Handwriting Reconstruction For a Camera Pen Using Random Dot Patterns

Matthias Sperber
Osaka Prefecture University
German Research Center for Artificial Intelligence
matthias@m.cs.osakafu-u.ac.jp

Martin Klinkigt
Osaka Prefecture University
klinkigt@m.cs.osakafu-u.ac.jp

Koichi Kise
Osaka Prefecture University
kise@cs.osakafu-u.ac.jp

Masakazu Iwamura
Osaka Prefecture University
masa@cs.osakafu-u.ac.jp

Benjamin Adrian
German Research Center for Artificial Intelligence
Benjamin.Adrian@dfki.de

Andreas Dengel
German Research Center for Artificial Intelligence
Andreas.Dengel@dfki.de

Abstract

This paper proposes a new method of handwriting reconstruction using a camera pen. We print random dot patterns on the document background to enable retrieval of both the current document and the pen position on this document. Dot arrangements are stored in a hash table using Locally Likely Arrangement Hashing. For retrieval, they are extracted from the camera image and matched to the corresponding points in the hash table. We were able to achieve high retrieval accuracy (81.1%∼100.0%), given a sufficient amount of visible dots. Using a two-step homography approximation, an accurate image of handwriting can be reconstructed. By using knowledge about document context and a client-server architecture, our method allows real-time processing on ordinary hardware.

1. Introduction

Despite many efforts to realize the so-called paperless office, today people in fact use an increasing amount of paper for their daily work [6]. This is motivated by various reasons. Paper is independent from computer technology and much more portable. Often, it is considered more convenient for making notes. It allows very flexible creation of content, since arbitrary types of content can be mixed freely. Finally, reading printouts is often less tiring than on-screen reading.

On the other hand, digital files also offer many advantages. They allow very easy storage, organizing, and sharing of documents. Larger amounts of text-based content can be created fast and conveniently, including effective means of editing. Furthermore, digital content permits automatic processing, ranging from simple searching to extraction of semantic information.

Pen computing aims at combining advantages of both, by simultaneously providing paper-based and digital versions of the same content. Several systems are available, including tablet-based, ultrasonic, and Anoto technology. All of these aspire natural usage, but bring along some problems: Some of them provide no absolute position information, or knowledge about the currently used document (*document context*). Others require extra equipment, or come at high costs.

In this paper, we propose a method of using a camera-based pen and randomized dot patterns. The dot patterns are printed on paper in an unobtrusive color such as yellow, and indexed in a hash table using Locally Likely Arrangement Hashing (LLAH) [5]. We determine the pen's current position by analyzing the image provided by a camera mounted on the pen, close to its tip. For this purpose, we extract the yellow dots from the image and use them to query the LLAH database.

This method can be applied at low costs. In the experiments, we achieved a high accuracy (81.1%∼100.0%), using a cheap USB camera and fairly sparse dot patterns. The requirement for good results is that only a small number of dots are occluded by document foreground. Handwriting can be reconstructed precisely by using a two-step homography estimation,

consisting of an application of RANSAC [2], and using obtained matches to calculate an optimal homography. Processing speed is reasonably fast, and allows for real-time application using some techniques we propose. The discussion suggests a solution for how to deal with too much disturbance by document foreground.

2. Related Work

Several commercial products realizing pen computing already exist, including tablets and ultrasonic pens. A number of methods for camera pens have also been proposed. *PaperLink* [1] is a marker equipped with a pen-tip camera. When marking certain parts on a paper, these can be automatically recognized and associated with electronic content. They can furthermore be used as input to an OCR routine. PaperLink is an interesting camera pen technology, though not capable of handwriting reconstruction. Seok et al. [7] suggest using an "overlooking" camera to trace the pen tip and reconstruct handwriting. This approach can achieve good results, but suffers from a lack of portability.

Some pen-tip camera systems allowing handwriting reconstruction have also been suggested. Iwata et al. [4] propose a camera pen using LLAH and features from document foreground for position detection. Both the absolute position and the document context are recognized. One disadvantage is that this technology does not permit writing on empty sheets of paper, or blank parts of documents. Another method is suggested by Uchida et al. [8]. It uses the microscopic fiber structure of paper to trace handwriting. Its main disadvantage is its limitation to capture only relative movement, but has been solved by Iwata et al. [3]. Video-mosaicing is applied to a number of camera images to obtain one larger piece of the document. When enough document foreground is available, the context can be extracted using LLAH.

