Correspondences between Parts of Shapes with Particle Filters

Rolf Lakaemper
Temple University
Philadelphia, USA
lakamper@temple.edu

Marc Sobel
Temple University
Philadelphia
marc.sobel@temple.edu

Abstract

Given two shapes, the correspondence between distinct visual features is the basis for most alignment processes and shape similarity measures. This paper presents an approach introducing particle filters to establish perceptually correct correspondences between point sets representing shapes. Local shape feature descriptors are used to establish correspondence probabilities. The global correspondence structure is calculated using additional constraints based on domain knowledge. Domain knowledge is characterized as prior distributions expressing hypotheses about the global relationships between shapes. These hypotheses are generated during the iterative particle filtering process. Experiments using standard alignment techniques, based on the given correspondence relationships, demonstrate the advantages of this approach.

1. Introduction and Related Work

The goal of our work is to find correspondences between visually similar shape features. Feature correspondences are used as a pre processing procedure for shape recognition in Computer Vision. For example,[2] define the steps to distinguishing shape similarity as, (i) correspondence, (ii) alignment, and (iii) similarity measurement. Devising consistent feature correspondences, in accordance with human perception, is necessary for this task.

Our research aims at a general description of correspondence. This includes correspondences between shapes and parts of shapes. In the sequel, we equate shapes with (appropriately sampled) sets of points; each such point has one or more (local) features assigned to it. Examples of such local features include curvature (if the shape is defined by a surface, [13]), point density, shape context [2], or features based on the Poisson equation [5]. Selecting which local features to use depends on the particular setting and goals involved. The challenge is to determine correspondences between shapes using local descriptors in a globally consistent manner. This challenge includes resolving ambiguities caused by combining the information contained in each local descriptor.

We approach this problem using Particle Filters (PF). Particle filters employ probabilities and multiple hypotheses to build correspondences between shapes. Particle filters have been successfully utilized in both Computer Vision (mostly for object tracking [16] and image segmentation [3]) and robot mapping [21]. General information about particle filtering is given e.g. in [4], [11].

In our setting, we assign a probability to each possible configuration of single point correspondences. It is based on the fitness of each participating single point-point correspondence. The probabilities model the visual correctness of correspondences; visually better correspondences are assigned higher probability values. Under certain assumptions, this assignment leads to the calculation of probabilities for sequences of correspondences, called particles below. Probabilities attached to particles are, in conformity with statistical terminology, called 'likelihoods', below. The global consistency of particles is enforced using prior distributions. The search space consists of the set of all possible particles.

The goal of particle filters (PF) is to estimate the posterior distribution over the entire search space using discrete distributions (constructed dynamically at each of a number of different iterations) based on a limited number of particles. In this sense, particles represent hypotheses about what the true relationship between shapes really is; we occasionally use this terminology below. The best correspondence is the most likely particle (Maximum Likelihood Particle, MLP) surviving at the end of the PF process [11].

All particles compete in an iterative process, each iteration consisting of two steps: prediction and evaluation. In the prediction step, particles are augmented by adding single correspondences; the resulting set of particles is called the 'preliminary sample' below. Correspondences are selected based on a correspondence-weight distribution, which represents single correspondence probabilities as defined by the local feature descriptors. Using Bayes rule, each particle is assigned a weight representing its strength,

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

proportional to its posterior probability. The evaluation step selects particles from the preliminary sample using these weights. We evaluate particles using residual sub-sampling [10]. This selection causes stronger hypotheses (particles) to dominate weaker ones, yet randomly permits some weak hypotheses (outliers) to survive and possibly prosper in later iterations. In this way we protect against choosing particles which are only local maxima [10].

The PF algorithm converges to particles with nearly optimal posterior probability because: (i) at each stage correspondences with greater weight tend to be chosen, and (ii) residual re-sampling has the effect of removing particles with relatively small posterior probability. Although we slightly extend the PF process by a *recede* step, we do not focus, below, on the question of how best to improve it. There are many different ways of designing particle filters; each such design is associated with techniques which optimize their performance [14].

The **main contribution of our research** is to introduce the use of PF to solve the correspondence problem, and, more specifically, the problem of correctly designing the prediction step. Prediction is based on an iteratively updated feature probability distribution. This distribution is composed of two parts: (i) the local part, representing the correspondence probability based on the local feature descriptors, and (ii) the global part, representing non local constraints. The constraints can describe topological or geometrical features of the shape and are used to achieve global consistency. The constraints are built into each iteration of the PF process; they are enforced using the (conditional) information available from the already constructed correspondence sequence. This has the advantage of allowing us the easy task of learning constraints at each iteration, rather than the hard task of predicting them a priori. We use prior distributions, built conditionally at each iteration, to enforce the learning constraints.

