Evaluating Contour Segment Descriptors

Cong Yang[1], Oliver Tiebe[1], Kimiaki Shirahama[1], Ewa Łukasik[2]
and Marcin Grzegorzek[1]

Abstract

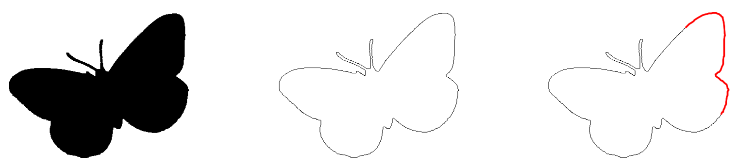
Contour Segment (CS) is the fundamental element of partial boundaries or edges in shapes and images. So far, CS has been widely used in many applications, including object detection/matching and open curve matching, etc. To increase the matching accuracy and efficiency, a variety of CS descriptors have been proposed. A CS descriptor is formed by a chain of boundary or edge points and is able to encode the geometric configuration of a CS. Because many different CS descriptors exist, a structured overview and quantitative evaluation are required in the context of CS matching. This paper assesses 27 CS descriptors in a structured way. Firstly, the analytical invariance properties of CS descriptors are explored with respect to scaling, rotation and transformation. Secondly, their distinctiveness is evaluated experimentally on three datasets. Lastly, their computation complexity is studied. Based on results, we find that CS length and matching algorithm affect the performance of CS matching while matching algorithms have more affection. The results also reveal that, with different combinations of CS descriptors and matching algorithms, several requirements in terms of matching speed and accuracy can be fulfilled. Furthermore, a proper combination of CS descriptors can improve the matching accuracy over the individual descriptors.

Keywords: Contour Segment, Contour Segment Descriptor, Open Curve Matching

[1][1]Cong Yang, Oliver Tiebe, Kimiaki Shirahama and Marcin Grzegorzek are with the Research Group for Pattern Recognition, Institute for Vision and Graphics, University of Siegen, Siegen, D-57076, Germany. E-mail: cong.yang@uni-siegen.de.
[2]Ewa Łukasik is with the Laboratory of Operational Research and Artificial Intelligence, Institute of Computing Science, Poznan University of Technology, Pl. Marii Skłodowskiej-Curie 60-965 Poznan, Poland.

1. Introduction

A Contour Segment (CS) is a fragment of shape boundary which is constructed by a chain of connected boundary points. As shown in Figure 1, compared to the shape boundary which is defined as a circular sequence of boundary points, a CS only describes the partial information of a shape boundary. Each boundary point in the CS is called a CS point. The main motivation for CS is that the connectedness of shape boundary points is not ensured in practice due to the noise affections [1] and manual operations [2]. Thus, viewing CSs as local patches with any length provides more flexibility against boundary instabilities. Moreover, psychophysical studies [3] show



(a) Shape (b) Shape Boundary (c) Contour Segment

Figure 1: Shape boundary and a contour segment (marked with the red line).

that we can recognise objects using CSs alone. Therefore, CS is an important element for computer vision tasks [4, 3].

Benefit from these properties, CS plays a key role in many different applications including object detection [5] and matching [6], etc. This is because an object shape cannot be ideally segmented from an image due to the background clutter [1] and object overlapping (occlusion) [7]. To overcome these, one typical method is to represent the object boundaries and background using CSs [8]. Moreover, CS is also commonly used for the application of sketch-based object retrieval [2] since partial matching is only expected between a sketched curve and an actual object boundary. Not only in the specific applications, CS is also regarded as a fundamental element of general applications like open curve matching [9, 6]. Similar to CS, open curve is a fragment of a shape boundary but with larger deformation and length than CSs. Open curve matching aims to find similar parts between two open curves and then calculate their similarity. Essentially, open curve matching is employed by both object detection and recognition applications since the lengths and deformations of generated edges in an image are not uniformed. Thus, we use CS to represent the partial features of an open curve, then search the similar parts by CS matching. This strategy is employed for shape matching by searching the similar parts among shape boundaries [10]. With these motivations, in this paper, one of our experiments is applied by the application of open curve matching.

In order to efficiently match CSs, proper CS descriptors and matching algorithms are required. Such a descriptor is required to encode the geometric configuration of those CS points. More specifically, among all CS points, we normally select some sample points and generate CS descriptors by considering the geometrical relationship between those points. If sample points are selected roughly, CS descriptors represent the coarse-grained CS features. If sample points are selected densely, CS descriptors describe the fine-grained CS features. For CS matching, traditional matching algorithms like Hungarian [11], Dynamic Programming (DP) [12] or Dynamic Time Warping (DTW) [13] are normally employed for searching the correspondences between sample points. For some CS descriptors [6, 14], due to the structure of their feature vectors, distance functions like correlation [15], histogram intersection (HI) [16], χ^2-statistics [17] and Hellinger [18] are used for calculating their distances.

Despite the extensive usage of CS [5, 6, 2, 9] and its extensions [2, 19, 20, 14], most of existing works only introduce CS descriptors without any comparison. In addition, some survey papers focus on shape representation techniques [21, 22], or only compare a handful of CS descriptors in specific applications [23]. There is no paper that integrally surveys and evaluates CS descriptors. In other words, it is unclear how the discrimination power of each descriptor is, and how similar or different it is to the other descriptors. Furthermore, deep insights are needed regarding the suitable combination between CS descriptors and matching algorithms, and the computational efficiency of each descriptor. Therefore, this paper studies the invariance properties, matching performance and computation complexity of 27 CS descriptors and their matching algorithms in a structured way.

Figure 2 illustrates the pipeline of the evaluating steps and their correlated sections (red colour). Specifically, our evaluation is applied by taking four properties into account. Firstly, we analyse and compare a taxonomy of invariant properties. Secondly, we theoretically analyse the computational complexity of both feature generation and CS matching on 27 CS descriptors. Thirdly, the matching performance of CS descriptors is analysed experimentally using different combinations of CS descriptors and matching algorithms via one general and two application-oriented scenarios. Lastly, the runtime in the CS matching experiment is evaluated and compared. With discussions and observations on the four properties above, we draw the recommendation for different application scenarios by balancing the matching accuracy and speed.

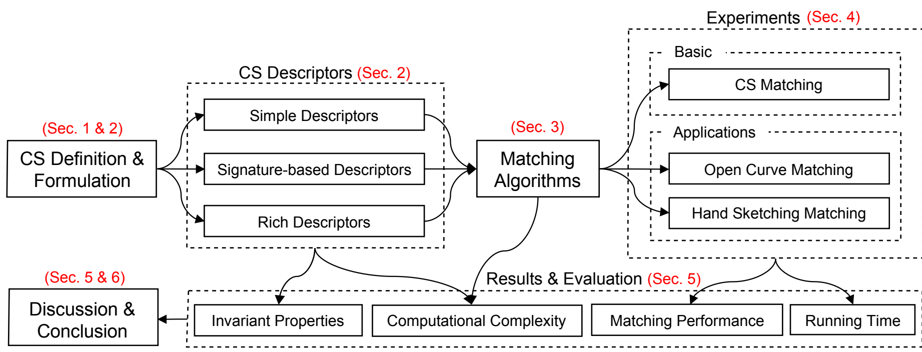


Figure 2: Pipeline for evaluating CS descriptors. Correlated sections are marked with red.

The most significant scientific contributions of this paper include: (1) We survey and evaluate the existing CS descriptors in a structured way. (2) In order to evaluate CS descriptors in different scenarios, we design and introduce 3 datasets for CS matching, open curve matching and hand sketching matching. (3) Formed on evaluations, we recommend the combinations of CS descriptors and matching algorithms for meeting different requirements in terms of accuracy and speed.

2. Contour Segment Descriptors

In this section, we first formulate the definition of CS. After that, CS descriptors are put into three groups, "simple", "signature-based" and "rich". Here, CS descriptors are classified and grouped based on the structure of their feature vectors. In order to regulate and standardise our next descriptions, for a CS, we assume the following restrictions: (1) With one pixel width. (2) With two endpoints, which only have one neighbour point. (3) No intersection point: except two endpoints, the rest points only have two neighbour points. Formed on these restrictions, a CS C is represented as a sequence of CS points $\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_N$ along the boundary path. Here, a CS point $\boldsymbol{p}_i$ $(1 \leq i \leq N)$ is expressed as a point in the Cartesian coordinate system, that is, $[x_i, y_i]$. We set CSs to have the same number of points for two reasons: First, it is easier for us to fairly evaluate the performance of CS descriptors using the same matching methods. Second, it also makes open curve matching easier since open curves can be decomposed into multiple CSs with the same number of points. Then, an open curve matching task is accomplished by multiple CS matching tasks.

2.1. Simple CS Descriptors

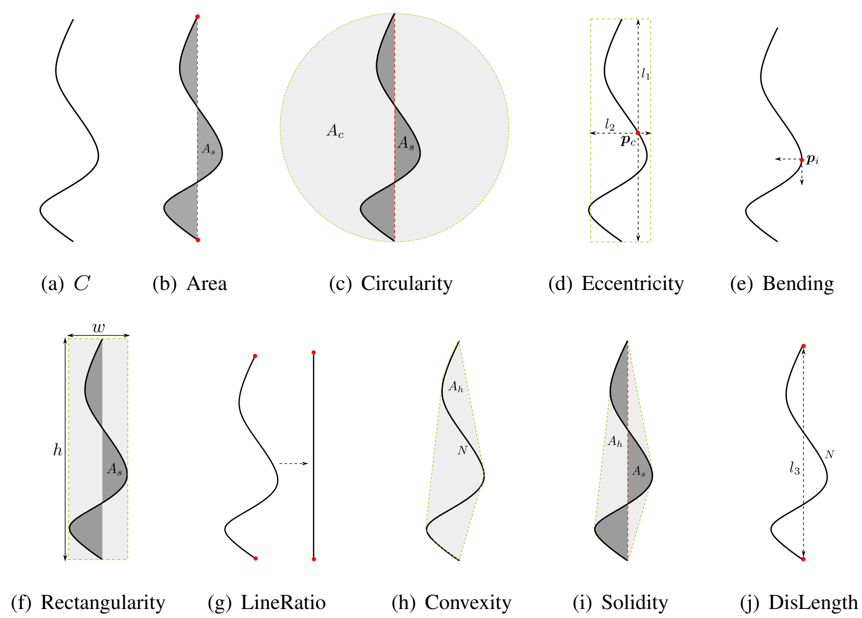


Figure 3: Simple descriptors for a CS C.

A simple CS descriptor is a scalar that represents a global feature of a CS. CS descriptors in this group are normally generated by considering the global CS geometry. The motivation of simple CS descriptors is that some practical problems [24] only need simple and coarse-grained features for fast calculations. Thus, it is desired to find descriptors that are both simple and generally applicable. Moreover, combing descriptors should introduce a new perspective. Thus, we survey and revise nine simple CS descriptors from shape survey papers [22, 21] and applications [24, 25]. In general, these descriptors usually can only discriminate CS with large differences. Therefore, they are not used as standalone descriptors but usually used as filters to eliminate false hits or combined with other rich descriptors.

Area: As shown in Figure 3(b), the area descriptor [22, 24] $\boldsymbol{f}_1$ is calculated as the area A_s (dark grey area) between the straight line (red dotted line) connecting the CS endpoints (red points) and the CS itself (Figure 3(a)). In order to ensure the scale invariance, $\boldsymbol{f}_1$ is normalised by the length of CS N, that is $\boldsymbol{f}_1 = A_s/N$.

Circularity: Circularity [22, 24] $\boldsymbol{f}_2$ illustrates how similar the CS C is to a circle. As shown in Figure 3(c), a circularity $\boldsymbol{f}_2$ is calculated as A_s/A_c where A_c denotes the area of the minimum CS surrounding circle (light grey area).

