An Improved Shape Matching Algorithm for Deformable Objects Using a Global Image Feature

Jibum Kim and Suzanne M. Shontz

Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
{jzk164,shontz}@cse.psu.edu

Abstract. We propose an improved shape matching algorithm that extends the work of Felzenszwalb [3]. In this approach, we use triangular meshes to represent deformable objects and use dynamic programming to find the optimal mapping from the source image to the target image which minimizes a new energy function. Our energy function includes a new cost term that takes into account the center of mass of an image. This term is invariant to translation, rotation, and uniform scaling. We also improve the dynamic programming method proposed in [3] using the center of mass of an image. Experimental results on the Brown dataset show a 7.8% higher recognition rate when compared with Felzenszwalb's algorithm.

1 Introduction

Shape matching is an important problem in many computer vision applications such as object tracking and image-based searches [1, 2]. The goal of shape matching is to match the source image to the target image, i.e., the deformed image. Felzenszwalb proposed a shape matching algorithm for deformable objects using triangular meshes and dynamic programming in [3]. Felzenszwalb's algorithm was novel in that it does not require any initialization to detect the target image unlike previous algorithms (e.g., [4]). A modification of Felzenszwalb's algorithm using flexible shape priors was proposed in [5]. In [5], large deformations on flexible regions were allowed through the use of the shape priors. However, it is hard to know which parts should be made flexible in advance, and thus significant user knowledge is required. Moreover, [3, 5] do not use global image features to detect deformable objects.

Recently, several papers have used global image features for shape matching. For example, hierarchy-based shape matching algorithms for deformable objects were proposed in [6, 7]. In [6], the authors identified shape parts, composed of multiple salient points, and used many-to-many matching for shape matching. The authors in [7] used a shape tree to capture both local and global shape information and employed dynamic programming to perform shape matching.

In this paper, we use local and global shape information to perform shape matching in a complementary method compared with that of [7]. In particular, we use triangular meshes and dynamic programming for shape matching, as in [3, 5], and extend the algorithm in [3] through the use of an added global image feature. The rest of this paper is organized as follows. In Section 2, we describe the three-step shape matching process. In Section 3, we present experimental results, and compare our experimental results with those of algorithm in [3] (i.e., the algorithm to which ours is the most similar). Finally, we give some conclusions and plans for future work in Section 4.

2 Shape matching process

Our shape matching process is composed of three steps. The first step is to determine boundary vertices which approximate the source image boundary. The second step is to generate a triangular mesh using the constrained Delaunay triangulation method to represent the deformable object. The third step is to find the optimal mapping from the source image to the target image which minimizes our energy function. We now describe each of the three steps in more detail.

2.1 Determination of the boundary vertices approximating the source image boundary

This step determines the number of boundary vertices to use in the source image. In order to find a polygonal approximation to a boundary curve in the source image, S, we create nearly equally-spaced vertices on the boundary of S. We select vertices such that the distance between them is as close to an ideal distance parameter as possible, and the boundary curve is close to the original curve. This step is illustrated in Figure 1(a).

2.2 Generation of the triangular mesh on the source image using the constrained Delaunay triangulation method

This step combines the boundary vertices into triangles to represent the non-rigid objects. The constrained Delaunay triangulation method [8] implemented in Triangle [9] is used to generate a triangular mesh, M, that respects the boundary of S, without adding any interior vertices, to represent the deformable parts of the image. This step is shown in Figure 1(b). Note that a triangular mesh without interior vertices can be represented using a dual graph. The vertices in the dual graph of M can be ordered and eliminated using a perfect elimination scheme [10].

2.3 Solution of the shape matching problem

In order to determine the optimal placement of the boundary vertices on a target image, T, we formulate and solve an optimization problem with the goal of

