Structure-oriented Contour Representation and Matching for Engineering Shapes

Suyu Hou [a], Karthik Ramani [a,b,*]

[a] Purdue Research and Education Center of Information Sciences in Engineering (PRECISE)
School of Mechanical Engineering, Purdue University
West Lafayette, U.S.A

[b] School of Electrical and Computer Engineering (by Courtesy), Purdue University
West Lafayette, U.S.A

Abstract

Conventional shape matching for engineering models primarily considers rigid shape similarity. They do not seek global shape similarity while considering large local deformations. However, engineering models created by some parametric-based design can involve large parametric changes. As a result, they do not share similarity in their global shape. Hence our goal is to develop shape representations for global matching of part models that can have large dissimilarity through stretching and/or bending.

This paper presents a strategy of an integrated shape matching for contours of engineering drawings inspired by the divide and conquer paradigm. The original shape is decoupled into two levels of shape representations namely, higher level structure and lower level geometry. The higher level structure matching is then achieved driven by optimal integrated solutions from matching of lower level local geometry. Feature points are first extracted using curve evolution to attain the two levels of representations. In order to suit engineering semantics, a new significance function for a point is defined to suppress small features using discrete curve evolution. To conduct the integrated shape matching, a mechanism of using lookup tables is employed to associate these two levels of representations. Dynamic Time Warping and Elastic Matching are employed at different levels of shape representations in order to achieve the optimal integration. To demonstrate the advantages of the proposed work for engineering shapes, experiments for contour evolution, feature point registration, and shape-based similarity for retrieval are conducted. They are also compared with the existing methods. The experimental results show that the structure-oriented contour representation and matching is more meaningful and consistent from an engineering perspectives.

Key words: Shape matching; Feature point extraction; Non-rigid deformations; Contour; Engineering drawings

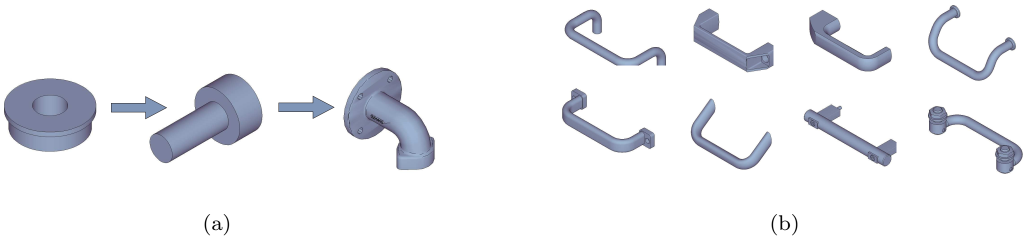
1. Introduction

Engineering drawings have played an essential role in carrying engineering specifications throughout the product lifecycle. As the start of 2D and most 3D part modeling, sketches and embodiment drawings serve to express and communicate ideas at either the earlier stage of design or the manufacturing process, respectively. Among the areas of specified knowledge, 2D geometric constraints and related structure information possess strong associations with product modeling including design intent, manufacturing constraints and operations, functional parameters, and so on. Therefore, retrieving and reusing past engineering drawings automatically is an important task for the engineering community.

Matching the shape of engineering drawings [1] has caught attentions from many research fields in recent years [1–5]. However, a primary shortcoming of existing engineering shape matching is that they seldom seek global similarity while considering large local deformations. Besides, there is limited practice in conducting shape matching by incorporating the notion of the geometric constraints or the structure content. Therefore, engineering parts with large local disparity, although similar in global structure, do not share similar shape representations. Hence, it is necessary to de-

[*] Corresponding author.
Email address: ramani@purdue.edu (Karthik Ramani).

[1] In this paper, we refer engineering drawings as all kinds of 2D visual data in binary form for engineering parts including CAD drawings, scanned images and orthogonal projection views of 3D engineering parts.


(a)

(b)

Fig. 1. Motivation for structure-oriented shape matching. (a) Examples resulting from parametric-based modeling. (b) Examples from the handle class of ESB [6] to illustrate the structure-based shape classification.

velop shape representations and corresponding matching strategies that can retrieve engineering drawings for such a purpose. In order to achieve this goal, we explore the structure content embedded in the shape while allowing large local variations through stretching and bending. For this study, we define local deformation as the change of form by stretching or bending of local features, such that the result of the deformation is distinct from the original. Here local features refer to those geometric entities that encompass topology and geometry. In this paper, shape is defined as a closed external boundary or contour from engineering drawings. First, the contour carries engineering significance by embedding the topology through vertex/point connectivity, and some geometric constraints such as perpendicularity and tangency by local geometry. Second, the contour captures significant visual parts of the objects including both the global structure and the local geometry. Besides, it articulates the deformations from both the structural and geometric perspectives. Hence, the study of the contour will support the targeted structure-oriented shape representation and matching.

The structure-oriented shape matching proposed in this paper is motivated by its engineering utilities as described below:

Parametric-based modeling : Geometric constraints and related structure information serve important roles in variant-based design such as feature-based parametric modeling. In Fig. 1(a), the part in the middle is derived from the part on the left by local stretching, and becomes the part on the right by further local bending. However, these local deformations throughout the process do not change the structure or even some of the local geometric constraints completely at each stage. It is because of the insensitiveness to local geometry alterations of the structure that drives the reuse of the existing parts. The feasibility of adapting local geometry to create new parts motivates retrieval of parts using structural similarity.

Partial dissimilarity recognition : It has been an important task to identify what has been changed on different versions of the same part during product lifecycle. For example, design engineers may change the geometry to optimize the performance during analysis, while manufacturing engineers may revise the geometry to address manufacturing concerns.

Shape-based classification : Shape-based structure information has been an essential property for similarity judgment and therefore, a shape-based classification criteria. Examples in Fig. 1(b) not only share a similar appearance, but also possess a similar functionality and maybe manufacturing process. In many classes of shapes, users may have classification criteria not just based on rigid similarity but also allowing stretching, bending, or even adding and removing local features within the classes.

Partial matching : It is a common interest for people to look for global shape similarity, local similarity or dissimilarity for potential knowledge reuse. Therefore partial matching is a good motivation to this study.

2. Related Work

In this section, we review some of the related work from the fields of computer vision and engineering.

2.1. *Technical drawing retrieval*

This paper addresses the problem of matching engineering drawings using shape content. 2D engineering shape matching has been under study using various shape descriptors.

Graph-based methods : Fonseca *et al.* [2] proposed a graph-based shape descriptor to encode spatial relations such as adjacency and inclusion among the components extracted from the drawing. Graph spectrum was then used for matching. Park *et al.* [4] also proposed a graph-based descriptor to represent spatial compositions of union and subtraction among the dominant components extracted from the drawing. However, the components defined have a limited capability to deal with freeform shape. Both methods are built upon the idea that the content stored in each node and each edge of the graph can be extracted from the drawings. However, these methods may encounter difficulties when the drawings under study do not possess the properties to construct the graph.

Feature vector-based methods : Compared with the graph-based shape representation, this category has more varieties in representing the shape. Among the various approaches, one of them is *string-based* representation: Gero *et al.* [7] represented the shape of an architecture drawing by encoding the convexity and concavity of the boundary in a string representation. Then they used string matching to retrieve similar drawings. Bai and Xu [8] recognized elements along a boundary by simulating a chain code for boundary representation. In addition to convexity and concavity, geometries such as angle, length, or area are also important properties for contour-based shape matching. Berchtold *et al.* [9] represented the contour using the above geometric properties under different *parameterization-based* methods such as Mehrotra-Gary, angular profile, and section coding. In addition, Berchtold *et al.* [3] developed an extended feature objects built upon Fourier descriptor to perform partial matching. Besides the above two groups of methods, *statistics-based* methods have also been applied in engineering drawing retrieval. Pu and Ramani [1] used spherical harmonics and histograms to conduct engineering drawing retrieval, which showed robustness for most kinds of 2D drawings. Müller and Rigoll in [5] employed pseudo 2D Hidden Markov Model to retrieve user-specified shapes. However, the feature vector-based approaches represent the overall shape while mixing global structure and local geometry. Therefore, they resulted in unexpected and inconsistent retrievals.

2.2. *Contour-based multimedia image retrieval*

There are some methods using skeleton graph [10] and shock graph [11] for structure matching. However, to be more specific to this context, we only review those using contour/boundary as the subject for the structure-based shape matching. Many efforts have been taken to compare contour-based shape descriptors such as Safar's work in [12]. Well known classical methods for 2D contour matching include curvature scale space [13,14], chain code [15], turning function [16], and Fourier descriptors [17]. Specifically, there are two major approaches for the structure-oriented contour matching: one is the *similarity distance-based* method, which relies on matching selected points using distance functions. The other is the *morphing-based* method which typically involves finding the mapping between two sets of the selected points that minimizes the energy needed to deform a set of points into another set of points. For both, the first step is to sample the feature points from the boundaries which can either be random/uniform points or points with geometric significance. The second step is to find an optimal warping path that the corresponding feature points transverse.