Eccentricity: Eccentricity [25, 24] $\boldsymbol{f}_3$ can be uniquely defined as the ratio of length of major axis to minor axis that cross each other orthogonally in the middle of the CS. We first find the middle point $\boldsymbol{p}_c$ of CS C (the red point in Figure 3(d)), then eccentricity is calculated as $\boldsymbol{f}_3 = l_1/l_2$ where l_1 and l_2 are the lengths of major axis and minor axis to the CS minimum bounding rectangle on $\boldsymbol{p}_c$, respectively.

Bending: Bending [21, 25] $\boldsymbol{f}_4$ is defined by the average bending energy. It captures the degree of a CS bending energy. For instance, the circle is the shape with the minimum bending energy. Bending is calculated as $\boldsymbol{f}_4 = \dfrac{1}{N}\sum\limits_{i=1}^{N} K(i)^2$ where $K(i)$ denotes the curvature of point $\boldsymbol{p}_i$ (Figure 3(e)) which is approximated by:

$$K(i) = \frac{2|(x_i - x_{i-1})(y_{i+1} - y_{i-1}) - (x_{i+1} - x_{i-1})(y_i - y_{i-1})|}{\sqrt{((x_i - x_{i-1})^2 + (y_i - y_{i-1})^2)((x_{i+1} - x_{i-1})^2 + (y_{i+1} - y_{i-1})^2)((x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2)}} \quad (1)$$

where $\boldsymbol{p}_{i-1}$, $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$ are three successive points.

Rectangularity: Rectangularity [24] $\boldsymbol{f}_5$ presents how rectangular a CS is , i.e. how much the CS fills its minimum bounding rectangle (Figure 3(f)). Rectangularity is calculated as $\boldsymbol{f}_5 = A_s/(w \cdot h)$ where w and h are the width and height of the CS minimum bounding rectangle.

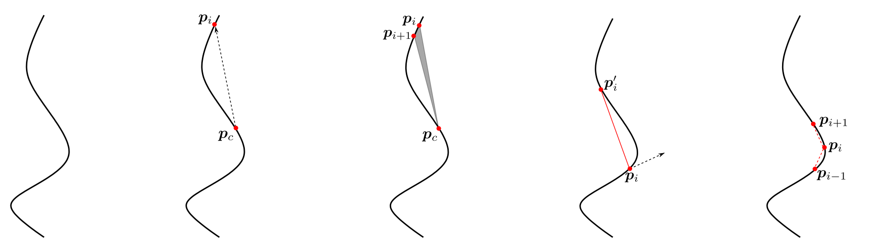
LineRatio: LineRatio [25, 24] $\boldsymbol{f}_6$ uses a straight line as a template, and illustrates how similar a CS is to a straight line (Figure 3(g)). LineRatio is calculated as $\boldsymbol{f}_6 = h/N$ where h is the height of the CS minimum bounding rectangle.

Convexity: Convexity [24] $\boldsymbol{f}_7$ is defined as the ratio of the convex hull [26] over that of the CS length. Convexity captures the minimal convex covering of a CS. A straightforward measure for Convexity can be calculated as $\boldsymbol{f}_7 = A_h/N$ where A_h denotes the CS convex hull area (Figure 3(h)).

Solidity: As shown in Figure 3(i), solidity [24] $\boldsymbol{f}_8$ describes the extent to which the CS is convex or concave and is defined as A_s/A_h. Solidity is an indicator that captures the concave-convex condition of a CS.

Dislength: Dislength [6] $\boldsymbol{f}_9$ illustrates the skewness power of CS (Figure 3(j)). It is defined by the ratio between distance of endpoints l_3 and the CS length.

2.2. Signature-based CS Descriptors



(a) C (b) Cendistance (c) AreaFunction (d) ChordLength (e) Turning Angle

Figure 4: Some signature-based descriptors for a CS C.

CS signature is the modified version of Shape Signature that represents a shape by a vector representing various spatial relations among shape boundary points [21]. Signature-based descriptors can capture the perceptual features of CSs and are often combined with some other feature extraction algorithms like Fourier descriptors [27, 28, 29, 30, 31] and Wavelet Descriptors [32, 33, 34]. The motivation of signature-based CS descriptors is to represent a CS with a midterm order of feature vector than the simple and rich CS descriptors. Thus, signature-based CS descriptors could offer a proper way for balancing the matching accuracy and speed.

Comcoor: Comcoor (Complex Coordinates) [22] descriptor $\boldsymbol{f}_{10}$ is mainly designed for transforming a CS represented by a two-dimensional sequence (sequence consisting of two-dimensional points) into one-dimensional sequence. Let $\boldsymbol{p}_c$ be the middle point for a CS $\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_N$. Complex Coordinates function is:

$$f_{10}^i = [x_i - x_c] + [y_i - y_c] \quad . \tag{2}$$

where $[x_c, y_c]$ is the coordinate of $\boldsymbol{p}_c$ and $[x_i, y_i]$ is the coordinate of any point in C, $i = 1, 2, \cdots, N$. Finally, $\boldsymbol{f}_{10} = [f_{10}^1, f_{10}^2, \cdots, f_{10}^N]$.

Cendistance: Cendistance (Centroid Distance Function) [21] descriptor $\boldsymbol{f}_{11}$ is the representation of centroid-based time series. The main characteristics are simplicity and flexibility since the accuracy can be controlled by the density of sample points. Similar to Comcoor, as shown in Figure 4(b), $\boldsymbol{f}_{11}$ is defined as $[f_{11}^1, \cdots, f_{11}^i, f_{11}^N]$ where f_{11}^i is the angle between $\boldsymbol{p}_c$ and the CS point $\boldsymbol{p}_i$.

Tangent: Tangent (Tangent Angle Function) [21] descriptor $\boldsymbol{f}_{12}$ is defined as follows:

$$f_{12}^i = \tan^{-1}\left(\frac{y_i - y_{i-w}}{x_i - x_{i-w}}\right) \quad , \tag{3}$$

where $\boldsymbol{p}_i = [x_i, y_i]$ is the ith CS point, and w is a window for controlling the accuracy. In our experiment, we set $w = 5$ and $(i - w) = 1$ if $(i - w) <= 0$. Finally, $\boldsymbol{f}_{12} = [f_{12}^1, f_{12}^2, \cdots, f_{12}^N]$. Since Tangent CS descriptor requires a window for feature generation, it is sensitive to noise. However, this is more flexible than most signature-based CS descriptors since we can control its accuracy by changing the size of the window.

Curvature: Curvature [21] descriptor $\boldsymbol{f}_{13}$ is very important for capturing salient perceptual characteristics. Superficially, Curvature descriptor is calculated as $\boldsymbol{f}_{13} = [K'(1), K'(2), \cdots, K'(N)]$ where $K(i)$ be the curvature of a CS point $\boldsymbol{p}_i$ similar to Bending ($\boldsymbol{f}_4$). In order to ensure the scale invariance, $K(i)$ is normalised by the mean absolute curvature.

Area Function: Area Function [21] descriptor $\boldsymbol{f}_{14}$ is calculated by the triangle area between the middle point $\boldsymbol{p}_c$ and two consecutive CS points $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$. (Figure 4(c)). In other words, $\boldsymbol{f}_{14} = [f_{14}^1, \cdots, f_{14}^N]$ where f_{14}^i is the area of the triangle consisting of $\boldsymbol{p}_c$, $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$. Area Function is simple and can be used to collect fine-grained features since the distance between $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$ is small and can capture the small deformations of a CS. Moreover, the coarse-grained features are also preserved as we collect the whole areas from $(\boldsymbol{p}_i, \boldsymbol{p}_{i+1})$ to a fix point $\boldsymbol{p}_c$.

Triangle Area: Different from Area Function, Triangle Area [35] descriptor $\boldsymbol{f}_{15}$ is computed directly from area of triangles formed by $\boldsymbol{p}_{i-t_s}$, $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+t_s}$ where $i \in$

$[1, N]$ and $t_s \in [1, \frac{N}{2} - 1]$. For each CS point $\boldsymbol{p}_i$, the triangle area is formed by:

$$f_{15}^i = \frac{1}{2} \det \begin{pmatrix} x_{i-t_s} & y_{i-t_s} & 1 \\ x_i & y_i & 1 \\ x_{i+t_s} & y_{i+t_s} & 1 \end{pmatrix} \quad . \tag{4}$$

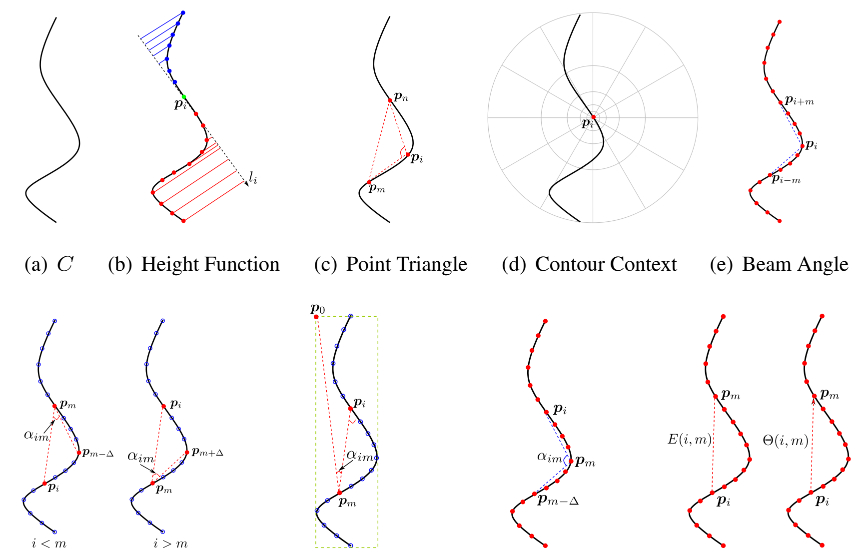
With this equation, when the CS path is traversed in the clock-wise direction, positive, negative and zero values of triangle area mean convex, concave and straight-line points, respectively. Triangle area provides useful information like the convexity/concavity at each CS point. Therefore, this descriptor provides high discrimination capability. Comparing to Area Function, Triangle Area is more flexible since the gap between three points can be modified by varying their positions.

Chord Length: Chord Length [36] descriptor $\boldsymbol{f}_{16}$ is derived by the distance between a CS point and its reference point. As shown in Figure 4(d), the chord length f_{16}^i of $\boldsymbol{p}_i$ is its shortest distance to the CS point $\boldsymbol{p}_i'$ so that $\boldsymbol{p}_i\boldsymbol{p}_i'$ is perpendicular to the tangent vector at $\boldsymbol{p}_i$. Finally, $\boldsymbol{f}_{16} = [f_{16}^1, \cdots, f_{16}^N]$ where f_{16}^i is normalised by the CS length N for scale invariance. Chord Length descriptor is robust to fine-grained deformations since chord lengths are calculated using different reference points rather than only one middle point like Area Function, etc. For example, if the middle point of a CS is changed because of deformation or noise, most chord lengths remain the same since each CS point may have a different reference point for calculating its chord length.

Turning Angle: As shown in Figure 4(e), a turning angle f_{17}^i represents the angle of $\angle \boldsymbol{p}_{i-1}\boldsymbol{p}_i\boldsymbol{p}_{i+1}$ defined by $\boldsymbol{p}_i$ and its neighbouring points $\boldsymbol{p}_{i-1}$ and $\boldsymbol{p}_{i+1}$. The turning angle descriptor $\boldsymbol{f}_{17}$ can be represented by the whole CS points as a set of feature vectors: $\boldsymbol{f}_{17} = [f_{17}^1, \cdots, f_{17}^N]$. Similar to Area Function, Turning angle captures the fine-grained features of a CS by using adjacent points. However, it has less ability for preserving coarse-grained deformation since the generated turning angles are globally isolated. On the contrary, area functions are not isolated since they are all connected by the middle point of a CS.