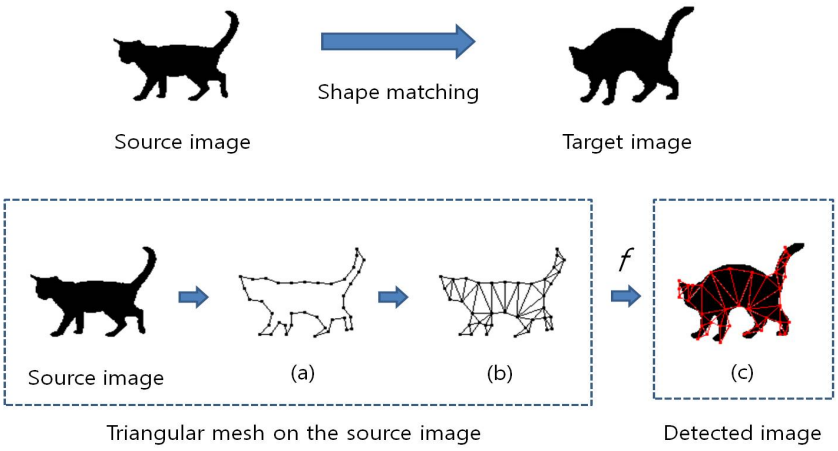


Fig. 1. Overview of the shape matching process. The function f maps triangles in the triangular mesh on the source image to a triangular mesh on the target image. (a) Equally-spaced boundary vertices are generated. (b) The triangular mesh is created. (c) The detected image is illustrated on the target image.

determining the mapping, f, from the triangles in S to the triangles in T, which has the lowest energy. Unlike the energy (cost) functions in [3, 5], our energy function is composed of three terms: an edge cost, a triangular deformation cost, and a triangular edge length distortion cost. The edge cost of the i^{th} triangle, $E_{edge,i}$, corresponds to a simple edge detector by assigning high costs to a low image gradient magnitude in T. Thus, $E_{edge,i}$ increases if edges detected on T are not placed on the boundary. The edge cost of the i^{th} triangle is given by

$$E_{edge,i} = \frac{1}{\lambda + |\nabla I|},$$

where λ is a constant.

The triangular deformation cost, $E_{def,i}$, represents how far the original triangle is transformed from a similarity transform [11] when the mapping from the source image to the target image occurs. The affine transformation of each triangle from S to T takes a unit circle to an ellipse with major and minor axes of lengths, α and β, respectively. Here, α and β are the singular values of the matrix associated with the affine transformation. The squared log-anisotropy of α and β is used to measure the triangular deformation as in [3]. The triangular deformation cost of the i^{th} triangle is given by

$$E_{def,i} = log^2\left(\frac{\alpha}{\beta}\right).$$

Unlike [3, 5], we introduce a new cost, the triangular edge length distortion cost, $E_{dis,i}$, which penalizes the sum of the edge length distortions of each triangle. Let $l_{j,s}$ and $l_{j,t}$ be the j^{th} edge lengths of a triangle in S and T, respectively.

Then, the sum of the three edge lengths of the triangle in S and T, respectively, are given by

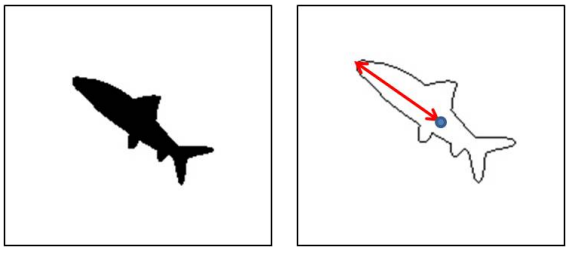
$$l_{sum,s} = \sum_{j=1}^{3} l_{j,s}, \quad l_{sum,t} = \sum_{j=1}^{3} l_{j,t}.$$

Then, γ is defined as

$$\gamma = \begin{cases} l_{sum,s}/l_{sum,t} & \text{if } l_{sum,s} > l_{sum,t} \\ l_{sum,t}/l_{sum.s} & \text{if } l_{sum,t} > l_{sum,s}. \end{cases}$$

To make the triangular edge length distortion cost invariant to uniform scaling from the source image to the target image, we use the center of mass of an image. Let $d_{max,s}$ and $d_{max,t}$ be the maximum distances from the center of masses to the boundary vertices in S and T, respectively. Figure 2 illustrates $d_{max,s}$ and $d_{max,t}$ for a fish shape in the Brown dataset. Then, δ is defined as follows:

$$\delta = \begin{cases} d_{max,s}/d_{max,t} & \text{if } \ d_{max,s} > d_{max,t} \\ d_{max,t}/d_{max,t} & \text{if } \ d_{max,s} < d_{max,t}. \end{cases}$$



(a) (b)