The process of finding correct correspondences can be seen as a labeling process. The features of one shape correspond to the labels; the features of the other have to be labeled. In 1976, Rosenfeld et al. [17] introduced the technique of relaxation labeling (RL) to approach this class of problems. It has since been a topic of successful ongoing research ([7], [8], [22]. In the soft version, a weight matrix defines the correspondence probabilities between data point and label. The property of consistency is defined in terms of certain given neighborhoods. RL is used to solve an optimization problem subject to the given consistency requirements. RL is a gradient descent method which guarantees convergence towards some local optimum. It is an iterative, deterministic approach, highly dependent on the initial correspondence probability matrix. In this same connection, our approach can be interpreted as sequential, non-deterministic, multiple hypotheses relaxation labeling. Sequential, because of the aforementioned prediction step, which assigns a *single* label *unchangeably* to a data point. By contrast, in each iteration, relaxation labeling *re-labels* all the data *simultaneously*. PF enhances strong local feature properties, while RL subsumes them, enhancing the global labeling structure. For these reasons particle filters have the drawback of being easily lured into local optima. This drawback is counterbalanced by the assumption, at each iteration, of multiple hypotheses which compete with one another.

Our research is designed as a general framework, yet the examples and results explore the particular case in which correspondences are built between 2-dimensional boundaries. Sebastian et al. [20] showed the importance of ordering constraints in matching curves. We use this constraint as an example. It shows how to integrate the constraint of contiguity conservation for the simpler case of non-partial shape matching. This special case can also be solved using Dynamic Programming (DP); successfully applied e.g. in [12], [19],[18]. Particle filters PF guarantee only a nearly optimal solution, as compared with DP methodology which guarantees an optimal solution in this case. Nonetheless, particle filters generalize in cases where DP methodology does not. For example, particle filters generalize easily to the task of building shape and part correspondences. Our final result shows how we learn constraints in the example of partial shape matching, used to build similarity measures for purposes of querying shape databases with shape parts.

Since local descriptors are imprecise, the optimum, calculated using these descriptors, is not necessarily a more accurate solution; an example is given in section 6.1. We note that the prediction step for particle filters, described above, samples a new correspondence from the possible correspondences; this selection does not depend on any implicit or explicit ordering of features. In contrast to the DP approach in [19], we do not need a designated start correspondence and handle shape-reflection (clockwise/anti clockwise order of boundary points) automatically. We also don't need additional parameters characterizing the minimum number of correspondences.

RL is a common way of computation for Markov random fields [15]. Typically, this involves specifying a prior over the search space of particles at any given iteration conditional on all others. This differs from the priors advocated here which are defined conditional on previous iterations. We leave comparisons between this approach and ours to future work. Another single hypothesis probabilistic approach to find correspondences is given in [1]. It evaluates shape matchings based on properties of the correspondence set itself, therefore focusing on the global shape properties.

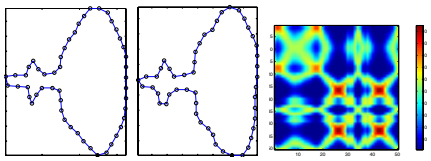

Figure 1. 2 shapes defined by a $2d$ polygonal boundary and their correspondence weight matrix. The shapes are indexed from the bottom, counter clockwise. The bottom right area of the weight matrix shows distinct correspondence probabilities, these are strong correspondences between the cross-shaped parts of the objects based on equal and distinct curvature.

1.1. Notation

If we deal with closed shape boundaries, all index-math is understood module n_i=number of points on boundary. Throughout the paper, $\phi_\sigma(x)$ will denote the mean 0 Gauss distribution with standard deviation σ of a random variable x. $|g|$ denotes the cardinality of a set g. For matrices M_1, M_2, $M_1 \odot M_2$ denotes the element-wise multiplication. For all colored figures the color scale is equal to fig.1, right.

2. Shape Features and Probability Matrix W

The shapes, $S_1 = v_1, .., v_{n_1}$ and $S_2 = u_1, ..., u_{n_2}$ are sets of $n_{1,2}$-dimensional points $v_i, u_j \in \mathbb{R}^{n_{1,2}}$. To describe a shape, we assign local feature descriptors to each of its points. Throughout the paper we use $2d$ polygonal boundaries as examples for shapes, the points are the vertices of the polygon. The local descriptors used for the examples are *local centroid distance* and *curvature*. The power of these descriptors is naturally limited, which is reflected in the correspondence ambiguity between vertices: a single vertex of shape S_1 might have a large number of possible partners (vertices with similar features) in the second shape S_2. The power of the system comes from its ability to solve these ambiguities in a logical manner. Insufficient regional or global description by local feature descriptors is, in consequence, not a problem [1].

