Intrinsic Mean for Semi-metrical Shape Retrieval Via Graph Cuts

Frank R. Schmidt[1], Eno Töppe[1], Daniel Cremers[1], and Yuri Boykov[2]

[1] Department of Computer Science
University of Bonn
Römerstr. 164, 53117 Bonn, Germany
{schmidtf,toeppe,dcremers}@cs.uni-bonn.de
[2] Computer Science Department
University of Western Ontario
London, ON, Canada
yuri@csd.uwo.ca

Abstract. We address the problem of describing the mean object for a set of planar shapes in the case that the considered dissimilarity measures are semi-metrics, i.e. in the case that the triangle inequality is generally not fulfilled. To this end, a matching of two planar shapes is computed by cutting an appropriately defined graph the edge weights of which encode the local similarity of respective contour parts on either shape. The cost of the minimum cut can be interpreted as a semi-metric on the space of planar shapes. Subsequently, we introduce the notion of a mean shape for the case of semi-metrics and show that this allows to perform a shape retrieval which mimics human notions of shape similarity.

1 Introduction

To decide whether two given objects are similar to one another and to cluster subsets of similar objects is an important challenge in Computer Vision. In the last years, this problem has been tackled for shapes by defining *dissimilarity measures* [6]. These measures proved themselves as useful in the context of shape recognition, clustering, classification and statistical modeling [5,9]. In particular, the study of metrics has been very promising to generalize statistical concepts like average objects or standard deviations [9]. But not every useful dissimilarity measure is a metric [3,2]. Indeed, most of them are semi-metrics, i.e. they violate the triangle inequality. But how can the statistical concept of a mean shape be defined, if there is no metric at hand? Since an embedding into an Euclidean space is not possible, we will approach the question of defining a template for a given collection of shapes only by studying the given semi-metric. The semi-metric that we like to study exemplarily is an energy functional that arises in the context of shape matching.

1.1 Dissimilarity Measures for Shapes

In order to abstract from location and rotation, the term *shape* refers to a complete class $\mathcal{C}$ of closed curves $c : \mathbb{S}^1 \to \mathbb{R}^2$ embedded in the plane $\mathbb{R}^2$. This

F.A. Hamprecht, C. Schnörr, and B. Jähne (Eds.): DAGM 2007, LNCS 4713, pp. 446–455, 2007.

class shall be invariant under rigid body transformations, i.e. translations and rotations. The set of all these shapes form a Riemannian manifold [8]. Any continuous transformation from one shape into another can be represented by a path in this curved space and geodesics, i.e. paths of minimal length, define a metric on these spaces [8,5]. Since only such paths are allowed that are placed *inside* the manifold, we are talking of *intrinsic paths*.

From a practical point of view, we often like to compare two different planar shapes. This task of matching two different shapes has been approached by minimizing a given functional [7]. Such a matching functional can be used as dissimilarity measure, just like the intrinsic length presented above.

1.2 Dissimilarity Measures and Statistics

Since the classical mean for a collection of objects is only defined if these objects are elements of a vector space, the generalization of a mean object has been of broad interest. In the case of manifolds, the Karcher mean [4] was introduced as minimum of an energy function and the concept of this metric-oriented mean has been applied to shape spaces [5]. Since distance functions that are robust to outliers will typically violate the triangle inequality [3], we are interested in such semi-metrical distance functions. Semi-metrics have been already considered for segmentation tasks [7,2]. But to the best of our knowledge, there are no statistical approaches for semi-metrics since there is no canonical definition of a mean object. In this paper, we will overcome this limitation by presenting a generalization of the Karcher mean for semi-metrics to which we will refer as *shape template*. These templates will be used to describe the *center of a cluster* and to perform the task of retrieving similar shapes from a given database. Because the definition of a mean *within* a manifold does not use the exterior vector space, such a template provides an *intrinsic mean*.

This paper is organized as follows. In Section 2, we propose an approximation scheme for matching an arbitrary collection of shapes. In Section 3, the result of this synchronistic shape matching will be used to construct a template for a given shape cluster. This cluster template will be used to retrieve similar shapes from a database. In Section 4, we analyze the runtime of the given methods and show some retrieving results for the well known LEMS database. In particular, we experimentally verify that this proposed intrinsic mean gives rise to superior retrieval rates. In Section 5, we will provide a conclusion of our work.

