pen in real time with sufficient spatio-temporal resolution and accuracy to enable handwritten character recognition. The system is tested on a large and heterogeneous set of examples and its performance is compared to that of three human operators and a commercial high-resolution pressure-sensitive tablet.

Keywords: Systems and applications, Active and real-time vision, Handwriting acquisition.

1 Introduction and Motivation

Computers are getting faster and smaller every day. Notebook and laptop personal computers, pen-based computers and personal organizers, are designed to be as small and portable as possible. While until now their size was limited by hard disk, memory chips, battery and power supplies, the lower bound is now increasingly dependent on the size of the input/output devices. The resolution of the human eye limits the size of the screen, and the dimensions of the fingers fix the minimum size of keyboards and mice. The desire to lower these bounds motivates the search for alternative ways for humans to communicate with computers and the development of new input/output devices such as audio and visual interfaces.

Audio and vision-based interfaces present two significant advantages. First they can be implemented as very small devices with the current VLSI technology. Second, in certain circumstances, they will allow the design of more natural interfaces than keyboards and mice. From this point of view, one of the natural ways of inputting data into the computers is by use of handwriting. So far, there are some devices that interface between handwriting and computers, such as electronic tablets or digitizers for on-line capturing and optical scanners for off-line conversion (see [10]). However, all of them are bulky, increasing the size and complexity of the whole system. This paper will present a visual interface that can be built using video technology and computer vision techniques. Some related work can be found in references [7, 5] and our work can also be integrated in the so-called "Digital Desk" [14, 13] being developed at XEROX PARC Cambridge Laboratory.

Out input system will consist of a camera, a common piece of paper and a common pen (and of course, a computer to do all the calculations). The camera, focused on the sheet of paper, will image the handwriting thereby tracking the trajectory of the pen in order to recognize the handwriting.

Section 2 describes the system, section 3 presents the experimental setup and the results of the different experiments, and section 4 summarizes the results and discusses new efforts and possibilities.

2 Overview of the system

Figure 1 shows a basic block diagram of the system and the experimental setup. The preprocessing stage performs the initialization of the algorithm, i.e., it finds the position of the pen on the first frame of the sequence and selects the template corresponding to the pen tip to be tracked. In subsequent frames, the preprocessing stage has the only function of cutting a piece of image around the predicted position of the pen tip and feeding it into the next block. The pen tip tracker has the task of finding the position of the pen tip on each frame of the sequence. The filter is a Kalman filter that predicts the position of the tip in the next frame based on an estimate of the current position, velocity and acceleration of the pen. Finally, the last block of our system divides the trajectory into segments (strokes) and classifies them as pen-up or pen-down.

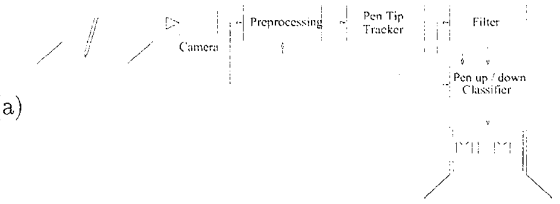
Initialization The first problem to solve is locating the position of the pen tip in the first image of the sequence and selecting the kernel to be tracked. There are three possible scenarios: 1) In a batch analysis the tracker is initialized manually by mouse-clicking on the pen tip in the first frame. 2) The user writes with a pen that is familiar to the system. 3) An unknown pen is used.

The familiar-pen case is easy to handle: the system may use a previously stored template representing the pen tip and detect its position in the image by correlation. There are a number of methods to initialize the system when the pen is unknown, we do not describe them here.

Tracking the pen The second block of the system has the task of finding the position of the pen tip in the current frame of the sequence. The solution of

1015-4651/96 $5.00 © 1996 IEEE
Proceedings of ICPR '96

(a)



(b)

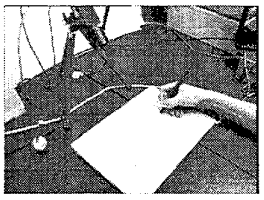


Figure 1. *(a) Block Diagram of the system. The camera feeds a sequence of images to the preprocessing stage. This block initializes the algorithm and selects the kernel to perform the tracking of the pen tip. The tip tracker obtains the position of the pen tip in each image of the sequence. The filter predicts the position of the pen tip in next image. The pen up/down classifier divides the recovered trajectory of the pen into segments and classifies each segment as pen up or pen down. (b) Experimental setup. One subject is writing on the tablet.*

this task is well known in the optimal signal detection literature. The optimal detector is a matched filter to the signal (in our case a part of the image) and the most likely position of the pen is given by the best match between the signal and the optimal detector.