The matrix, W, consisting of correspondence probabilities for each pair of points (u_i, v_j) $v_i \in S_1$; $u_j \in S_2$, using the local features of v_i, u_j is described below. Let $F(v_i) : \mathbb{R}^n \rightarrow \mathbb{R}$ be a feature (e.g. curvature) of a shape point $v_i \in S$. We first compute the correspondence likelihood for all mutual point correspondences based on the distance between features:

$$L = [l_{ij}],\ l_{ij} = \phi_{\sigma_l}(D(F(v_i), F(u_j))); \qquad (1)$$

with, $D(.,.)$ being a distance measure in the feature space and σ_l a given standard deviation. We create a row-normalized version L^r of L as well as a column-normalized version L^c. $L^r = P(u_j|v_i)$, $1 \leq i \leq n_1$; $1 \leq j \leq n_2$ is the conditional correspondence probability in the direction $S_1 \rightarrow S_2$; L^c, defined analogously, describes the conditional correspondence probability in the direction, $S_2 \rightarrow S_1$. We define the correspondence probability W as the normalized joint probability:

$$W = \frac{L^r \odot L^c}{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} (l^r_{ij} l^c_{ij})} \qquad (2)$$

The definition of W as a joint probability ensures cognitive symmetry. This means that the correspondence between u_i and u_j is order independent, i.e. there is no distinction between query and target shape. We therefore determine the symmetric correspondence between shapes, not the (one-sided) correspondence of one shape to another. If the shapes are defined by multiple feature descriptors $F_1, .., F_d$ we define W^{F_i} according to eq.2 and define W as the joint probability $W = W^{F_1} \odot ... \odot W^{F_d}$. Figure 1 gives an example for two shapes and their correspondence probability matrix W.

3. Correspondences and Groupings

Given two shapes S_1,S_2 with n_1, n_2 vertices, we define the *set of correspondences* $\mathcal{C}$ as the set of all pairs of vertices of S_1 and S_2:

$$\mathcal{C} = \{\{v_i, u_j\} | v_i \in S_1, u_j \in S_2\} = S_1 \times S_2.$$

The matrix W defines a probability $P_\mathcal{C}$ over the set $\mathcal{C}$ of correspondences: $P_\mathcal{C}(v_i, u_j) = w_{ij}$.
A *Grouping* $g \in \mathcal{G}$ is a member of the power set $\mathbb{P}\mathcal{C}$ of $\mathcal{C}$. A grouping defines a configuration of correspondences. $\mathcal{G}$ defines the search space for our PF process. Each element $g \in \mathcal{G}$ takes the form $g = \{\{v_{i_1}, u_{j_1}\}, .., \{v_{i_k}, u_{j_k}\}\}$. Further constraints (e.g. contiguity conservation, see section 5) on groupings can limit the search space to a subset $\mathcal{G}^- \subset \mathcal{G}$. A grouping g is *complete*, if it is maximal with respect to the containment ordering in the set $\mathcal{G}^-$. In quantitative terms, $\forall g' \in \mathcal{G}^- : \ g \subset g' \rightarrow g = g'$

Figure 2 shows an example of a complete grouping, computed under the constraint of strong contiguity conservation (defined in section 5), i.e. here $\mathcal{G}^-$ is the set of contiguity conserving groupings. Defining a grouping as edges of a graph with vertices $S_1 \cup S_2$, fig.2 visualizes this graph by drawing edges as well as by the connectivity matrix (the connectivity matrix is superimposed over the probability matrix W).

3.1. Optimal Sets of Correspondences

We define the weight of a grouping as,

$$W_\mathcal{G}(g \in \mathcal{G}) = \prod_{l=1}^{|g|} \exp\left(P_\mathcal{C}(v_{i_l}, u_{j_l})\right) = \prod_{l=1}^{|g|} \exp\left(w_{i_l, j_l}\right) \qquad (3)$$

[1] ambiguity can even be seen as a positive descriptor property, allowing a particle to choose between multiple local correspondences, all of which do not prematurely claim to be certain

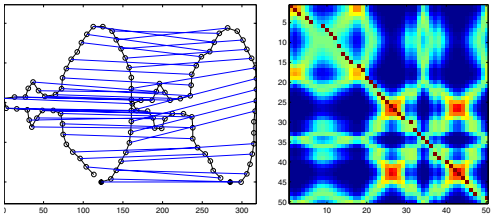

Figure 2. Left: A complete, strongly contiguity conserving grouping. Each blue line is a single correspondence. Right: Connectivity matrix of the same grouping, superimposed over correspondence weight matrix W. Each red dot is a single correspondence (red dot means conn.matrix=1. Note that the grouping is complete although not all points are participating.