2 Shape Matching

In this section, we will present a method to solve the shape matching task for more than two given shapes. To this end, we first present the shape matching method developed in [10] for *two* different shapes by cutting a specific planar graph. Subsequently, we consider the more general problem of simultaneously matching *multiple* shapes and propose an efficient approximative solution.

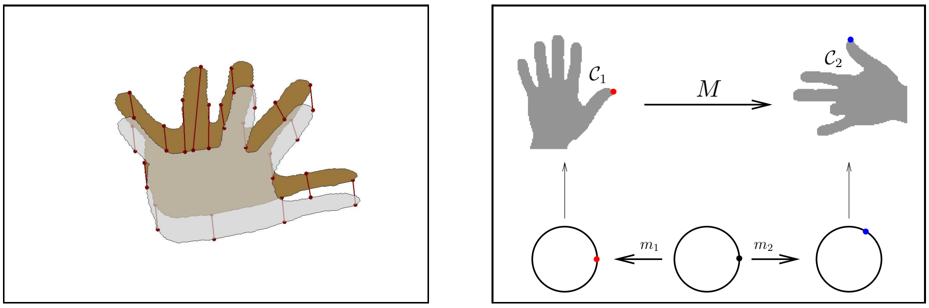

Fig. 1. Matching. *Left hand side:* Matching two shapes amounts to computing a correspondence between pairs of points on both shapes. *Right hand side:* Instead of looking for a mapping $M : \mathcal{C}_1 \to \mathcal{C}_2$, a matching $m = (m_1, m_2) : \mathbb{S}^1 \to \mathbb{S}^1 \times \mathbb{S}^1$ is defined on the parameterization domains.

2.1 Matching of Two Shapes Via Graph Cut

As a shape $\mathcal{C}$ we understand the class of closed curves $c : \mathbb{S}^1 \to \mathbb{R}^2$ that is invariant under rigid body motions. These shapes form the shape space $\mathcal{S}$. Since it is well known that the curvature $\kappa : \mathbb{S}^1 \to \mathbb{R}$ is a unique description of every shape $\mathcal{C}$, the shape space can be described in terms of these curvature functions[1]:

$$\mathcal{S} := \left\{ \kappa : \mathbb{S}^1 \to \mathbb{R} \,\middle|\, \int_{\mathbb{S}^1} \exp\left[i \int_0^t \kappa(\tau)\mathrm{d}\tau \right] \mathrm{dt} = 0 \right\} \tag{1}$$

By the definition of $\mathcal{S}$, all rigid body motions are eliminated and we can focus on the non-rigid shape transformations. To decide whether two shapes are similar, we want to detect local transformations like stretching and contraction. Therefore, we are looking for a correspondence mapping that maps the points of one shape to the corresponding points on the second shape. Since the points of a shape define an arbitrary subset of the plane $\mathbb{R}^2$, it is much simpler to find the correspondence directly on the parameterization domain $\mathbb{S}^1$ – see also Figure 1. To ensure that a matching covers both parameterization domains exactly once, a matching consists of two orientation preserving bijective mappings $m_1, m_2 : \mathbb{S}^1 \to \mathbb{S}^1$ that simultaneously sample the points of both parameterization domains. The space of all these sampling mappings will be called $\mathrm{Diff}^+(\mathbb{S}^1)$. Given two shapes $\mathcal{C}_1$ and $\mathcal{C}_2$ with their curvature functions κ_1 resp. κ_2, we are interested in a matching $m \in \mathrm{Diff}^+(\mathbb{S}^1) \times \mathrm{Diff}^+(\mathbb{S}^1)$ that minimizes the following functional

$$E_{\kappa_1}^{\kappa_2}(m) = \int_{\mathbb{S}^1} [(\kappa_1 \circ m_1 - \kappa_2 \circ m_2)(s)]^2 \mathrm{d}m(s). \tag{2}$$

In this functional, the *data term* $(\kappa_1 - \kappa_2)^2$ is therefore integrated along the *matching* $s \mapsto (m_1(s), m_2(s))$. Since $\mathrm{d}m(s) = \|m'(s)\|\,\mathrm{d}s$ holds, the *smoothness*

[1] For a detailed study of this manifold, we are referring to [5].