2.3. Rich CS Descriptors

Rich CS descriptors capture the CS geometrical features in both fine- and coarse-grained levels. Compared to simple and signature-based CS descriptors, the feature



(a) C (b) Height Function (c) Point Triangle (d) Contour Context (e) Beam Angle

(f) Partial Contour (g) Opt Partial Contour (h) Chord Distribution (i) Length Direction

Figure 5: Some rich descriptors for a CS C.

vector of rich descriptors has more dimensions and varieties. Therefore, rich CS descriptors carry more information of the original CS. Specifically, rich descriptors $\boldsymbol{f}_{18} - \boldsymbol{f}_{25}$ are usually generated based on feature vector F_i on each sample point $\boldsymbol{p}_i$. By collecting F_i on every sample point $(F_1, F_2, \cdots, F_N)$, we can get the rich descriptor for representing a CS C. In order to simplify the description, we only introduce the way of F_i generation on rich descriptors $\boldsymbol{f}_{18} - \boldsymbol{f}_{25}$. In contrast, rich descriptors $\boldsymbol{f}_{26}$ and $\boldsymbol{f}_{27}$ have their own feature structures, they are introduced independently.

Height Function: As shown in Figure 5(b), a height function [37] F_i is defined based on the distances of the other sample points to its tangent line, $i = 1, \cdots, N$. The motivation of this descriptor ($\boldsymbol{f}_{18}$) is to represent CS points by considering their relations to all other sample points in the same direction.

Point Triangle: Point Triangle [38] descriptor $\boldsymbol{f}_{19}$ is inspired by Carlsson [39] in which they only consider qualitative orientations of each triangle: oriented clock or counterclockwise. Point triangle descriptor provides a full quantitative description of each triangle. As shown in Figure 5(c), a point triangle descriptor F_i is a histogram of all triangles spanned by $\boldsymbol{p}_i$ and all pairs of points $\boldsymbol{p}_m, \boldsymbol{p}_n$. Especially, for each triangle, F_i contains three values: angle $\angle \boldsymbol{p}_m \boldsymbol{p}_i \boldsymbol{p}_n$, distance $\boldsymbol{p}_i \boldsymbol{p}_m$ and distance $\boldsymbol{p}_i \boldsymbol{p}_n$. This leads to a significant increase in descriptive power since both orientation and distance features are captured in the point triangle.

Contour Context: For each point $\boldsymbol{p}_i$, Contour Context descriptor $\boldsymbol{f}_{20}$ [40] considers the $N-1$ vectors obtained by connecting $\boldsymbol{p}_i$ to all other points. The key motivation is that the distribution over relative positions of each CS point is a robust, compact, and highly discriminative descriptor. Similar to shape context [41], in order to capture the geometrical information of $\boldsymbol{p}_i$, a log-polar histogram is defined by five sections on the radius and 12 sections on the angle. Thereafter, a contour context F_i for $\boldsymbol{p}_i$ can be represented as a 60-dimensional feature vector.

Beam Angle: The basic idea of Beam Angle [4] descriptor $\boldsymbol{f}_{21}$ is to represent each CS point $\boldsymbol{p}_i$ by a weighted Beam Angle Histogram (BAH). This descriptor represents a CS point using multiple angles with different weights. With this, beam angle descriptor can mitigate the uncertainty in CS representation since it down-weights the interaction of distant CS points. Specifically, at CS point $\boldsymbol{p}_i$, the beam angle F_i is subtended by lines $(\boldsymbol{p}_{i+m}, \boldsymbol{p}_i)$ and $(\boldsymbol{p}_i, \boldsymbol{p}_{i-m})$ (Figure 5(e)), where $m = 1, \cdots, N'$ and N' is determined experimentally. Therefore, $\boldsymbol{p}_i$ is represented by N' weighted angles.

Partial Contour: Partial Contour [42] descriptor $\boldsymbol{f}_{22}$ is calculated by the relative orientations between lines that connect the CS sampled points. As shown in Figure 5(f), for a CS point p_i, F_i is the angle α_{im} which is formed by a line connecting p_i and p_m, and a line to a third point relative to the position of the previous two points. This third point is chosen depending on the position of the other two points to ensure that the selected point is always inside the CS. This allows them to formulate the descriptor as a self-containing descriptor of any of its parts. The Partial Contour descriptor has several important properties, such as rotation and translation invariance, covering both local and global characteristics, and partial matching based on self-containing.

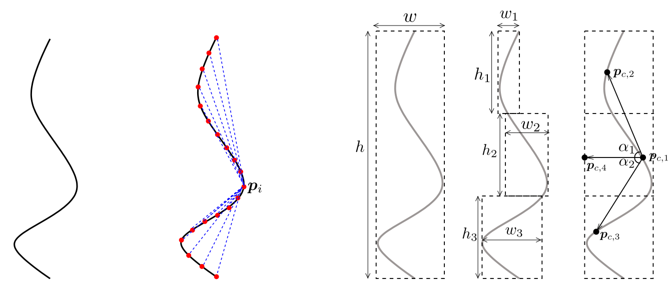
Opt Partial Contour: Opt Partial Contour [23] descriptor $\boldsymbol{f}_{23}$ analyses angles defined by lines connecting a CS-dependent reference point and the CS points. Based on this, both fine- and coarse-grained features can be captured since $\boldsymbol{f}_{23}$ considers relative angles between CS points and the reference point which is defined by the upper left corner of the CS surrounding box (See Figure 5(g)). For a CS point $\boldsymbol{p}_i$, F_i is formed by the angles $\angle \boldsymbol{p}_i \boldsymbol{p}_m \boldsymbol{p}_0$, where $\boldsymbol{p}_m$ is also a CS point.

Chord Distribution: A chord is a line joining two points of a region boundary, and the distribution of chords' lengths and angles is often used as shape descriptor [43]. Chord Distribution descriptor $\boldsymbol{f}_{24}$ [20] uses chords to exploit the available point ordering information for the subsequent order-preserving assignment matching. In comparison, the contour context descriptor $\boldsymbol{f}_{20}$ loses all the ordering information due to the histogram binning and does not consider the influence of the local neighbourhood on single point matches. As shown in Figure 5(h), for a CS point $\boldsymbol{p}_i$, a chord distribution F_i is computed based on angles α_{im} which describe the relative spatial arrangement of the sampled points. For a single CS, the angles are calculated over all possible point combinations, yielding the descriptor matrix $\boldsymbol{f}_{24}$.

Length Direction: As shown in Figure 5(i), for a CS point $\boldsymbol{p}_i$, Length Direction descriptor $\boldsymbol{f}_{25}$ [44] consists of F_i representing both the length (Euclidean distance in the log space) and direction (four quadrant inverse tangent) of the vector from $\boldsymbol{p}_i$ to other points in C. The length and direction features are independently saved.

Line Segment: Line Segment [14] descriptor $\boldsymbol{f}_{26}$ is generated based on straight-line segment statistics. As shown in Figure 6(b), for each CS point, the descriptor considers a continuous portion of the CS with length equal to a pre-defined percentage of the CS size. Then, the length of the straight-line segment between the reference point and other CS points is computed using the Euclidean distance. For the set of straight-line segments, statistical movements (average and standard deviation) are calculated. By performing this for different lengths of contour portions, a CS C is represented by a feature vector which describes the CS at different sizes. There are several characteristics of this descriptor: (1) It is simple and intuitive as it is directly generated based on connection lines between CS points. (2) It preserves the coarse-grained features hierarchically using different scales. (3) Built on the feature vectors, the distance between CSs can be quickly calculated by vector distance.

Sub Box: Sub Box descriptor [6] $\boldsymbol{f}_{27}$ preserves the medium-grained CS features by using the full- and sub-bounding boxes. As shown in Figure 6(c), in order to generate sub box descriptor, CS bounding box and sub-boxes with the same height ($h_1 = h_2 =$



(a) C (b) Line Segment (c) Sub Box

Figure 6: Some rich descriptors for a CS C.

h_3). Then a CS C is represented by a 11-dimensional feature vector including the CS length N, endpoint distance, area, angles, and height-width ratios of the full- and the sub-boxes.

3. Contour Segment Matching

We introduce the methods for calculating the distance $D(C_1, C_2)$ between two CSs C_1 and C_2, depending on types of CS descriptors presented in the previous section. After that, we introduce the open curve matching approach for our experiments.

3.1. CS Matching: Simple Descriptors

Since the simple descriptors $\boldsymbol{f}_1, \boldsymbol{f}_2, \cdots, \boldsymbol{f}_9$ are scalar, we calculate the distance $D(C_1, C_2)$ by:

$$D(C_1, C_2) = \frac{|\boldsymbol{f}_{1,j} - \boldsymbol{f}_{2,j}|}{\boldsymbol{f}_{1,j} + \boldsymbol{f}_{2,j}} \quad , \tag{5}$$

where $\boldsymbol{f}_{1,j}$ and $\boldsymbol{f}_{2,j}$ are the jth simple descriptors of C_1 and C_2, respectively. Note that simple descriptor values significantly vary depending on CSs. Thus, it is needed to make their differences independent of their descriptor values. To this end, Eq. 5 is designed to normalise the difference between two simple descriptor values.

3.2. CS Matching: Signature-based Descriptors

There are two methods for calculating $D(C_1, C_2)$ between C_1 and C_2 represented by signature-based descriptors: Point matching and vector distance. Let $\boldsymbol{f}_{1,j} = [f_{1,j}^1, \cdots, f_{1,j}^N]$

and $\boldsymbol{f}_{2,j} = [f_{2,j}^1, \cdots, f_{2,j}^N]$ be the jth signature-based CS descriptors for C_1 and C_2, respectively.

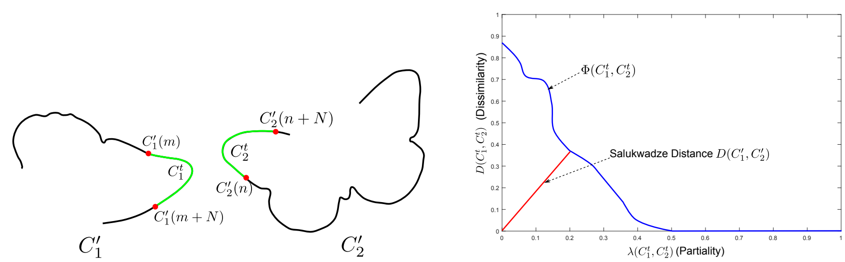
With the point matching method, we first calculate the differences $d(f_{1,j}^m, f_{2,j}^n)$ between $f_{1,j}^m$ in $\boldsymbol{f}_{1,j}$ and $f_{2,j}^n$ in $\boldsymbol{f}_{2,j}$ $(m, n = 1, 2, \cdots, N)$. Then, a matrix of differences between all CS points in C_1 and C_2 is generated. In order to find an optimum match of CS points between C_1 and C_2, we use the matching algorithms like DTW [13], DP [12] and Hungarian [11] on the matrix. Specifically, in order to avoid the brute-force approach [12] of the standard DTW algorithm, we employ the FastDTW [45] for CS point matching. For the DP, we employ the solution proposed by Sellers [46] to reduce the time complexity of traditional approach [12]. Hungarian algorithm [11] solves the assignment problem in a weighted bipartite graph. With this approach, the correspondence between CS points is generated by minimising the global cost between CS point distances. The resulting distance values of the matched CS points can be denoted as $s_1, \cdots, s_N$ and the similarity between C_1 and C_2 is calculated as the mean value.

With vector distance computation, we directly calculate the distance between $\boldsymbol{f}_{1,j}$ and $\boldsymbol{f}_{2,j}$ without considering any point matching. In particular, we employ the following distance measures: (1) Correlation [15], (2) Histogram Intersection (HI) [16], (3) χ^2-Statistics [17] and (4) Hellinger (or Bhattacharyya Coefficient) [18]. These distance measures are selected based on their simpleness and applicability evaluated in [47]. Since there are many other existing distance measures, in Section 6, we will discuss their adoption as our future work.