When there is no possibility of confusion, we use the simplified notation, $W(g)$ for this quantity. We formulate the correspondence problem as one of choosing the complete grouping, $\hat{g} \in \mathcal{G}^-$ from the set of constrained groupings, $\mathcal{G}^-$ with maximal weight or, more specifically,

$$\hat{g} = \arg\max_{g \in \mathcal{G}^-}(W_{\mathcal{G}}(g)) \tag{4}$$

The optimal grouping is complete as a consequence of the fact that correspondence weights are defined to be larger than 1. Thus, groupings with more correspondences dominate those with fewer correspondences. This is in contrast to the joint probability, $\prod(w_{ij})$, which decreases as the number of correspondences increases. The optimal grouping can therefore be understood as a search for a grouping with as many connections as possible, subject to optimizing the overall weight. Dynamic Programming methodology [19] solves a special case of this optimization problem: entire shape matching under the constraint of contiguity conservation. In contrast to that approach, we need no additional parameters to ensure that the optimal grouping has some minimal number of correspondences. This is a consequence of our use of complete groupings.

Specific tasks of shape matching, like the matching of parts, requires far stronger constraints. Typically, in this case, additional domain knowledge is required to achieve success. Particle filters are well designed to properly formulate and use this knowledge; we employ them below for this purpose. It will be seen that this approach provides more flexibility in solving the optimization problem when these additional constraints are present. PF enable us to (interactively) learn constraints during the iteration process leading to optimization.

4. Near Optimal Labeling using Particle Filters

Below, we refer to a grouping $g \in \mathcal{G}^-$ as a single particle. We employ the notation, $g_{1:t}$ for a particle at time (iteration) t. Particles are built by adding single correspondences at each iteration. Correspondences are selected based on an updated version W_t of the correspondence-weight distribution W. The update of $W \rightarrow W_t$ defines additional constraints. The following sections will explain the prediction and evaluation step of PF in our setting, as well as the new step of *recede*. In what follows we use the definition: all correspondences are *admissible* at iteration $t = 1$. At iteration $t > 1$, a correspondence $c \in \mathcal{C}$ is admissible if, for a given particle, $g_{1:t-1} \in \mathcal{G}^-$ (at iteration $t-1$), the particle $g_{1:t} = g_{1:t-1} \cup c$ is in $\mathcal{G}^-$.

4.1. Prediction Step

The prediction step consists of 2 parts: a) select a correspondence based on the updated probability distribution W_t over all admissible correspondences $c \in \mathcal{C}$ at iteration t. b) compute the posterior probability of the resulting augmented particle.

4.1.1 Distribution for Correspondence Selection

W^t is the updated version of W (at iteration t) incorporating constraints given by a matrix, $C(g_{1:t})$. $C(g_{1:t})$ depends on the particle $g_{1:t}$ at each iteration t. In this sense, we have,

$$W^t = W \odot C(g_{1:t}) \tag{5}$$

Note that $C(g_{1:t})$ changes from one particle g to the next and from one iteration t to the next. This enables us to adjust or learn constraints during the PF process. Section 5 will give examples of such constraints.

4.1.2 Posterior Distribution Estimation

We define w^t_{ij} of W^t as the log-likelihood for the selection of the correspondence c_{ij} at time t to update the grouping $g_{1:t} \rightarrow g_{1:t+1} = g_{1:t} \cup c_{ij}$. Hence c_{ij} is our new observation at time (iteration) t with log-likelihood $\log(P(c_{ij}|g_{1:t})) = w^t_{ij}$. The posterior distribution of $g_{1:t+1}$ at time (iteration) t is given, up to a constant of proportionality, by:

$$P(g_{t+1}|c_{ij}) \propto P(c_{ij}|g_t)P(g_t) = exp(w^t_{ij})P(g_t) \tag{6}$$

The prediction step sequentially estimates a near optimal grouping as a near solution to eq.4.

4.2. Evaluation Step

We use the standard evaluation technique of residual resampling [10].

4.3. Extension of Classical PF: Recede Step

Since we assign correspondences 'statically', i.e. once a correspondence is established it does not change, the basic predict-evaluate sequence of PF algorithms has the disadvantage that it sometimes fails to converge to optimal or

near optimal particles. This is a consequence of the fact that correspondences chosen at the beginning do not necessarily, in spite of having large weight, lead to particles which share this property. For this reason we augment the algorithm by incorporating a *recede* step which 'destroys' correspondences in particles. This is, in reality, a 'jump' move (i.e., one in which new particle neighborhoods are selected). In general jump moves serve to change the initial correspondences in order to create more viable particles (i.e., those with larger total weight). The dependence of later on earlier correspondences makes these moves imperative. The recede step extends the classical PF sequence of prediction and evaluation to prediction, evaluation, recede. Jump moves can be implemented in different ways. In our setting it is important to insure that the particles have similar numbers of correspondences; since they have a major influence on particle survival. We have therefore designed the recede step in such a way that every r'th iteration, a certain number $d < r$ of correspondences are destroyed for each and every one of the particles. This guarantees that, in early iterations, all particles have the same number of correspondences. In later iterations, the number of correspondences per particle depends on the property of completeness; we allow this to vary so that particles with larger numbers of correspondences dominate. In the current implementation, r is set to 10, 5 particles are destroyed. Figure 3 shows an