Assuming that the changes in size and orientation of the pen tip during the sequence are small, the most likely position of the pen tip in each frame will be given by the maximum of the correlation between the kernel and the image neighborhood. A pyramidal [2] scheme is used to calculate correlation in a coarse to fine approach in order to be computationally efficient and robust when dealing with large motions. The neighborhood at the lowest level of the pyramid is centered in the predicted position of the centroid. The maximum of correlation is found at each level of the pyramid and then propagated to the next level.

Filtering Using the output of the correlation-based tracker, the filter predicts the position of the pen tip in the next frame based on an estimate of the position, velocity and acceleration of the pen tip in the current frame. This filter improves the performance of the system since it allows us to reduce the size of the neighborhood used to calculate correlation. The measurements will be acquired faster and the measured trajectory will be smoothed by the noise rejection of the filter. A Kalman Filter [6, 1, 4] is a recursive estimation scheme that is suitable for this problem. We assumed a simple random walk model for the acceleration of the pen tip on the image plane. The model is given by(1).

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{x}(t) + \mathbf{v}(t) + \mathbf{n}_x(t) \\ \mathbf{v}(t+1) &= \mathbf{v}(t) + \mathbf{a}(t) + \mathbf{n}_v(t) \\ \mathbf{a}(t+1) &= \mathbf{a}(t) + \mathbf{n}_a(t) \\ \mathbf{y}(t) &= \mathbf{x}(t) + \mathbf{n}_y(t) \end{cases} \quad (1)$$

where $\mathbf{x}(t)$, $\mathbf{v}(t)$ and $\mathbf{a}(t)$ are the two dimensional-components of the position, velocity and acceleration of the tracked point, and $\mathbf{n}_a(t)$, $\mathbf{n}_v(t)$ and $\mathbf{n}_x(t)$ are additive zero-mean, Gaussian, white noises respectively. The state of the filter $\mathbf{X}(t)$ includes three 2-dimensional variables, $\mathbf{x}(t)$, $\mathbf{v}(t)$ and $\mathbf{a}(t)$. This model is a second order model that is appropriate to describe the dynamics of a punctual object moving on a plane. The output of the model $\mathbf{y}(t)$ is the estimated position of the pen tip.

Pen up/down classification The trajectory obtained by the pen tip tracker and the filter is not suitable to perform handwriting recognition since most of the recognition systems up to date assume that their input data is only formed by pen down strokes. The detection of the areas where the pen is lifted and therefore, not writing, is accomplished by using the additional information given by the ink path on the paper.

The segmentation block divides the trajectory into handwriting segments or strokes that contains at most twenty points. The ink-path linker stage associates each segment to the ink on the paper. The linker produces a set of features for each segment that will be used by the classifier to decide if the segment corresponds to a pen up or a pen down case.

Segmentation The trajectory is segmented using two features, the velocity of the pen tip and the curvature of the trajectory. Selection of these two features was inspired by the work of Viviani [11, 12] and Plamondon [9, 8] and by the intuitive idea that on the limit points between two different handwriting strokes the velocity of the pen is very small and/or the curvature of the trajectory is very high. The set of segmentation points is the result of applying a threshold on each of the mentioned features. Figure 2 (a) shows the trajectory obtained for the sequences "Webster"; in (b) the segmentation points are marked with an 'o'; and (c) shows the resulting segments.

Figure 2. *(a) trajectory obtained for the sequence "Webster", (b) segmentation points (marked with 'o'), (c) shows the resulting segments.*

Connection between the segments and the ink on the paper This module associates each segment of the trajectory with the additional information provided by the ink trace on the paper, i.e., this module finds whether there is a part of the ink

trace on the paper in correspondence to each segment. The algorithm can be applied sequentially on the whole sequence of frames or in batch mode to the last frame after the writer has finished writing a word. Due to the complexity of the problem, we work in batch mode as a first approach.

Ideally, for each pen down segment there should be a matching stroke of ink trace on the paper while for each pen up there should none. In reality, it is very difficult and computationally expensive to find an exact match between segments of the trajectory and the ink trace since the noise in the acquisition of the trajectory, the noise in the image, the change in local brightness due to change in illumination, the errors in the segmentation, and the fact that sometimes the pen retraces a previous ink trace, conspire against the efficiency of a simple correlation algorithm used for matching.