Anoto pens are commercial camera pens that allow reconstruction of handwriting using special dot paper. Dots are printed using carbon-based ink and detected using an infrared camera. This technology is very portable and able to distinguish a large number of documents, yet rather pricey. Anoto uses regular position-encoding patterns consisting of a large number of densely placed black dots. Most commonly, documents are printed on special dot paper, although the dot pattern can also be printed on-the-fly provided a sufficient printing resolution and suitable ink.

3. The Proposed Dot Pattern

Concerning practical use, the proposed system is most closely related to Anoto in that it uses a background dot pattern for position determination. The attempt is to make our system affordable, by using sparse dot patterns printed in yellow so as not to compromise readability. The position is not directly encoded into the pattern, instead position detection relies solely on its uniqueness. The pattern proposed in this paper is generated using a randomized method. Patterns can be printed using a cheap color printer, and are extracted using images from a low-resolution camera.

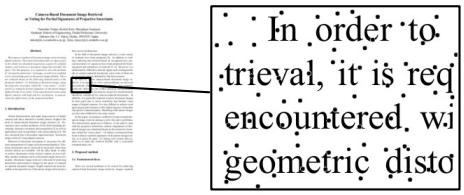


Figure 1. Sketch of dot pattern.

To enable handwriting reconstruction, these dot patterns are printed on each document, as shown in Fig. 1. To generate dot patterns, our method initially produces a regular grid of dots, with a fixed distance in between. These are then displaced both horizontally and vertically by a random offset according to a Gaussian distribution. Offsets are required to lie within the bounding square of the point, to preserve a certain level of regularity. This is important since for all possible positions of the pen on the paper, enough dots must be available for position detection. In other words, no "holes" are allowed. An additional check is performed to ensure a minimum distance between dots, equal to their diameter. This is necessary to recognize when two connected components (see Sect. 4) belong to the same dot, which can be the case if the dot is "split in half" by document foreground.

This method of generating patterns was chosen as a tradeoff between two factors: The dot pattern should appear fairly regular to the eye, in a way that readability of document contents is not disturbed. On the other hand, each pattern must be distinctive so as to enable our method to retrieve the correct position.

Simultaneously, LLAH is used to index and store dot arrangements in a hash table. Two key properties are the use of *local* arrangements of feature points, and *geometric invariants*. The former enables correct retrieval even for the highly limited viewing area of the camera pen, under the condition that enough feature points can be extracted. Furthermore, calculated features are invariant to effects resulting from skewed or rotated camera images.

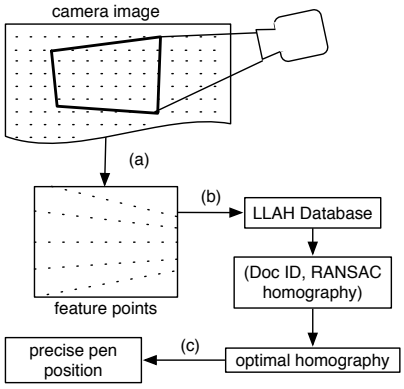


Figure 2. Retrieval steps: (a) Extraction of feature points from camera image, (b) LLAH query (c) determination of pen tip position using optimal homography.

4. Retrieving the Pen Position

The pen tip position is determined from the provided camera image for each frame. Figure 2 shows the steps of retrieval. First, the yellow dots are extracted from the image (Fig. 2, step (a)) as follows: A *distance image* d to the target color yellow is created using RGB representation and applying the following formula to each pixel (x, y):

$$d(x,y) = \sum_{c \in \{r,g,b\}} (w_c \cdot |t_c - p_c(x,y)|) \qquad (1)$$

where c runs through the three color channels, t_c denotes the channels' values of the target color and p_c those of the provided camera image. The three color channels are weighted as $w_r = \frac{2}{9}, w_g = \frac{3}{9}, w_r = \frac{4}{9}$, based on experimental results. Next, *adaptive thresholding* is performed on the result, and noise is removed using *dilation*. Finally, feature points are determined as centroids of connected components.

In the following step, these feature points are used to query the LLAH database [5] and retrieve the document (Fig. 2, step (b)). Feature points are matched and used to determine the perspective transformation (or *homography*) which transforms the original points into the arrangement captured by the camera. This is done using RANSAC [2] and ultimately enables determining the pen tip position. By ignoring outliers, RANSAC is able to estimate a homography that is very close to the actual one. However, the RANSAC approximation of the current position tends to be unstable when used at a very precise level. In other words, reconstructed handwriting is not smooth.