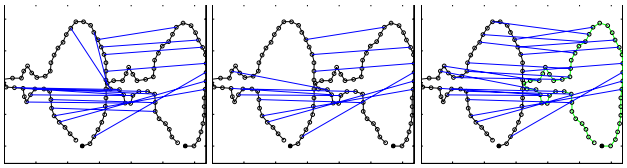

Figure 3. Result of *recede* step: Left: before recede. Middle: directly after recede, 8 randomly selected correspondences removed from particle. Right: 8 iterations later. The particle is re-built in a more consistent way, containing the same number as the grouping left. The final result can be seen in fig. 2

example of a particle undergoing a recede step. The improved performance due to the recede step is demonstrated in fig.6, section 6.1.

4.4. Extended PF Algorithm for estimating the MLP

We will now define the general PF process to estimate the Maximum Likelihood Particle. $g^i_{1:t} \in \mathcal{G}^-$ denotes the ith particle in iteration t, $G_{1:t}$ the set of all particles in iteration t. The algorithm follows the classic steps of prediction and evaluation and is extended by the additional recede step.

---

1)**INIT**: t=1, $g^i_{1:t} = \emptyset \ \ \forall i = 1..m$= number of particles. $W^t = W$. Init r for the recede-step (see section 4.3).

2)Prepare the constraint matrices $C(g^i_{1:t})$ for $i = 1..m$ and compute $W^t_i = W \odot C(g^i_{1:t})$

3)Select a correspondence $c^i \in \mathcal{C}$ based on the distribution W^t_i.

4)**PREDICTION**: compute posterior distribution (weight of particle) $P(g^i_{1:t+1}|c^i)$ using eq. 6.

5)normalize weights: $P(g^i_{1:t+1}) \leftarrow \frac{P(g^i_{1:t+1})}{\sum_{j=1}^{m} P(g^j_{1:t+1})}$

6)**EVALUATION**: compute new set of particles $G_{t+1} \leftarrow RRS(G_t)$ using residual re-sampling (RRS) preserving most probably those particles with dominant weight.

7)**RECEDE**: if $\mod(t, r) = 0$ delete $n < r$ correspondences in each particle in $G_{1:t}$ (see section 4.3).

8) LOOP: **if** not all particles are complete: $t \leftarrow t+1$, return to step 2 **else** return particle $\hat{g}_{1:t} = argmax_{g_{1:t} \in G_t}(P(g_{1:t}))$ with maximum weight to represent a near optimal solution.

---

5. Adding Constraints to Solve Specific Correspondence Tasks

We now show how to define constraint matrices to model different tasks.

5.1. One to One Correspondences

To guarantee one to one correspondences (in contrast to one to many), the probability of selecting a correspondence containing a point that is already part of an existing correspondence in particle $g_{1:t}$ must be set to 0. A matrix $C(g_{1:t}) = [c_{ij}]$ defined by:

$$c_{ij} = 0 \iff (v_i \cup u_j) \cap \left(\bigcup_{i=1}^{|g_{1:t}|} g^i_{1:t} \right) \neq \emptyset, else\ c_{ij} = 1$$

guarantees the one to one constraint. Example (particle $g_{1:t}$ is represented by its connectivity matrix):

$$g_{1:t} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} C(g_{1:t}) = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

5.2. Strong Contiguity Preservation

g is be said to be **contiguity conserving** if, for a clockwise ordering of $(v_{i_1}, .., v_{i_k})$ in S_1, $(u_{j_1}, .., u_{j_k})$ is ordered either clockwise or anti-clockwise in S_2 (allowing both clockwise and anti-clockwise orderings in S_2 makes it possible to find correspondences in the reflected version of S_2). g is **strongly contiguity conserving**, if it is contiguity conserving and contains one to one correspondences only. The implementation of such a constraint using matrices is straightforward; we give an example below:

$$g_{1:t} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} C(g_{1:t}) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

An example illustrating strong contiguity conserving grouping is in fig.2.

5.3. Adding Domain Knowledge for Partial Shape Matching

The constraint of strong contiguity conservation is sufficient to find good groupings e.g. for closed shape boundaries. However, it fails in finding part correspondences, see example fig.4, left. Additional domain knowledge has to be implemented. In this example, the constraint matrix is stated as a hypothesis based on the grouping $g_{1:t}$ and weight matrix W. As the number of iterations increases, the hypothesis becomes stronger. This can be interpreted as *learning the constraint* during the PF process.

