Segmentation free spotting of Cuneiform using part structured models

Bartosz Bogacz
Interdisciplinary Center for Scientific Computing, Heidelberg University, Germany
bartosz.bogacz@iwr.uni-heidelberg.de

Nicholas Howe
Smith College
Northampton, MA, USA
nhowe@cs.smith.edu

Hubert Mara
Interdisciplinary Center for Scientific Computing, Heidelberg University, Germany
hubert.mara@iwr.uni-heidelberg.de

***Abstract*—Cuneiform scripts constitute an immense source of information about ancient history, dating back almost four thousand years. Documents were written by imprinting wedge-shaped impressions into wet clay tablets, and current scholarly practice typically transcribes the resulting markings by hand with ink on paper. This work develops algorithmic methods for cuneiform script, combining feature extraction for cuneiform wedges with prior work on segmentation-free word spotting using part-structured models. We adapt the inkball model used for word spotting to treat wedge features as individual parts arranged in a tree structure. The geometric relationship between query and target is measured by the energy necessary to deform the tree structure. We also introduce an optimizing method for wedge feature extraction based on optimally assigning tablet structuring elements to hypothesized wedge models. Finally, we evaluate the method on a real-world dataset, and show that it outperforms the state of the art in cuneiform character spotting.**

***Keywords*-Cuneiform script, Word spotting, Symbol spotting, Spatial pattern recognition, Part structured models, Optimal assignment, Feature extraction, Gaussian mixture models, Hidden Markov models**

I. Introduction

For more than three millenia in the ancient Middle East, scribes wrote documents using cuneiform script [1]. Characters were typically written on clay tablets by imprinting a rectangular stylus and leaving a wedge (*cuneus* in Latin) shaped trace, i.e., triangular markings. As clay was always cheaply and easily available, those capable of writing could produce a multitude of documents. Therefore, the content of cuneiform tablets ranges from mundane shopping lists to treaties between empires. Figure 1 shows an excerpt from one of our cuneiform tablets used in our source data.

The *Cuneiform Digital Library Initiative* [2] incorporates a number of projects aimed at cataloging cuneiform documents and making them available online as tracing, 2D image and sometimes as transliteration. However, the library cannot be searched using cuneiform characters as queries. Only the transliterations can be searched using Latin query words.

Our main contributions in this work are the optimal extraction of wedge features given a wedge model, the introduction of two new wedge feature representations and the adaptation of part structured models by Howe [3]. We combine these approaches to propose a new method for segmentation-free spotting of cuneiform characters.

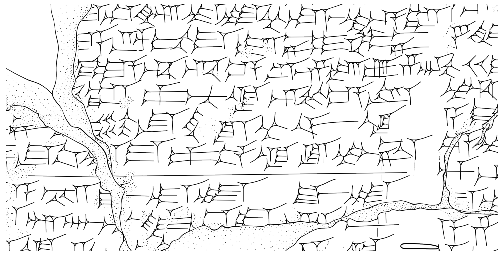


Figure 1. Excerpt of our source data. A SVG providing a born-digital cuneiform tablet.

Previous Work

In previous work we presented methods homogenizing various sources of cuneiform tablets [4], [5], such as photographs, 3D scans [6] of original tablets and transcriptions created with a vector graphics editor. We transformed each representation into a 12 dimensional feature vector of keypoints of a wedge-shaped impression. We evaluated different methods for comparing cuneiform characters and introduced a comparison method based on linear assignment [7], [8].

Previous work on part structured models [3] by Howe introduced a method for segmentation free word spotting in handwritten documents that models query words using overlapping balls of ink, with a representation of their 2D spatial layout based upon neighbor offsets. The same spatial representation can be used with many components besides inkballs; here we apply the method to our representation of cuneiform wedges to achieve an algorithm for segmentation free spotting of cuneiform characters. The representation, which we describe as a *structured wedge model*, allows computation of an approximation to the optimal one-to-one wedge assignment that is possible for segmented characters, except that the algorithm described in Section IV efficiently computes the optimal approximate match over an entire tablet at once.