Similarity distance-based : Various papers have been published on contour matching using similarity distance between feature points. Belongie *et al.* [18] developed a local shape descriptor, called shape context, to identify the best correspondence between the sampling points from the query and the ones from the target. Various studies have extended Belongie's work: Jain and Zhang [19] used geodesic shape context to find the feature point correspondence; Zheng and Doermann [20] incorporated relaxation labelling to optimize the graph matching of the shape context; Grauman and Darrell [21] used Earth Mover's distance to calculate the shape context similarity between two feature points. Latecki [22] developed a strategy of using a group of convex or concave segments to recognize the visual correspondence between the query and the target. The similarity between the corresponding parts is computed using the classical turning function. Liu *et al.* [23] developed a list of local geometric properties as the shape descriptors for the feature points. Then they used Dynamic Time Warping (DTW) to find the best feature point correspondence. Gdalyahu *et al.* [24] treated the contours as strings and employed string edit operations to identify the optimal feature point alignment during the DTW matching process. Grigorescu *et al.* in [25] proposed to use a local shape descriptor of a feature point, called distance set, to find the feature point correspondence. The distance set was composed of the relative distances between the feature point and its K nearest neighbors. Both Petrakis *et al.* [26] and Scott [27] devised strategies to adapt DTW for the special purpose of contour matching. Petrakis excluded cases where the feature points have no possibility of matching, while Scott enforced order preserving conditions during the matching process.

Morphing-based methods : These methods have been used recently in curve matching under non-rigid deformations. The idea was derived from a physics analogy to identify the minimum total work needed to deform one curve into another. Two major mathematical models to calculate the morphing energy have been used in recent years, one of which is Elastic Matching (ET) and the other is Thin Plate Spline (TPS) model. Basri [22] developed a spring model, a linear model, and a continuous model to include both stretching and bending energy into the cost functions of continuous contour matching. Sebastian [28] employed another ET function for continuous contour matching. The alignment of the starting points was considered for better matching. Sederberg [29] developed a deformation cost function including stretching and bending energy for discrete contour blending. Singh [30] used the deformation cost defined in [29] for contour matching. Strategies based on triangle inequality were used to reduce the computational cost for matching. The TPS model has been popular for non-rigid registration recently. Unlike the ET, the TPS model does not include the computation of the stretching energy explicitly. Chui and Rangarajan [31] proposed an optimization framework of the TPS model for robust point matching using simulated annealing and a softassign algorithm. Similarly, Wang *et al.* [32] optimized the TPS model by landmark sliding. However, the existing techniques have only been applied to multimedia contour matching.

In summary, there has been no distinctive effort to first decouple the structure from the overall shape by excluding some geometry, and then retrieve the geometry ignored before for structure matching optimization. In addition, geometry constraints are not of any concern in the existing work and there has been no study in engineering shape matching society to address non-rigid shape similarity.

3. Overview of Contour Representation and Matching

In this section, we present the major procedures involved in developing the proposed work. Our target is to conduct the contour matching in conformity with engineering semantics in addition to be invariant to scaling, translation, and rotation. To be more specific, the shape representation should conform to the topology and the geometry supported by engineering semantics; the matching result should be consistent with engineering interpretation: in this case overall structure similarity with tolerance to local stretching and bending. The following part presents the algorithm along with the pseudo code and the explanation for each procedure. The details are presented in Section 4.

Contours are first extracted from images of the engineering drawings. Polygon representations of the contours are constructed next. The primitives of the polygon are line segments with attributes being length and absolute orientation. We scale the image before polygonization for the purpose of normalization. Our algorithm then captures the higher-level structure by extracting the feature points which manifest themselves as the end positions of the geometric primitives in an engineering drawing. In this paper, structure is reflected from the topology defined by these feature points and the local geometry carried by these points. Matching these feature points indicates possible structure conformity between two contours. Nevertheless, the information carried by the feature points does not support every detail of the shape matching. A geometry comparison is conducted based on the results from the candidate feature point correspondence. Each case of the feature point correspondence depends on the choice of starting points from both contours, where rotational alignment is considered. Therefore, iterative processes of the feature point correspondence and the geometry comparison are performed in order to exhaust the rotational alignments stored beforehand. The structure matching is optimized when the system achieves its minimal integrated cost at local geometry comparison.

More specifically, let P and P' be the representations of two contours, with $P = \{p_i | i = 1, ..., N\}$, $p_i \in R^2$, a list of N consecutive points and $P' = \{p'_{i'} | i' = 1, ..., N'\}$, $p'_{i'} \in R^2$, a list of N' consecutive points. A cyclically ordered list of N segments $S = \{s_1, s_2, ...s_N\}$ is derived from P with $s_i = (p_i, p_j)$, i and j are two consecutive indexes in P. Similarly, $S' = \{s'_1, s'_2, ...s'_N\}$ can be derived from P'.

Algorithm 1 Contour Matching

Input Contour representation P and Contour representation P'.

Output Correspondence Γ between feature points and similarity cost $cost^*$.

```
$F, F' \leftarrow$ FEATURE-POINT-EXTRACTION($P, P'$)
$\Psi \leftarrow$ INITIALIZE-STARTING-PAIRS-FEATURE-POINTS($F, F'$)
$cost^* \leftarrow \infty$
for $i = 0$ to $|\Psi|$ do
    $\hat{\Gamma} \leftarrow$ FEATURE-POINT-REGISTRATION($P$,$P'$,$F$,$F'$,$\Psi_i$).
    $\hat{cost}^* \leftarrow$ INTEGRATED-GEOMETRY-COMPARISON($P$,$P'$,$\hat{\Gamma}$)
    if $\hat{cost}^* < cost^*$ then
        $\Gamma \leftarrow \hat{\Gamma}$ and $cost^* \leftarrow \hat{cost}^*$.
    end if
end for
return Correspondence $\Gamma$ and similarity cost $cost^*$
```

The procedure FEATURE-POINT-EXTRACTION extracts those points that contain engineering implications. In image processing, feature points are defined as points with curvature extreme, cusp, inflection points, and the discontinuities of curvature because they usually carry perceptual significance [23]. In this paper, we define feature points as those points that are end points of lines and arcs. [2] Based on this definition, points carrying geometric constraints such as perpendicularity and tangency are included in feature points. On the other hand, the initial data P and P' at hand usually includes some features of small protrusions or depressions as well as noise. Their existence perturbs the intended structure matching. In this paper, we adapt the idea of discrete curve evolution from [33] to extract defined feature points. Given P at the initial stage, we delete the point carrying the least significance in a stepwise manner until the curve evolution breaks the stopping condition. Full descriptions are given in Section 4.1.1 to explain the defined significance function and the stopping condition. The end of the evolution gives a set of feature points $F = \{f_k | k = 1, ...K\}$, where k maps to the index of P through a look-up table described in Section 4.1.2. The same procedure applies to P' and all the other database models as well. The result from extracting the feature points of P' is $F' = \{f'_{k'} | k' = 1, ...K'\}$.

The procedure of INITIALIZE-STARTING-PAIRS-FEATURE-POINTS prunes the search of the starting pairs of feature points and returns a list of candidate pairs $\Psi_i = (f_k, f'_{k'})$ where f_k and $f'_{k'}$ share similar local geometry. Matching P and P' uniquely determines the relative global rotation between two contours. We assume that the optimal alignment is well approximated by at least one of the KK' possible selections of f_k and $f'_{k'}$. The details are presented in Section 4.1.1. This procedure leaves us with a set of $\Psi \subseteq F \times F'$ candidate global alignments, such that normally $|\Psi| \leq KK'$ and $|\Psi| \ll NM$.

Given a member $\Psi_i = (f_k, f'_{k'})$ from Ψ, the procedure of FEATURE-POINT-REGISTRATION will establish

[2] In this paper, we regard polylines as a list of line segments. In addition,we only deal with simple curves of uniform curvature or compound curves composed of simple curves.

the candidate correspondence path $\hat{\Gamma} = \{(f_k, f'_{k'})_r | r = 1, ..., \hat{R}, f_k \in F, and\ f'_{k'} \in F'\}$. In reality, not every feature point has its corresponding point for matching. We therefore employ the relaxed DTW to identify the optimal path such that the matching cost between F and F' is minimum. Full descriptions of this procedure are presented in Section 4.2.1. The procedure of INTEGRATED-GEOMETRY-COMPARISON computes the total matching cost between two contours fed by the path of $\hat{\Gamma}$ from FEATURE-POINT-REGISTRATION. The result of $\hat{\Gamma}$ will segment P and P' into $|\hat{R}|$ groups of geometric entities. We use ET to quantify the dissimilarity of each pair of groups. The total cost integrated from overall geometry comparison is then used as an indicator of the best correspondence between F and F'.

The search for the best Γ and the least $cost^*$ continues until Ψ is exhaustively visited. The set of procedures inside the loop is the core part of the structure-oriented shape matching. The prior intention of the structure-oriented shape matching is clearly indicated inside the algorithm. That is, we establish the feature point correspondence using only structure-related content and evaluate the similarity cost by retrieving all the geometric content involved. We do not use the cost from the structure matching as the function for evaluation because its influence is embedded in the computation of the integrated geometry comparison. In addition, the integrated geometry comparison is a more comprehensive evaluation of the overall matching.