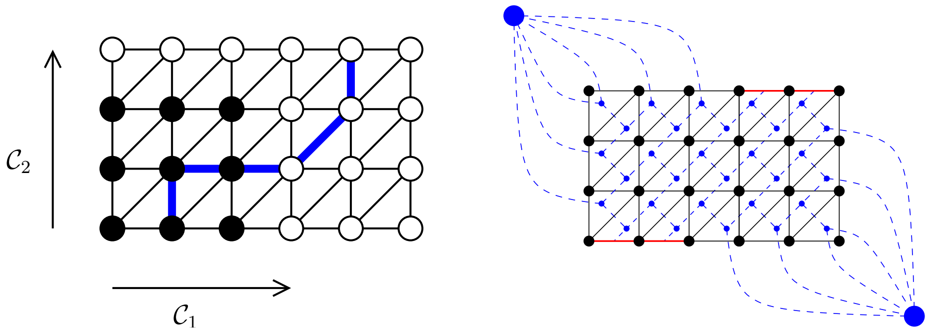

Fig. 2. Left hand side: Sampling two shapes $\mathcal{C}_1$ and $\mathcal{C}_2$ by N points, we receive a squared graph (filled vertices). If we copy the bottom line onto the top and the received construction to the right (blank vertices), every matching can be represented by a path from the matching vertex $(a, 0)$ to the vertex $(a + N, N)$. By identifying $(a, 0)$ with $(a + N, N)$, every matching becomes a shortest cycle. **Right hand side:** Every cycle in G describes a cut on the dual graph G^* (dashed edges). A minimal graph cut in G^* has therefore the same weight as the shortest cycle in G.

term m' is directly coupled to the data term. Using (2), a distance function on the shape space $\mathcal{S}$ can be defined as follows.

Definition 1 (Shape Distance). *Given two shapes* $\mathcal{C}_1, \mathcal{C}_2 \in \mathcal{S}$ *with their curvature functions* $\kappa_1 : \mathbb{S}^1 \to \mathbb{R}$ *and* $\kappa_2 : \mathbb{S}^1 \to \mathbb{R}$ *resp., we will call*

$$\mathrm{dist}(\mathcal{C}_1, \mathcal{C}_2) := \min_{m \in \mathrm{Diff}^+(\mathbb{S}^1)^2} E_{\kappa_1}^{\kappa_2}(m)^{\frac{1}{2}} \tag{3}$$

the distance *of these shapes. Every matching fulfilling this minimum will be called* a minimal matching *of* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*.*

It is well known that the calculation of this semi-metrical distance can be done by finding the shortest path in a graph. In Figure 2, the appropriate graph $G = (V, E, w)$ is sketched. The vertices $(x_1, x_2) \in V$ represent a possible match between $c_1(x_1)$ and $c_2(x_2)$ and the data term of this vertex is $(\kappa_1(x_1) - \kappa_2(x_2))^2$. Therefore, the weight w of any edge $(x_1, x_2) \to (y_1, y_2)$ carries the value of the path integral along this edge. If we sample each shape by N points, any path from $(a, 0)$ to $(a + N, N)$ describes a matching. Hence, $\mathrm{dist}(\mathcal{C}_1, \mathcal{C}_2)$ can be calculated by finding an initial correspondence $(a, 0)$ and afterwards the path of minimal weighted length from $(a, 0)$ to $(a + N, N)$. Given an initial correspondence $(a, 0)$, the classical way to calculate the shortest path length is the Dynamic Time Warping (DTW) method which takes linear time in the size of the given graph. Testing all initial correspondences leads therefore to a runtime of $\mathcal{O}(N^3)$ [3,7].

On the other hand, if we identify any possible initial matching $(a, 0)$ with $(a+N, N)$, the graph becomes a cylinder and the formerly shortest path describes a shortest cycle on this cylindrical graph. Whitney proved in [12] that for any planar graph G, there is a one-to-one relationship between cycles on G and cuts in the dual graph G^*. Therefore, the value of a minimal edge cut will be $\mathrm{dist}(\mathcal{C}_1, \mathcal{C}_2)^2$. Mathematically, this can be summarized in the following theorem.

