9-2019

Symmetric Inkball Alignment with Loopy Models

Nicholas Howe
Smith College, nhowe@smith.edu

Ji Won Chung
Smith College

Symmetric Inkball Alignment with Loopy Models

Nicholas R. Howe
Smith College
Northampton, Massachusetts, USA
nhowe@smith.edu

Ji Won Chung
Smith College
Northampton, Massachusetts, USA
goldgwon@gmail.com

***Abstract*—Alignment tasks generally seek to establish a spatial correspondence between two versions of a text, for example between a set of manuscript images and their transcript. This paper examines a different form of alignment problem, namely pixel-scale alignment between two renditions of a handwritten word or phrase. Using loopy inkball graph models, the proposed technique finds spatial correspondences between two text images such that similar parts map to each other. The method has applications to word spotting and signature verification, and can provide analytical tools for the study of handwriting variation.**

***Index Terms*—alignment; inkball; handwriting; pattern recognition;**

I. Introduction

Alignment refers to the process of bringing two or more corresponding patterns into juxtaposition, such that areas of similar nature overlap. Within the realm of document analysis, one familiar form of this problem is the alignment of a document image with its transcript text. This is an example of a one-dimensional alignment problem, since the positions of the transcript characters are adjusted only along the horizontal dimension. One-dimensional sequence alignment has been well studied, and may be solved via algorithms such as dynamic time warping, hidden Markov models or connectionist temporal classification.

This paper examines a different problem, namely flexible fine-grained alignment in two dimensions at once. We seek alignment at the pixel level between different versions of a handwritten word. Figure 1 shows an example of such an alignment, where each portion of one handwritten version of a word is mapped to a corresponding portion of a second. Some features of a word may be present in only one image; rather than generating spurious matches for these portions, they are flagged in red as unmatched components.

A. Related Work

Pixel-scale alignment of offline handwriting has received some attention, albeit less than other alignment problems and mostly for rigid or affine transformations. Alignment has long been significant for online signature matching [1]. In other contexts such as word spotting, pixel-scale alignment has been treated more as a means to an end than as a goal in its own right.

Some scholars have presented work which generates pixel alignment between handwriting images under a limited set of possible transformations. Manmatha et al. perform affine alignment of word images using the general cost-weighted bipartite matching algorithm of Scott and Longuet-Higgins [2], [3]. Learned-Miller also uses affine alignments of letter instance samples [4]. Work on alignment using flexible (e.g., non-affine) transformations is less common. Hassner et al. use an alignment based upon dynamic time warping to perform transcript alignment, with intermediate steps of the algorithm working at the pixel level [5]. Howe uses tree-structured inkball models as a tool for offline word spotting, and also to solve flexible keypoint alignment problems [6], [7]. Leung and Suen describe a pattern alignment algorithm similar in spirit to that proposed here [8]. It decomposes a signature into line and curve primitives, then matches them symmetrically via an iterative spatial warping process with a gradually decreasing neighborhood of influence. Fang et al. later apply this method to the signature verification task, not for directly measuring similarity but to generate synthetic training samples by interpolation [9].

Fig. 1. Alignment of one image with a second (shown to scale). Areas with red dots have no match detected in the opposite image.

This paper makes two novel contributions to research on handwriting alignment. First, it drops the requirement found in prior inkball work that all models of handwriting be structured as a tree. Since most handwriting does include loops and other cyclical structures, this allows for better handling of realistic common scenarios. Second, it searches for a match between the two structures in a symmetric bidirectional manner, whereas prior work matches in only one direction at a time. Symmetric bidirectional matching explicitly penalizes many-to-one matches, and provides a powerful corrective against many common errors seen in prior inkball work. Figure 2 shows an example.

The next section proposes a novel algorithm for symmetric flexible matching of handwriting keypoints. Section III describes experiments that explore its properties. The last section concludes by considering the meaning of the results and proposing future work.