The bottleneck of Algorithm 1 is from the **for** loop where K and K' are the main factors for the computational complexity. The size of the **for** loop is dependent on the size of Ψ which has an upper bound of KK'. Inside the **for** loop, it is $O(KK' + max(K, K'))$ to compute the matching cost using DTW, and $O(min(K, K'))$ to compute the integrated geometry comparison using ET. Overall, the complexity from the **for** loop is $O(K^2K'^2)$. Outside the **for** loop, the complexity is $O(N + N')$ for feature point extraction and $O(KK')$ for the procedure of initialization. Section 4 will explain the reasons. Finally, the overall computational complexity of Algorithm 1 is $O(N + N' + K^2K'^2)$. The real computational time of the proposed algorithm is lower than what the complexity indicates. The number of feature points K is limited in reality. This is because our work only focuses on engineering shapes with obvious structure and low frequency of small features, as is explained in Section 5. Therefore, the real computation involved is manageable, even though the complexity is $O(K^2K'^2 + N + N')$.

4. Methods of Contour Representation and Matching

In this section, we present the details of the various procedures involved in Algorithm 1. Section 4.1 explains how the two levels of shape representation are constructed and associated using extracted feature points. Section 4.2 presents the process of matching with two levels of shape representation.

4.1. *Structure-oriented contour representation*

4.1.1. *Feature point extraction*

Feature points are extracted to formalize geometric entities to reflect the structure. It is important to locate those desired feature points without changing their local geometry. Therefore, it is not applicable to extract interesting points by involving smoothing which will change the positions of the points. There are several related works on finding so-called critical points from polygonal curves without changing the positions of these critical points. Marji and Siy [34] used Region of Support to identify dominant points. Latecki and Lakämper [33] evolved polygon contours by removing the least significant point stepwise. Similar to [33], Zhuang *et al.* [35] employed a method called Conservative Bounding Contour for the same purpose. They employed a sliding circle to remove points on contours of CAD drawings. However, the simplification was dependent on the radius of the sliding circle.

We adopt the method of discrete contour evolution from [33] because it helps to remove small features and noises while extracting the feature points. In addition, this method does not depend on any extra parameters, but only the intrinsic geometry. At each evolution step, a pair of consecutive segments s_i and s_{i+1} is replaced with a single segment connecting the endpoints of s_i and s_{i+1} if the relevance measure defined in Eq. (1) is the minimum.

$$K(p_i) = \frac{\beta(s_i, s_{i+1})l(s_i)l(s_{i+1})}{l(s_i) + l(s_{i+1})}\ , \tag{1}$$

where $\beta(s_i, s_{i+1})$ is the turning angle at the joint point p_i of s_i and s_{i+1}; $l(s_i)$ is the length of s_i normalized with regard to the perimeter of the contour. The main property of the relevance measures quantifies the contribution of the joint point of s_i and s_{i+1} to the overall shape. However, the relevance measure does not consider the relative region around the interesting points. Instead, it only takes into account of the absolute relevance at the points in question during the evolution. Therefore, smoothness on an arc is not well-maintained and will cause false positive feature points later on. At the same time, small perturbations may resist evolution because of their high relevance measures compared with those of points on smooth curves. (see Fig. 2(a) and Fig. 2(c))

The algorithm of how feature points are attained is described in Algorithm 2, with details of each procedure explained in the following parts. Let $P^m = \{p_i | i = 1, ...N^m\}$ be a contour representation at the m_{th} stage of the evolution. Initially, m is set to 0.

Relative Relevance Measure RK: We compute K of each point in P using Eq. (1) at first. In this paper, we use the signed turning angle instead of the absolute turning angle. The sign of the turning angle is related to the traverse direction of the contour, which is indicated by the direction vector $\overrightarrow{s_i}$ of s_i. Top n candidate points that have the smallest $|K|$ are selected to compute RK, where $n = 25\%N$ by default. We then choose the point that has the biggest RK

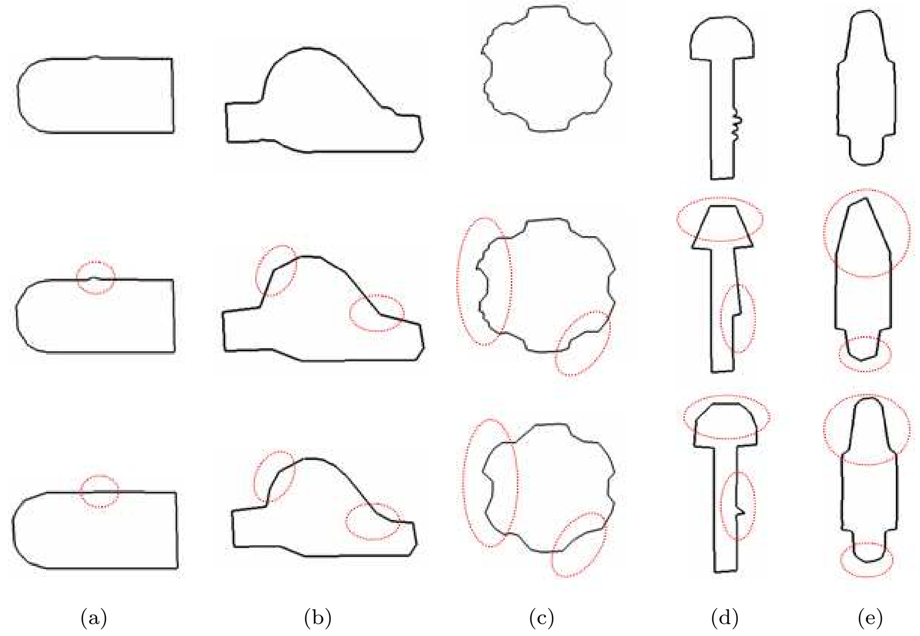


Fig. 2. (a)-(e) Comparison of contour evolution results for examples (1^{st} row) using the Relevance Measure in [33] (2^{nd} row) and the proposed Relative Relevance Measure (3^{rd} row). Red circles highlight the portions where the process of evolution are different between these two methods.

Algorithm 2 Feature Point Extraction

Input A contour representation P^m

Output A set of feature points F.

```
$k \leftarrow m$
repeat
    $p_i \leftarrow$ RELATIVE-RELEVANCE-MEASURE($P^k$)
    $P^{k+1} \leftarrow P^k$ with $p_i$ removed from $P^k$
    $k \leftarrow k+1$
until STOPPING-CONDITION-CHECK($P^k$,$P^{k-1}$,$P^0$)
$F \leftarrow$ FINALIZE-FEATURE-POINT($P^k$)
return $F$
```

for the stepwise evolution until the steady state defined later is broken.

The Relative Relevance Measure RK defined in Eq. (2) is obtained as follows: we first pick up a region around the point p_i using a circle of radius r, where $r = \rho$ if $\rho > \max(l(s_i), l(s_{i+1}))$ or $r = \max(l(s_i), l(s_{i+1}))$ if $\rho < \min(l(s_i), l(s_{i+1})), (\rho > 0)$. As a result, a number of L points from P are included in the region on the left-hand-side of p_i, and a number of R points from P are included in the region on the right-hand-side of p_i [3].

$$RK(p_i) = \max_{k=-L}^{R}(|K(p_i) - K(p_{i+k})|) \; . \tag{2}$$

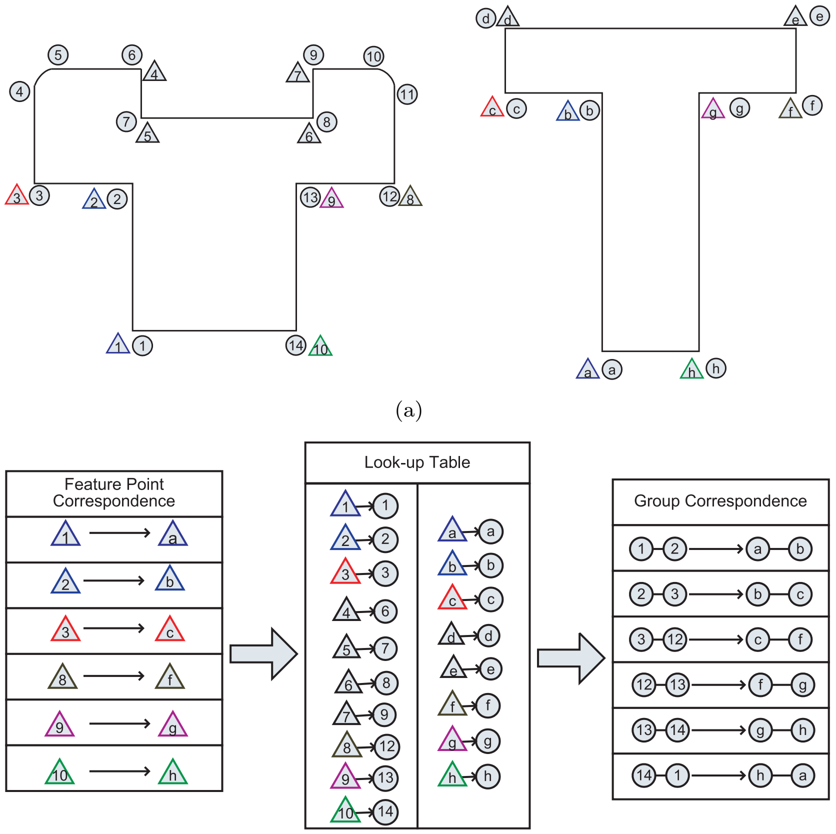
The idea behind RK is to quantify the dominance of those points carrying small visual significance measured by K. The reason to choose max function is to find out the maximal visual difference between the points in question and their neighbors. Points on an arc usually share similar K with its neighbors, while points on small perturbations have opposite signs of K to at least one of their neighbors, which will incur a high value of RK. This is especially true when small perturbations are located on long line segments (see Fig. 2(a)).