Theorem 1. *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two shapes with their curvature functions $\kappa_1 : \mathbb{S}^1 \to \mathbb{R}$ and $\kappa_2 : \mathbb{S}^1 \to \mathbb{R}$ resp. Then, the following equation holds*

$$\mathrm{dist}(\mathcal{C}_1, \mathcal{C}_2)^2 = \min_{\substack{X^* \subset E^* \\ X^* \textit{ edge set of} \\ \textit{a graph cut in } G^*}} \sum_{e^* \in X^*} w(e) \tag{4}$$

Proof. For a detailed proof, we are referring to [10]. □

To calculate the graph cut, we use the algorithm presented in [1]. In Section 4, we will analyze the runtime of this method in comparison to the shortest path method. We will demonstrate that for similar shapes the graph cut method is favorable over the shortest path method.

2.2 Synchronistic Shape Matching

After introducing the shape matching of two shapes, the question arises how a whole collection of shapes can be set in correspondence. Since any shape carries some artifacts according to the chosen discretization, a matching between two shapes could emphasize these artifacts and hence provide a matching that does not coincide with the human notion of point correspondence. If a synchronistic matching of a whole collection of shapes is to be achieved, the noisy artifacts of one shape shall be inhibited by the other shapes. To provide a synchronistic shape matching is therefore a challenging task and the goal of this subsection.

Analogously to (2), we define a functional for the *synchronistic shape matching*. Given a collection $\mathcal{T} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ of n shapes, a matching m consists of n different mappings $m_i \in \mathrm{Diff}^+(\mathbb{S}^1)$ that minimize the pairwise curvature differences. Let $\kappa_1, \ldots, \kappa_n$ be the curvature functions of the shapes $\mathcal{C}_1, \ldots, \mathcal{C}_n$ resp. Then, we like to minimize the following functional.

$$E_{\mathcal{T}}(m) = \int_{\mathbb{S}^1} \sum_{i,j=1}^{n} [(\kappa_i \circ m_i - \kappa_j \circ m_j)(s)]^2 \mathrm{dm}(s) \tag{5}$$

To calculate any synchronistic shape matching is a computationally challenging task. Analogously to Section 2.1, we can find the matching mapping $m \in \mathrm{Diff}^+(\mathbb{S}^1)^n$ by searching for a closed circle in a n-dimensional grid. If we use a sampling rate of 100 points for every shape, we would need ten billion grid points to match a small collection of five shapes. Since this is too expensive, we are interested in an approximation scheme.

Given a matching mapping $m = (m_1, \ldots, m_n)$, the mapping $(m_i, m_j) \in \mathrm{Diff}^+(\mathbb{S}^1)^2$ describes a matching between the two shapes $\mathcal{C}_i, \mathcal{C}_j \in \mathcal{T}$. Since $m_{i,j} := (m_i, m_j)$ does not necessarily minimize (2), we can reformulate the functional (5) as a compromise between (2) and the property that $m_{i,j}$ and

Fig. 3. Synchronistic Shape Matching. If we compare the first two shapes, we receive a matching that cannot detect the two missing fingers. If we add a third shape (middle), the matching can be improved using (6). The last two images represent the matching according to $\gamma = 10^{-6}$ and $\gamma = 10^{-4}$. Note that the location of the sixth and eighth shape point have been changed.

$m_{j,k}$ describe $m_{i,k}$. As abbreviation, we like to introduce $\widetilde{m}_{i,j} := m^2_{i,j} \circ (m^1_{i,j})^{-1}$ for a given pairwise matching $m_{i,j} = (m^1_{i,j}, m^2_{i,j})$. With this notation, $m_{i,j}$ becomes the graph of $\widetilde{m}_{i,j}$ and the described compromise can be formulated as the following functional.

$$E_{\mathcal{T}}\left((m_{i,j})_{i,j=1,\dots,n}\right) = \sum_{i,j=1}^{n} E^{\kappa_j}_{\kappa_i}(m_{i,j}) + \gamma \cdot \sum_{i,j,k=1}^{n} \int_{\mathbb{S}^1} \|(\widetilde{m}_{i,j} - \widetilde{m}_{k,j} \circ \widetilde{m}_{i,k})(s)\|^2 \,\mathrm{d}m_{i,j}(s), \tag{6}$$

Note that (6) is a major relaxation of (5). Instead of the n matching functions $m_1, \dots, m_n$, we are dealing now with the n^2 binary matching functions $m_{i,j} = (m^1_{i,j}, m^2_{i,j}) \in \mathrm{Diff}^+(\mathbb{S}^1)^2$. To solve (6), we start with the matchings $m_{i,j}$ that minimize (2). Then iteratively, every matching $m_{i,j}$ is improved according to (6) using the predefined $m_{i,j}$. Since we assume that all shapes are similar, we use the proposed graph cut method to solve the binary matchings during the whole iteration process. This is done according to the result in Section 4 that for similar shapes the graph cut method outruns the DTW method. In Figure 3, we see how the synchronistic shape matching improves a given matching.