II. Algorithm

What does it mean to align two images of writing at the pixel level, assuming that both images represent the same word or piece of text? For documents produced by typewriter a natural ground truth definition presents itself: ink pixels should align if they were produced by contact with the same portion of a given typewriter key. This concept may be expanded to all printed text in a single font, under the mild assumption that printed characters intended to be identical in shape are thus functionally interchangeable.

For handwritten characters, whose appearance varies with each instantiation, we can posit the existence of a canonical character form (analagous to a Platonic ideal). Each pixel in a particular written instance of a character may be attributed to an imperfect reproduction of some corresponding point on the ideal form. Pixel-scale alignment between two handwritten characters thus means pairing points from each such that both correspond to the same position on the ideal form.

In practice, ideal character forms are both hypothetical and unknown. Yet they provide conceptual guidance about the qualities of a good alignment between actual character examples: distinctive points marked by curvature, extremity, or juncture should correspond, while assignment of the the less notable points between them should be smoothly continuous and evenly distributed. These considerations guide qualitative assessment of an alignment. Quantitative indicators can be derived from applying the alignment toward some other task, such as word spotting or signature verification.

With these considerations, an alignment of the ink skeleton, or even a finite set of keypoints evenly spaced along it, can stand in for a pixel-by-pixel alignment. Given an alignment of keypoints, interpolation between them extends the alignment to the entire skeleton. Points not on the skeleton may then be aligned by interpolation along the segment between their nearest skeleton point and the ink boundary. Figure 3 illustrates this process. In practice, a full alignment rarely requires computation because the keypoint alignment itself suffices for most purposes. Note that the alignment specified by a keypoint matching is more complex than warping and stretching of the 2D plane, since it allows the relative topology of the points to change.

Keypoint alignment lends itself to an *inkball* representation of handwriting, where written symbols are modeled as overlapping disks of ink arranged in a 2D spatial pattern. Each keypoint represents the center of a disk of ink in this model. Prior work has studied one-way matches between inkball models of handwriting and observed markings [6], [7]. It employs an asymmetric matching process: a flexible model adapts to a static target, and multiple model parts can match to the same target structure without penalty. The new algorithm proposed herein differs in that both sides of the comparison are inkball models, and the goal is to find something close to a one-to-one matching while also respecting the inherent geometry of the two sides. In this sort of symmetric bidirectional match, each side actively matches to the other, and the strongest bonds

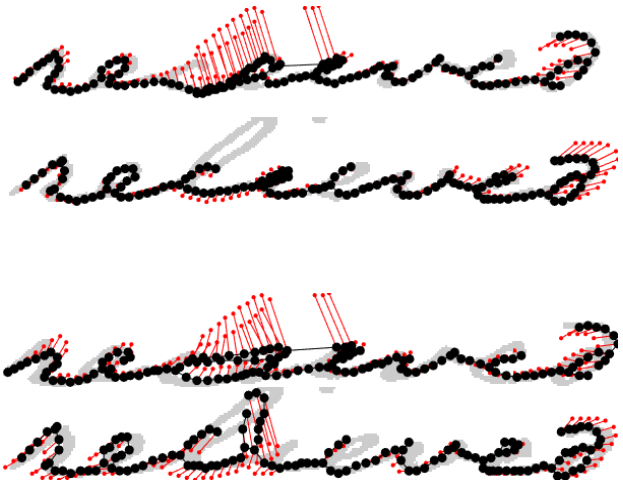

Fig. 2. Difference between asymmetric (top pair) and symmetric matching (bottom pair). The two asymmetric matches are inconsistent with each other, yet do not pay any penalty as a result. The symmetric match enforces mutual consistency, correctly assessing a penalty for aligning the top of the 'l' with the letter 'e'.

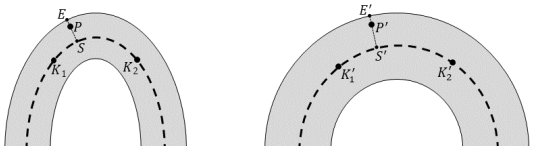

Fig. 3. Interpolation of a full pixel-scale alignment from keypoints. P is aligned with P' when the proportional distances between E and S and between k_1 and k_2 match those between E' and S' and between k'_1 and k'_2 respectively.

form where the attraction is mutual on both sides. Figure 2 illustrates the difference between the two approaches.