3.3. CS Matching: Rich Descriptors

Since feature vectors of rich descriptors $\boldsymbol{f}_{18}, \cdots, \boldsymbol{f}_{27}$ are not uniformed, they have their own way of matching CS points as well as calculating CS distances: (1) For Height Function descriptor $\boldsymbol{f}_{18}$, to match two CSs C_1 and C_2, the distance between any pair of points are computed by the weighted difference of their height features [37]. Then, a cost matrix is generated. Here, high weights are put on CS points near to the center to tolerate CS deformations. Finally, a matching algorithm like Hungarian, DP or DTW is applied on the cost matrix to get the similarity between C_1 and C_2. (2) For the descriptors $\boldsymbol{f}_{19}$ - $\boldsymbol{f}_{24}$, they are all constructed by the histograms of CS points. Thus, a cost matrix is obtained by computing the distance between any pair of points using the histogram intersection. Then, the overall similarity between C_1 and C_2 is calculated by applying a matching algorithm like Hungarian, DP or DTW to the cost matrix. (3) For Length Direction descriptor $\boldsymbol{f}_{25}$, the distance between C_1 and C_2 is calculated by the method in [44] in which the distance and orientation matrices are fused to get a affinity matrix that represents the similarities of corresponding point pairs. Then, the optimal correspondence and overall similarity between C_1 and C_2 are calculated with Hungarian, DP or DTW. (4) For Line Segment $\boldsymbol{f}_{26}$ and Sub Box $\boldsymbol{f}_{27}$, both descriptors are organised into vectors, so we can directly use the vector distance methods.

3.4. Open Curve Matching



(a) Open curve matching using CSs (green lines)

(b) Salukwadze distance (the red line)

Figure 7: Open curve matching and the open curve similarity method.

Given two open curves C_1' and C_2' with size N_1 and N_2, respectively, searching similar parts between C_1' and C_2' is equivalent to finding two CSs C_1 and C_2 with size N starting at the curve point $C_1'(m)$ and $C_2'(n)$ which yield the smallest distance $D(C_1, C_2)$, $m \leq N_1$, $n \leq N_2$, $N \leq \min(N_1, N_2)$ (Figure 7(a)). Therefore, the problem is reduced to searing the proper (m, n, N) combination which minimises the CS distance [20]. Note that the similarity between C_1 in C_1' and C_2 in C_2' depends on their length N. In other words, a smaller N often leads to a higher similarity. To avoid this undesirable effect of N, the Pareto-framework [48, 20] is employed for quantitative interpretation of partial similarity. Pareto-framework illustrates a way to select a multi-constrained path that can meet the optimal requirements.

As shown in Figure 7(b), we define two quantities: partiality $\lambda(C_1^t, C_2^t)$, which describes the length of the CS (the higher the value the smaller the CS), and the distance $D(C_1^t, C_2^t)$ which measures the dissimilarity. A Pareto optimum is defined by $\Phi(C_1^t, C_2^t)$ which is a pair of partiality and dissimilarity values that fulfill the criterion of the lowest dissimilarity for the given partiality. Finally, to achieve a similarity value between C_1' and C_2', the so-called Salukwadze distance is employed which measures the minimum distance from the origin $(0, 0)$ to the point on the Pareto optimum. The Salukwadze distance is then returned as the open curve similarity value.

4. Experimental Design

This section outlines our experimental setup to evaluate CS descriptors.

4.1. Datasets

To the best of our knowledge, there are no suitable datasets for evaluating the performance of CS descriptors. This is because in most existing CS-related applications [49, 37, 38, 40, 4, 42, 23], researchers propose CS descriptors only for their own specific scenarios. Moreover, we cannot directly employ the existing datasets [20, 44, 14, 6] for CS evaluation since they contain only images or shapes rather than CS. Thus, we have designed and collected three datasets for our experiments.

MPEG7 CS: MPEG7 [50] dataset is a standard and commonly used shape dataset for evaluating shape matching and classification. The total number of images in the MPEG7 database is 1400: 70 classes of various shapes, each class with 20 images. We create the MPEG7 CS dataset using its shape contours. For the shapes from the same class in MPEG7, we first extract their contours and then manually remove the same part from contours. Finally, the MPEG7 CS dataset is generated with 1400 CSs. For easy and fair performance evaluation of CS descriptors, the CSs we created have the same number of CS points. With this property, this dataset is also used for comparing the runtime between CS descriptors.

ETHZ CS: ETHZ [51] is a dataset for testing object class detection algorithms. It contains 255 test images and features five diverse shape-based classes (apple logos, bottles, giraffes, mugs and swans). Based on the shapes from ground truth, we manually generate the following open curves by keeping their contour parts in the parentheses: 44

open curves for apple logos (the right half part), 55 open curves for bottles (the right half part), 91 open curves for giraffes (the upper neck part), 48 open curves for mugs (the right half part with handle) and 32 open curves for swans (the upper half part with head). In total, there are 271 open curves. This dataset is used for evaluating the open curve matching.

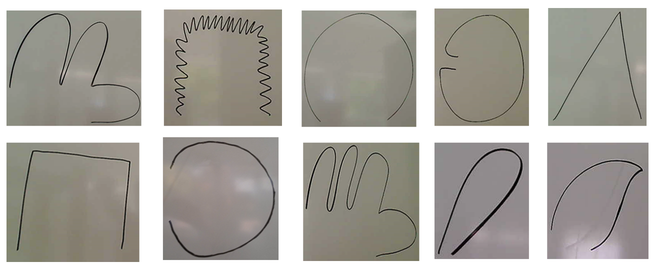


Figure 8: Sample sketches in the Sketching CS dataset.

Sketching CS: Sketching CS is a dataset collected by ourselves including 18 open curves of sketches which are commonly used for the application of sketch-based object retrieval [52]. We first collect images showing sketches drawn on a white board (see Figure 8). After that, all sketches are segmented and processed into open curves using the methods introduced in [53, 54]. This dataset is used for evaluating the difference between CS descriptors and human perception for sketching matching.

4.2. Experiment Types

Accurate matching requires effective CS descriptors to find perceptually similar curves from a database, irrespective of their rotations, translations and scales. A desirable CS descriptor should have stable performance on different types of datasets. Low computation complexity is also an important factor of a CS descriptor. Considering these requirements, we establish three types of experiments.

4.2.1. Experiment 1: Invariance Properties

Descriptors are often classified according to their invariance levels to certain geometrical transformations. We mainly assess three invariance properties of each CS descriptor: rotation, scaling and translation invariances. Rotation and scaling invariances mean that the CS features remain the same even when the CS is rotated and rescaled. Translation invariance means that two CSs can still be correctly matched even when every CS point is moved into a constant distance in a specified direction.

4.2.2. *Experiment 2: Matching Performance*

Table 1: Experiment settings for evaluating the matching performance of CS descriptors. L1, L2, L3 and L4 denote the sampling densities that are used for generating CS descriptors. NULL means this part is not considered in our experiments.

<table>
<tr><th>Descriptors</th><th colspan="2">Distance Functions</th><th>Matching Algorithms</th><th>Lengths</th><th>Datasets</th></tr>
<tr><td>Simple
($\boldsymbol{f}_1$ - $\boldsymbol{f}_9$)</td><td colspan="2">Difference Value</td><td>NULL</td><td>L1, L2, L3, L4</td><td>(1) ETHZ CS
(2) Sketching CS
(3) MPEG7 CS</td></tr>
<tr><td rowspan="2">Signature
($\boldsymbol{f}_{10}$ - $\boldsymbol{f}_{17}$)</td><td colspan="2">Difference Value</td><td>(1) DTW (2) DP (3) Hungarian</td><td rowspan="2">L1, L2, L3, L4</td><td rowspan="2">(1) ETHZ CS
(2) Sketching CS
(3) MPEG7 CS</td></tr>
<tr><td colspan="2">(1) Correlation (2) HI
(3) χ^2-Statistics (4) Hellinger</td><td>NULL</td></tr>
<tr><td rowspan="2">Rich
($\boldsymbol{f}_{18}$ - $\boldsymbol{f}_{27}$)</td><td>$\boldsymbol{f}_{18}$ - $\boldsymbol{f}_{25}$</td><td>Proposed Distances</td><td>(1) DTW (2) DP (3) Hungarian</td><td rowspan="2">L1, L2, L3, L4</td><td rowspan="2">(1) ETHZ CS
(2) Sketching CS
(3) MPEG7 CS</td></tr>
<tr><td>$\boldsymbol{f}_{26}$, $\boldsymbol{f}_{27}$</td><td>(1) Correlation (2) HI
(3) χ^2-Statistics (4) Hellinger</td><td>NULL</td></tr>
</table>

As illustrated in Table 1, we evaluate descriptors under multiple settings. For all descriptors, we generate their features using four sampling densities for selecting sample points from a CS: L1 = 25%, L2 = 50%, L3 = 75% and L4 = 100%. Based on this strategy, we aim to check the influence of sample point density to the the CS matching performance. With different combinations of distance functions, matching algorithms and CS lengths (sampling densities), the experiments are conducted on the proposed three datasets:

MPEG7 CS: We use this dataset to evaluate the CS matching performance in a retrieval scenario. We employ the so-called bulls-eye score [50] as an evaluation measure. Given a query CS, we retrieve the 40 most similar CSs from the database and count the number of CSs belonging to the same class as the query. The bulls-eye score is the ratio of the total number of correctly matched CSs to the number of all the possible matches (which is 20×1400). Thus, the best score is 100 percent.

ETHZ CS: The experiment on this dataset aims to evaluate the open curve matching performances using different CS descriptors. We conduct an open curve retrieval scenario using the method introduced in Section 3.4. Specifically, we use each of 271 curves as a query and retrieve the 100 most similar curves among the whole dataset. The final evaluation is composed of two parts: recall and precision. Both parts are averaged over curves in each class [55]. For example, since there are 44 open curves in the Apple logo class, the precision (or recall) for this class is calculated by taking the average of precisions (or recalls) obtained using each of 44 open curve as a query.

Sketching CS: Sketching CS dataset is used for evaluating the matching performance based on CS descriptors with respect to human perception. To implement this, we collect the ground truth by first carrying out an on-line questionnaire for selecting the most similar sketches to the query. Given a query sketch, we retrieve the most similar sketch and examine whether it is the most voted sketch by humans. Then the accuracy is calculated based on the ratio between the number of matched sketches and the total number of sketches.

4.2.3. Experiment 3: Computation Complexity

We examine the computation complexity of each CS descriptor in terms of the time complexity analysis and the real runtime. We theoretically analyse the computational time complexity for the generation and matching of each CS descriptors [56]. The real runtime is evaluated based on the full retrieval time of 1400 queries on the MPEG7 CS dataset as the runtime.

4.3. Experimental Environment

The experiments in this paper are delivered on two platforms: Cluster and laptop. Feature generation and full object retrieval experiments are accomplished on *Horus*, a cluster provided by the University of Siegen, which includes 136 nodes, each consisting of 2 Intel Xeon X5650 with 2,66 GHz and 48 GB memory. This enables us to efficiently finish our massive experiments. In order to fairly compare the runtime of each CS descriptor, the runtime estimation experiments are finished on a laptop with Inter Core i7 2.2GHz CPU, 8.00GB memory and 64-bit Windows 8.1 OS. All methods in our experiment are implemented in Matlab.

5. Results

This section presents results in the experimental settings defined in Section 4.

Table 2: Invariance properties of different CS descriptors. SI, RI and TI denote the scale invariance, rotation invariance and translation invariance. The existence and lack of an invariance are indicated with '+' and '-', respectively. Regarding $+^*$ of $\boldsymbol{f}_{14}$, if the sampling is dense enough, then it is rotation invariant, and vice versa.