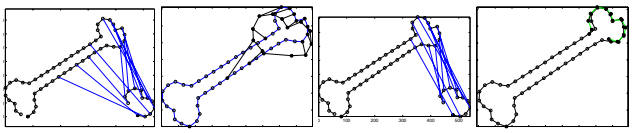

Figure 4. Part correspondences using different constraints. From left to right: 1) strong contiguity only. 2) Procrustes alignment (see section 6 based on (1). 3) windowed strong contiguity as introduced in section 5.3. The window was learned during the PF process. 4) Procrustes alignment of (3)

In the following we assume shape S_1 *entirely* contains part S_2. This is not a strong assumption, since if S_1 only contains parts of S_2, we can exchange both shapes and use an appropriate subset of S_2 as the part. This assumption is above and beyond the constraints used to compare entire (not parts of) shapes. For purposes of simplification, we assume S_1, S_2 have the same scale and are sub-sampled equally (ignoring the these constraints is done only at the cost of a less intuitive constraint matrix). These assumptions jointly imply that the corresponding vertices in S_1 have to be 'close together', and not spread out all over S_1. In other words, we focus exclusively on a certain connected region (or window) of S_1.

Figure 4 shows the effect of this constraint. If it is not applied (left), only the center vertices of part S_2 are held together by the contiguity constraint, hence they lead to correct correspondences. The outer vertices of S_2 have the opportunity to correspond to wide regions of shape S_1, since contiguity is a local neighborhood constraint which is weaker for the (intuitively) weaker neighborhood-bound outer vertices. Focusing on a connected region in S_1 leads to the correct correspondences (right).

If the region of focus is known a priori (i.e. the position of the part in the shape is roughly given but the exact single correspondences are not explicitly known), the optimization process is simply a windowed version of entire shape matching. Unfortunately the window is not known. *The strength of our approach is that it can estimate the window during the PF process by analysis of the particle given in each iteration.* Below, we use the notation, 'diagonal matrices' to refer to matrices with elements on either the main or side-diagonals. To highlight a certain region $r = (v_r, v_{r+1}, .., v_{r+k})$ in S_1, correspondences with $v_i \in r$ have to be masked in W. Due to the equidistant sub sample and equal scale assumptions, the mask is a diagonal matrix. The position of the diagonal defines the allowed region. We use a soft mask, i.e. a gaussian blurred diagonal, see fig.5. The kernel size, or standard deviation σ_t of the blurring defines the distinctiveness of the region. To learn

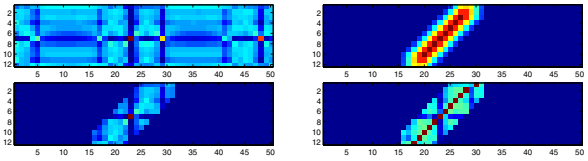

Figure 5. Windowing for partial shape matching. In reading order: 1) correspondence matrix W of shape and part shown in fig.4. 2) The final window, learned during the PF process. 3) Windowed version of (1). 4) The final connectivity matrix (superimposed over (3)). (4) corresponds to the grouping shown in fig.4, right.

the window during the PF process, we analyze the connectivity matrix of a particle $g_{1:t}$ at time t for the presence of diagonals. That is to say, we determine the most likely diagonal of its connectivity matrix, taking into account both the connectivity matrix of $g_{1:t}$ as well as the underlying correspondence weights of the matrix W. We gauss-filter the estimated diagonal to obtain a window $C_{\sigma_t}(g_{1:t})$. The diagonal is a hypothesis for our focus of attention, namely, the region r. The strength of the hypothesis is modeled by the filter kernel size σ_t. A large σ_t defines a weak hypothesis (or indistinct region). Decreasing σ_t during the iterative process, increases the trust in the diagonal hypotheses at later iterations, since it narrows the diagonal constraint window $C_{\sigma_t}(g_{1:t})$. The final constraint matrix $C(g_{1:t})$ is then given by:

$$C(g_{1:t}) = C_{\sigma_t}(g_{1:t}) \odot C_C(g_{1:t})$$

with $C_(g_{1:t})$ modeling the strong contiguity preservation as defined in section 5.2. To ensure that the final particle $\hat{g}_{1:t}$ is entirely placed in the window $C_{\sigma_t}(\hat{g}_{1:t})$, the particle process is repeated: from the first PF process, we only use the final constraint matrix $C_{\sigma_t}(\hat{g}_{1:t})$ of the winning particle. We then run a second entire particle filtering pass using $C_{\sigma_t}(\hat{g}_{1:t})$ in each iteration (i.e. we finally run a windowed version of the optimization process). The second pass needs only a few particles since the window limits the search space drastically (in our tests we decreased from 80 particles in the window-learning pass to 8 particles in the windowed pass). Note that especially the interplay between $K_C(g_{1:t})$ and the recede-step already builds a near diagonal solution in the

first run, since the domain knowledge gained in iteration t helps to rebuild destroyed connections appropriately inside of the window. With this approach, parts could be matched successfully. For results see section 6.

6. Experiments and Results

6.1. Performance of Particle Filtering

We evaluate the performance of the PF process in the optimization process with respect to different numbers of particles, with and without recede-step.