To formalize this intuition, suppose that L and R represent two inkball models to be compared, referred to respectively as the left and right model. Models may be derived automatically from images of handwriting by thinning to a single pixel width skeleton and placing inkballs (keypoints) at the endpoints, junctions, and at regularly spaced intervals on all branches. Label the keypoints k_i^L and k_j^R, where $1 \leq i \leq n_L$ and $1 \leq i \leq n_R$. (Throughout this paper the designations L and R will consistently indicate the model to which a particular entity belongs.) Each keypoint has a set of neighbors as indicated by the connectivity of the skeleton and denoted H_i^L or H_j^R.

A perfect alignment would pair each k_i^L with exactly one k_j^R and vice versa. The possibility that $n_L \neq n_R$ means that perfect alignment may not be attainable in practice. Furthermore, either image may contain extraneous structure not present in the other, meaning that some keypoints should not be assigned a corresponding node even if the numbers allow it. To handle this situation, we can define alignment as a directional bipartite match between the expanded sets $\{k_i^L\} \cup \{\emptyset^L\}$ and $\{k_i^R\} \cup \{\emptyset^R\}$, where pairings to $\emptyset^L$ and $\emptyset^R$ indicate null matches. We can represent the assignment using two functions $A^L(k_i^L) \rightarrow k_j^R$ and $A^R(k_j^R) \rightarrow k_i^L$.

Many assignments are possible within the framework just

described, but few will respect the geometrical arrangement and connectivity of the original keypoints. Qualitatively speaking, the best alignment maps keypoints to locations that preserve their relative positions as compared to neighboring points. Suppose that the keypoint locations in the source image are $\mathbf{v}_i^L$ and $\mathbf{v}_j^R$ respectively, and their new configurations under a proposed alignment are given by $\mathbf{c}_i^L$ and $\mathbf{c}_j^R$. Simple rigid translation by a vector $\mathbf{w}$ is one possible alignment that perfectly preserves geometry.

$$\mathbf{c}_i^L = \mathbf{v}_i^L + \mathbf{w} \Rightarrow \mathbf{c}_i^L - \mathbf{c}_{i'}^L = (\mathbf{v}_i^L + \mathbf{w}) - (\mathbf{v}_{i'}^L + \mathbf{w}) = \mathbf{v}_i^L - \mathbf{v}_{i'}^L \tag{1}$$

More likely there will be some deformation δ between the default and aligned configurations but for desirable alignments the magnitude should be small, particularly between neighboring keypoints.

$$\mathbf{c}_i^L - \mathbf{c}_{i'}^L = \mathbf{v}_i^L - \mathbf{v}_{i'}^L + \delta_{ii'}^L \tag{2}$$

The magnitudes of the observed deformations $\delta_{ii'}^L$ and $\delta_{jj'}^R$ will be used below (see Equation 10) to compute a deformation mismatch energy E_Δ.

A. Measurement

Given proposed configurations for each side, a precise definition of the alignment quality can now be given. Broadly speaking the chosen definition encompasses three different pieces: the proximity of configured nodes to a corresponding target on the opposite side, the consistency of the implied pairing given model connectivity, and the deformations embodied in the configuration itself. It is convenient to formulate quality as an energy to be minimized.

$$E = \lambda_M E_M + \lambda_A E_A + \lambda_\Delta E_\Delta \tag{3}$$

The first term is computed from the configuration vectors under the assumption that each keypoint maps to its closest point on the opposite side. This provides both the keypoint assignment and a set of distances to sum.

$$A^L(k_i^L) = k_j^R : j = \arg\min_k \|\mathbf{v}_k^R - \mathbf{c}_i^L\| \tag{4}$$

$$\epsilon_i^L = \min_k \|\mathbf{v}_k^R - \mathbf{c}_i^L\| \tag{5}$$

$$E_M = \frac{1}{n_L}\sum_i \epsilon_i^L + \frac{1}{n_R}\sum_j \epsilon_j^R \tag{6}$$