Stopping condition for contour evolution: The contour will finally converge to a line segment if there is no stopping condition exerted on the evolution process. We employ a self shape similarity measure Υ defined in Eq. (3) to determine the stage at which the steady state is broken, i.e. the change of the overall shape is significant enough to be larger than the threshold of Υ_k.

$$\Upsilon_k = \frac{\Delta(f(P^k), f(P^{k-1}))}{f(P^0)} \quad (k >= 1), \tag{3}$$

where Δ is a distance function and f is a shape function on contour. In this paper, we use L_1 distance and the area of the contour as the default. The self shape similarity measure defined above takes the relative change into account. The evolution process stops at $\Upsilon_k > \gamma$ where γ is an experimental parameter. Fig. 2 shows the comparisons of the contour evolution using the method in [22] and the proposed method on engineering drawings. The stopping condition defined by Eq. (3) is the same for both methods regarding each example. The performance demonstrates its advantages over the existing method and the results are more meaningful from an engineering perspectives. The proposed contour evolution with the definitions of RK is more stable at smooth curves than the method in [33]. It gives more priority to remove small perturbations than to merge segments on curves during evolution as is demonstrated from Fig. 2(a) to Fig. 2(d). Based on our observation, the pro-

[3] In the implementation, $k = (k+N) \mod N$ for an arbitrary index k where N is the number of points in P.



(a)

(b)

Fig. 3. Two levels of shape representation and association: indexes inside circles are from P; indexes inside triangles are from F; colors indicate feature point correspondence. (a) Lower level original point set and higher level feature point set. (b) Lookup table to associate two levels of data representations.

posed method is also more stable when Υ_k is relaxed (see Fig. 2(e)). This is because the method in [33] may select a point on a smooth curve to be deleted at one step. The deletion may cause a small change of the shape which may be within the acceptance of Υ_k. However, a small relaxation of Υ_k may cause a significant change of the shape by allowing accumulation of small changes of the shape. On the contrary, the proposed evolution method takes priority to select a point from a perturbation during one evolution step, which may cause a large change of the shape. Therefore, a small relaxation of Υ_k is not large enough to allow the deletion of the selected point, thus keeping the evolution stable under small relaxation of Υ_k.

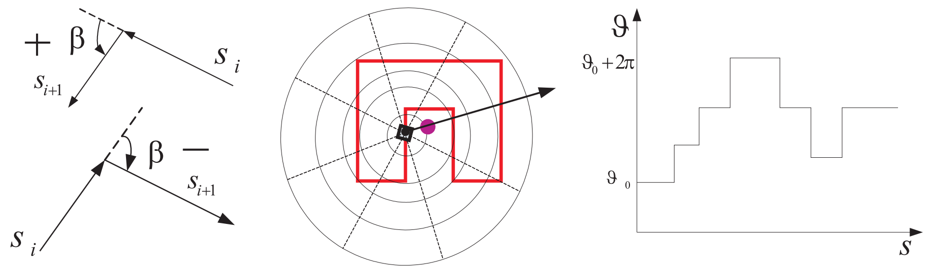
Feature point finalization: Recall from the feature point definition that points on an arc except the end points are not regarded as the feature points because their existence is mainly due to the process of polygonization and the evolution. Besides, our desire of the structure matching without influence from local bending will have difficulties if these points are regarded as the feature points. In the procedure of FINALIZE-FEATURE-POINT, points with turning angle $|\beta(p_i)| < \alpha$ are not included in F if they also satisfy the heuristic conditions below:

$$\beta(p_{i-1}) \approx \beta(p_i) \approx \beta(p_{i+1}) \cap \frac{\min(l(s_i), l(s_{i+1}))}{\max(l(s_i), l(s_{i+1}))} \geq \varsigma\,. \quad (4)$$

The bottleneck of Algorithm 2 is from the repeated processes of the contour evolution. The computation cost inside each process is $O(N)$. The number of iterations is dependent on the result from STOPPING-CONDITION-CHECK. In this implementation, Υ_k is set to be a tight threshold in order not to miss any feature point. Therefore, it is unlikely that the times of the iterations is close to N. Hence, the system takes only a polynomial complexity of $O(N)$ in this procedure.

4.1.2. *Two levels of shape representation and their association*

We decouple the original shape into two levels of representations: a set of feature points $F = \{f_k | k = 1, ...K\}$ and groups of segments $G = \{g_r | r = 1, ..., R\}$, where g_r is a list of segments derived from P. The key to link F and G is the use of a look-up table $T = \{t_k | k = 1, ..., K\}$, where t_k stores the corresponding index of F in P. The association is established during the decoupling process in Section 4.1.1. The relationship is that the end points of g_r and g'_r are two consecutive pairs of feature points on the correspondence path Γ. For example, let $(f_k, f_{k'})$ and $(f_{k+l}, f_{k'+l'})$ be two consecutive pairs of feature points on Γ resulted from the structure matching of F and F'. As a result, two groups $g_r = \{s_r | r = t_k, ..., t_{k+l}\}$ and $g'_{r'} = \{s'_{r'} | r' = t'_{k'}, ..., t'_{k'+l'}\}$



(a) (b) (c)

Fig. 4. Shape functions: a) Turning angle: ± indicates convexity/concavity [16]. b) Shape context [18]: the reference axis defined by the feature point and the center of the shape makes the shape context rotation invariant. c) Turning function [16].

from P and P' are formed. A geometry comparison between the corresponding groups is conducted after the feature point correspondence Γ between two shapes is established. The system regroups the segments each time after establishing the correspondence path in order to evaluate the overall similarity. As a result, $|G| = |G'|$. Fig. 3 illustrates the relationship between the two shape representations. It illustrates the role of the lookup tables between the feature point correspondence from the structure matching and the group correspondence for the geometry comparisons. Notice that the third corresponding group is obtained by ignoring some feature points during the structure matching. The advantage of using two level of shape representation is that the structure matching is not affected by local variations, while the local dissimilarities are resolved through the comparison inside the lower level geometry representations which are groups of segments defined by the feature point correspondence. In this way, we seek a system-level solution by leveraging the higher level abstract with the lower level detail first, and then optimize the solution by resorting to the content ignored before.

Geometric properties of feature points: We adopt two popular shape functions to describe the local geometric property of the feature points. One is the turning angle [16] and the other is the shape context [18]. These two geometric quantities are both invariant to scaling, translation and rotation. Turning angle is an absolute measure of the local geometry. However, it does not consider the relation between the feature point and the global shape. Therefore, two non-corresponding feature points which have the same turning angle may falsely match. Shape context encodes the relative shape of the local feature point with regard to the global shape into a two dimensional histogram. However, a statistical measure can also get a false match due to the nature of statistical computation. The advantage of combining them is to achieve more accuracy for the structure matching. For each feature point in F, we use $\beta(f_k)$ to denote its signed turning angle (see Fig. 4(a)) and $\pi(f_k)$ to its shape context (see Fig. 4(b)).

Geometric properties of a group: The geometry within each group is represented by its turning function $\vartheta(g_r)$ [16] (see Fig. 4(c)). Turning function is an aggregation of turning angles at a set of consecutive sampling points parameterized by the length of the segments. The merits of turning function are that it captures the characteristics of convexity/concavity, perpendicularity/tangency, and smoothness/sharpness along the contour. Besides, it is invariant under translation and scaling. In this paper, rotational invariance is not of concern since each group has a linear sequence of segments. A circle is the only place where the end points of a group are the same. In such a case, the turning function *per se* is rotation invariant. The vertical variation along the turning function can be used to infer the deformation caused by bending, while the horizontal information along the turning function reflects the effects from stretching. A single representation of $\vartheta(g_r)$ can therefore be used for the geometry comparison.

4.2. *Integrated matching with two levels of shape representation*

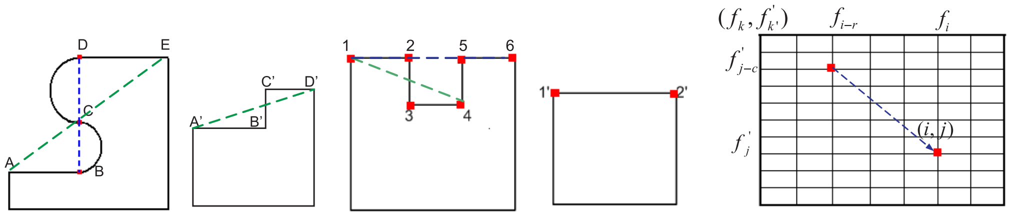
In this part, Section 4.2.1 describes the process of how feature point registration is established for potential matching of the structure. Section 4.2.2 then presents the geometry comparison which is used to select the best structure matching.