Related work

Rothacker et. al [9] employ their word-spotting framework based on bag-of-features Hidden Markov Models (HMM)

[10] for segmentation-free character spotting of cuneiform. A 3D scan of a cuneiform tablet is transformed into a 2D representation using the curvature of the tablet surface. Then, the authors learn a HMM on a single example query word and spot cuneiform characters by decoding the learned HMM on a grid of possible positions on the cuneiform tablet.

HMMs are usually used for segmentation-free word-spotting [11]. They model the query word as sequence of vertical, sometimes overlapping, slices and a state machine with transition probabilities between the slices that are learned given the example query.

Another approach to matching words, also in historical context, is the work of Leydier et. al [12] that uses features with an elastic matching method. They use a concept of zones of interest, like vertical stem lines in latin script, to extract differential features, a histogram of oriented gradients, that are then matched using cohesive elastic matching.

Other work on word-spotting is not applicable since the very varied geometric position of the wedges induces a lot of vertical complexity. Word-spotting methods commonly assume that discriminative features are primarily encoded on the horizontal axis [13].

II. Extraction of Keypoints

In this work we derive our dataset from SVG files [5] providing born-digital cuneiform tablet transcriptions. This saves us the task of detecting the strokes used to draw wedges, these are directly enumerable from the source data, but we still have to identify which set of strokes denotes a cuneiform character.

Our input data is a set of spline paths that we call strokes. Wedges consist of up to six strokes, three for the triangular wedge-head and three for the wedge-arms. We detect wedge-heads by finding three pairwise intersecting strokes. Wedge-arms are any additional strokes that intersect any stroke of the wedge-head. This description of wedges is general enough to match all wedges on a born-digital cuneiform tablet. It also matches many more structures which are not proper wedges as seen in Figure 2. One difficulty is that cuneiform script is written very densely. Strokes from different wedges may intersect and create false positive wedge-heads or false positive wedge arms.

We approach this challenge by assuming that most strokes have been drawn to indicate proper wedges. We assign strokes to detected candidate wedges. Strokes cannot fill two roles at once. Either, *i)* a stroke is assigned to be one of the three sides of a wedge-head or *ii)* it is assigned to be one of the three wedge-arms. Strokes can also be left unused if drawn by error on a transcription. This task can be expressed as an optimal assignment problem, facilitating a computationally efficient solution.

Let $s_1 \dots s_n$ be the set of strokes, $w_1 \dots w_m$ the set of candidate wedges by finding walks of three pairwise intersecting strokes. Each stroke also has endpoints, $s = \{v_{s,1}, v_{s,2}\}$, these are two points that are most distant from each other lying on the boundary that describes the spline of the stroke. Each wedge-head has three points $w = \{v_{w,1}, v_{w,2}, v_{w,3}\}$, the centers of the intersection areas of the strokes describing this wedge, and a center $v^c_w = (v_{w,1} + v_{w,2} + v_{w,3})/3$.

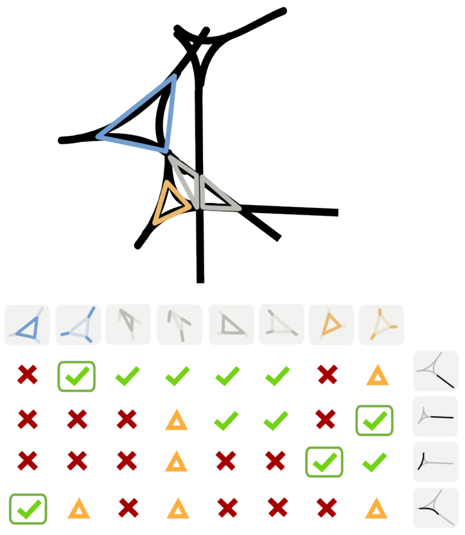


Figure 2. A simplified visual representation of the score matrix used for optimally assigning strokes to wedges. False positive candidate wedges are gray triangles. True positive candidate wedges are orange and blue. The green check mark signifies a match with high score, the orange triangle a match with low score and the red cross a match that is not allowed.