The second term captures the self-consistency of the implied alignment. An ideal alignment arranges keypoints in perfect pairs, e.g., $A^R(A^L(k_i^L)) = k_i^L$. In cases of misalignment, where $A^R(A^L(k_i^L)) \neq k_i^L$, the severity of the mismatch depends on the separation of the matched nodes, i.e., how far one must travel through the keypoint graph to find a return path, quantified via the geodesic distance functions G^L and G^R.

$$E_A = \frac{1}{n_L}\sum_i T^L(k_i^L) + \frac{1}{n_R}\sum_j T^R(k_j^R) \tag{7}$$

$$T^L(k_i^L) = \min_k \left[G^R(A^L(k_i^L), k_k^R) + G^L(A^R(k_k^R), k_i^L)\right] \tag{8}$$

$$T^R(k_j^R) = \min_k \left[G^L(A^R(k_j^R), k_k^L) + G^R(A^L(k_k^L), k_j^R)\right] \tag{9}$$

The functions T^L and T^R measure the round trip distance from a keypoint to its associated node and back again via geodesic paths (always zero in the case of a perfect pairing).

The third term in the energy measures how much the original structure must be deformed to achieve the chosen configuration.

$$E_\Delta = \frac{1}{n_L}\sum_i \sum_{h\in H_i^L} \|\delta_{ih}\| + \frac{1}{n_R}\sum_j \sum_{h\in H_j^R} \|\delta_{jh}\| \tag{10}$$

The above equations do not account for keypoints that may have no proper match in the opposite image. Without proper handling these may be assigned a spurious match at very high energy. To achieve more stable results, a modified energy formula limits the maximum effect of unmatched nodes. For brevity, we define $\xi_i^L = \sum_{h\in H_i^L} \|\delta_{ih}\|$, and also assume below that $\lambda_M = \lambda_A = \lambda_\Delta = 1$. Thresholds τ_1 and τ_2 set the standard maximum per-node energy contribution.

$$\begin{aligned} E^* = & \\ & \frac{1}{n_L}\sum_i \left[\min\left(\tau_1, \epsilon_i^L + T^L(k_i^L)\right) + \min\left(\tau_2, \xi_i^L\right)\right] \\ + & \frac{1}{n_R}\sum_j \left[\min\left(\tau_1, \epsilon_j^R + T^R(k_j^R)\right) + \min\left(\tau_2, \xi_j^R\right)\right] \end{aligned} \tag{11}$$

B. Optimization

Finding a good alignment under the conditions above is not easy, since moving one point in a configuration affects each of its neighbors. Loop connections in written symbols create cycles in the neighbor graph, introducing the potential for complicated feedback. Perfect optimization in similar systems has been shown to be NP hard, yet nevertheless good results have been obtained in practice using message passing algorithms [10], [11]. This paper therefore seeks to develop a set of promising optimization heuristics, with the goal of achieving a high-quality final outcome despite a lack of theoretical guarantees.

The algorithm operates in rounds. Each keypoint maintains a state ψ_i^L or ψ_j^R representing a probability distribution for its possible location over 2D image space. (For both convenience and numeric stability, all 2D distributions are represented computationally as negative log probabilities sampled on a pixel-resolution grid.) During a round, each keypoint receives messages from its neighbors and also incorporates information from the oppositely aligned model to update its state. The positions of most keypoints usually stabilize after just a half dozen rounds or so, while a few take longer to settle.

1) Initialization: Because the optimization is heuristic rather than exact, the initialization of the keypoints influences the end result. A slight bias towards likely pairings can help the solution to converge quickly, but overcommitment at this stage can also lead to suboptimal results. The proposed initialization strategy starts with plausible relative probabilities for matching at each of the opposite model's keypoints, and expands this to

a full 2D probability distribution via the following process: (1) interpolate squared log probabilities between keypoints on the handwriting skeleton; (2) use a generalized distance transform (GDT) [12] to extend to all other points by adding their squared distance from the skeleton as a penalty; (3) normalize the entire 2D probability distribution to sum to 1.