We thus use an additional step in which the inverse of the *RANSAC homography* from the previous step is applied on points extracted from the camera image. This means to position the extracted pattern "above" its corresponding part in the original pattern. The transformed dots are then matched to the nearest database points. The number of matched points is much higher than the number of matches retrieved from LLAH. Also, this time it is simple to ignore outliers by imposing a distance threshold when performing matching. From these matches, a second homography is calculated which is optimal in terms of the least-squares error. Because of the high number of matches and the unlikeliness of outliers, results can be highly improved using this *optimal homography*, rather than the RANSAC homography to calculate the pen tip position (Fig. 2, step (c)). This technique creates much smoother handwriting for negligibly longer processing time.

Finally, some basic error detection is applied, using the observation that often, within a streak of correctly recognized documents, a few individual erroneous ones can be found and should be omitted.

To enable fast retrieval speed, we suggest a combination of two techniques. The retrieval process is *parallelized* using a client-server architecture. The client is responsible for determining coordinates of dots from the camera image, whereas the server performs the LLAH query and retrieves document ID and pen position. In other words, the dots of the current frame can already be extracted, while the LLAH query for the previous frame is still running. This requires either a dual CPU, or a dedicated network server as might be appropriate in organizations. Furthermore, once the current document context has been established with some probability, LLAH queries are performed on *tiny hash tables* containing only one document. Once the determined position gets unstable, the large hash table is used again to retrieve a new, possibly different document context. Loading effort for the tiny hash table is justified by the fact that the context usually remains constant for a longer period of time.

5. Experimental Results

To investigate the usefulness of our method, we evaluated performance as well as each of the retrieval steps as shown in Fig. 2.

For the experiments, LLAH parameters were set to $n = 7$, $m = 6$, $k = 15$ [5]. Affine invariants were used. The size of the hash table was 6.7×10^7 bins (256MB), and for collisions, the corresponding hash entry was marked invalid. For the dot pattern, initial distance between dots was set to 2.7mm, which is equiva-

Figure 3. The camera pen, containing an ordinary ballpoint pen and a tiny USB camera.

Table 1. Runtime needed for individual retrieval steps, with respect to database size.

Extraction of dots	13.7ms		
LLAH query	*num. of documents in DB*		
(assuming 80	*1*	*100*	*1,000*
feature points)	7.2ms	10.7ms	38.2ms
Homographies	2.7ms		
Loading tiny DB	(195ms)		

lent to 7918 dots per document. Dot diameter was set to 0.2mm. The pattern was printed using a laser printer.

The computer hardware featured an Intel Core CPU clocked at 2.13GHz, and 3GB RAM. For the pen, we used a cheap USB camera with a resolution of 720×576, providing 30 frames per second. It should be noted here that the speed is too slow to track fast writing. In the experiments, we performed only slow handwriting, but for realistic use the camera should be replaced by a one with higher speed. The construction of the pen can be seen in Fig. 3. When facing straight down, the camera's distance to the document was about 3.3cm, providing a captured area of about 2.5×2.0 cm^2, or equivalently $\sim$68.8 dots. The actual number of dots was often higher because of a steeper camera angle when writing.

5.1. Performance

We measured performance of the retrieval step. Extraction of dot coordinates from the camera image took up a fixed amount of time, about 13.7ms. The remainder of runtime was primarily needed by the LLAH query and depended strongly on the number of documents in the database and the number of dots used for the query, as can be seen in Table 1. For the more desirable database size of 1,000 documents, about 38.2ms of CPU time were needed. Even using parallel processing, this would allow only 25 frames per second when always querying the large database, which is insufficient for realtime capture of appropriately fast writing.

Since runtime is much faster for the case of only one document in the database, document context should be used to load a tiny database. When using such a tiny database, in combination with parallel processing by means of a client-server architecture, real-time application is achievable: The client process needs 13.7ms per frame for dot extraction, the server process a total of 9.9ms, plus an occasional database reload. This would allow more than 70 frames per second on our used machine, assuming an appropriately fast camera.

5.2. Extraction of the Dot Pattern

To evaluate our method of extracting the dot pattern (Fig. 2, step (a)) from the camera image, we prepared three videos, capturing the trajectory of a smaller amount of handwriting: (a) No document content was visible, (b) a small amount, (c) a larger amount of content (see Fig. 4). Care was taken to impose minimal changes to lighting conditions and camera angle between the three videos.