4.2.1. *Feature point correspondence*

Similarity cost of two feature points: Similar feature points should have similar geometric properties. In this paper, we use turning angle to narrow down the necessity for the feature point comparison. If two feature points have opposite polarities, i.e. convexity and concavity, the cost between them is set as infinity. Otherwise, the similarity cost between two feature points f_k on F and $f'_{k'}$ on F' is measured by:

$$FeaSimDis(f_k, f'_{k'}) = \chi^2(\pi(f_k), \pi(f'_{k'})) , \qquad (5)$$

where χ^2 is the Hungarian distance between two shape context measures [18]. The similarity distance defined in Eq. (5) is not just used for the structure matching discussed in the later part of this section, but is also helpful in the procedure of INITIALIZE-STARTING-PAIRS-FEATURE-POINTS. In this paper, we choose only convex feature points with $FeaSimDis(f_k, f'_{k'}) < \delta$ as the starting pair of feature points.



(a) (b) (c)

Fig. 5. Examples of merging feature points: (a) $A \Rightarrow A'$ and $E \Rightarrow D'$ by discarding both $B \rightarrow D$ and $B' \rightarrow C'$, or $B \Rightarrow B'$ and $D \Rightarrow C'$ by discarding C. (b): $1 \Rightarrow 1'$ and $6 \Rightarrow 2'$ by discarding $2 \rightarrow 5$, or $1 \Rightarrow 1'$ and $5 \rightarrow 2'$ by discarding $2 \rightarrow 4$; No discarding of 2 or $2 \rightarrow 3$ because of the polarity conflict between them. (c) Computation of the merging cost in DTW implementation.

Merging cost of discarding feature points: In reality, it is unnecessary to enforce every feature point to be registered with another during the structure matching. These constraints may bring undesirable results since the size of F and F' may not be the same in most of the cases. Fig. 5(a) and Fig. 5(b) illustrate the motivation to discard some feature points during the structure matching. However, ignoring some feature points does not work without a penalty. The strategies to define the merging costs for each shape descriptor are different. Eq. (6) and Eq. (7) give the definitions of the merging costs.

$$M_\theta = \sum_{k=p}^{q} \theta(f_k) \tag{6}$$

$$M_\pi = \chi^2\left(\frac{1}{q-p-1}\sum_{k=p+1}^{q-1} \pi(f_k),\ \ \frac{1}{2}(\pi(f_p)+\pi(f_q))\right) \tag{7}$$

The final merging cost to discard feature points from p to q in F is defined as following:

$$MergeCost(p,q,F) = \omega_\theta M_\theta + \omega_\pi M_\pi \ , \tag{8}$$

where ω_θ and ω_π are parameters to leverage the effects from these two shape functions. The merging cost is defined as above because the ignorance of feature points from p to q is equivalent to connecting feature point from $p-1$ to $q+1$. The algebraic sum of the signed turning angle aggregates the cost if all feature points share the same polarity (convex or concave) and cancels the structure change if there is any difference in the polarity (see Fig. 5(a) and Fig. 5(b)). The way it calculates the merging cost from the shape context is to reflect the difference before and after the merge. The intuition behind it is that the difference will be little if the gap from $p-1$ to $q+1$ is small. We then combine the costs using weighted summation to measure the overall effects from merging.

Structure matching by relaxed DTW: The purpose of this part is to establish a path $\hat{\Gamma} = \{(f_k, f'_{k'})_r | r = 1, ..., \hat{R}, f_k \in F, and\ f'_{k'} \in F'\}$ to register F with F' so that the matching cost is minimal. Thus the formulation of the problem is

$$\min_{\Gamma} MatchingCost(F, F', \Gamma, (f_k, f'_{k'})) \ , \tag{9}$$

where $(f_k, f'_{k'})$ is the pair of the starting feature points fed by Ψ. We solve the problem by adopting DTW. In this approach, all point-wise relationships are presented in a $K \times K'$ table where K is the size of F and K' is the size of F'. Each node (i,j) of the DTW table characterizes the relationship between a pair of feature points (f_i, f'_j). It includes its own index (i,j), the index (i_p, j_p) of its parent node, which is the pair of feature points previous to this node along the path, and $node(i,j)$, the minimal cost of matching at this node so far along the path.

In this paper, several considerations are included to relax the matching by DTW: 1) there is no necessity to enforce each feature point to be registered with another, except the first pair of feature points which is designated by Ψ. 2) the parent node is not necessary to be the adjacent previous node to the node of interest on DTW table. The optimal cost of the incomplete path ending at node (i,j) is defined in Eq. (10).

$$node(i,j) = \min_{r,c}(Prevnode + Merge) + FeaSim \tag{10}$$

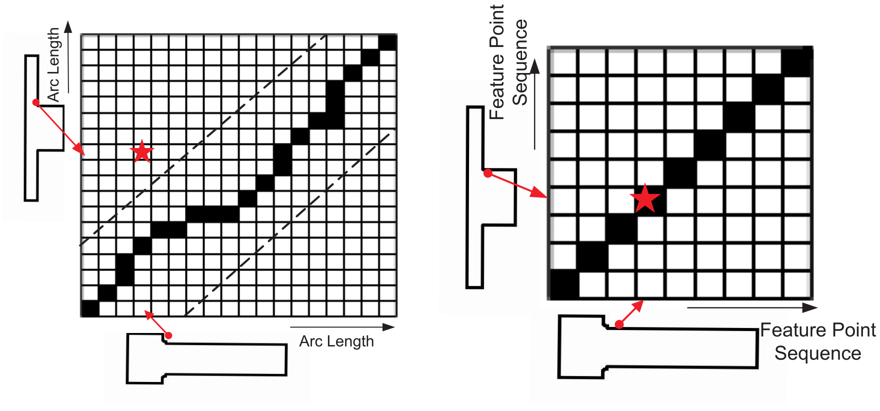
$$Prevnode = node(i-r, j-c) \tag{11}$$

$$FeaSim = FeaSimDis(f_i, f'_j) \tag{12}$$

$$Merge = MergeCost(i-r, i, F) + MergeCost(j-c, j, F') \ , \tag{13}$$

where the optimal cost at node (i,j) is obtained by traversing all possible values of (r,c), $0 \le r \le C_r, 0 \le c \le C_c$. C_r and C_c are constants designated as the number of feature points that can be skipped at most. We choose appropriate C_r and C_c in the implementation in order to make the feature point registration less constrained by its position in F and F'. The specific (r,c) that gives the minimal cost is used to locate the parent node (see Fig. 5(c)). As a result, the complete path that has the minimal cost computed from the DTW table represents the optimal structure matching.

It is worthwhile to point out that the proposed DTW matching uses feature points that are not uniformly sampled from the original shape. They are not even close to each other in geodesic distance space in most of the cases. Therefore, the parameter behind the DTW table in this paper is the feature point sequence rather than the geodesic distance which is the arc length for the 2D contour. Traditional matching of uniform sampling points using DTW is highly dependent on the arc length-based parameterization. Hence, two feature points which correspond to each other in their structure representations may reside too far



(a)

(b)

Fig. 6. Examples to illustrate the purpose of using feature points for structure matching: (a) DTW matching using uniform sampling points: corresponding points do not match because they are out of the constraint region for matching (b) DTW matching using feature points: corresponding feature points match because they occupy close positions in the feature point sequences.

away and not be registered as the result (see Fig. 6(a)). An example in Fig. 6 explains the motivation to use the proposed structure matching strategy for engineering drawings. We only use the geometry properties from the feature point for the structure matching, thus ignoring the effects caused by geodesic distance-based parameterization.

4.2.2. *Geometry comparison*

The process of ET can be treated as deforming a group of segments into another group of segments. In order to conduct ET with both stretching and bending, a pair of corresponding segments has to be provided. In reality, the numbers of segments inside the corresponding groups are different from each other. Therefore, it is hard to conduct ET when the segment correspondences are unknown.