Simple CS Descriptors				Signature-based CS Descriptors				Rich CS Descriptors			
Descriptors	SI	RI	TI	Descriptors	SI	RI	TI	Descriptors	SI	RI	TI
$\boldsymbol{f}_1$	+	+	+	$\boldsymbol{f}_{10}$	-	-	-	$\boldsymbol{f}_{18}$	+	+	+
$\boldsymbol{f}_2$	+	+	+	$\boldsymbol{f}_{11}$	-	+	+	$\boldsymbol{f}_{19}$	+	+	+
$\boldsymbol{f}_3$	+	+	+	$\boldsymbol{f}_{12}$	+	+	+	$\boldsymbol{f}_{20}$	+	-	+
$\boldsymbol{f}_4$	+	+	+	$\boldsymbol{f}_{13}$	+	+	+	$\boldsymbol{f}_{21}$	+	+	+
$\boldsymbol{f}_5$	+	+	+	$\boldsymbol{f}_{14}$	-	$+^*$	+	$\boldsymbol{f}_{22}$	+	+	+
$\boldsymbol{f}_6$	+	+	+	$\boldsymbol{f}_{15}$	-	+	+	$\boldsymbol{f}_{23}$	+	+	+
$\boldsymbol{f}_7$	+	+	+	$\boldsymbol{f}_{16}$	+	+	+	$\boldsymbol{f}_{24}$	+	+	+
$\boldsymbol{f}_8$	+	+	+	$\boldsymbol{f}_{17}$	+	+	+	$\boldsymbol{f}_{25}$	+	+	+
$\boldsymbol{f}_9$	+	+	+					$\boldsymbol{f}_{26}$	+	+	+
								$\boldsymbol{f}_{27}$	+	+	+

5.1. Experiment 1: Invariant Properties

Table 2 illustrates the theoretical invariance properties of different CS descriptors. Firstly, all the CS descriptors are invariant to translation of CSs, except for Comcoor ($\boldsymbol{f}_{10}$) which is represented by coordinates of CS points ($\boldsymbol{f}_{10}$ is also neither scaling nor rotation invariant). Secondly, we can clearly observe that all the simple CS descriptors are invariant to rotation since they are generated by only considering the overall feature of a CS. Moreover, normalisation process can ensure the scaling invariance of simple descriptors. Thirdly, some signature-based descriptors, such as the Area Function ($\boldsymbol{f}_{14}$) and Triangle Area ($\boldsymbol{f}_{15}$), do not perform well for scaled CSs. In practice, we can normalise them by the CS length. These descriptors thereby become scale-invariant. Lastly, except Contour Context ($\boldsymbol{f}_{20}$), all the rich descriptors comply with the three invariance properties. In practice, we can sound the rotation property of Contour Context descriptor by the preprocessing method introduced in [6].

5.2. Experiment 2: Matching Performance

Here, we report the matching performance on three datasets.

5.2.1. Evaluation on MPEG7 CS dataset

Table 3 illustrates the CS retrieval results on MPEG7 CS dataset using simple CS descriptors. We can see that Eccentricity ($\boldsymbol{f}_3$), Bending ($\boldsymbol{f}_4$) and Rectangularity ($\boldsymbol{f}_5$)

Table 3: CS matching results (%) using Simple CS descriptors on MPEG7 CS Dataset.

Descriptors	L1	L2	L3	L4
$\boldsymbol{f}_1$	3.7	3.7	3.7	3.7
$\boldsymbol{f}_2$	3.7	3.7	3.7	3.7
$\boldsymbol{f}_3$	**22.3**	**22.8**	**22.8**	**22.8**
$\boldsymbol{f}_4$	**22.9**	**24.0**	**24.1**	**23.4**
$\boldsymbol{f}_5$	**22.3**	**22.8**	**22.8**	**22.8**
$\boldsymbol{f}_6$	18.7	18.9	18.9	18.9
$\boldsymbol{f}_7$	16.6	15.9	15.7	16.0
$\boldsymbol{f}_8$	3.7	3.7	3.7	3.7
$\boldsymbol{f}_9$	18.7	18.9	18.9	18.9

outperform the other descriptors in which Bending ($\boldsymbol{f}_4$) achieves the best performance on MPEG7 CS dataset. Moreover, considering the performance on different lengths, we can see that for each simple descriptor, the scores are almost the same. Therefore, all the simple CS descriptors are robust to CS length changes. The rationale behind this is that simple CS descriptors are calculated by considering only global coarse-grained CS features. Even if some fine-grained features get lost because of small number of sample points, the global features remain the same.

Table 4: CS matching results (%) with Signature CS descriptors using point matching methods on MPEG7 CS Dataset.

Descriptors	DTW				DP				Hungarian			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{10}$	62.1	62.3	62.3	62.4	50.0	49.9	50.6	50.9	**63.3**	**64.1**	**64.2**	**64.4**
$\boldsymbol{f}_{11}$	60.4	61.6	61.8	61.8	58.1	60.7	61.1	61.4	48.0	48.3	48.3	48.3
$\boldsymbol{f}_{12}$	54.5	55.9	60.0	61.4	54.9	55.7	59.6	60.8	51.1	51.4	50.0	49.8
$\boldsymbol{f}_{13}$	50.3	52.8	54.8	55.9	49.3	52.1	54.2	55.0	46.0	48.9	49.6	48.9
$\boldsymbol{f}_{14}$	55.0	57.7	57.7	59.4	50.8	54.5	53.7	56.4	41.9	43.6	38.6	42.4
$\boldsymbol{f}_{15}$	58.3	59.6	60.0	59.9	53.8	55.7	56.3	56.6	49.0	49.4	49.6	49.7
$\boldsymbol{f}_{16}$	3.2	2.9	2.9	2.9	3.3	3.0	2.9	2.9	3.3	2.9	2.9	2.9
$\boldsymbol{f}_{17}$	15.4	14.5	12.4	10.8	41.1	40.3	36.3	32.4	38.5	39.9	39.9	40.6

For the signature-based CS descriptors on MPEG7 CS dataset, in Table 4, Comcoor ($\boldsymbol{f}_{10}$) with Hungarian matching method achieves the best bulls-eye score. In Table 5, both Comcoor ($\boldsymbol{f}_{10}$) and Cendistance ($\boldsymbol{f}_{11}$) with Hellinger distance method obtain the best score. Among all results, Comcoor ($\boldsymbol{f}_{10}$) with Hungarian matching method achieves the best performance (64.4% bulls-eye score). Similar to the ETHZ dataset,

Table 5: CS matching results (%) with Signature CS descriptors using vector distance methods on MPEG7 CS dataset.

Descriptors	Correlation				HI				χ^2-Statistics				Hellinger			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{10}$	8.2	7.1	7.1	6.9	13.5	14.0	14.5	14.3	55.7	55.8	56.5	56.6	**63.7**	**63.6**	**63.7**	**63.5**
$\boldsymbol{f}_{11}$	5.6	5.6	5.6	5.5	5.5	5.4	5.4	5.4	63.0	62.7	62.8	62.6	**63.7**	**63.6**	**63.7**	**63.5**
$\boldsymbol{f}_{12}$	5.9	5.8	5.9	5.9	7.1	7.0	6.8	6.6	44.2	47.4	50.0	49.4	42.7	41.8	44.8	43.7
$\boldsymbol{f}_{13}$	6.0	6.3	6.3	6.2	6.0	8.7	14.0	19.0	35.1	20.6	16.3	12.2	55.3	52.6	49.6	45.7
$\boldsymbol{f}_{14}$	5.6	5.6	6.2	5.6	5.1	5.1	5.1	5.1	54.4	54.8	54.8	54.3	49.7	49.2	47.7	46.3
$\boldsymbol{f}_{15}$	5.8	5.9	5.7	5.6	5.5	5.5	5.5	5.7	36.2	34.5	35.3	32.1	51.1	51.8	51.9	51.3
$\boldsymbol{f}_{16}$	3.3	3.0	2.9	2.9	2.9	2.9	2.9	2.9	3.2	2.9	2.9	2.9	2.9	2.9	2.9	2.9
$\boldsymbol{f}_{17}$	44.6	39.5	33.0	27.3	48.9	37.5	28.0	24.7	0.07	0.15	0.17	0.18	42.0	30.4	22.3	18.1

we can observe that the CS matching performance is enhanced and damaged dramatically if the matching algorithms or vector distance methods are changed. Moreover, for most signature-based CS descriptors, CS retrieval using point matching methods performs much better than the vector distance methods.

Table 6 and Table 7 illustrate the CS retrieval performance using rich CS descriptors. Among all rich descriptors, Beam Angle ($\boldsymbol{f}_{21}$) achieves promising results in all three matching methods, in which Hungarian matching method yields the best score (79.6% bulls-eye score). For most rich descriptors ($\boldsymbol{f}_{18}$ - $\boldsymbol{f}_{26}$), they are relatively robust to the CS length and matching algorithm changing. However, the performance of Sub Box ($\boldsymbol{f}_{27}$) significantly relies on the vector distance methods in which the Hellinger method outperforms the other vector distance methods. One main reason is that feature values in a 11-dimensional Sub Box descriptor are significantly varied. Hellinger method can handle biased value distributions over dimensions while the other distance methods are easily affected by large values. Moreover, as Sub Box descriptor only capture the medium-grained features, its matching accuracies are worse than the Line Segment ($\boldsymbol{f}_{26}$) descriptor which preserve the fine-grained features.

Comparing the performance of three types of CS descriptors in MPEG7 CS dataset, we can draw the following observations: (1) Most rich descriptors have a better performance than the signature-based descriptors. (2) Most signature-based CS descriptors outperform the simple CS descriptors. (3) For most signature-based CS descriptors, point matching strategy performs better than the vector distance strategy.

Table 6: CS matching results (%) using Rich CS descriptors and point matching methods on MPEG7 CS Dataset.

Descriptors	DTW				DP				Hungarian			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{18}$	64.9	64.7	63.7	63.4	70.8	74.8	76.2	76.9	62.9	70.7	72.7	74.0
$\boldsymbol{f}_{19}$	40.9	39.0	36.9	35.8	68.5	69.5	70.3	70.5	69.8	70.1	70.4	70.3
$\boldsymbol{f}_{20}$	64.2	64.5	64.5	64.5	62.4	63.9	64.2	64.4	66.2	67.7	68.1	68.4
$\boldsymbol{f}_{21}$	72.8	75.1	74.6	73.6	73.1	75.1	75.3	73.8	**79.6**	**79.6**	**77.4**	**73.9**
$\boldsymbol{f}_{22}$	64.6	64.6	64.8	64.4	69.2	73.3	76.6	77.4	64.8	66.4	67.2	66.5
$\boldsymbol{f}_{23}$	57.6	60.1	61.0	61.2	48.8	52.6	54.2	54.2	47.4	51.5	53.4	53.5
$\boldsymbol{f}_{24}$	66.3	66.0	65.9	65.3	70.6	73.8	76.5	77.3	69.4	72.8	74.3	75.0
$\boldsymbol{f}_{25}$	64.2	64.6	64.7	64.8	64.1	64.3	64.3	64.4	64.1	64.3	64.4	64.4

Table 7: CS matching results (%) using Rich Descriptors and vector distance methods on MPEG7 CS Dataset.

Descriptors	Correlation				HI				χ^2-Statistics				Hellinger			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{26}$	70.8	70.7	70.5	70.3	70.0	71.0	71.2	71.2	72.7	73.2	73.3	73.3	74.5	74.8	74.8	74.8
$\boldsymbol{f}_{27}$	5.8	5.9	5.7	5.6	5.5	5.5	5.5	5.7	36.2	34.5	35.3	32.1	51.1	51.8	51.9	51.3

5.2.2. Evaluation on ETHZ CS dataset

Table 8: Open curve matching results (%) using Simple CS descriptors on ETHZ CS Dataset.

Descriptors	L1		L2		L3		L4	
	P	R	P	R	P	R	P	R
$\boldsymbol{f}_1$	33.9	56.8	34.0	56.9	34.1	57.1	34.0	57.0
$\boldsymbol{f}_2$	42.5	75.1	42.8	75.5	42.7	75.5	42.7	75.4
$\boldsymbol{f}_3$	**55.0**	**90.0**	**55.1**	**90.1**	**55.1**	**90.1**	**55.1**	**90.1**
$\boldsymbol{f}_4$	48.1	80.7	48.8	81.7	48.9	81.9	49.3	82.4
$\boldsymbol{f}_5$	**55.0**	**90.0**	**55.1**	**90.1**	**55.1**	**90.1**	**55.1**	**90.1**
$\boldsymbol{f}_6$	52.5	87.9	52.5	87.8	52.5	87.9	52.5	87.9
$\boldsymbol{f}_7$	37.1	61.1	37.1	60.8	36.1	58.8	36.4	59.7
$\boldsymbol{f}_8$	38.1	65.7	38.1	65.6	38.2	65.8	38.2	65.8
$\boldsymbol{f}_9$	52.5	87.9	52.5	87.8	52.5	87.9	52.5	87.9

Table 8 shows Precisions (P) and Recalls (R) of simple CS descriptors. As seen from this figure, Eccentricity ($\boldsymbol{f}_3$) and Rectangularity ($\boldsymbol{f}_5$) outperform the other descriptors (90.1% recall). Moreover, considering the performance on different lengths, like MPEG7 CS dataset, all simple CS descriptors are robust to CS length changing.