Let A be a set of allowed assignments $(s, w) \in A$ of strokes s to wedges w. An assignment is allowed if a stroke s touches one of the three pairwise intersecting strokes of a wedge-head w. For the two roles a stroke can fill, we define two score functions. The score function c^h computes the score of assigning a stroke to a candidate wedge-head as one of the three pairwise intersection strokes.

$$c^h_{sw} = \begin{cases} \text{area}(w) * \gamma, & \text{if } (s, w) \in A, \\ -1, & \text{else} \end{cases} \tag{1}$$

We preferentially assign strokes to wedge-heads. Different weighting parameters were tested $\gamma = (0.1, 5, 10, 20)$ with $\gamma = 10$ giving the best results.

When weighting the assignment of strokes as wedge-arms, we want to minimize the angle between the direction an edge of the wedge-head is pointing to, $\vec{v^w}$, and the direction of the stroke, $\vec{v^s}$. This is described by the score function c^a. Wedges typically have obtuse-angled wedge-arms, therefore, we penalize acute angles and disallow angles smaller than $45°$.

$$c^a_{sw} = \begin{cases} \alpha = \max \vec{v^s} \cdot \vec{v^w}, & \text{if } (s, w) \in A \wedge \alpha < 45° \\ -1, & \text{else} \end{cases} \tag{2}$$

We arrange the costs in a $m \times (n + m)$ matrix for optimization. The matrix has n rows for each of the strokes

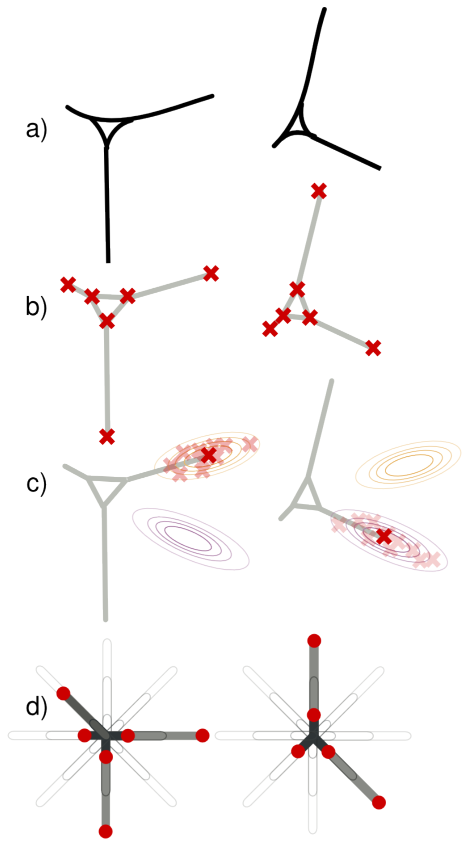


Figure 3. a) The original wedge as present on the cuneiform tablet. b) Keypoints model, each red cross marks one of the six extracted keypoints. c) Gaussian mixture model, contour lines for two Gaussian mixtures for one keypoint are shown. d) Template model, each red dot indicates a bit in the feature vector. The count of set bits varies depending on the shape of each character.

to be uniquely assigned and $m * 6$ columns for m wedges with 6 possible positions and orientations each. Additionally, there are m columns for strokes that have not been assigned to any candidate wedge. Figure 2 shows a simplified and illustrative version of this matrix.

$$C = \begin{array}{cccccc} & w_1 & \dots & w_m & s_1 \dots s_n & \\ \left(\right. & c^h \dots c^a & \dots & c^h \dots c^a & 1 \dots 0 & \left.\right) s_1 \\ & c^h \dots c^a & \dots & c^h \dots c^a & 0 \dots 0 & s_2 \\ & c^h \dots c^a & \dots & c^h \dots c^a & 0 \dots 0 & s_3 \\ & \vdots & \ddots & \vdots & \ddots & \vdots \\ & c^h \dots c^a & \dots & c^h \dots c^a & 0 \dots 1 & s_n \end{array} \quad (3)$$

Finding an optimal assignment using the Hungarian method [14] returns a subset of viable wedges, where each candidate wedge has been assigned at least three strokes in wedge-head roles.

III. Wedge Models

From the extracted subset of viable wedges, see Section II, we compute and evaluate the three following wedge models. Each of the models assumes a prototypical shape of a wedge. Due to the manner in which wedge-shaped impression are created, that is, by impressing a rectangular stylus in wet clay, all wedges share the shape of a triangle, the cut of a rectangle's edge.

Additionally, only few orientations and shapes of wedges were used for writing. Therefore, we aim to keep our models very simple and describe only the most necessary features of a wedge-shaped impression to discern its meaning.