During this implementation, we propose a simple idea to compare the difference between g_r and g'_r. Recall from Section 4.1.2 that turning function $\vartheta(g_r)$ is used to represent the geometry inside a group g_r, so is it for every group in G. Let $\vartheta(G)$ be the concatenation of $\vartheta(g_r)$s. We therefore collect $U \gg N$ values from $\vartheta(G)$ by uniform sampling without breaking $\vartheta(g_r)$s. Similarly, we collect U values from $\vartheta(G')$. Let N_r and N'_r be the number of sampling values inside $\vartheta(g_r)$ and $\vartheta(g'_r)$ respectively, and $N_R = average(N_r, N'_r)$. We then recollect a number of N_R values from $\vartheta(g_r)$ and $\vartheta(g'_r)$ respectively by uniform sampling. In this way, the correspondences are automatically built to support pairwise ET without creating new segments. Let $l_r = L/N_R$, where L is the total length of the segments in g_r, $l'_r = L'/N_R$ for g'_r; $\theta_{r,i}$ be the i_{th} sampling value from the turning function $\vartheta(g_r)$ of g_r, and $\theta'_{r,i}$ for $\vartheta(g'_r)$ of g'_r. The ET adopted from [29] is then defined as follows:

$$ET(g_r, g'_r) = \sum_{i=1}^{N_R} EB_i + \sum_{i=1}^{N_R} ES_i \tag{14}$$

$$EB_i = \kappa_b(\theta_{r,i} - \theta'_{r,i})^2 \tag{15}$$

$$ES_i = \kappa_s \frac{(l_r - l'_r)^2}{(1-c_s)\min(l_r, l'_r) + c_s \max(l_r, l'_r)} \ , \tag{16}$$

where κ_s, κ_b and c_s are user-defined parameters. In this paper, we favor more on finding parts revised by local stretching by adjusting κ_s over κ_b in the implementation. This is because stretching is a more common operation to change the local geometry of engineering models. The overall matching cost is simply the aggregation from each geometry comparison between the corresponding groups.

$$\hat{cost}^* = \sum_{r=1}^{R} ET(g_r, g'_r) \ . \tag{17}$$

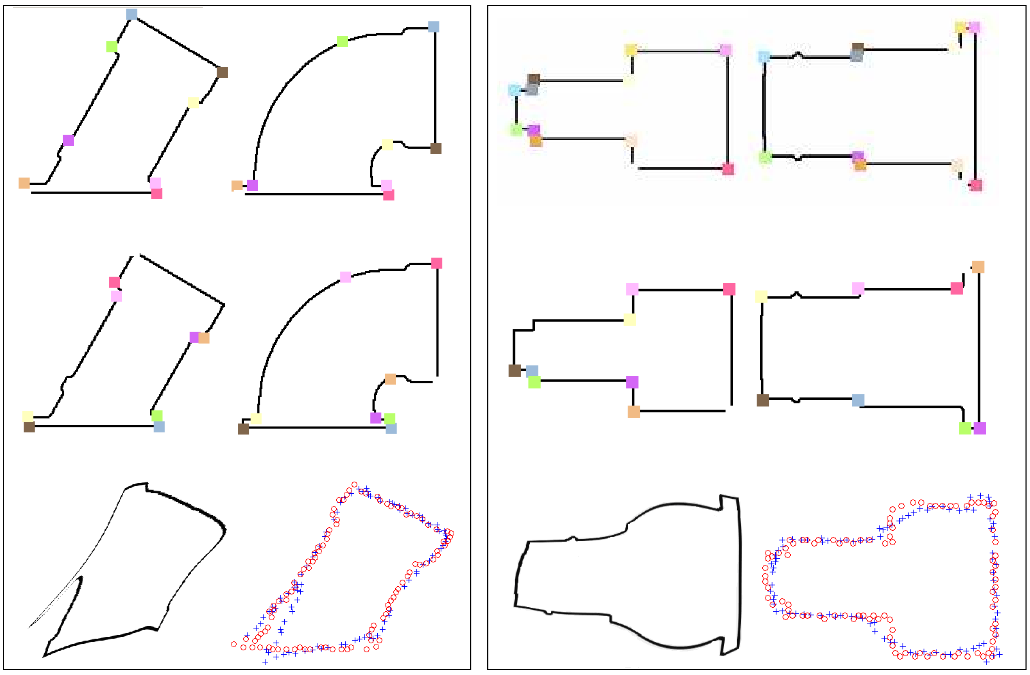
5. Experimental Comparisons and Discussions

Two kinds of experiments were conducted to demonstrate the quality of the structure-oriented shape matching. The first experiment in Section 5.1 demonstrated the capability of establishing the feature point correspondence using the proposed method. The second experiment in Section 5.2 showed the feasibility of the structure-oriented shape matching when used for the shape-based similarity for retrieval. Real data from our Engineering Shape Benchmark database [6] were employed here to examine the quality. At the same time, we compare the results using the proposed method with the ones using some existing methods.

5.1. *Feature point correspondence*

Fig. 7 shows the comparison of feature point correspondence using the proposed method, the method from the perceptually-based shape morphing [23], and the method from shape context-based shape matching [18,36].

The target in [23] is to identify the feature point correspondence for contours of multimedia shapes. In order to have a fair ground for comparison, we used our own method to extract the feature points, while using two different methods for the feature point correspondence. There



(a) Example with local bending

(b) Example with local stretching

Fig. 7. Comparison of the feature point correspondence using the proposed method (1^{st} row), the method in [23] (2^{nd} row), and the shape context-based method in [18] (3^{rd} row). In the first two rows, the left figures in each window are the queries while the right figures in each window are the targets. Color points indicate feature point correspondence. In the 3^{rd} row, the figure on the left is the deformed query after the warping process and the right figure indicates the correspondence between the query and the target with the minimum warping cost. Red $\diamond$ represents the target shape while the blue $+$ represents the query shape.

are two major differences between the proposed method and the method in [23] even though they share similarities in both the process and the algorithm. First, the proposed method is built upon two levels of shape representation and matching, while the method in [23] has only one level of shape matching which performs only on the shape representation of those feature points. Therefore the method in [23] does not consider the geometry information located between any two feature points. Second, we chose shape context as the shape descriptor for the feature points, because shape context represents the content of the feature point with regard to the overall shape. The relative shape representation even follows human perception of the feature point correspondence. This can be explained that when human decides correspondence, he/she not only uses local geometry at each feature point, but also refers to the relative positions of the feature points regarding the overall shape. However, the shape descriptor for the feature point in [23] only considers the geometric information within a limited neighboring area of the feature point, thus ignoring the relation of the feature point to the overall shape.

Shape context-based shape matching in [18] has been used as a benchmark method for non-rigid shape matching and correspondence identification recently. It has shown good performance for non-rigid image registration for multimedia shapes. However, this method is built upon uniform sampling. Therefore it mixes topological information with geometry details, which is not applicable for matching engineering shapes when topological structure matching is desired. Fig. 7 shows two comparisons of the feature point correspondence: an example with large local bending and small protrusions on the left and an example with large local stretching and small protrusions on the right. In both examples, the proposed method determined the feature point correspondence better than the method from [23] even when there were large local deformations and small features. The result shows that the method from [23] falsely matched some feature points since they shared similar local geometry. On the other hand, the result using the method from [18] did not reflect the structural correspondence because the uniform sampling mixes the topology with the geometry. Both examples shows that the deformations did not conserve the original topology of the query shapes during the warping process. The examples in Fig. 7 justifies the notion that the structure-oriented shape matching can achieve results in better conformity with engineering semantics. The proposed method has overall better matching of the feature points than the method in [23]. It is also apparent from the examples that the system can identify the best structure matching by ignoring some unnecessary feature points using the relaxed structure matching.

The proposed matching strategy works well when two shapes share large portions of structure similarity even though local geometry deformation and local structure dissimilarities coexist (see Fig. 8(a) to Fig. 8(e)). However, the performance of correspondence identification is under the

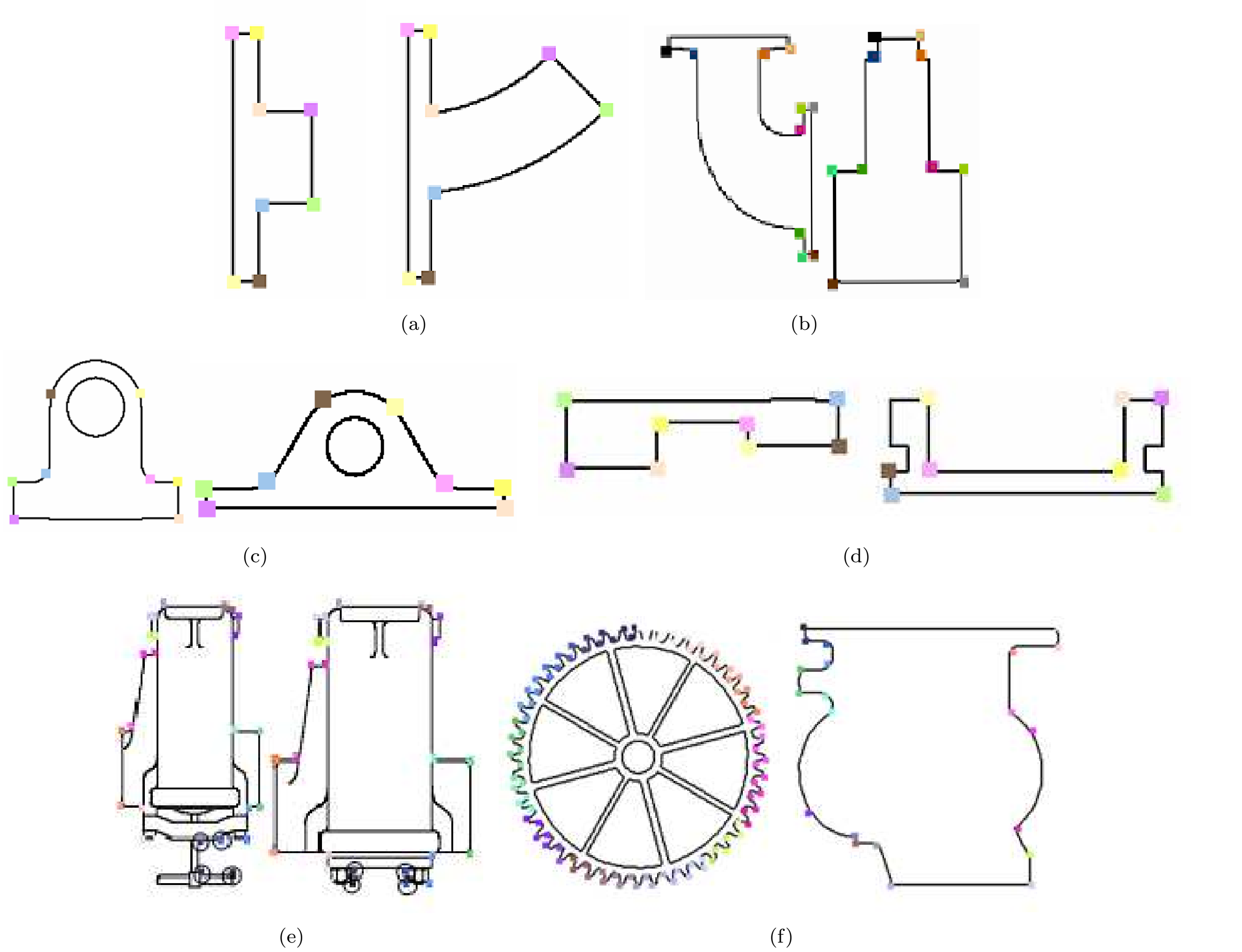


Fig. 8. (a)-(e) Feature point correspondence with different kinds of deformations. (f) Cases when feature point correspondence may fail.