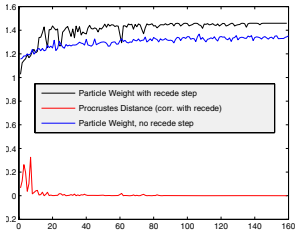

Figure 6. Performance of PF system with different number (1-160) of particles (x-axis) with and without recede step. Black: with recede. Blue: no recede step. Red: Procrustes Distance.

Given a shape S defined by a $2d$ polygonal boundary with 50 equidistant vertices, we compute the correspondence matrix W of S with *itself* under strong contiguity conservation. This setting provides a ground truth optimum grouping $\hat{g} = \{\{v_1, v_1\}, .., \{v_{50}, v_{50}\}\}$. We perform different runs the optimization process using 1 to 160 particles, the result of each run is the weight of the best particle. To see the influence of the recede step, we repeat the experiment skipping this step. To see the influence in the alignment resulting from finding a near optimal solution only, we compute the shape similarity based on the best particle correspondence using Procrustes Analysis (PA) [6]. PA aligns two shapes S_1, S_2 based on one to one correspondences. It finds the best scaling, reflection, rotation and translation of S_2 to minimize the sum of squared distances between corresponding points. This sum, the PA distance, is used in our experiment as the shape similarity measure. PA is extremely sensitive to outliers and can only be used with a robust correspondence computation. The usability of PA as similarity measure shows the stability of the PF correspondences. Figure 6 illustrates the result. The PF system with recede step computes weights near the optimum (1.45) robustly already with > 60 particles. Interesting is the fact that the Procrustes distance drops to 0 (perfect alignment) much faster, proving that the additional precision gained is visually unimportant, hence the precise optimum, as found by DP approaches, would not enhance the alignment here. The graph also shows the importance of the recede step: skipping it, the performance of the system is significantly less than optimal.

The time complexity of the algorithm is determined by the number of iterations and the complexity of the prediction step for each particle[2]. The latter one depends on the complexity to build the constraint matrix. In the case of partial matching the constraint matrix can be determined in $O(n^2)$. The number of iterations is n, since we aim for completeness. Hence, in this case, the time complexity is $O(n^3)$.

6.2. Part Correspondences

In this experiment we provide visual proof of the performance of partial matching. The data are shapes from the standard MPEG-7 similarity data set [9]. The data set provides 70 classes of 20 similar shapes each. We use the data set in the form of boundary polygons; each such polygon contains 50 vertices. We randomly select shape S_1 from the data and S_2 as a random *part* (size: 15 vertices) from a different shape in the same class. The PF process (50 particles, constraint: windowed strong contiguity as presented in section 5.3) leads to correspondences which are used for Procrustes alignment. The figure shows some typical examples of partial matchings.

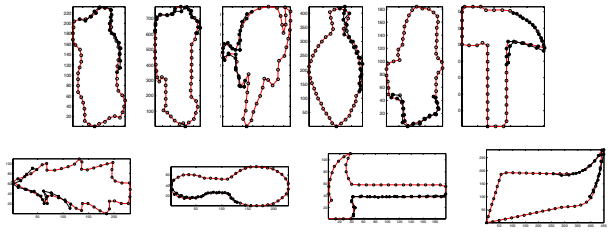

Figure 7. Examples of partial matching using different shapes from the MPEG-7 dataset.

6.3. Database Retrieval using Parts

We compare our PF correspondence to part matching techniques based on time series algorithms described in [9]. As in [9], we took 10 parts of shapes of different classes from the MPEG-7 data set, the parts being identical to the ones described in [9]. We then matched each part to the entire database (all 1400 shapes), based on the PF correspondence (PFPA) and Procrustes Alignment. To achieve a fair comparison, we used curvature as shape descriptor, as in [9]. Fig. 8 shows the parts (left column) and the top 5 matches for each part. In nearly all cases (96%) the query resulted in shapes of the same class as the query part.

	PFPA	OSB	DTWCW	LCSS
Top 1	100	100	90	90
Top 5	96	92	72	42
Top 10	91	84	67	34
Top 20	74	67	59	26

The table shows the intra class hit percentage for the top $1, 5, 10, 20$ results, i.e. how many of the top-n results are from the same class as the query part. PFPA is the described

[2] the constant factor depends on the number of particles and number of correspondences destroyed in the recede step

algorithm, OSB the algorithm described in [9]. DTWCW is a windowed version of Dynamic Time Warping, LCSS stands for Longest Common Sub Sequence. The percentages of OSB, DTWCW and LCSS are taken from [9], details about these algorithms are described there, too. Due to our robust partial PF correspondence, even a simple similarity measure like PA outperforms the competing approaches.