For the signature-based CS descriptors, as shown in Table 9 and Table 10, we can clearly observe that for both point matching and vector distance strategies, matching

Table 9: Open curve matching results (%) with Signature CS descriptors using point matching on ETHZ CS Dataset.

Descriptors		DTW				DP				Hungarian			
		L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{10}$	P	52.6	52.8	52.8	52.8	34.7	34.6	34.6	34.6	46.7	49.6	49.6	49.6
	R	86.0	86.2	86.3	86.4	58.3	58.0	58.1	58.0	81.3	81.3	81.2	81.2
$\boldsymbol{f}_{11}$	P	32.4	32.4	32.5	32.5	32.3	32.3	32.4	32.5	30.4	30.2	30.3	30.2
	R	53.3	53.2	53.4	53.4	52.8	52.7	52.9	53.0	50.3	49.9	50.1	49.8
$\boldsymbol{f}_{12}$	P	52.6	53.9	53.3	53.9	**54.4**	**53.0**	**52.8**	**53.9**	53.0	52.6	53.1	53.7
	R	89.0	90.4	89.4	90.1	**90.9**	**89.2**	**88.8**	**90.0**	85.8	85.7	86.8	87.8
$\boldsymbol{f}_{13}$	P	48.0	49.8	46.8	44.1	46.6	44.9	42.9	41.0	46.7	44.4	41.4	40.3
	R	79.3	81.7	78.2	74.7	77.5	74.9	72.4	70.1	77.0	73.4	40.3	67.8
$\boldsymbol{f}_{14}$	P	33.1	32.2	33.8	31.7	31.4	30.8	32.4	30.7	32.7	32.9	30.8	32.3
	R	54.2	52.7	55.0	52.0	50.9	49.9	52.5	49.9	53.6	53.9	50.7	53.2
$\boldsymbol{f}_{15}$	P	38.6	38.3	38.3	38.1	32.9	32.6	32.7	32.3	34.4	34.5	34.6	34.6
	R	62.8	62.4	62.4	61.7	53.7	53.1	53.0	52.6	57.2	57.4	57.4	57.4
$\boldsymbol{f}_{16}$	P	22.4	21.8	19.3	19.5	22.3	21.8	19.4	19.5	22.3	21.7	19.3	19.5
	R	38.1	37.7	37.1	37.0	38.0	37.6	37.1	37.0	38.0	37.6	37.1	37.0
$\boldsymbol{f}_{17}$	P	23.1	21.3	22.5	22.3	27.3	24.8	25.5	26.6	30.9	28.4	29.0	29.7
	R	43.4	38.9	40.8	38.8	52.2	50.0	48.9	49.4	57.0	51.6	53.2	54.9

Table 10: Open curve matching results (%) with Signature CS descriptors using vector distance on ETHZ CS Dataset.

Descriptors		Correlation				HI				χ^2-Statistics				Hellinger			
		L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{10}$	P	49.0	49.6	49.5	49.7	29.7	27.8	28.5	27.0	**53.3**	**53.3**	**52.5**	**52.4**	49.4	49.4	49.2	48.8
	R	83.6	84.3	84.2	84.4	48.9	45.6	46.8	44.9	**90.4**	**90.4**	**89.5**	**89.4**	84.3	84.1	84.0	83.6
$\boldsymbol{f}_{11}$	P	5.8	6.0	6.0	6.0	17.0	17.2	17.1	17.1	35.1	34.9	35.0	35.0	49.4	49.4	49.2	48.8
	R	9.9	10.2	10.2	10.3	27.9	28.2	28.0	28.1	57.6	57.3	57.5	57.5	84.3	84.2	84.0	83.5
$\boldsymbol{f}_{12}$	P	12.7	15.1	14.1	14.0	4.5	5.7	5.9	6.2	51.4	49.9	49.8	49.8	43.4	39.0	38.9	38.0
	R	17.2	20.5	19.3	19.3	6.6	8.3	8.6	9.1	84.4	82.8	82.7	82.9	74.8	69.3	69.1	67.8
$\boldsymbol{f}_{13}$	P	4.7	5.1	6.7	7.9	45.4	29.2	11.5	9.7	47.4	42.0	34.1	30.2	53.2	55.2	52.1	50.0
	R	8.2	9.2	11.5	13.0	76.7	52.5	23.3	20.3	77.4	67.9	56.4	50.7	87.8	88.8	84.3	81.2
$\boldsymbol{f}_{14}$	P	4.3	4.6	3.6	5.4	14.0	14.0	14.1	14.3	38.3	38.1	37.8	37.5	49.1	47.3	47.2	47.2
	R	7.7	7.8	5.6	9.5	23.0	23.1	23.2	23.4	62.6	62.4	62.0	61.6	82.9	81.2	81.0	80.6
$\boldsymbol{f}_{15}$	P	9.0	9.3	9.0	8.4	27.5	26.8	27.1	26.6	40.2	39.3	39.3	39.2	43.5	43.9	43.9	44.8
	R	15.3	15.7	15.3	14.3	47.6	46.7	47.0	45.9	66.6	65.4	65.4	65.0	73.2	73.7	73.7	74.9
$\boldsymbol{f}_{16}$	P	22.2	21.4	19.3	19.6	18.5	18.5	18.6	18.6	22.0	21.8	19.3	19.5	18.0	18.5	18.7	18.7
	R	37.8	37.0	37.1	37.1	36.7	36.6	36.9	36.9	37.2	37.3	37.0	37.0	35.4	36.1	36.9	36.9
$\boldsymbol{f}_{17}$	P	33.7	26.0	25.8	29.1	34.6	28.6	29.7	29.9	19.5	15.8	16.0	21.1	33.3	26.5	24.4	24.3
	R	59.7	47.0	48.0	54.7	65.6	55.6	57.7	57.9	36.0	27.2	28.4	39.6	63.0	52.1	47.8	47.0

performances are highly related to the matching methods. For two strategies, point matching methods perform better than the vector distance methods. Specifically, Tangent ($\boldsymbol{f}_{12}$) with dynamic programming and Comcoor ($\boldsymbol{f}_{10}$) with χ^2-Statistics achieve the best performance in two strategies, respectively. However, considering their best recall and precision scores, $\boldsymbol{f}_{12}$ and $\boldsymbol{f}_{10}$ are very close to each other though $\boldsymbol{f}_{12}$ with dynamic programming is slightly better.

For the rich CS descriptors, similar to signature-based descriptors, matching algorithms have big influences on recall and precision scores (Table 11 and Table 12). Compared to other CS rich descriptors, Point Triangle ($\boldsymbol{f}_{19}$), Contour Context ($\boldsymbol{f}_{20}$) and Length Direction ($\boldsymbol{f}_{25}$) are stable for CS length changing and also have good matching performance (more that 90% recall). Among all these descriptors, Point Triangle ($\boldsymbol{f}_{19}$) with dynamic programming achieves the best performance (95.4% recall). For Line Segment ($\boldsymbol{f}_{26}$) and Sub Box ($\boldsymbol{f}_{27}$) descriptors (Table 12) which are using the vector distance methods, both of their matching performances are not as good as Point Triangle ($\boldsymbol{f}_{19}$) with dynamic programming.

Table 11: Open curve matching results (%) with Rich CS descriptors and point matching methods on ETHZ CS Dataset.

Descriptors		DTW				DP				Hungarian			
		L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{18}$	P	40.8	40.3	40.0	39.3	46.0	46.5	46.5	46.3	46.8	47.2	47.2	47.2
	R	77.6	76.8	76.0	75.1	87.5	88.6	88.5	88.1	88.3	89.0	89.2	89.0
$\boldsymbol{f}_{19}$	P	43.0	41.1	41.0	41.0	**51.2**	**50.4**	**50.2**	**50.0**	51.0	50.1	50.0	50.0
	R	83.5	81.0	80.2	80.0	**95.4**	**94.2**	**94.0**	**94.0**	94.5	94.0	93.6	94.0
$\boldsymbol{f}_{20}$	P	41.5	41.5	41.5	41.5	50.0	50.0	49.5	49.5	49.0	49.0	49.0	49.0
	R	79.0	78.6	78.5	78.5	94.0	94.0	94.1	94.1	93.0	93.0	93.0	93.0
$\boldsymbol{f}_{21}$	P	45.3	41.2	37.3	35.3	41.0	37.0	34.0	32.4	43.5	40.1	37.3	35.3
	R	86.2	79.0	72.0	67.8	78.5	71.1	65.5	62.3	83.0	76.4	71.3	67.1
$\boldsymbol{f}_{22}$	P	41.1	40.2	40.0	39.6	46.1	46.3	46.2	46.1	44.6	43.6	43.1	42.9
	R	76.6	75.3	74.6	74.2	87.4	88.1	87.9	87.8	83.9	82.3	81.5	81.3
$\boldsymbol{f}_{23}$	P	48.8	48.7	48.7	48.8	30.1	28.4	27.8	27.7	30.8	29.0	28.5	28.3
	R	90.1	90.1	90.1	90.2	59.5	57.1	56.5	56.5	60.8	58.3	57.5	57.5
$\boldsymbol{f}_{24}$	P	41.5	40.5	40.0	39.7	46.7	46.9	47.0	47.0	46.4	46.7	46.8	46.9
	R	77.3	75.8	75.1	74.4	88.6	89.1	89.3	89.4	87.7	88.1	88.3	88.5
$\boldsymbol{f}_{25}$	P	40.6	40.6	40.6	40.6	48.7	48.7	48.7	48.7	40.4	40.4	40.5	40.4
	R	76.5	76.6	76.6	76.4	90.9	91.0	91.0	91.0	76.3	76.3	76.3	76.2

Comparing the performance of open curve matching using simple, signature-based and rich CS descriptors on ETHZ CS dataset, we can draw the following observations: (1) Most of rich CS descriptors outperform signature-based CS descriptors. (2) Most

Table 12: Open curve matching results with Rich CS descriptors and vector distance methods on ETHZ CS Dataset.

Descriptors		Correlation				HI				χ^2-Statistics				Hellinger			
		L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
$\boldsymbol{f}_{26}$	P	39.3	39.0	39.2	39.1	33.2	33.2	33.4	33.2	29.2	29.2	29.3	29.2	43.7	43.5	43.7	43.5
	R	73.9	73.8	74.3	74.2	61.9	61.8	62.3	61.9	55.2	55.2	55.4	55.3	80.7	80.3	80.8	80.4
$\boldsymbol{f}_{27}$	P	5.6	3.7	3.3	3.4	2.6	2.2	2.2	2.3	51.6	53.7	54.1	53.7	52.3	53.5	53.6	53.4
	R	10.0	7.6	6.9	6.9	5.2	4.7	4.6	4.8	83.1	85.9	86.3	85.6	84.1	85.6	85.9	85.6

signature-based CS descriptors performs better than the simple CS descriptors. (3) For most signature-based CS descriptors, point matching strategy performs better than the vector distance strategy for open curve matching. (4) For open curve matching on ETHZ dataset, Point Triangle ($\boldsymbol{f}_{19}$) with dynamic programming achieves the best performance (95.4% recall). (5) Considering CS descriptors with the best performance in Table 3- 12, we can observe that the precision and recall values of the best performance in those tables are close to each other. The main reason is that for open curve matching tasks, the influence of a individual descriptor is relatively reduced because the open curve similarity value is calculated based on the statistics of plentiful CS lengths and distances. Moreover, we can also observe that for the most CS descriptors, if one has good performance for CS matching in the MPEG7 CS dataset, it also achieves promising results for open curve matching in the ETHZ CS dataset. Thus, signature-based and rich descriptors with proper matching algorithms can fulfil different requirements in terms of speed and accuracy for open curve matching.