influence from the feature points recognized. The matching will have difficulty to identify exact correspondence when the feature points recognized are not as good as what is defined in Section 4.1.1. First of all, the number of feature points recognized depends on the frequency of local features appearing. When the shape has a large number of feature points, it takes relatively more computational time for matching. In such a case, the structure representation becomes highly coupled with the geometry because high frequency of small features are dominant (see left of Fig. 8(f)). Second, the target of the structures matching almost has no meaning when the shape has no apparent structure, such as the example on the right of Fig. 8(f). Therefore, the real intentions of the proposed strategy are difficult to be reflected in such examples.

5.2. *Shape-based similarity for retrieval*

In this experiment, we demonstrated the performance of the shape-based similarity for retrieval using the proposed shape matching strategy. Fifty-five orthogonal projection images from ESB models were used in the test. We designed our own data set for testing instead of using a public engineering drawing database such as the one from [9]. This was because existing public data seldom reflect the intention of matching global structures with the presence of large local variations. The selected engineering drawings from ESB were different from each other with obvious visual distinctions. We chose those examples in order to demonstrate the robustness of matching higher level structures even when local geometries vary distinctively. These examples were categorized into five categories (see Fig. 9). However, this categorization was not for the purpose of building a ground truth database for retrieval. This is because first, the evaluation of the non-rigid shape similarity has more flexibilities than the evaluation of the rigid shape similarity. The result of categorization is highly dependent on how user evaluates the non-rigid similarity and how deformation evolves. Engineering shapes have so many variations derived from large local deformations. For example, the shape in Fig. 9(f) can be categories into the same group as the rectangle shape (see Fig. 9(c)) because it results from the bending of the rectangle. However it can also be put into Category IV because we can stretch it radially into the shape in Fig.

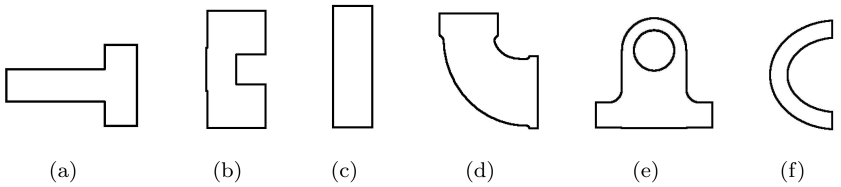


Fig. 9. (a)-(e) Representative shapes from five shape-based categories I to V. (f) An example of an ambiguous shape.

9(d). Second, this categorization was only to help us to understand the applicability and the performance of the proposed method. The categorization did not have a direct impact on evaluation because the evaluation was observation-based rather than quantitative analysis-based. Retrieval of different categories of parts did not mean low quality performance. Instead, it can help us to understand the non-rigid shape similarity better if the retrievals can be created by certain deformations from the query shape. Nevertheless, we present this work to the user not only for the non-rigid shape similarity for retrieval, but also to encourage more efforts to understand how to improve the method to evaluate non-rigid shape similarity for engineering shapes.

Fig. 10 lists the results of the top 6 retrievals from 10 engineering drawings using the proposed structure-oriented contour matching and the shape context-based shape matching. Two examples from each category are involved in the comparison such as the first two are from Category I in Fig. 9(a); the second two are from Category II in Fig 9(b), and so on. The results demonstrated the superiority of the proposed method and even showed some interesting results.

First, the overall performance of the proposed method was better than the method in [18] for engineering drawing retrieval. This was especially obvious for queries with apparent structure such as those in the first six rows of Fig. 10. The proposed method obtained more consistent retrievals than the method in [18]. Shape context-based matching has been tested for freeform multimedia image retrieval such as digit matching [18] where relatively small deformation happens. However, it has not been tested on engineering drawings before. Based on our observations, it did not give good output when local variations involve stretching such as the cases in Fig. 7(b), or when two images look very different such as the cases in Fig. 7(a). The reason for it not to work well for engineering drawings is because engineering models can have large local variations through stretching or bending while sharing the similar structure.

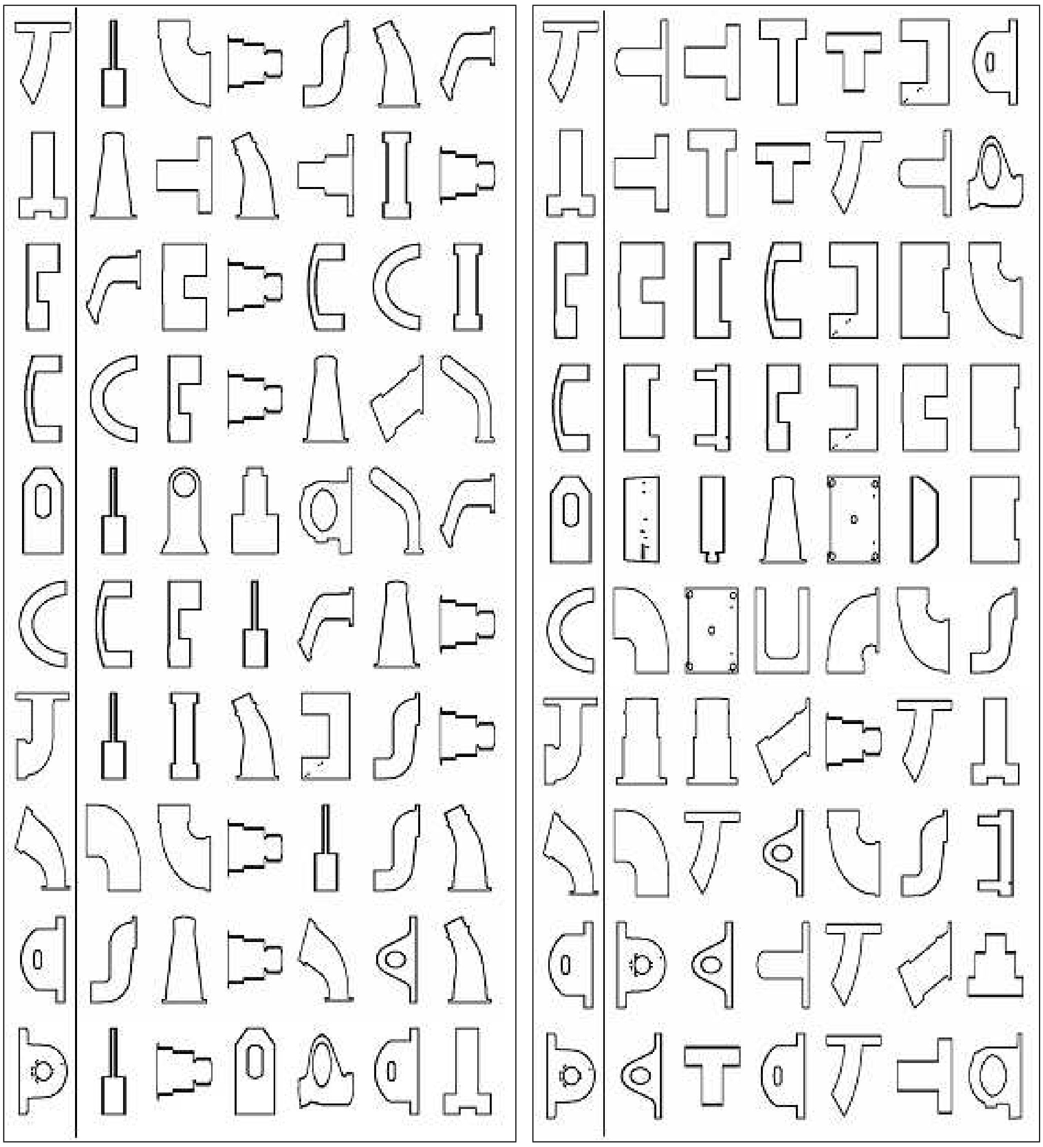
Second, the proposed method worked better when the results of feature point detection have fewer false positives. This was demonstrated by the examples from the first five rows which have more consistent retrievals than the ones from the last five rows (left) of Fig. 10. The examples from the first five rows had less freeform shapes than the ones from the last five rows. Therefore, the examples from the last five rows may have more false positive feature points. Most of the retrievals from the first five rows were from the same categories as the queries, and they shared a similar process of local deformations. The remaining retrievals were from one or two other categories which can be obtained by involving different deformations. However, the retrievals from the last five rows were mixtures of data from several categories. Most of the retrievals coming from the different categories as the query can be deformed from the query either by stretching or bending or even both. However, some of them can be hardly deformed from the query shape

One interesting finding was that queries from the Category V retrieved data from Category I commonly. The opposite also happened to queries from the Category I. Based on the notion of the structure-based shape similarity, it can be explained that shapes from both categories shared large structural similarity defined by the feature points. Only one geometric entity changed from line segment in Category I to an arc in the Category V, which can be explained as the result of local bending. These two kinds of shapes can coexist in the retrievals because their similarity is allowed in the proposed shape matching. Another interesting observation from the experiments is about the ambiguous query in Fig. 9(f). This example was originally put in Category III. However, it obtained more data from Category IV than from its preliminary category. This fact indicates that stretching is more favored during the computation of the matching cost in the current implementation. In the future, user will be enabled to leverage the cost from the stretching and the cost from the bending.