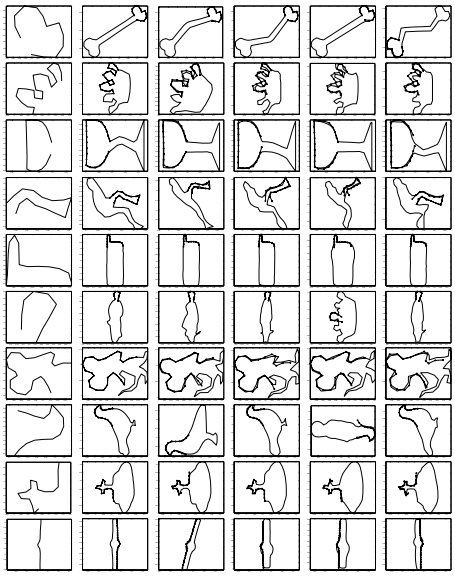

Figure 8. Results of database query with parts. Left column is the query part, which is compared against the entire MPEG-7 data set (1400 shapes). Each row shows the respective top 5 hits.

7. Conclusion and Outlook

We presented a Particle Filter framework to solve the correspondence problem. The performance was demonstrated on partial shape matching of $2d$ polygonal boundaries, which was solved through a windowing approach. The window was learned during the iterative particle filter process. Though the examples of this paper were restricted to $2d$ boundary shape representation, the PF framework can handle any shapes represented by point sets of arbitrary dimensionality. Future work will therefore focus on, (i) $2d$ point sets representing shapes with inner structures, and (ii) $3d$ shapes.

8. Acknowledgements

This work was partly supported by the NSF under Grant No. IIS-0534929 and by the DOE under Grant No. DE-FG52-06NA27508.

References

[1] H. Alt, L. Scharf, and S. Scholz. Probabilistic matching of sets of polygonal curves. In *Proceedings of the 22nd European Workshop on Computational Geometry (EWCG)*, pages 107–110, Delphi, Greece, March 2006. 2

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24:705–522, 2002. 1

[3] M. de Bruijne and M. Nielsen. Image segmentation by shape particle filtering. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3*, pages 722–725, Washington, DC, USA, 2004. IEEE Computer Society. 1

[4] A. Doucet, N. De Freitas, and N. Gordon, editors. *Sequential Monte Carlo methods in practice*. Springer. 1

[5] L. Gorelick, M. Galun, and A. Brandt. Shape representation and classification using the poisson equation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(12):1991–2005, 2006. Member-Eitan Sharon and Member-Ronen Basri. 1

[6] J. Gower and G. Dijksterhuis. *Procrustes problems*, volume 70. Springer New York, December 2005. 7

[7] R. M. Haralick and L. G. Shapiro. The consistent labeling problem: Part i. *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-1(2):173–184, 1979. 2

[8] J. Kittler and J. Illingworth. Relaxation labelling algorithms-a review. *Image Vision Comput.*, 3(4):206–216, 1986. 2

[9] L. J. Latecki, Q. Wang, S. Koknar-Tezel, and V. Megalooikonomou. Optimal subsequence bijection. *IEEE Int. Conf. on Data Mining (ICDM)*. 7, 8

[10] J. Liu, R. Chen, and T. Logvinenko. A theoretical framework for sequential importance sampling and resampling, 2000. 2, 4

[11] J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, October 2002. 1

[12] E. Milios and E. Petrakis. Shape retrieval based on dynamic programming, 2000. 2

[13] F. Mokhtarian, N. Khalili, and P. Yuen. Estimation of error in curvature computation on multi-scale free-form surfaces. *Int. J. Comput. Vision*, 48(2):131–149, 2002. 1

[14] P. D. Moral, A. Doucet, and A. Jasra. Sequential monte carlo samplers. *Journal Of The Royal Statistical Society Series B*, 68(3):411–436, 2006. 2

[15] W. Qian and D. M. Titterington. Stochastic relaxations and em algorithms for markov random fields. *Journal of Statistical Computing and Simulation*, 40, 1992. 2

[16] Y. Rathi, N. Vaswani, and A. Tannenbaum. A generic framework for tracking using particle filter with dynamic shape prior. 16(5):1370–1382, May 2007. 1

[17] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Trans. Syst., Man, Cybern.*, SMC-6(6):420–433, 1976. 2

[18] F. Schmidt, D. Farin, and D. Cremers. Fast matching of planar shapes in sub-cubic runtime. In *IEEE International Conference on Computer Vision (ICCV)*, 2007. 2

[19] C. Scott and R. Nowak. Robust contour matching via the order-preserving assignment problem. *IEEE Transactions on Image Processing*, 15(7):1831–1838, 2006. 2, 4

[20] T. Sebastian, P. Klein, and B. Kimia. On aligning curves. *PAMI*, 25(1):116–125, January 2003. 2

[21] S. Thrun. Particle filters in robotics. In *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)*, 2002. 1

[22] Yefeng Zheng and David Doermann. Robust Point Matching for Nonrigid Shapes By Preserving Local Neighborhood Structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):643–649, April 2006. 2