Let $S(...)$ denote the function from a keypoint value set to 2D probability distribution just described. We use the difference in the local skeleton tangent angle between the left and right keypoints (denoted α below) to generate the keypoint value set, and thence the full probability distributions.

$$\psi_i^L = S^R(\alpha(k_i^L, k_1^R), \alpha(k_i^L, k_2^R), ...) + \lambda_P P^R(k_i^L) \quad (12)$$

The second term $P^R(k_i^L)$ is a simple Gaussian potential, centered at the relative x and y percentile position of k_i^L in the image. It favors solutions that match keypoints in similar positions.

2) Update: During a round, keypoints update sequentially in a randomly chosen order. Each keypoint incorporates the current state of its neighbors, translated by an offset taken from the original model configuration, diffused to account for flex in the model, and renormalized to sum to 1.

$$\psi_i^L \leftarrow N\left(\psi_i^L + \sum_{h \in H_i^L} \Upsilon(\psi_h^L, \mathbf{v}_h^L - \mathbf{v}_i^L)\right) \quad (13)$$

Here $N(\cdot)$ signifies normalization, and $\Upsilon(\psi_h^L, \mathbf{v}_h^L - \mathbf{v}_i^L)$ first translates ψ_h^L by $\mathbf{v}_h^L - \mathbf{v}_i^L$ and then applies the GDT. Note that the GDT here is used for tractability, in lieu of a more exact computation of the probability diffusion.

At the end of the round, the state of every node is updated further using information from the opposite model. First we find the cross probabilities p_{ij}^L and p_{ji}^R by evaluating the a keypoint's distribution at the locations of the nodes in the opposite model.

$$p_{ij}^L = N^*(\psi_j^R(\mathbf{v}_i^L)) \quad (14)$$

$$p_{ji}^R = N^*(\psi_i^L(\mathbf{v}_j^R)) \quad (15)$$

Here $N^*(\cdot)$ indicates that normalization is applied over the set of keypoints only. Thus p_{ij}^L gives the probability that keypoint k_j^R aligns with k_i^L.

This information is used in two ways. First, all states are updated by a distribution derived from the partner probabilities.

$$\psi_i^L \leftarrow N\left(\psi_i^L + S^R(p_{i1}^L, p_{i2}^L, ...)\right) \quad (16)$$

Second, we can detect keypoints that have no corresponding partner on the other side.

$$q_i^L = \max(0, 1 - \sum_j p_{ij}^L) \quad (17)$$

The state of these nodes is updated further, mixing with a uniform distribution over all locations at total probability q_i^L.

3) Finalization: The message passing stages run for R rounds, after which the marginal configuration can be read directly from the node states.

$$\mathbf{c}_i^L = \arg\min_w \psi_i^L(w) \quad (18)$$

Depending upon the complexity of the patterns to be matched, the probability maps for some points may exhibit multiple competing modes even after iteration. In this case the marginal configuration may end up with large discontinuities between the final positions of neighboring nodes where the most likely mode undergoes a transition. To mitigate this, a globally consistent configuration can be estimated by fitting a traditional tree-structured inkball model, using the node states as the data term. This technique essentially applies an asymmetric process on top of the symmetric match results, so we refer to it as *hybrid symmetric*. It tends to produce configurations with smaller displacements between neighbors, but reintroduces a small possibility of inconsistencies between the matches in each direction.

III. Experiments

Pixel-scale keypoint alignment can serve as a diagnostic and analytical tool for handwriting comparison. It also offers potential applications in signature verificaton and word spotting, although current implementations are too slow for practical use in the latter role. This paper aims to demonstrate the basic capabilities of the method in various areas. Exhaustive testing for any particular application is left as future work.