Overall, the success of the proposed matching strategy gives us a new perspective on how to evaluate shape similarity for engineering shapes. It lays a good foundation to develop new classes of shapes that allows certain specific deformations by stretching and bending.

6. Conclusions

A perspective to evaluate shape similarity for engineering models is developed using a shape matching strategy driven by the structure from the contours of engineering drawings. The contribution of this paper is threefold. First, two levels of shape representation and matching are developed with the structure and the geometry decoupled, associated, and optimally integrated throughout the process. Second, a new Relevant Relevance Measure is defined for the feature point extraction using the discrete contour evolution. Third, the Dynamic Time Warping (DTW) algorithm is adapted to perform the relaxed feature point correspondence. Two levels of shape matching which are conducted



(a) Results using [18]

(b) Results using the proposed method

Fig. 10. Shape-based similarity retrieval comparison. Pictures in the first column of each figure are the queries.

using different and appropriate approaches synergies with each other to achieve the optimal matching. Experiments show better results against the existing methods for real engineering shapes with large local variations, at both feature point correspondence and shape-based similarity for retrieval. The goodness of structure-oriented shape matching is reflected from the results that the retrievals are more meaningful and consistent from the engineering standpoint. In addition, the results show that shape matching is less sensitive to large local deformations when global structure matches. The proposed strategy works better when structure can be decoupled from local geometry. Therefore, it provides a strong incentive to further improve the overall shape-based similarity retrieval by complementing the structure-based shape matching with the statistics-based shape matching.

Acknowledgements

This material is partly based upon work supported by the National Science Foundation under Grant IIS No. 0535156. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. The authors thank the anonymous reviewers for their comments.

References

[1] J. Pu, K. Ramani, On visual similarity based 2D drawing retrieval, Computer Aided Design 38 (3) (2006) 249–259.

[2] M. J. Fonseca, A. Ferreira, J. A. Jorge, Content-based retrieval of technical drawings, International Journal of Computer Applications in Technology 23 (2) (2005) 86–100.

[3] S. Berchtold, D. A. Keim, H.-P. Kriegel, Using extended feature objects for partial similarity retrieval, The International Journal on Very Large Data Bases 6 (4) (1997) 333–348.

[4] J. H. Park, B. S. Um, A new approach to similarity retrieval of 2D graphic objects based on dominant shapes, Pattern Recognition Letters 20 (6) (1999) 591–616.

[5] S. Müller, G. Rigoll, Searching an engineering drawing database for user-specified shapes, in: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR'99), 1999, pp. 697–700.

[6] S. Jayanti, N. Iyer, Y. Kalyanaraman, K. Ramani, Developing an engineering shape benchmark for CAD models, Special Issue on Shape Similarity Detection and Search for CAD/CAE Applications, Computer Aided Design 38 (9) (2006) 939–953.

[7] J. S. Gero, V. Kazakov, Entropic similarity and complexity measures for architectural drawings, in: B. T. J.S. Gero, T. Purcell (Eds.), Visual and Spatial Reasoning in Design II, University of Sydney, Sydney, 2001, pp. 147–161.

[8] Y. Bai, X. Xu, Object boundary encoding - a new vectorisation algorithm for engineering drawings, Computers in Industry 46 (1) (2001) 67–54.

[9] S. Berchtold, H.-P. Kriegel, S3: similarity search in CAD database systems, in: SIGMOD '97: Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, ACM Press, New York, NY, USA, 1997, pp. 564–567.

[10] T. B. Sebastian, B. B. Kimia, Curves vs skeletons in object recognition, in: Proceedings of International Conference on Image Processing, Vol. 3, 2001, pp. 22–25.

[11] T. B. Sebastian, P. N. Klein, B. B. Kimia, Recognition of shapes by editing their shock graphs, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (5) (2004) 550–571.

[12] M. Safar, C. Shahabi, C. H. Tan, Resiliency and robustness of alternative shape-based image retrieval techniques, in: Proceedings of IEEE International Database Engineering and Applications Symposium, 2000, pp. 337–348.

[13] F. Mokhtarian, S. Abbasi, J. Kittler, Robust and efficient shape indexing through curvature scale space, in: Proceedings of British Machine Vision Conference, 1996.

[14] T. Adamek, N. E. OConnor, A multiscale representation method for nonrigid shapes with a single closed contour, IEEE Transactions on Circuits and Systems for Video Technology 14 (5) (2004) 742–753.

[15] H. Freeman, Computer processing of line-drawing images, ACM Computing Surveys 6 (1) (1974) 57–97.

[16] E. Arkin, L. Chew, D. Huttenlocher, K. Kedem, J. Mitchell, An efficiently computable metric for comparing polygonal shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (3) (1991) 209–216.

[17] H. Kauppinen, T. Seppanen, M. Pietikainen, An experimental comparison of autoregressive and Fourier-based descriptors in 2D shape classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (2) (1995) 201–207.

[18] S. J. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (4) (2002) 509–522.

[19] V. Jain, R. H. Zhang, Robust 2D shape correspondence using geodesic shape context, in: Proceedings of 13th Pacific Conference on Computer Graphics and Applications, no. 121-124, 2005.

[20] Y. Zheng, D. Doermann, Robust point matching for nonrigid shapes by preserving local neighborhood structures, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (4) (2006) 643–649.

[21] K. Grauman, T. Darrell, Fast contour matching using approximate earth mover's distance, in: Proceedings of Computer Vision and Pattern Recognition, 2004, pp. 220–227.

[22] L. J. Latecki, R. Lakämper, Shape similarity measure based on correspondence of visual parts, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (10) (2000) 1185–1190.

[23] L. Liu, G. Wang, B. Zhang, B. Guo, H.-Y. Shum, Perceptually based approach for planar shape morphing, in: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference on, 2004, pp. 111–120.

[24] Y. Gdalyahu, D. Weinshall, Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (12) (1999) 1312–1328.

[25] C. Grigorescu, N. Petkov, Distance sets for shape filters and shape recognition, IEEE Transactions on Image Processing 12 (10) (2003) 1274–1286.

[26] E. G. M. Petrakis, A. Diplaros, E. Milios, Matching and retrieval of distorted and occluded shapes using dynamic programming, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (11) (2002) 1501–1516.

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

[28] T. B. Sebastian, P. N. Klein, B. B. Kimia, Alignment-based recognition of shape outlines, in: IWVF-4: Proceedings of the 4th International Workshop on Visual Form, 2001, pp. 606–618.

[29] T. W. Sederberg, E. Greenwood, A physically based approach to 2D shape blending, Computer Graphics 33 (10) (1992) 1683–1699.

[30] R. Singh, N. P. Papanikolopoulos, Planar shape recognition by shape morphing, Pattern Recognition 33 (10) (2000) 1683–1699.

[31] H. Chui, A. Rangarajan, A new point matching algorithm for non-rigid registration, Computer Vision and Image Understanding 89 (2-3) (2003) 114–141.

[32] S. Wang, T. Kubota, T. Richardson, Shape correspondence through landmark sliding, Computer Vision and Pattern Recognition 1 (2004) 143–150.

[33] L. J. Latecki, R. Lakämper, Convexity rule for shape decomposition based on discrete contour evolution, Computer Vision and Image Understanding 73 (3) (1999) 441–454.

[34] M. Marji, P. Siy, A new algorithm for dominant points detection and polygonization of digital curves, Pattern Recognition 36 (10) (2003) 2239–2251.

[35] Y. Zhuang, K. Goldbergt, M. Picket, Simplifying complex CAD geometry with conservative bounding contours, in: Proceedings of IEEE International Conference on Robotics and Automation, Vol. 3, 1997, pp. 2503–2508.

[36] S. J. Belongie, J. Malik, J. Puzicha, Matching with shape contexts, Retrieved online from *http : //www.eecs.berkeley.edu/Research/Projects/CS/vision /shape/sc_digits.html* (March 2007).