Fig. 2. (a) A sample image shape from the Brown dataset [12]. (b) The dot in the middle represents the center of mass for the image (this is denoted as C in the target image), and the arrow represents the maximum distance from the center of mass in the image to the boundary vertices (i.e., $d_{max,s}$ or $d_{max,t}$)

The triangular edge length distortion cost, $E_{dis,i}$, is large when δ and γ are different. This occurs when the transformation of the triangles from the source image to the target image does not correspond to uniform scaling. Finally, $E_{dis,i}$ is defined as follows:

$$E_{dis,i} = \begin{cases} log(\gamma/\delta) & \text{if } \ \gamma > \delta \\ log(\delta/\gamma) & \text{if } \ \gamma < \delta. \end{cases}$$

The energy function, $E(f)$, is defined as a weighted sum of the three cost terms over the N triangles and is given by

$$E(f) = \sum_{i=1}^{N} E_i = \sum_{i=1}^{N} (E_{edge,i} + a * E_{def,i} + b * E_{dis,i}), \tag{1}$$

where, a and b are scalars. Note that small values of a and b are desirable for highly deformable objects. Let n be the number of vertices in M. Our goal is to determine the f that minimizes $E(f)$. To solve the optimization problem, the shape matching algorithm uses a dynamic programming method given in Algorithm 1. For computational efficiency, a discrete set of grid locations, G, in T is assumed. In line 3 of Algorithm 1, vertices i and j are the parents of the k^{th} vertex. The vertices in M are eliminated in order using a perfect elimination scheme [10].

The optimal matching image (detected mesh) with the lowest possible energy is found using the dynamic programming method shown in Algorithm 1. Note that, for some cases, matching based on the smallest energy does not guarantee a good match. For those cases, we choose the matching image with the next smallest energy. In order to choose the criteria for the sub-optimal cases, we employ the center of mass of an image, which is a global feature of an image. Let the center of mass in T be C and the center of the matched vertices in T be $\hat{C}$. We define D as follows:

$$D = \left\| C - \hat{C} \right\|_2 .$$

Algorithm 1 *Shape Matching Algorithm : Dynamic Programming*

for $k = 1$ to $n - 2$ **do**
  Eliminate the k^{th} vertex in M
  $\{i, j\} \leftarrow$ parent(k)
  for p, $q \in G$ **do**
    $V\,[i, j]\,(p, q) \leftarrow \min_{r \in G} E(i, j, k, p, q, r) + V\,[i, k]\,(p, q) + V\,[j, k]\,(p, q)$
    Choose $p, q \in G$ minimizing $V\,[n-1, n]\,(p, q)$. Trace back to find the optimal mapping for other vertices and compute $\hat{C}$.
  end for
end for
while $D > \theta$ **do**
  Choose new $p, q \in G$ with the next smallest $V\,[n-1, n]\,(p, q)$. Trace back to find the optimal mapping for other vertices and compute $\hat{C}$.
end while

If D is greater than a threshold value of θ, it is likely that the detected mesh on T is poor as shown in Figure 3. The threshold value, θ, can be defined using $d_{max,t}$, the maximum distance from C to the boundary vertices in T. In Figure 3, θ is set to $d_{max,t}/3$. In this case, we select instead the next smallest energy, and

find a new optimal mapping (i.e., by executing the while statement in Algorithm 1) until $\hat{C}$ is such that

$$D \leq \theta. \tag{2}$$

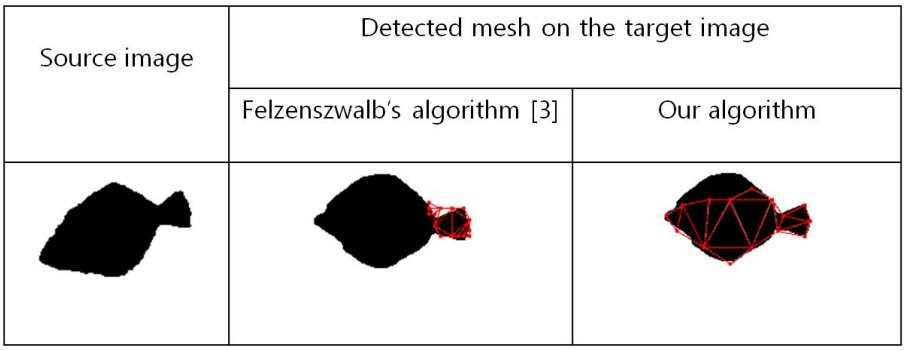


Fig. 3. Poor shape matching result of the algorithm in [3] (left) and improved matching result using the center of mass in the target image (right). For these figures, the detected mesh is illustrated on the target image. For this experiment, $\theta = d_{max,t}/3$.