A. Word Spotting

Although current implementations are too slow for full-scale word spotting applications, the proposed technique may be useful as a tool for reranking images retrieved by another faster method. The George Washington dataset [13] makes a useful test case for this hypothesis because it is familiar and well studied. We examine one-shot single-word queries, without training. Each word image can form a query in leave-one-out mode and there is no need for a train/test split. Of the 4857 segmented words in the 20-page set, 4161 appear more than once and therefore can serve as useful test queries.

The experimental protocol begins with an initial ranking of all the target words produced using an asymmetric part-structured inkball model match, as described in prior work [6]. Following this, the top k words are reranked using six rounds of the symmetric two-way match described in Section II. The mean average precision over these k retrievals serves to measure the quality of the reranking. Figure 4 shows the relative performance of the proposed technique as compared to the original ranking, for various values of k. Both the fully symmetric and hybrid symmetric results greatly improve on the asymmetric result, with the fully symmetric match slightly ahead.

Reranking using the proposed technique greatly improves on the earlier inkball method for word spotting. It does not compete with learning-based approaches to this problem, several versions of which achieve mean average precision in

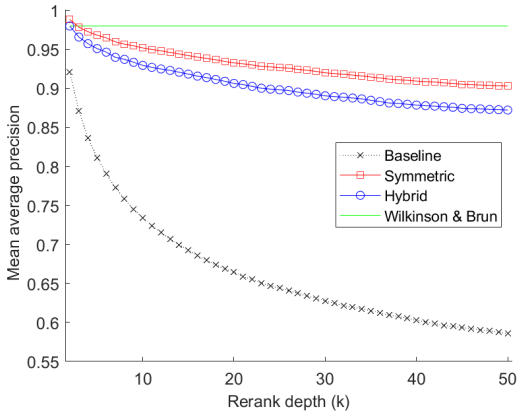

Fig. 4. Mean average precision within top k reranked results for one-shot query by image protocol. (Computed over queries with at least one hit in the original top k results.) State of the art shown for comparison [14].

the high 90s [15], [16]. On the other hand, such methods typically rely on offline training with data from the target collection or something similar. By contrast, inkball methods work on a single example without offline training.

B. Signature Matching

Signatures offer a promising area for alignment analysis. While each instance is unique, genuine signatures come from a single writer and thus presumably share a strong underlying prototype. They are often complex relative to ordinary writing, and may include flourishes and complicated overwriting. These features mean that individual instances may differ markedly in topology, specifically in terms of which strokes cross each other and where.

Signature matching comes in two modes, termed *online* and *offline*. The former allows use of temporal sequence information about the strokes that form the writing, while the latter provides only images of a complete signature. Online data typically includes a the pen tip location over time, and may include pressure data as well, or at least pen up/pen down indications.

Online data present an opportunity to develop inkball models in a more realistic manner. With offline images, the stroke order at crossings is unclear, resulting a model with junctions even though the actual pen trajectory is a 1D curve. Although some crossings are intentional, others may result incidentally from a flourish that can intersect at different points in each rendition of a signature. Online data offer a way to model different topologies by using pen trajectory itself as the model. Choosing keypoints at regular intervals along the trajectory, and creating neighbor relations only with the previous and next points on the trajectory, we create a model that closely mimics the behavior of the actual writing. In particular, points that result from distant portions of the pen trace will not be strongly contrained to cross at particular locations.

Given inkball models built from both online and offline signatures, it becomes possible to compare online versions to offline within a single framework. This offers an advantage in realistic applications: even if online signatures are not generally available, procuring a single online sample for each user can be achieved at a much lower cost. Matching an online sample to an offline version may offer many of the advantages of online algorithms, without as many constraints on practical implementation. The computational cost of online and offline models are similar.

We test these hypotheses on a portion of the GPDS synthetic online/offline signature data set [17], [18]. This set comprises 100 writers, with 24 genuine signatures and 30 skilled forgeries for each. In addition to the signature images, online trace information is available, including 2D position and pressure information at regular time intervals. To convert the latter to inkball models, cubic spline interpolation renders the pen trace as a skeleton of 8-connected pixels. Keypoints are selected at regular intervals covering the pen-down regions, defined as any position where pressure exceeds a low threshold (10% of the maximum range). Offline images of signatures are also converted to inkball models using the procedure described in Section II. Although the online and offline models have very different structures, they can still be matched against each other.