A. *Key-point Model*

The key-point model derives directly from the way wedges are drawn in transcriptions. It models wedges using six two-dimensional points as shown in Figure 3. The first three points are the vertices of the three pairwise intersecting strokes forming the wedge-head. The last three points are endpoints of the wedge-arms attached to the respective wedge vertices. This model is described in detail in [8].

Let f^k be the feature vector of a wedge in the key-point model and $x_1, y_1 \dots x_6, y_6$ the respective key-points.

$$f^k = (\overbrace{x_1y_1\ x_2y_2\ x_3y_3}^{\text{Wedge-head}}\ \overbrace{x_4y_4\ x_5y_5\ x_6y_6}^{\text{Wedge-arms}}) \quad (4)$$

Wedges in the key-point model are compared using the Euclidean distance. The 12-dimensional features are not being normalized as the sizes and lengths carry meaning.

B. *Binary Templates Model*

While the key-point model accurately describes the shape of a wedge, it can be simplified since the exact angle of the wedge-arms and their exact length do not change the meaning of a wedge. The binary templates model is a compromise between an exact visual representation and a purely semantic representation of wedges. The wedge-head and the wedge-arms are matched against a set of templates and successful matches are recorded in the binary feature vector. This approach is similar to standardized cuneiform wedges used for writing in Assyriology and encoded in Unicode [15].

The templates are sets of vectors extending from the center of the wedge-head. These vectors point in eight different directions and have three different lengths to accommodate different scales of wedge-heads and wedge-arms. This arrangement is shown in Figure 3

Let t_k be a vector in the set of template vectors. The templates are enumerated by the set of angles $\{\alpha = 45°n \mid 0 \leq n < 8 \wedge n \in \mathbb{N}\}$ and template sizes $\{\hat{t} \in 1, 5, 10\}$. Using this discretization our binary vectors are 24-dimensional.

$$t_k = \left\{ \begin{pmatrix} \sin\alpha \\ \cos\alpha \end{pmatrix} \hat{t} \mid \alpha, \hat{t} \right\} \quad (5)$$

Then, the feature vector f^b is a binary vector defined as follows, where the notation $[i = j]$ indicates the Kronecker delta.

$$f_k^b = [\text{argmax}_k \;\; t_k \cdot f_l^k = k] \tag{6}$$

The resulting binary feature vectors are compared using the Euclidean distance. Testing with the cosine distance yielded worse results.

C. Gaussian Mixtures Model

We can imagine written wedges are imperfect manifestations of a small set of prototype wedges. We model this assumption by learning and expressing the feature vector of the keypoint model using a mixture of high-dimensional Gaussian distributions. Therefore, wedge feature-vectors in the Gaussian mixture model express the probabilities of being one of these prototype wedges. Figure 3 shows the modeling of wedges using Gaussian distributions.

Let μ be randomly initialized prior means and σ^2 prior variances of our data for K Gaussian mixtures. We model the prior distribution of the parameters θ as follows.

$$\theta_{i=1\ldots K} = \{\mu_{i=1\ldots K}, \sigma^2_{i=1\ldots K}\} \tag{7}$$

Then, the posterior distribution $p(\theta \mid x)$ given the feature vectors x is estimated using the expectation maximization method [16].

$$p(\theta \mid x) = \sum_{i=1}^{K} \phi_i \mathcal{N}(\mu_i, \Sigma_i) \tag{8}$$

Each component of the feature representing wedges in this model is the mean of one of the Gaussian distributions.

$$f_i^g = \mu_i \tag{9}$$

This feature vector of a wedge instance models the probability that this wedge belongs to one of the learned wedge classes. The feature vectors are compared using the Euclidean distance. In future work we will investigate the influence of other distance metrics.

IV. Part Structured Spotting

Because the identity of cuneiform symbols lie in the precise arrangements of their wedge components, part-structured models offer a very natural framework for representation. Models of this sort were developed for photographic object recognition before being applied to handwritten word spotting [17]. The inkball representation by Howe uses simple parts arranged in a geometric pattern to form characters and words. In this work we use individual wedges as the parts in the model, and represent their expected geometric relationship as default offsets in a tree structure.