3 Experiments

This section describes the experimental evaluation of our shape matching algorithm and provides a comparison of our results with those in [3], i.e., the shape matching algorithm which is the most similar to ours. We use the well-known Brown [12] dataset to test our algorithm. The Brown dataset has 99 images divided into 9 categories (each of which has 11 images). The Brown dataset is challenging because it includes both occluded images and images with missing parts. Sample images from the Brown dataset are illustrated in Figure 4. We use the standard evaluation method to test our algorithm. For each image, we count how many of the 10 best matches (i.e., as defined by the smallest cost) belong to the same category. For our algorithm, the detected mesh with the smallest cost must also satisfy (2). We use 25 pixels for the distance between successive boundary vertices. The grid size in the target images is set to 30x30. We use a=4b in (1). The threshold value θ is set to $d_{max,t}/3$. Our experimental results show that choosing a value of θ between $d_{max,t}/3$ and $d_{max,t}/2$ gives good matching results.

Figure 5 shows some of the good matching results obtained with our algorithm. For these source images, both our algorithm and Felzenszwalb's algorithm show good matching results. However, our algorithm shows slightly better matching results. Felzenszwalb's algorithm shows incorrect matching results for some

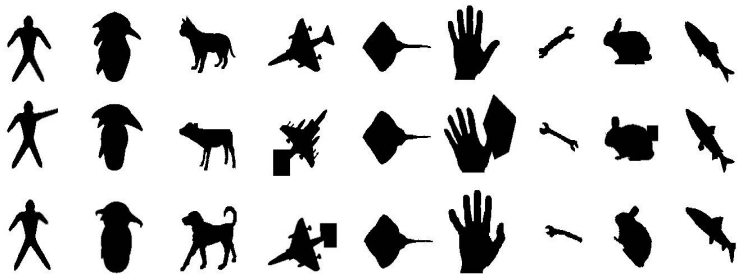


Fig. 4. Sample images in the Brown dataset [12]. Three sample images are shown per category.

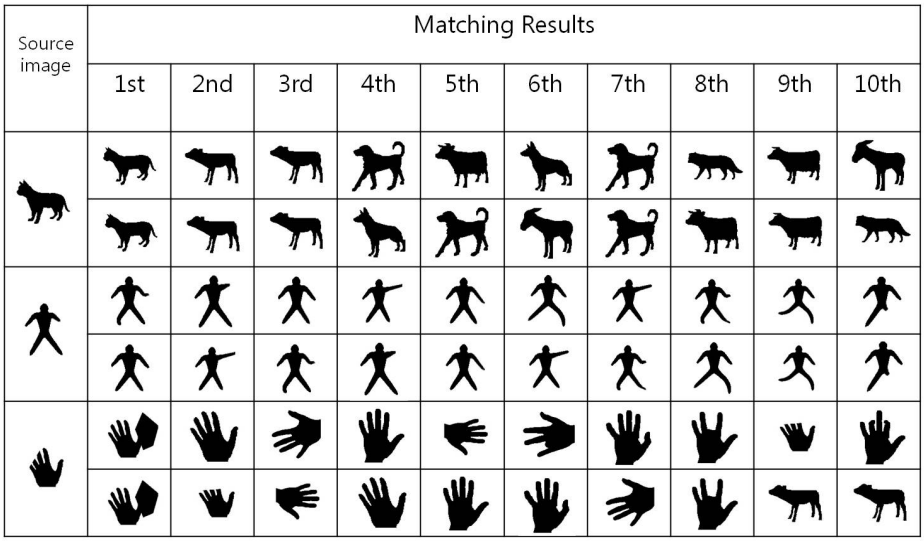


Fig. 5. Good shape matching results for the Brown dataset on three source (query) images and comparison with [3]. For each source image, the 10 best matching results are shown with the smallest (left) to the largest (right) energy. The top figures in each group represent the matching results obtained from our algorithm, whereas the bottom figures in each group represent matching results using the algorithm in [3]. For these experimental sets, only two matching results of [3] (i.e., the bottom right images) fail to match.

cases (e.g., the bottom right images). Interestingly, we see that our algorithm and Felzenszwalb's algorithm show different rank orderings among the 10 best matching results. Figure 6 shows some poor matching results for our matching algorithm on the Brown data set. For these images, Felzenszwalb's algorithm fails in most cases and succeeds in only a few instances.