Our goal was to measure how well points extracted from the image match to points in the LLAH database. For this purpose, we applied the *optimal homography* on database points, and matched these to the extracted points using nearest neighbors with a distance threshold. The number of documents in the LLAH database was 100. Table 2 shows that for more document content, the number of correct matches drops, due to occlusion. Also, the number of falsely extracted points increases, partly because additional content distorts the adaptive thresholding.

Finally, we also measured the mean square error of detected positions, by using the matches to calculate a homography from the correct points to the extracted points which is optimal in terms of the least-squares error. Table 2 shows that extraction with little document content is also more accurate in terms of this value. This is (1) because of effects of the two previously discussed observations, and (2) because with more content, yellow dots are more likely to be partly hidden, moving the extracted centroid of the dot away from their actual center.

5.3. LLAH Accuracy

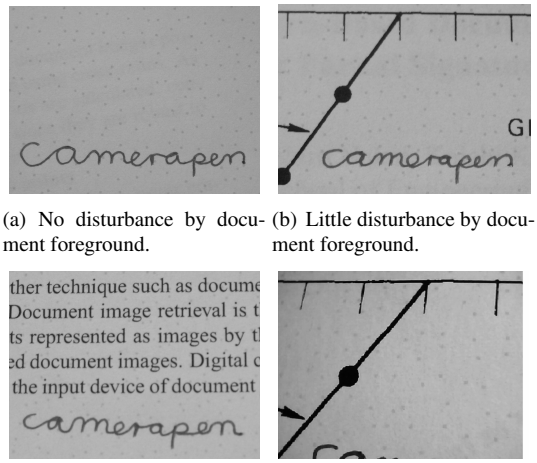
For evaluating the accuracy of LLAH (Fig. 2, step (b)), we used video frames for the handwritings showed in Fig. 4. For each of the three examples, we determined the number of frames that met both of two conditions: (1) The correct document was recognized by LLAH, (2) the determined position was within a bounding rectan-

Table 2. Extraction accuracy for the examples in Fig. 4(a)–4(c). The number of correct points and falsely extracted points, and the mean square error are denoted, each averaged over all frames marked as correct.

	visible document content		
Property	none	little	much
Avg. # correct points	77.7	72.6	65.9
Avg. # false positives	3.1	6.5	10.9
Mean square error	1.9 px	2.5px	3.7px

Table 3. LLAH accuracy for the examples in Fig. 4, and different database sizes.

	visible document content		
DB size	none	little	much
1 document	100.0%	100%	65.2%
100 documents	100.0%	99.8%	48.7%
1,000 documents	85.8%	81.1%	6.4%



(a) No disturbance by document foreground.

(b) Little disturbance by document foreground.

(c) Much disturbance by document foreground.

(d) Example of actual camera view for Fig. 4(b).

Figure 4. Handwriting to be recognized with three levels of difficulty. Note that the foreground above the handwriting is that of interest, since this is what the camera captures, as can be seen in Fig. 4(d).

gle of $4.0 \times 0.8\text{cm}^2$, drawn around the actual handwritten word. This experiment was performed using three database sizes.

Table 3 shows almost perfect result for the case of no disturbance by document contents, and good results for moderate disturbance. However, for the third example, accuracy dropped heavily, especially for the large database. It also shows that for only 1 document in the database, accuracy is by far the highest.

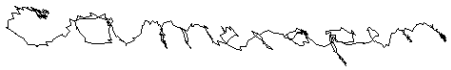
5.4. Handwriting Reconstruction

After showing potential and limitations of the proposed method on a more theoretical level, we demonstrate the actual quality of reconstructed handwriting (Fig. 2, step (c)). We used the videos from the previous section. For visualization, a naïve approach of drawing straight lines between consecutively determined pen coordinates was employed. Coordinates "far off" (outside the bounding box, as defined in Sec. 5.3) were ignored, since in realistic applications they can be easily detected using distance thresholds. Figure 5 shows the reconstructed handwriting for different setups. The difference between using only RANSAC homographies and an additional optimal homography can be seen in Figs. 5(a) and 5(b). Figures 5(b) – 5(e) show increasingly difficult configurations. It can be seen that for no document foreground, and little foreground but small database size, no reconstruction errors were made, the result looks smooth. For little foreground but larger database sizes, as well as lots of foreground, a small number of errors were introduced.