Let $Q = \{q_1, ..., q_n\}$ represent a set of wedges that form some character or an other unit of interest, and let $\{v_1, ..., v_n\}$ represent their positions, taken as the mean of the three wedge head vertices. We assemble the parts into a tree structure by greedily forming pairwise links between disconnected units. Without loss of generality let q_1 be the wedge closest to the group's center of mass; this wedge becomes the root of the tree. Letting $q_{i\uparrow}$ indicate the parent of q_i in the tree structure, we define the default offset m_i for each child node. Note that m_1 and $q_{1\uparrow}$ are undefined, since q_1 has no parent. Figure 4 shows the model induced for one example query.

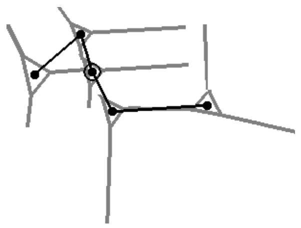


Figure 4. Tree model for a five-wedge query, with root at the center.

$$m_i = v_i - v_{i\uparrow} \tag{10}$$

Having identified the model's tree structure and default offsets, we define a deformation energy E_ξ for any proposed configuration of the model, $Z = \{z_1, ..., z_n\}$. This energy is a quadratic function of the difference between the observed offsets and the model default, and is invariant under rigid translations.

$$E_\xi(Z) = \sum_{2}^{n} \|(z_i - z_{i\uparrow}) - m_i\|^2 \tag{11}$$

Given a cuneiform document, word spotting attempts to identify locations that are likely matches to the query model. More precisely, if $T = \{t_1, ..., t_N\}$ is the set of wedges in the target, with positions $U = \{u_1, ..., u_N\}$, then we seek locations x where the following energy function reaches a minimum and lies below some target threshold.

$$\mathcal{E}(x) = \min_{Z|z_1=x} [E_\xi(Z) + E_\omega(Z, Q, T, U)] \tag{12}$$

The second term in this expression measures the proximity of model wedges to suitable target wedges under the proposed configuration. A parameter α trades off between spatial proximity and wedge match quality D, as computed using one of the methods from the previous section.

$$E_\omega(Z, Q, T, U) = \sum_{i=1}^{n} E_\omega(z_i, q_i, T, U) \tag{13}$$

$$E_\omega(z_i, q_i, T, U) = \min_{j=1}^{N} \left[\|z_i - u_j\|^2 + \alpha D(q_i, t_j)\right] \tag{14}$$

The energy function in Equation 12 can be minimized via an efficient dynamic programming algorithm, as in prior work on part-structured models [18]. We first write an expression for the minimum energy of arbitrary subtrees of the model, where $Z_{i\downarrow}$ represents the positions of wedge q_i and all its descendants.

$$\mathcal{E}_i(x) = \min_{Z_{i\downarrow}|z_i=x} [E_\xi(Z_{i\downarrow}) + E_\omega(Z_{i\downarrow}, Q_{i\downarrow}, T, U)] \quad (15)$$

This in turn can be rewritten recursively as the wedge match at the root plus the minimum energy over all possible child configurations, as adjusted by the model offsets.

$$\mathcal{E}_i(x) = E_\omega(x, q_i, T, U) + \min_{Z_{i\downarrow}|z_i=x} \mathcal{E}_{\downarrow i}(x) \quad (16)$$

$$\min_{Z_{i\downarrow}|z_i=x} \mathcal{E}_{\downarrow i}(x) = \sum_{j|j\uparrow=i} \Gamma\left(\mathcal{E}_j(x - m_j)\right) \quad (17)$$

Here Γ denotes the generalized distance transform (GDT) [17], which performs the minimization over the deformation term. At the leaves of the model, the child energy contribution $\mathcal{E}_{\downarrow i}$ is zero and the energy function $\mathcal{E}_i$ can be computed simply via a GDT, with the function values with respect to x represented on a discrete grid. Moving up the tree, parent node energies $\mathcal{E}_i$ can be computed using the results at the leaves, translated by the offset m_j and passed through another generalized distance transform before being added up. This process eventually yields the energy of the entire model over the grid of possible root positions. Strong local minima on this grid are the locations where the model matches well. We place bounding boxes around these minima and perform a non-minimum suppression of the results.