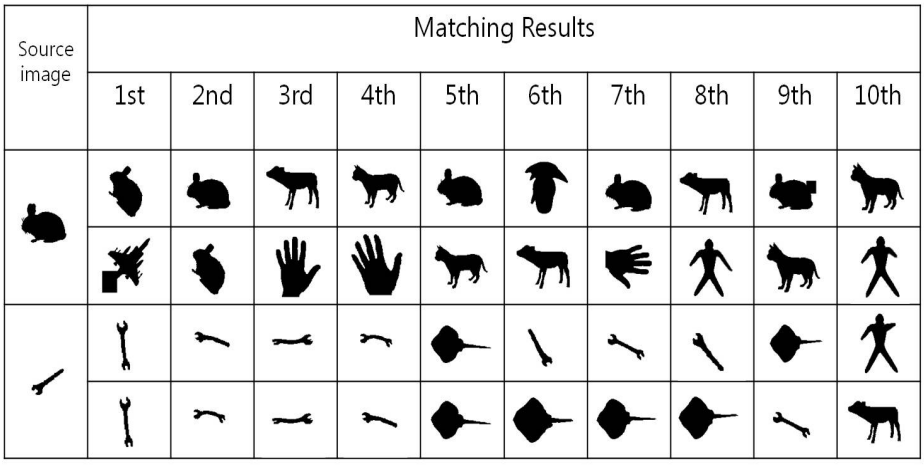


Fig. 6. Poor shape matching results for the Brown dataset on three source (query) images and comparison with [3]. For each source image, the 10 best matching results are shown with the smallest (left) to the largest (right) energy. The top figures in each group represent the matching results obtained from our algorithm, and the bottom figures in each group represent matching results using the algorithm in [3]. For this experimental data set, both our algorithm and [3] show poor matching results. However, our algorithm shows better matching results than does the method in [3].

The recognition rates comparing our algorithm with Felzenszwalb's algorithm are shown in Table 1. The recognition rate is defined as the ratio of the total number of correct hits to the total number of correct hits possible [7]. For the Brown data set, the total number of correct hits possible is 99*10 since there are 99 images, and we find the 10 best matches. Our algorithm yields a 7.8% higher recognition rate when compared with Felzenszwalb's algorithm on the Brown data set. Several matching results, which include detected meshes on the target image, are shown in Figure 7. We observe that the algorithm in [3] produces poor matching results when detected meshes (triangles) are placed in poor positions within a target image. This often results in a large D value. This occurs because [3] matches the target image only by considering the local properties of each triangle such as shape similarity and the edge boundary. However, our algorithm produces better matching results by using global image features as shown in Figure 7.

Table 1. Recognition rate results on the Brown dataset.

Method	Recognition Rate
Felzenszwalb's algorithm [3]	64.4 %
Our algorithm	72.2 %

4 Conclusions and Future Work

We have proposed a shape matching algorithm for deformable objects using both local and global shape information. In particular, we employ the center of mass of an image as a global image feature. Similar to the algorithm in [3], our algorithm does not need initialization to detect deformable objects. Experimental results show that Felzenszwalb's algorithm [3] sometimes produces poor matching results because it only considers the local properties of each triangle such as shape similarity and the edge boundary. However, our method produces improved matching results using a new energy function term and improved dynamic programming based upon global image features. Experimental results on the Brown dataset show a 7.8% higher recognition rate when compared to Felzenszwalb's algorithm. To further improve the recognition rate, we plan to use the symmetry of an image for deformable shape matching as in [13].

Acknowledgements

The authors have benefitted from helpful conversations with Robert Collins of The Pennsylvania State University. The work of the second author was funded in part by NSF grant CNS 0720749 and an Institute for CyberScience grant from The Pennsylvania State University.