3 Shape Classification Given a Synchronistic Matching

In this section, we will present a way to describe a shape cluster using the synchronistic shape matching of Section 2.2. For this purpose, we present a generalization of the Karcher mean [4] for the space $\mathcal{S}$ in respect to the semi-metric $\mathrm{dist}(\cdot,\cdot)$. Afterwards, we show that this template improves the retrieval result considerably.

3.1 Cluster Template

Given a set $\mathcal{T} = \{\mathcal{C}_1, \dots, \mathcal{C}_n\}$ of training shapes, we want to tackle the problem of finding the center of this shape cluster. Thus, we are looking for a *template*

$\mathcal{C}_\mathcal{T}$ that is close to all shapes in $\mathcal{T}$. The next definition describes this task as a minimization problem.

Definition 2 (Template). *Given a collection $\mathcal{T} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ of shapes with the curvature function $\kappa_1, \ldots, \kappa_n$ resp. Moreover, let $(m_{i,j})_{i,j=1,\ldots,n}$ be a minimum of* (6)*. Then a minimum $\kappa_\mathcal{T}$ of the functional*

$$\kappa \mapsto \sum_{i=1}^{n} \int_{\mathbb{S}^1} [(\kappa \circ m^1_{1,i} - \kappa_i \circ m^2_{1,i})(s)]^2 \mathrm{dm}_{1,i}(s). \tag{7}$$

is called a template *of the cluster $\mathcal{T}$.*

Note that we used the synchronistic shape matching for a realignment of all shapes in the training set $\mathcal{T}$. Therefore, the Euler-Lagrange equation of (7) is linear and the template is therefore easy to calculate. While the intrinsic mean constructed above will generally not correspond to a meaningful shape, we shall demonstrate that it form an excellent basis for shape retrieval.

3.2 Cluster-Based Retrieval

In the Section 3.1, we presented a method to find a template $\mathcal{C}_\mathcal{T}$ for a given cluster $\mathcal{T}$. Now, we want to retrieve from a database those shapes that are similar to the shapes of $\mathcal{T}$. Therefore, we have to decide if an arbitrary shape fits to a given cluster $\mathcal{T}$. We are doing this by calculating the distance of a given shape $\mathcal{C}$ to the template $\mathcal{C}_\mathcal{T}$. If this distance is small enough, we classify $\mathcal{C}$ as an element of $\mathcal{T}$. If different clusters $\mathcal{T}_1, \ldots, \mathcal{T}_k$ are at hand, we choose the following algorithm:

1. Given a shape $\mathcal{C} \in \mathcal{S}$, calculate for every $i = 1, \ldots, k$ the distances $d_i = \mathrm{dist}(\mathcal{C}, \mathcal{C}_{\mathcal{T}_i})$.
2. Find $i_0 := \arg\min_i d_i$.
3. If $d_{i_0} < \lambda_{i_0}$, classify $\mathcal{C}$ as a shape of cluster $\mathcal{T}_{i_0}$.
4. Otherwise, state that $\mathcal{C}$ cannot be classified properly.

The choice of λ_i for a given cluster $\mathcal{T}_i$ is important for the appropriate description of the class $\mathcal{T}_i$. In fact, we may choose λ_i differently for different clusters. In Section 4, we will present a representative example to show how well this classification method works.

4 Experimental Results

In this section, we will analyze the proposed methods on real shapes. For this purpose, we use the shapes that are provided by the LEMS laboratory of the Brown University [11] and apply the curvature descriptor introduced in [10]. In detail, we analyze the runtime of the graph cut algorithm in comparison to the classical method using Dynamic Time Warping (DTW). Afterwards, we demonstrate how well shapes can be retrieved with the help of the introduced cluster template.