Table I summarizes the results of several experiments run on the GPDS set, using the first ten genuine and ten forged signatures for each. The first genuine signature is used as the test in each case. Condition 1 does symmetric matching using a left model built from the non-branching online pen trace of the test signature, and a right model built from the offline target image. Condition 2 uses both left and right models built from the offline images. Conditions 3 and 4 repeat experiments using the models from 1 and 2 with a hybrid symmetric match. Finally, conditions 5 through 7 are control experiments based on prior work: conditions 5 and 6 perform two opposite asymmetric inkball matches using the models from conditions 1 and 2 and taking the maximum of their energies, while condition 7 uses only a one-way offline match. (Since asymmetric matches require a tree structure, any loops in the model are broken arbitrarily.) In each case, the model fitting gives an energy score that measures the quality of the match. The table shows the equal error rate accuracy of a threshold classifier based upon the fitting energy, with the appropriate threshold tuned for each writer. A right-tailed T test indicates that condition 3 is significantly better than both condition 1 ($p < 0.01$) and conditions 5-6 ($p < 0.025$) but the improvement on conditions 2 and 4 is only marginal ($p < 0.1$). All conditions 1-6 are significantly better than 7 ($p < 0.025$).

The results here appear better than published error rates on this data set [18], but no firm conclusions can be drawn because the experiments use just a subset of the available signatures. The one-way offline-to-offline asymmetric technique (condition 7) has previously been found to be competitive with state of the art methods on other datasets [19].

The GPDS is particularly challenging for the proposed technique because many signatures feature overwriting with closely spaced and nearly parallel lines, which are difficult

TABLE I
SIGNATURE VERIFICATION RESULTS ON GPDS SUBSET

Condition	Equal Error Rate
1. Online to offline symmetric	16.6%
2. Offline to offline symmetric	14.8%
3. Online to offline hybrid	**12.4** %
4. Offline to offline hybrid	14.6 %
5. Online to offline asymmetric	16.2 %
6. Offline to offline asymmetric	15.6 %
7. Offline to offline one-way	20.8 %
*Ferrer et al. [18] (*full set)	16.4%

Fig. 5. Seven versions of the term '16th' with superimposed keypoint color coding based on satisfaction scores. Green areas are well matched in other images, while red indicates rare or poorly matched features.

for the algorithm to distinguish from each other. This may explain the difference in results observed between conditions 1 and 3, which both use models built from the online pen trace. Examination of the condition 1 results shows that the matched position occasionally jumps between parallel lines. The hybrid symmetric result in condition 3 suppresses such artifacts and therefore can produce a better result. Note that although these two conditions start with online pen traces, all models are purely geometric and do not make use of velocity information.

C. Analytical Tool

Properties of the keypoint match can be measured and used to draw useful analytical conclusions. One example is keypoint *satisfaction*, defined as the extent to which a keypoint's match is mutually returned.

$$s_i^L = \max_j p_{ij}^L p_{ji}^R \tag{19}$$

Figure 5 shows an example visualization created based upon satisfaction scores. Multiple examples of a selected word are compared against each other, and the satisfaction is tallied at each keypoint. The resulting scores are displayed in the figure by means of color codes placed on top of the original word images. Features that do not commonly appear in the other renditions get low scores, shown in red. The technique identifies unusual features such as stray marks, long extensions on the crossbar of the letter t, and the split stroke on the numeral one which appears in some examples but not others.

IV. CONCLUSION

This paper has proposed to perform pixel-scale alignment via a symmetric keypoint matching algorithm, using message passing on a structural graph. Over all the results look promising, even if the method does present some limitations. It currently runs relatively slowly due to the many image translations that must be computed during the update stage (Equation 13). This computation can be parallelized, and in the future a GPU implementation of the alignment algorithm should perform significantly faster and allow much more thorough experimentation.