References

1. A. Yilmaz, O. Javed, and M. Shah, *Object Tracking: A Survey*, ACM COMPUT SURV, 38(4), pp. 1–45, 2006.
2. T. Yeh, K. Tollmar, and T. Darrell, *Searching the Web with Mobile Images for Location Recognition*, IEEE CVPR, pp. 76-81, 2004.
3. P.F. Felzenszwalb, *Representation and Detection of Deformable Shapes*, IEEE PAMI., 27(2), pp. 208-220, 2005.
4. M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active Contour Models*, Int. J. Comput. Vision, 1(4), pp. 321-331, 1988.
5. T. Chang and T. Liu, *Detecting Deformable Objects with Flexible Shape Priors*, IEEE ICPR, pp. 155-158, 2004.
6. N. Payet and S. Todorovic, *Matching hierarchies of deformable shapes*, GBR, pp. 1-10, 2009.
7. P.F. Felzenszwalb and J.D. Schwartz, *Hierarchical matching of deformable shapes*, IEEE CVPR, pp. 1-8, 2007.

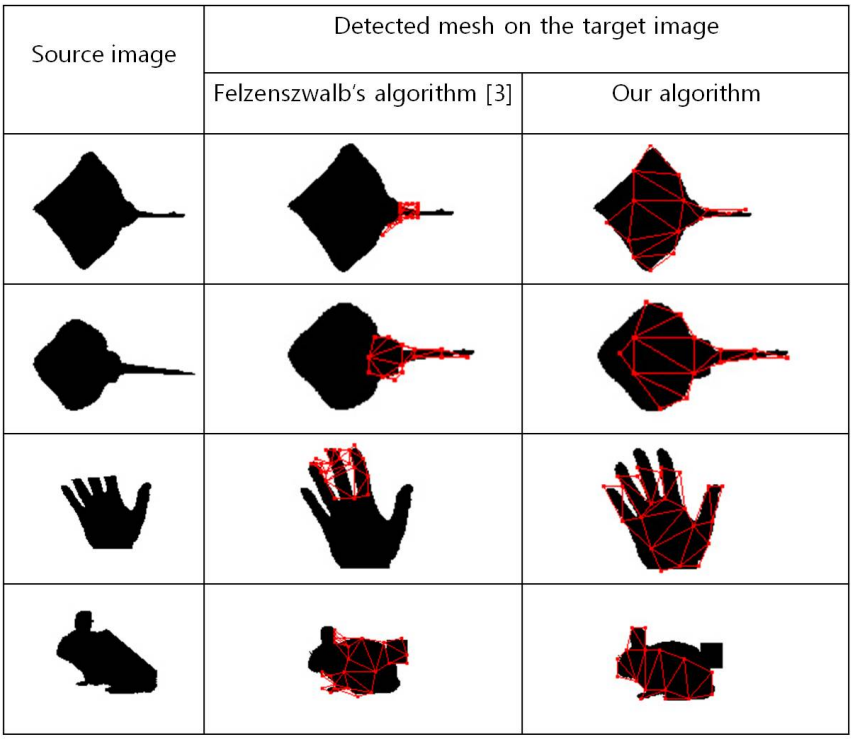


Fig. 7. Example matching results including detected meshes on the target image using images from the Brown dataset [12]. For this experiment, Felzenszwalb's algorithm shows poor matching results because triangles are placed at poor positions.

8. R. Seidel, *Constrained Delaunay Triangulations and Voronoi Diagrams with Obstacles*, Technical Report 260, Inst. for Information Processing, Graz, Austria, pp. 178-191, 1988.
9. J.R. Shewchuk, *Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator,* Lecture Notes in Computer Science, vol. 1148, pp. 203-222, 1996.
10. M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.
11. B. Widrow, *The rubber mask technique*, Pattern Recognition, 5(3), pp. 174-211, 1973.
12. T.B. Sebastian, P.N. Klein, and B.B. Kimia, *Recognition of shapes by editing their shock graphs*, IEEE PAMI., 26(5), pp. 550-557, 2004.
13. S. Lee and Y. Liu, *Symmetry-Driven Shape Matching*, Penn State Technical Report CSE 9(11), pp. 1-22, 2009.