V. Evaluation

We evaluate our methods on a dataset of two cuneiform tablets line traced by professional Assyriologists using a graphics editor. These tablets contain around 500 identifiable cuneiform characters. The tablets are only incompletely manually labeled and segmented. This precludes a precise evaluation of the performance of the presented methods on the given dataset and offers us no possibility to exactly analyze the methods on their recall performance.

However, our evaluation scheme makes the relative differences in retrieval performance still valid and allows for a comparison of the methods. We perform retrieval queries by example using the set of segmented cuneiform characters. Given the size of our dataset, we assume that each character class has in average 30 instances. Perfect recall is achieved when 30 instances are retrieved. An expert then decides for each returned result whether it belongs to the class of the query and tags it with either true positive or false positive.

We evaluate our method against the work of Rothacker et al. on word-spotting on Latin script [10]. Since we use vectorized transcriptions of cuneiform tablets as data, we

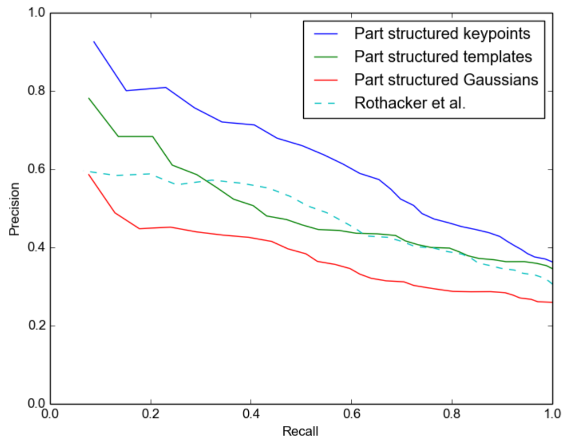


Figure 5. Precision recall graph of the four evaluated methods.

cannot employ their work on word-spotting on cuneiform tablets for comparison [9]. We have no 3D data or curvature data available for our dataset.

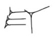
To make our data available to their bag-of-features word spotting framework, we first rasterize the vectorized dataset to raster images. The size of the raster images is chosen so that their choice of slice parameters and sliding windows advancement is optimal for our data. That is, we chose 40 pixel sized structuring elements for the dense SIFT [19] transformation with a 5 pixel wide regular grid. The query examples are taken from the incompletely segmented document and sliced with 5 pixel wide horizontal slices advancing 2 pixels. We use the same HMM topology as presented in their work.

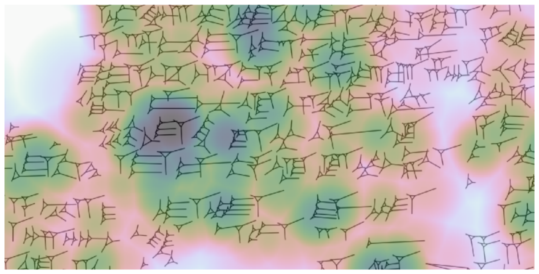
VI. Results

Figure 6 shows the score and energy distribution for the respective method for an exemplary query. The results of our evaluation are shown in Figure 5. The Gaussian mixture model yielded the least satisfactory results. Closer inspection of the inferred classes showed that they did not represent the space of different wedges well. We are currently investigating better clustering approaches for wedges.

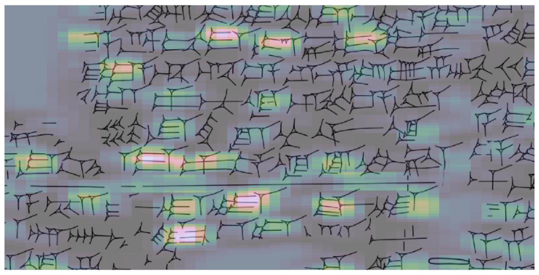
Binary template vectors offer more flexibility, nonetheless, they do not perform as well as the keypoint model. We attribute this to the rigidity of the defined templates. If many semantically different wedges fall into the same template, they no longer can be differentiated. A more flexible template model, more angles and more sizes, would lead to high-dimensional feature vectors that are hard to compare.

The keypoint model performs best and outperforms the approach presented by Rothacker et. al. The keypoints model uses a advantageous description of wedges when used in combination with the adapted part structured spotting. It too models two dimensional points that are compared using the Euclidean distance.