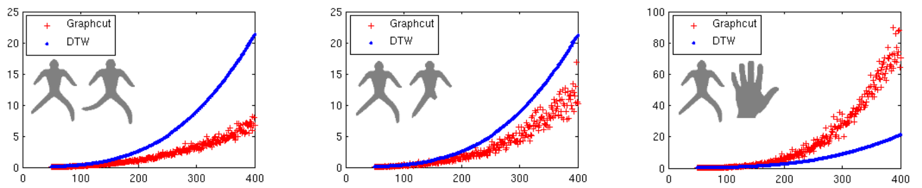

Fig. 4. Runtime comparison. The runtime of the DTW and the proposed graph cut method are plotted against the sampling rate of both shapes. In the first two cases the graph cut matching works faster than the classical DTW approach. The plots indicate that the graph cut method should be favored over DTW if one expects similar shapes. Otherwise, one should benefit of the granted constant DTW runtime.

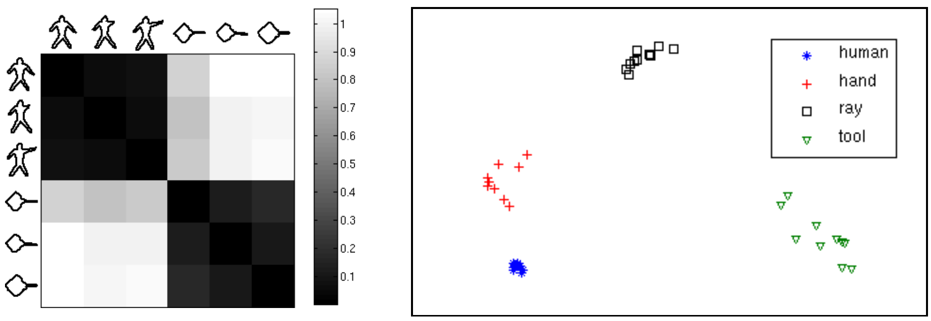

Fig. 5. Clustering. On the left hand side, the pairwise dissimilarity of six given shapes according to dist$(\cdot, \cdot)$ are color-coded. On the right hand side, 40 shapes are projected into the Euclidean plane based on their pairwise distance. In general, this projection will not preserve pairwise distances since dist$(\cdot, \cdot)$ is not a metric. But even this approximation indicates that the distance function incorporates the human notion of shape similarity.

4.1 Runtime Comparison

The bottleneck of the classical DTW method is the search for an initial correspondence. If a complete search over all possible initial matchings is done, the runtime is always $\mathcal{O}(N^3)$ for a fixed sampling rate of N points per shape. On the other hand, the runtime of the graph cut method depends very much on the input data. Figure 4 demonstrates the runtime of the graph cut method in respect to the DTW method. The plots indicate that the matching of two shapes is very fast with the graph cut method, if these shapes are similar to one another. One the other hand for distinctively different shapes, the classical DTW method outruns the graph cut method. Therefore, we used the DTW method to cluster the whole database. But for the template calculation of a given cluster, we always used the proposed graph cut method.

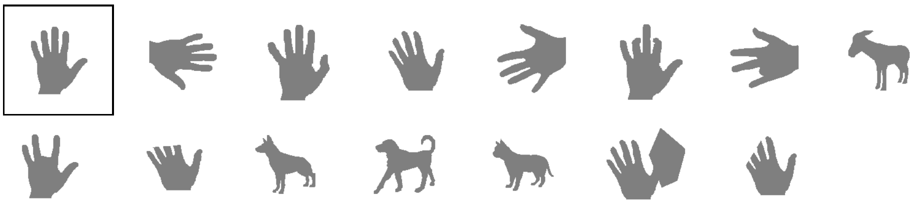

Fig. 6. Shape-based Retrieval. Using one of the training shapes (boxed) as a representative for retrieval gives an unsatisfactory retrieval performance: In order to extract all hands from the data base one needs to determine the 15 best hits.

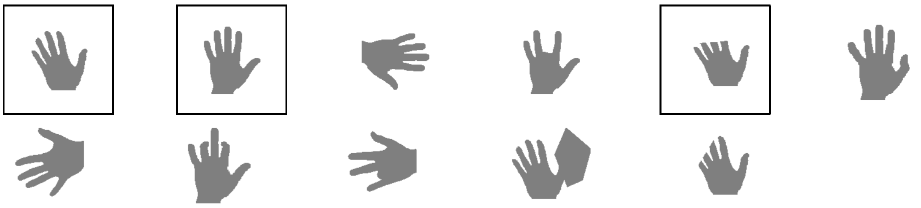

Fig. 7. Template-based Retrieval. We define a template based on the framed shapes. The eleven best hits correspond to all hand shapes in the database. This shows that the distance to the proposed intrinsic mean provides for superior retrieval performance than using an individual template as done in Figure 6.