The algorithm that we have developed to solve this problem is composed by the following steps:
1.- Divide each segment into subsegments defined by pairs of adjacent points in the segment, e.g. the first subsegment will be formed by the first and second point, the second subsegment will be formed by the second and third point and so on.
2.- Take the first subsegment of a segment and find the perpendicular that passes through the middle point of the subsegment. Sample the image along the perpendicular to obtain the image profile (see figure 3 (a)).

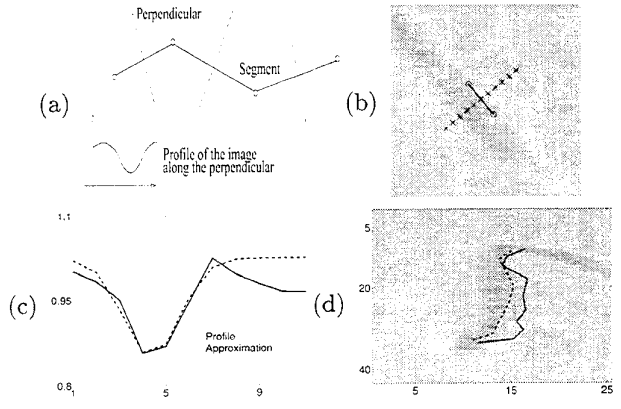


Figure 3. *(a) The profile of the image along the perpendicular in the middle point of each subsegment is used get features that will be used to classify the segment. (b) 'o' indicates the two points that define the subsegment and '*' indicates the sample points where the profile of the image is measured. (c) '-' profile of the image and '- -' approximation by fitting a mixture of Gaussians. (d) The recursive estimation of the position of the ink trace produces a shift of the segment towards the ink trace. '-' initial segment and '- -' shifted segment.*

3.- Fit a mixture of Gaussians to this profile. See figure 3 (c) (We model the ink trace profile with a superposition of Gaussian functions whose mean, sigma and height will give the position, the width and the contrast of the profile).
4.- Reestimate the displacement based upon the position and the contrast of the ink trace. The displacement will be recursively estimated in each iteration, with a gain of the estimation that is proportional to the the contrast of the ink trace, i.e., if the contrast is big, move the points towards the trace, otherwise, remain in the same position (see fig. 3 (d)).
5.- Take the next subsegment and recompute from the second step.
6.- Once a full segment is finished, calculate its corresponding features. They will be:
a) Mean contrast: average of contrast found for each subsegment.
b) Mean fitting error: average of the error in the fitting of the mixture of Gaussians.
c) Mean quality of the subsegment: The quality of the subsegment will be the distance from the subsegment to the ink trace.
d) Integral of brightness along the found ink trace.
e) Mean velocity along the segment.
7.- Compute the above features for each segment.

The computation of features require normalization so that we can use them for classification. We normalized the mean contrast per segment and the integral of brightness along the ink trace with respect to the average brightness on a section of the image that covered each segment. We also normalized the integral of brightness along the ink trace with respect to the number of points per segment.

Classification We use the k-Nearest Neighbors (kNN) algorithm to classify the segments in pen up or pen down. This algorithm is well known in the pattern recognition literature (see [3]). The kNN requires the existence of a database of segments that have already been classified in pen up or pen down. This database will correspond to a cloud of points in $\mathbb{R}^N$, where N is the number of features (in our case $N = 5$). Each segment to be classified will be another point in $\mathbb{R}^N$ and the algorithm consists of taking the distance from this point to all the ones in the database and find the closest k. Then, each of these k nearest neighbors will vote either as a pen down or as a pen up. The segment will be classified based on the result of this voting process, Fig. 6 shows an example of the results of this algorithm on three sequences, using a database of segments previously classified by hand as a reference.

It is clear that the success of the kNN depends on the goodness of the database, i.e. the capability of the database of cover all the possible cases with the proper clusters. We collected our classification database by selecting segments that we considered representatives of pen up and pen down, among segments of several sequences.

3 Experiments

Input Sequences We collected a database of handwriting sequences in order to test the algorithm under general conditions and types of hand writing. The database was formed with 30 sequences of one word, each written with different types of letters

(plain block letters, small letters, cursive letters, Chinese characters and drawings). Half of these sequences were simultaneously taken with an electronic tablet that provides ground truth while the other half was taken with the normal elements of the system, a simple piece of paper and a simple pen. We tried to check the robustness of the algorithm against different pens, lighting conditions and position of the camera. Fig. 1 shows the experimental setup.