6. Discussion and Outlook

The experiments show reasonable results when the dot pattern is occluded only to a certain extent. Problems arise when too much document content interferes with the extraction of the pattern. This can be solved by introducing additional features, based on document foreground [4]. Centroids of connected components (i.e., characters) are used as feature points, indexed as before, and extracted separately. It is beneficial to detect collisions between foreground and background before printing, and exclude respective dots from the pattern.

Our experiments also show decreasing accuracy for larger databases, due to their large number of features to be distinguished. Still, it is desirable to achieve database sizes well over 1,000 documents. When including features from document foreground, accuracy can be assumed to behave as for the example from the experiments with no disturbance by document foreground. This leaves room for further increasing of database size. Moreover, dealing with large databases is really only needed to retrieve document context, in which case less accuracy is needed than for determin-



(a) RANSAC homography, 100 documents in database, no foreground as in Fig. 4(a)

(b) optimal homography, 1000 documents in database, no foreground as in Fig. 4(a)

(c) optimal homography, 100 documents in database, little foreground as in Fig. 4(b)

(d) optimal homography, 1000 documents in database, little foreground as in Fig. 4(b)

(e) optimal homography, 100 documents in database, much foreground as in Fig. 4(c)

Figure 5. Reconstructed handwriting for different database sizes and amount of document foreground. Benefits of using the optimal homography can be seen.

ing the precise pen position. Finally, further potential of improvement might come from making features more discriminative, for example using higher quantization or more permutations for LLAH feature calculation. Care must be taken with robustness issues, however.

A possibility to avoid reloading tiny hash tables is to adapt LLAH to retrieve only points from the established document context despite using the large database.

On a final note, for practical usage it is best to replace the USB camera connection we used by either a wireless connection, or a memory stick storing coordinates of extracted dots. Also, for averagely fast writing, a high-speed camera is needed. For cameras with lower speed, too few positions are captured when writing fast.

7. Conclusion

We presented a new approach for a camera pen. Using this pen, handwriting applied on paper can be automatically transferred into digital form and used for further applications. Main advantages are the low costs, and the fact that no equipment besides the camera pen itself is necessary. For the pen, we used a cheap USB-camera. To enable handwriting reconstruction, a yellow dot pattern is printed on normal paper using an ordinary laser printer.

Our camera pen allows accurate reconstruction of handwriting, when enough yellow dots can be extracted from the camera image. Using a client-server architecture and knowledge about the document context, fast position determination is possible.

Future work includes incorporating document foreground into the feature space, to resolve issues of colliding background and foreground. Furthermore, methods should be investigated to increase database sizes, as well as further improve retrieval speed.

Acknowledgements. This work was in part supported by the Grant-in-Aid for Scientific Research (B) (19300062) and the Grant-in-Aid for Challenging Exploratory Research (21650026) from Japan Society for the Promotion of Science (JSPS).

References

[1] T. Arai, D. Aust, and S. E. Hudson. Paperlink: a technique for hyperlinking from real paper to electronic content. In *CHI '97: Proc. of the SIGCHI conference on Human factors in computing systems*, pages 327–334, New York, NY, USA, 1997. ACM.

[2] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[3] K. Iwata, K. Kise, M. Iwamura, S. Uchida, and S. Omachi. Camera pen system using feature tracking and document image retrieval. *IEICE Technical Report*, 109(418):34–44, February 2010.

[4] K. Iwata, K. Kise, T. Nakai, M. Iwamura, S. Uchida, and S. Omachi. Capturing digital ink as retrieving fragments of document images. *Proc. (ICDAR2009)*, pages 1236–1240, July 2009.

[5] T. Nakai, K. Kise, and M. Iwamura. Hashing with local combinations of feature points and its application to camera-based document image retrieval. *Proc. CBDAR2005*, pages 87–94, Aug. 2005.

[6] A. J. Sellen and R. H. R. Harper. *The Myth of the Paperless Office*, volume 1. The MIT Press, 1 edition, 2003.

[7] J.-H. Seok, S. Levasseur, K.-E. Kim, and J. H. Kim. Tracing handwriting on paper document under video camera. *Proc. ICFHR*, 2008.

[8] S. Uchida, K. Itou, M. Iwamura, S. Omachi, and K. Kise. On a possibility of pen-tip camera for the reconstruction of handwritings. *Proc. CBDAR2009*, pages 119–126, July 2009.