4.2 Proposed Retrieval Method

The presented retrieval method works in two phases – the learning phase and the retrieval phase. In the learning phase, the shapes that define a shape class were matched via the proposed synchronistic shape matching and thus define a template. During the second phase, the distance between the calculated templates and the unknown shapes from a database are calculated. According to this distance, the unknown shapes can be classified. On the left hand side of Figure 5, we see an example of how well the dissimilarity measure function $\text{dist}(\cdot,\cdot)$ divides the shape database into appropriate clusters. Nonetheless, the question how the number of cluster can be estimated is still unsolved. Therefore, we applied a 4-means run for a subset of the LEMS database that is projected via multidimensional scaling on the right hand side of Figure 5. Since Figure 5 illustrates that the class *ray* and the class *human* are very easy in respect to the given database, we want to analyze the retrieval for the class *hand*. Figure 6 shows the classical retrieval according to one selected shape. We need 15 shapes to find all eleven hand shapes. Since the database consists of only eleven hand shapes, it is remarkable that the first eleven hits according to the template-based (cf. fig. 7) retrieval are in fact these shapes. Note that the learned shapes are not necessarily the best hits. Due to the semi-metric, the template can be closer to some shapes than to others.

5 Conclusion

In this paper, we introduced a generalization of the Karcher mean for semi-metrical spaces. To this end, we approximate the computationally infeasible simultaneous matching of n shapes by a consistent iteration of pairwise matchings. The latter problem can be solved by computing the minimal cut through a graph whose nodes encode the local similarity of respective contour parts on each shape. The presented experiments indicate that for the matching of similar shapes, this graph cut approach provides a speed-up factor up to 4 relatively to the classical method using Dynamic Time Warping (DTW). Just as humans have no problem in finding the correspondence on two similar shapes, the proposed method finds an initial match and the complete correspondence simultaneously and faster than the usual approach via DTW.

In a shape retrieval experiment on the LEMS database, we demonstrated that the proposed intrinsic mean for semi-metrical shape spaces provides for superior retrieval performance than individual shape instances do.

References

1. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE PAMI 26(9), 1124–1137 (2004)
2. Cremers, D., Soatto, S.: A pseudo-distance for shape priors in level set segmentation. In: Paragios, N. (ed.) IEEE 2nd Int. Workshop on Variational, Geometric and Level Set Methods, Nice, pp. 169–176 (2003)
3. Gdalyahu, Y., Weinshall, D.: Flexible syntactic matching of curves and its application to automatic hierarchical classication of silhouettes. IEEE PAMI 21(12), 1312–1328 (1999)
4. Karcher, H.: Riemann center of mass and mollifier smoothing. Comm. Pure and Applied Math. 30, 509–541 (1977)
5. Klassen, E., Srivastava, A., Mio, W., Joshi, S.H.: Analysis of planar shapes using geodesic paths on shape spaces. IEEE PAMI 26(3), 372–383 (2003)
6. Latecki, L.J., Lakämper, R.: Shape similarity measure based on correspondence of visual parts. IEEE PAMI 22(10), 1185–1190 (2000)
7. Manay, S., Cremers, D., Hong, B.-W., Yezzi, A., Soatto, S.: Integral invariants for shape matching. IEEE PAMI 28(10), 1602–1618 (2006)
8. Michor, P., Mumford, D.: Riemannian geometries on spaces of plane curves. J. of the European Math. Society (2003)
9. Pennec, X.: Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements. Journal of Mathematical Imaging and Vision 25(1), 127–154 (2004) (A preliminary appeared as INRIA RR-5093, January 2004)
10. Schmidt, F.R., Töppe, E., Cremers, D., Boykov, Y.: Efficient shape matching via graph cuts. In: Energy Minimization Methods in Computer Vision and Pattern Recognition (to appear)
11. Sharvit, D., Chan, J., Tek, H., Kimia, B.: Symmetry-based indexing of image databases (1998)
12. Whitney, H.: Congruent graphs and the connectivity of graphs. Amer. J. Math. 54, 150–168 (1932)