Validation of the results A very rough and qualitative estimate of the accuracy of the experimental results can be obtained by just looking at the trajectory resulting from the tracking. However, a quantitative measurement of the accuracy can be obtained by computing the distance between the ground truth and the data given by the algorithm. An electronic tablet was used to register the ground truth. The tablet is a WACOM Digitizer, Active area: 153.6 x 204.8 mm, Resolution: 50 lpmm, Accuracy: $\pm$ 0.25 mm and Maximum report rate: 205 points/second. Also, a simple way of measuring the goodness of our algorithm is to compare its results with the ones produced by a person performing the same task that the algorithm. In this way, we test the information available in the image sequence. The system should ideally achieve a comparable accuracy.

Subjects' results Three subjects were asked to click a pointer over five different sequences of images of the data base (one sequence of each type). They were also asked to click three times over one of these five sequences in order to evaluate their consistency. Fig. 4 presents the results obtained by the three subjects JA, SZ and DB working on the sequence "Alphabetical".

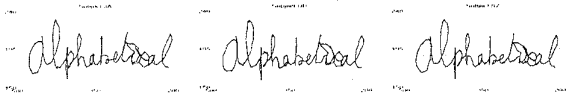


Figure 4. *"Alphabetical" sequence clicked by the three subjects JA, DB and SZ respectively*

The sequences that were tracked by the subjects have the ground truth given by the electronic tablet, therefore, it is possible to compute the accuracy of the human tracking with respect to the ground truth. Table 1 shows the distances (in pixels) between the subjects' results and the curve obtained with the tablet, for five different sequences. We can see that subject JA is the most consistent of the three.

Experimental Results The system was implemented in Matlab working on a Sun Sparc20. We evaluated the performance of the whole system in general and each separated block in particular working on the mentioned database of sequences. The experiments help us to find the best tuning of the system. We tested the algorithm with and without the filter in order to see if the filter provided any benefits. The code that implements the system requires 50.10^6 floating points operations per

Sequence	JA	DB	SZ	Mean
Automatically	1.15	1.05	1.05	1.08
Alphabetical	1.36	1.19	1.07	1.21
China2	1.42	1.11	1.99	1.51
Undergraduate	1.48	1.73	1.36	1.52
StarBoxCirc	1.44	1.46	1.60	1.50

Table 1. *Distances (in pixels) between the subjects' results and the curve obtained with the tablet, for five different sequences. The subjects' errors are in the same order of magnitude .*

cycle (measured with the command "flops" of Matlab). The algorithm has been tested with all the sequences of the data base, Fig 5 displays the results obtained on two of the sequences. The dotted points are the points given by the electronic tablet, after being projected onto the image plane.

Figure 5. *Trajectory recovered by the algorithm and points given by the tablet projected onto the image plane. The output of the system qualitatively agree with the points given by the tablet.*

Comparison Table 2 shows distances from some of the tracked curves to the corresponding path obtained with the electronic tablet and to the corresponding mean path given by the subjects.

Sequence	dist. to subjects	dist. to tablet	KF
Automatically	0.0232	0.478	-
Automatically	0.0072	0.4212	yes
Undergraduate	0.5626	1.356	-
Undergraduate	0.6570	0.4262	yes
Alphabetical	0.1539	1.186	-
Alphabetical	0.1160	0.1623	yes
Star-Box-Circle	0.2652	0.07097	-
Star-Box-Circle	1.0088	0.2995	yes
China2	1.5698	0.0192	-
China2	1.1811	0.0061	yes

Table 2. *Table of distances (in pixels) from the results obtained with and without the filter to subjects' data and tablet data.*

The data showed in table 2 lead to conclude that the system works quite well with equivalent performance to that of a human operator. The algorithm is giving slightly more accurate results than the data given by the subjects with respect to the ground truth (tablet data). The results presented in table 2 and in Fig. 5 show that the filter provide smoothing and noise rejection. Besides, the other important element that the filter brings to the system is the prediction that allows to cut computation by reducing the neighborhood used to do correlation.

Classification Figure 6 shows the results of the

pen up/down classification algorithm working on some of the sequences collected with the tablet. The database of reference segments used by the kNN was formed by 80 segments that we considered representative, chosen among the collection of all segments from all sequences but the sequence to be classified. We use $k = 15$, i.e., we use the 15 nearest neighbors to classify an incoming segment. The results are quite encouraging and we expect to improve them when the system will be implemented in real time.