5.2.3. Evaluation on Sketching CS dataset

The purpose of this experiment is to evaluate the difference between CS descriptors and human perception for sketching matching. Figure 5.2.3 illustrates the comparison of all accuracy values among three types of CS descriptors. Note that the horizontal axis represents each value's ID which is used only for visualisation. We focus on the vertical axis by analysing the accuracy values. Considering the value distribution, signature-based CS descriptors have the biggest variation ranging from 2.5% to 80%, followed by the rich CS descriptors which are in the range of $[10\%, 80\%]$. Simple CS descriptors have the smallest interval $[20\%, 50\%]$. One reason is that for each signature-based and rich CS descriptors, there are many algorithms that can be em-

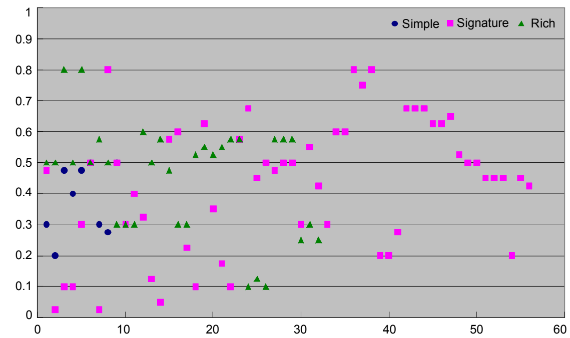


Figure 9: Comparison of accuracy values between three types of CS descriptors. Each mark represents a accuracy value with a different combination of a descriptor and matching algorithm.

ployed for calculating similarities between CSs. On the contrary, simple CS descriptors only have one. Therefore, rich and signature-based CS descriptors have more flexibility for handling different scenarios. Considering the mean accuracy values among three types of descriptors, rich CS descriptors (52.68%) is closer to the human perception than signature-based (47.68%) while the simple CS descriptors (40%) have the lowest one. Overall, there is still a gap between the human perception and CS descriptors (mean value 46.79%).

5.3. Experiment 3: Computation Complexity

Table 13: Theoretical analysis of computation complexity for each CS descriptor in terms of feature generation and matching. NULL means this part is not considered in our experiments.

<table>
<tr><th>Name</th><th colspan="2">Feature Generation</th><th colspan="3">Difference Value</th></tr>
<tr><td>$\boldsymbol{f}_1$ - $\boldsymbol{f}_9$</td><td colspan="2">$O(N)$</td><td colspan="3">$O(1)$</td></tr>
<tr><td rowspan="2">Name</td><td rowspan="2">Feature Generation</td><td colspan="3">Matching (Point Matching)</td><td rowspan="2">Matching (Vector Distance)</td></tr>
<tr><td>DTW [13]</td><td>DP [12]</td><td>Hungarian [11]</td></tr>
<tr><td>$\boldsymbol{f}_{10}$ - $\boldsymbol{f}_{14}$, $\boldsymbol{f}_{17}$</td><td>$O(N)$</td><td>$O(N^2)$</td><td>$O(N^2)$</td><td>$O(N^3)$</td><td>$O(N)$</td></tr>
<tr><td>$\boldsymbol{f}_{15}$, $\boldsymbol{f}_{16}$</td><td>$O(N^2)$</td><td>$O(N^2)$</td><td>$O(N^2)$</td><td>$O(N^3)$</td><td>$O(N)$</td></tr>
<tr><td>$\boldsymbol{f}_{18}$ - $\boldsymbol{f}_{20}$
$\boldsymbol{f}_{22}$ - $\boldsymbol{f}_{25}$</td><td>$O(N^2)$</td><td>$O(N^2)$</td><td>$O(N^2)$</td><td>$O(N^3)$</td><td>NULL</td></tr>
<tr><td>$\boldsymbol{f}_{21}$</td><td>$O(N)$</td><td>$O(N^2)$</td><td>$O(N^2)$</td><td>$O(N^3)$</td><td>NULL</td></tr>
<tr><td>$\boldsymbol{f}_{26}$</td><td>$O(N)$</td><td>NULL</td><td>NULL</td><td>NULL</td><td>$O(1)$</td></tr>
<tr><td>$\boldsymbol{f}_{27}$</td><td>$O(N^2)$</td><td>NULL</td><td>NULL</td><td>NULL</td><td>$O(1)$</td></tr>
</table>

Table 13 illustrates the theoretical analysis of computational complexity for each CS descriptor. All the simple CS descriptors have the same feature generation complexity $O(N)$ since they are generated by simply accumulating values computed for

N CS points. As simple CS descriptors are just scalar values, their distance can be computed by scalar subtraction. Thus, its computation cost is $O(1)$.

For signature-based descriptors, except Triangle Area ($\boldsymbol{f}_{15}$) and Chord Length ($\boldsymbol{f}_{16}$), most descriptors take $O(N)$ complexity for feature generation since each element is calculated only using one CS point. $\boldsymbol{f}_{15}$ and $\boldsymbol{f}_{16}$ are both calculated by considering one target CS point and other reference points selected by searching the whole CS path. Therefore, their computation complexity is $O(N^2)$. For CS retrieval, with the vector distance strategy, all four distance methods have the same computation complexity $O(N)$.

With the point matching strategy, three matching algorithms have different computation complexity. More specifically, (i) the computational complexity of DTW is $O(N^2)$ because it needs to compute distances for all possible point pairs in two CSs, each having N points. (ii) For solving the sequence alignment problem using DP, we reduce the time complexity from $O(N^3)$ with the traditional brute-force approach [12]) to $O(N^2)$ with the method introduced in [46]. This is because the method in [46] makes a complete list of all pairs of intervals using given CSs so that each pair displays a maximum local degree of similarity. Like this, the matching complexity is reduced by trading space for time. (iii) Hungarian algorithm solves our CS matching task in $O(N^3)$ time as introduced in [57].

For rich CS descriptors, we can observe that most descriptors require (or more than) $O(N^2)$ complexity for feature generation. In contrast, Beam Angle ($\boldsymbol{f}_{21}$) is calculated by the angles between CS points and its neighbouring points which can be captured directly. Line Segment ($\boldsymbol{f}_{26}$) is calculated by the statistics of a fixed range of straight-line scales n (n is set from 5% to 85% in all experiments). Since n is independent of CS point number N and $n \ll N$, the computational complexity of this method is determined by the number of CS points. Therefore, the complexity of $\boldsymbol{f}_{21}$ and $\boldsymbol{f}_{26}$ are $O(N)$. For CS matching, the point matching strategy is applied to descriptors ranging from $\boldsymbol{f}_{18}$ to $\boldsymbol{f}_{25}$, in which the matching algorithms have the same complexity as signature-based descriptors. For the vector distance strategy, Line Segment ($\boldsymbol{f}_{26}$) and Sub Box ($\boldsymbol{f}_{27}$) have the complexity $O(n)$ where n is the feature dimension. Since their feature dimensions are fixed and the distance between two vectors can be calculated in a constant time, their computational complexity is $O(1)$.

5.4. Discussion

Table 14: Matching performance and time (hour) comparison between selected CS descriptors which have outstanding matching performances on ETHZ and MPEG7 CS datasets.

Name	Notation	Type	Matching Method	FG	Matching	ETHZ CS*	MPEG7 CS*
Comcoor	$\boldsymbol{f}_{10}$	Signature	χ^2-Statistics	0.0001	0.0542	P: 52.9, R: 89.9	56.2
Comcoor	$\boldsymbol{f}_{10}$	Signature	Hungarian	0.0001	213.74	P: 48.9, R: 81.3	64.0
Comcoor	$\boldsymbol{f}_{10}$	Signature	Hellinger	0.0001	0.0531	P: 49.2, R: 84.0	63.6
Cendistance	$\boldsymbol{f}_{11}$	Signature	Hellinger	0.0001	0.0462	P: 49.2, R: 84.0	63.6
Tangent	$\boldsymbol{f}_{12}$	Signature	DP	0.0001	3.86	P: 53.5, R: 89.7	57.8
Point Triangle	$\boldsymbol{f}_{19}$	Rich	DP	6.82	121.8	P: 50.5, R: 94.4	69.7
Contour Context	$\boldsymbol{f}_{20}$	Rich	DP	0.04	6.7	P: 49.6, R: 94.1	63.7
Beam Angle	$\boldsymbol{f}_{21}$	Rich	Hungarian	0.01	147.1	P: 39.1, R: 74.5	77.6
Partial Contour	$\boldsymbol{f}_{22}$	Rich	DP	0.18	5.3	P: 46.2, R: 87.8	74.1
Opt Partial Contour	$\boldsymbol{f}_{23}$	Rich	DTW	0.17	22.1	P: 48.8, R: 90.1	60.0
Chord Distribution	$\boldsymbol{f}_{24}$	Rich	DP	0.18	5.4	P: 46.9, R: 89.1	74.6
Length Direction	$\boldsymbol{f}_{25}$	Rich	DP	0.02	212.1	P: 48.7, R: 91.0	64.3

* As there are four values for four different lengths, we take the mean value for comparison.

P: Precision, R: Recall, D-Value: Difference Value, FG: Feature Generation

Table 14 illustrates the comparison between the selected descriptors which have outstanding matching performances on ETHZ and MPEG7 CS datasets. We can observe that the selected signature-based and rich descriptors have robust performances in which Beam Angle ($\boldsymbol{f}_{21}$) with Hungarian [11] achieves the best bulls-eye score (77.6%). Tangent ($\boldsymbol{f}_{12}$) and Point Triangle ($\boldsymbol{f}_{19}$) with Dynamic Programming [12] obtain the best precision (53.5%) and recall (94.4%). However, considering the runtime and retrieval performance, Partial Contour ($\boldsymbol{f}_{22}$) and Chord Distribution ($\boldsymbol{f}_{24}$) are close to the best while taking less time for feature generation and matching.

Based on the observations above, we can draw the following recommendations: (i) When choosing a CS descriptor for open curve matching with time complexity not being of primary importance, the best choice is Point Triangle ($\boldsymbol{f}_{19}$) with Dynamic Programming [12] since it is scale, rotation and translation invariant (Table 2) and robust to CS length changing (Table 6 and Table 11). Moreover, it achieves promising results on both ETHZ and MPEG7 CS datasets (Table 14). (ii) Partial Contour ($\boldsymbol{f}_{22}$) and Chord Distribution ($\boldsymbol{f}_{24}$) with DP are the best choices to obtain a stable and promising performance while taking less computational time, as shown in Table 14. (iii) If we want to apply a fast open curve matching and obtain a relatively promising results, Comcoor ($\boldsymbol{f}_{10}$) and Cendistance ($\boldsymbol{f}_{11}$) with Hellinger [18] vector distance method are the best choice, for which the fast runtime for feature generation and matching can be ensured while the matching performance on both datasets is not the best but still promising.

Table 15: Retrieval Results on two datasets using the fused descriptors.

Descriptors	Matching Algorithm	ETHZ CS Dataset		MPEG7 CS Dataset
		Precision	Recall	
Eccentricity ($\boldsymbol{f}_3$)	D-Value	55.1	90.1	22.7
Point Triangle ($\boldsymbol{f}_{19}$)	DP	50.5	94.4	69.7
Fused	DP + D-Value	56.1	97.6	74.6

To obtain a state-of-the-art performance for open curve matching on a real-world dataset, multiple CS descriptors should be chosen and fused [58]. As discussed in [6], even a combination of simple shape descriptors improves the overall performance of individual descriptors. According to the matching performance on ETHZ and MPEG7 CS datasets, we fuse Eccentricity (f_3) that is the best simple CS descriptor on both datasets, with other signature-based and rich CS descriptors. To compute proper fusing weights, we first divide the dataset into two equal parts, one used for weight estimation and the other for testing. For fusion weight estimation, we employ a supervised optimisation scheme [59] in which two heuristic approaches are combined. We experimentally assess the matching performance by fusing the Point Triangle ($\boldsymbol{f}_{19}$) and Eccentricity ($\boldsymbol{f}_3$) descriptors. Our experiments shows that, comparing to Point Triangle ($\boldsymbol{f}_{19}$), the fused descriptor improves the matching accuracy by 4.4% on the ETHZ CS dataset and by 4.9% on the MPEG7 CS dataset (Table 15). Therefore, a proper combination of CS descriptors can improve the matching accuracy over the individual descriptors.