Query character: 



(a)



(b)

Figure 6. Result scores of both methods overlaid on the excerpt from Figure 1 of one of our source documents. (a) Decoding scores for the method of Rothacker et al. (b) Energy distribution of our approach using the keypoint model.

Summary & Outlook

In this work we presented a novel method of segmentation-free spotting of cuneiform characters. We introduced three different feature representations of wedges for evaluation. Spotting is performed using a part structured approach where characters are modeled by a tree structure of connected wedges, an adaptation and generalization of our previous work. Compared to the recent approach presented by Rothacker et al. [9] our best performing approach requires no learning, no prior examples from the dataset and works with only one parameter.

Currently, our optimizing extraction of wedge key-points uses an a-priori model of wedges for scoring assignments and modeling wedges. In future work, we want to build upon our Gaussian mixture model wedge features to provide a learned wedge model for scoring competing candidate wedges features for extraction. Additionally, we aim to extract wedges from the transcriptions available in the Cuneiform Digital Library Initiative database to provide a large scale search function for cuneiform characters.

References

[1] W. von Soden, *The ancient Orient: an introduction to the study of the ancient Near East*. Wm. B. Eerdmans Publishing Co., 1994.

[2] J. Kantel, P. Damerow, S. Köhler, and C. Tsouparopoulou, "3D-Scans von Keilschrifttafeln - ein Werkstattbericht," in *26. DV-Treffen der Max-Planck-Institute*. Ges. für wiss. Datenver., 2010.

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

[4] B. Bogacz, J. Massa, and H. Mara, "Homogenization of 2d & 3d document formats for cuneiform script analysis," in *Workshop on Hist. Doc. Imaging and Processing*, 2015.

[5] J. Massa, B. Bogacz, S. Krömker, and H. Mara, "Cuneiform detection in vectorized raster images," in *Comp. Vis. Winter Workshop*, 2016.

[6] H. Mara and S. Krömker, "Vectorization of 3d-characters by integral invariant filtering of high-resolution triangular meshes," in *Int. Conf. on Doc. Analysis and Rec.*, 2013.

[7] B. Bogacz, M. Gertz, and H. Mara, "Cuneiform character similarity using graph representations," in *Comp. Vis. Winter Workshop*, 2015.

[8] ——, "Character retrieval of vectorized cuneiform script," in *Int. Conf. on Doc. Analysis and Rec.*, 2015.

[9] L. Rothacker, D. Fisseler, G. G. W. Müller, F. Weichert, and G. A. Fink, "Retrieving cuneiform structures in a segmentation-free word spotting framework," in *Workshop on Hist. Doc. Imaging and Processing*, 2015.

[10] L. Rothacker, M. Rusinol, and G. A. Fink, "Bag-of-features HMMs for segmentation-free word spotting in handwritten documents," in *Int. Conf. on Doc. Analysis and Rec.*, 2013.

[11] J. A. Rodríguez-Serrano and F. Perronnin, "Handwritten word-spotting using hidden markov models and universal vocabularies," *Pat. Rec.*, 2009.

[12] Y. Leydier, F. Lebourgeois, and H. Emptoz, "Text search for medieval manuscript images," *Pat. Rec.*, 2007.

[13] T. M. Rath and R. Manmatha, "Word spotting for historical documents," in *Int. Conf. on Doc. Analysis and Rec.*, 2007.

[14] J. Munkres, "Algorithms for the assignment and transportation problems," *Soc. for Ind. and App. Math.*, 1957.

[15] Unicode Consortium, *The Unicode Standard, Version 5.0*. Addison-Wesley, 2006.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Jour. of the Royal Stat. Soc.*, 1977.

[17] P. Felzenszwalb and D. Huttenlocher, "Pictorial structures for object recognition," *Int. Journal on Comp. Vis.*, 2005.

[18] N. Howe, "Inkball models for character localization and out-of-vocabulary word spotting," in *Int. Conf. on Doc. Analysis and Rec.*, 2015.

[19] M. Bicego, A. Lagorio, E. Grosso, and M. Tistarelli, "On the use of sift features for face authentication," in *Comp. Vis. and Pat. Rec. Workshops*, 2006.