We sampled the image to get its profile along the perpendicular to the trajectory using 10 points (5 points for each side), this value was empirically determined as the best trade-off between missing the ink trace and catching multiple traces.

In figure 6 we plot the segments with a thickness that was proportional to the confidence of the classification, where the confidence is the percentage of votes for pen down out of the total number of votes. The segments classified as pen down are drawn with solid line while the segments classified as pen up are drawn with dashed line. The measurement of the confidence of the classification of the segments is a better parameter than the result of a binary decision, to be used as input to an algorithm that recognizes handwriting.

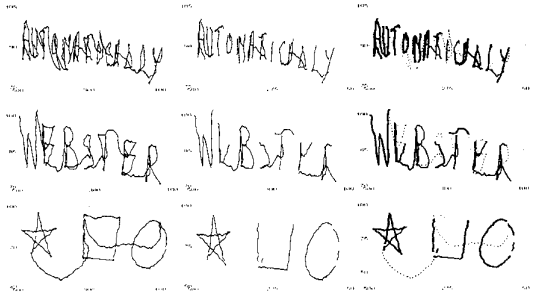


Figure 6. *Results of the classification algorithm on the several sequences of the database using 15-NN. The trajectories obtained with the pen tip tracker are shown in the first column. The segments classified as pen down are shown in the second column. The plots in the third column display segments with a thickness is proportional to the confidence in the classification, where the confidence is the percentage of votes for pen down out of the total number of votes . '-' pen down segments and '- -'pen up segments.*

4 Conclusions and further work

This paper has presented a new way of input data for computers. The handwriting path has been recovered successfully from its spatio-temporal representation given by the sequence of frames. The use of a filter provided two advantages, first, there is some rejection of noise and second, there is an improvement in the speed of the algorithm since the the use of the prediction allow us to reduce the size of the neighborhood where the correlation is calculated. There are some parts of the system that need to be improved such as the performance of the pen up/down classification algorithm.

We are also looking into implementing the system in real time on either a Pentium-based platform and/or VLSI. We believe that the filtering stage can be improved by using a model that describes the dynamics of handwriting and this is one of the areas that we plan to work on. Also, we consider very important to model the statistics of the pen up/down classification problem in order to be able to estimate the variation of the orientation of the pen. This estimation will add robustness to the system since the expected position of the ink trace will be known more accurately. Finally, we would like to include a block in the system that will perform handwriting recognition based upon the output of the pen up/down classifier.

5 Acknowledgments

This work is supported in part by the Center for Neuromorphic Systems Engineering as a part of the National Science Foundation Engineering Research Center Program; and by the California Trade and Commerce Agency, Office of Strategic Technology. We wish to thank all the persons that helped us throughout this work, in especially Jean-Yves Bouguet (who wrote many words for us), Stefano Soatto, and Luis Goncalves for the very useful discussions.

References

[1] R. Bucy. Non-linear filtering theory. *IEEE Trans. A.C. AC-10, 198*, 1965.

[2] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. *IEEE Trans. Commun.*, COM-31:532–540, 1983.

[3] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, Inc., 1973.

[4] A. Jazwinski. *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[5] F. B. J.L. Crowley and J. Coutaz. Finger tracking as an input device for augmented reality. *Int. Work. on Face and Gecture Recog.*, pages 195–200, 1995.

[6] R. Kalman. A new approach to linear filtering and prediction problems. *Trans. of the ASME-Journal of basic engineering.*, 35-45, 1960.

[7] M. Munich and P.Perona. Visual input for pen based computers. *CNS Technical Report CNS-TR-95-01, California Institute of Technology*, 1995.

[8] R. Plamondon and B. Clément. Dependence of peripheral and central parameters describing handwriting generation on movement direction. *Human Movement Science*, 10:193–221, 1991.

[9] R. Plamondon and F. J. Maarse. An evaluation of motor models of handwriting. *IEEE Transaction on systems, man, and cybernetics*, 19(5):1060–1072, 1989.

[10] C. Tappert, C. Suen, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:787–808, 1990.

[11] P. Viviani and G. McCollum. The relation between linear extend and velocity in drawing movements. *Neuroscience*, 10(1):211–218, 1983.

[12] P. Viviani and C. Terzuolo. Trajectory determines movement dynamics. *Neuroscience*, 7(2):431–437, 1982.

[13] P. D. Wellner. Adaptative thresholding for the digitaldesk. *Technical Report EPC-1993-110*, 1993.

[14] P. D. Wellner. Self calibration for digitaldesk. *Technical Report EPC-1993-109*, 1993.