6. Conclusion and Future Work

In this paper, we made a comprehensive comparison of 27 CS descriptors by correlating them with distance and matching methods. We also studied and evaluated the invariance properties, matching performance and computation complexity of CS descriptors. In order to cover various configurations for CS matching, the evaluation is carried out with respect to different matching algorithms and CS lengths (see Table 1).

From the theoretical and experimental results, it can be derived that the selection of the CS length and matching algorithm affects the matching performance while matching algorithms have a higher influence. The results further reveal that signature-based and rich descriptors with proper matching algorithms can fulfill different requirements in terms of speed and accuracy for CS matching. The overall recommendations for choosing CS descriptors and their settings are illustrated in Table 16. In addition, a proper

Table 16: Recommended choices of descriptors for different requirements.

Best Accuracy	Promising Accuracy and Less Runtime	Fast Speed and Relatively Promising Accuracy
Point Triangle ($\boldsymbol{f}_{19}$) + DP	Partial Contour ($\boldsymbol{f}_{22}$) + DP	Comcoor ($\boldsymbol{f}_{10}$) + Hellinger
	Chord Distribution ($\boldsymbol{f}_{24}$) + DP	Cendistance ($\boldsymbol{f}_{11}$) + Hellinger

combination of rich and simple CS descriptors can improve the matching accuracy over the individual descriptors without adding too much computational complexity. In the future, we will bring more CS descriptors and matching algorithms into our evaluation.

Acknowledgements: Research activities leading to this work have been supported by the China Scholarship Council (CSC) and the German Research Foundation (DFG) within the Research Training Group 1564 (GRK 1564).

References

[1] B. Hariharan, P. Arbelaez, R. Girshick, J. Malik, Hypercolumns for object segmentation and fine-grained localization, in: IEEE CVPR, 2015, pp. 447–456.

[2] F. Wang, L. Kang, Y. Li, Sketch-based 3d shape retrieval using convolutional neural networks, in: IEEE CVPR, 2015, pp. 1875–1883.

[3] J. Shotton, A. Blake, R. Cipolla, Multiscale categorical object recognition using contour fragments, IEEE Trans. PAMI 30 (7) (2008) 1270–1281.

[4] N. Payet, S. Todorovic, From a set of shapes to object discovery, in: ECCV, 2010, pp. 57–70.

[5] H. Riemenschneider, M. Donoser, H. Bischof, Using partial edge contour matches for efficient object category localization, in: ECCV, 2010, pp. 29–42.

[6] C. Yang, O. Tiebe, P. Pietsch, C. Feinen, U. Kelter, M. Grzegorzek, Shape-based object retrieval by contour segment matching, in: IEEE ICIP, 2014, pp. 2202–2206.

[7] Y. Liu, J. Gall, C. Stoll, Q. Dai, H.-P. Seidel, C. Theobalt, Markerless motion capture of multiple characters using multiview image segmentation, IEEE Trans. PAMI 35 (11) (2013) 2720–2735.

[8] G. Bertasius, J. Shi, L. Torresani, Deepedge: A multi-scale bifurcated deep network for top-down contour detection, in: IEEE CVPR, 2015, pp. 4380–4389.

[9] A. Maheshwari, J.-R. Sack, K. Shahbaz, H. Zarrabi-Zadeh, Improved algorithms for partial curve matching, Algorithmica 69 (3) (2014) 641–657.

[10] M. Baust, L. Demaret, M. Storath, N. Navab, A. Weinmann, Total variation regularization of shape signals, in: IEEE CVPR, 2015, pp. 2075–2083.

[11] H. W. Kuhn, The hungarian method for the assignment problem, Naval research logistics quarterly 2 (1955) 83–97.

[12] R. Bellman, The theory of dynamic programming, Bulletin of the American Mathematical Society 60 (6) (1954) 503–516.

[13] G. Al-Naymat, S. Chawla, J. Taheri, Sparsedtw: A novel approach to speed up dynamic time warping, in: Australasian Data Mining Conference, 2009, pp. 117–127.

[14] J. J. de Mesquita Sa Junior, A. R. Backes, Shape classification using line segment statistics, Information Sciences 305 (2015) 349–356.

[15] G. Yule, M. Kendall, An Introduction to the Theory of Statistic, 14th Edition, 1968.

[16] Y. Rubner, C. Tomasi, L. J. Guibas, The earth mover's distance as a metric for image retrieval, IJCV 40 (2) (2000) 99–121.

[17] R. L. Plackett, Karl pearson and the chi-squared test, International Statistical Review 51 (1) (1983) 5972.

[18] A. Bhattacharyya, On a measure of divergence between two multinomial populations, The Indian Journal of Statistics 7 (4) (1946) 401–406.

[19] T. Ma, L. Latecki, From partial shape matching through local deformation to robust global shape similarity for object detection, in: IEEE CVPR, 2011, pp. 1441–1448.

[20] M. Donoser, H. Riemenschneider, H. Bischof, Efficient partial shape matching of outer contours, in: ACCV, 2010, pp. 281–292.

[21] D. Zhang, G. Lu, Review of shape representation and description techniques, Pattern Recognition 37 (1) (2004) 1–19.

[22] R. J. Yang Mingqiang, Kpalma K. Idiyo, A survey of shape feature extraction techniques, in: Pattern Recognition, 2008, pp. 43–90.

[23] P. Kontschieder, H. Riemenschneider, M. Donoser, H. Bischof, Discriminative learning of contour fragments for object detection, in: BMVC, 2011, pp. 1–12.

[24] M. Peura, J. Iivarinen, Efficiency of simple shape descriptors, in: Aspects of visual form, 1997, pp. 443–451.

[25] I. T. Young, J. E. Walker, J. E. Bowie, An analysis technique for biological shape. i, Information and Control 25 (4) (1974) 357–370.

[26] A. Andrew, Another efficient algorithm for convex hulls in two dimensions, Information Processing Letters 9 (5) (1979) 216–219.

[27] P. J. Van Otterloo, A Contour-oriented Approach to Shape Analysis, Prentice Hall International Ltd., Hertfordshire, UK, 1991.

[28] R. Chellappa, R. Bagdazian, Fourier coding of image boundaries, IEEE Trans. PAMI 6 (1) (1984) 102–105.

[29] K. Arbter, W. E. Snyder, H. Burhardt, G. Hirzinger, Application of affine-invariant fourier descriptors to recognition of 3-d objects, IEEE Trans. PAMI 12 (7) (1990) 640–647.

[30] A. Krzyzak, S. Leung, C. Suen, Reconstruction of two-dimensional patterns from fourier descriptors, Machine Vision and Applications 2 (3) (1989) 123–140.

[31] H. Kauppinen, T. Seppanen, M. Pietikainen, An experimental comparison of autoregressive and fourier-based descriptors in 2d shape classification, IEEE Trans. PAMI 17 (2) (1995) 201–207.

[32] J.-R. Ohm, F. Bunjamin, W. Liebsch, B. Makai, K. Mller, A. Smolic, D. Zier, A set of visual feature descriptors and their combination in a low-level description scheme, Signal Processing: Image Communication 16 (12) (2000) 157–179.

[33] Q. M. Tieng, W. Boles, Recognition of 2d object contours using the wavelet transform zero-crossing representation, IEEE Trans. PAMI 19 (8) (1997) 910–916.

[34] H. S. Yang, S. U. Lee, K. M. Lee, Recognition of 2d object contours using starting-point-independent wavelet coefficient matching, VCIR 9 (2) (1998) 171–181.

[35] N. Alajlan, I. E. Rube, M. S. Kamel, G. Freeman, Shape retrieval using triangle-area representation and dynamic space warping, Pattern Recognition 40 (7) (2007) 1911–1920.

[36] J. W. Harris, H. Stocker, Segment of a circle, in: Handbook of Mathematics and Computational Science, New York: Springer-Verlag, 1998, pp. 92–93.

[37] J. Wang, X. Bai, X. You, W. Liu, L. J. Latecki, Shape matching and classification using height functions, PR Letters 33 (2) (2012) 134–143.

[38] C. Lu, L. Latecki, N. Adluru, X. Yang, H. Ling, Shape guided contour grouping with particle filters, in: IEEE ICCV, 2009, pp. 2288–2295.

[39] J. Thureson, S. Carlsson, Appearance based qualitative image description for object class recognition, in: ECCV, 2004, pp. 518–529.

[40] Q. Zhu, L. Wang, Y. Wu, J. Shi, Contour context selection for object detection: A set-to-set contour matching approach, in: ECCV, 2008, pp. 774–787.

[41] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. PAMI 24 (4) (2002) 509–522.

[42] H. Riemenschneider, M. Donoser, H. Bischof, Using partial edge contour matches for efficient object category localization., in: ECCV, 2010, pp. 29–42.

[43] T. F. Cootes, D. Cooper, C. Taylor, J. Graham, Trainable method of parametric shape description, Image and Vision Computing 10 (5) (1992) 289–294.

[44] T. Ma, L. Latecki, From partial shape matching through local deformation to robust global shape similarity for object detection, in: IEEE CVPR, 2011, pp. 1441–1448.

[45] S. Salvador, P. Chan, Fastdtw: Toward accurate dynamic time warping in linear time and space, in: KDD Workshop, 2004, pp. 70–80.

[46] P. H. Sellers, The theory and computation of evolutionary distances: Pattern recognition, Journal of Algorithms 1 (4) (1980) 359–373.

[47] P. Karczmarek, A. Kiersztyn, W. Pedrycz, P. Rutka, Chain code-based local descriptor for face recognition, in: CORES, 2015, pp. 10–20.

[48] A. M. Bronstein, M. M. Bronstein, A. M. Bruckstein, R. Kimmel, Partial similarity of objects, or how to compare a centaur to a horse, IJCV 84 (2) (2009) 163–183.

[49] L. Chen, R. Feris, M. Turk, Efficient partial shape matching using smith-waterman algorithm, in: CVPR Workshops, 2008, pp. 1–6.

[50] L. J. Latecki, R. Lakamper, T. Eckhardt, Shape descriptors for non-rigid shapes with a single closed contour, in: IEEE CVPR, 2000, pp. 424–429.

[51] V. Ferrari, T. Tuytelaars, L. V. Gool, Object detection by contour segment networks, in: ECCV, 2006, pp. 14–28.

[52] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, M. Alexa, Sketch-based shape retrieval, ACM Trans. Graphics 31 (4) (2012) 1–10.

[53] N. Otsu, A threshold selection method from gray-level histograms, Automatica 11 (285-296) (1975) 23–27.

[54] X. Bai, L. Latecki, W. Liu, Skeleton pruning by contour partitioning with discrete curve evolution, IEEE Trans. PAMI 29 (3) (2007) 449–462.

[55] T. Fawcett, An introduction to roc analysis, Pattern Recognition Letters 27 (8) (2006) 861–874.

[56] S. Homer, A. Selman, Introduction to complexity theory, in: Computability and Complexity Theory, Texts in Computer Science, Springer US, 2011, pp. 75–80.

[57] L. Liu, D. Shell, Assessing optimal assignment under uncertainty: An interval-based algorithm, in: Robotics: Science and Systems, 2010.

[58] K. van de Sande, T. Gevers, C. Snoek, Evaluating color descriptors for object and scene recognition, IEEE Trans. PAMI 32 (9) (2010) 1582–1596.

[59] C. Yang, O. Tiebe, P. Pietsch, C. Feinen, U. Kelter, M. Grzegorzek, Shape-based object retrieval and classification with supervised optimisation, in: ICPRAM, 2015, pp. 204–211.