Pixel-scale alignment of handwriting images deserves further study on its own merits, besides its use as a step toward accomplishing other tasks. Beyond the methods presented herein, it would be worthwhile to explore alternate techniques using other approaches, perhaps based upon deep learning or other trained methods. Reliable tools for such alignment can provide insight on handwriting style and its variations, and may yet lead to hitherto unforeseen applications.

REFERENCES

[1] X. Xia, Z. Chen, F. Luan, and S. Song, "Signature alignment based on GMM for on-line signature verification," *Pattern Recognition*, vol. 65, pp. 188–196, May 2017.

[2] R. Manmatha, C. Han, and E. Riseman, "Word spotting: A new approach to indexing handwriting," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 1996.

[3] G. Scott and H. C. Longuet-Higgins, "An algorithm for associating the features of two images," *Proceedings: Biological Sciences*, vol. 244, no. 1309, pp. 21–26, Apr. 1991.

[4] E. Learned-Miller, "Data-driven image models through continuous joint alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 2, pp. 236–250, 2006.

[5] T. Hassner, L. Wolf, and N. Dershowitz, "OCR-free transcript alignment," in *Int. Conf. on Document Analysis and Recognition*, 2013.

[6] N. Howe, "Part-structured inkball models for one-shot handwritten word spotting," in *Int. Conf. on Document Analysis and Recognition*, 2013.

[7] ——, "Inkball models for character localization and out-of-vocabulary word spotting," in *Int. Conf. on Document Analysis and Recognition*, 2015.

[8] C. H. Leung and C. Y. Suen, "Matching of complex patterns by energy minimization," *IEEE Trans. on Systems, Man, and Cybernetics – Part B*, vol. 28, no. 5, pp. 712–720, October 1998.

[9] F. B., L. C. H., Y. Y. Tang, P. C. K. Kwok, K. W. Tse, and Y. K. Wong, "Offline signature verification with generated training samples," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 149, no. 2, pp. 85–90, April 2002.

[10] G. Cooper, "The computational complexity of probabilistic inference using Bayesian belief networks," *Artificial Intelligence*, vol. 42, no. 2-3, pp. 393–405, March 1990.

[11] L. Sigal, M. Isard, H. Haussecker, and M. Black, "Loose-limbed people: Estimating 3d human pose and motion using non-parametric belief propagation," *Int. J. of Computer Vision*, vol. 98, no. 1, pp. 15–48, May 2012.

[12] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions," *Theory of Computing*, vol. 8, no. 19, 2012.

[13] V. Lavrenko, T. Rath, and R. Manmatha, "Holistic word recognition for handwritten historical documents," in *Proc. of the IEEE Workshop on Document and Image Analysis for Libraries DIAL'04*, 2004, pp. 278–287.

[14] T. Wilkinson and A. Brun, "Semantic and verbatim word spotting using deep neural networks," in *Int. Conf. on Frontiers in Handwriting Recognition*, 2016, pp. 307–312.

[15] T. Wilkinson, J. Lindström, and A. Brun, "Neural Ctrl-F: Segmentation-free query-by-string word spotting in handwritten manuscript collections," in *Int. Conf. on Computer Vision*, 2017.

[16] S. Sudholt, N. Gurjar, and G. Fink, "Learning deep representations for word spotting under weak supervision," 1 2018, arXiv:1712.00250v3.

[17] J. Fierrez *et al.*, "BiosecurID: a multimodal biometric database," *Pattern Analysis and Applications*, vol. 13, no. 2, pp. 235–246, May 2010.

[18] M. Ferrer, M. Diaz, C. Carmona-Duarte, and A. Morales, "A behavioral handwriting model for static and dynamic signature synthesis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1041–1053, 2016.

[19] P. Maergner, N. R. Howe, K. Riesen, R. Ingold, and A. Fischer, "Graph-based offline signature verification," 2019, arXiv:1906.10401.