Machine Vision and Applications (2017) 28:373–391
DOI 10.1007/s00138-017-0823-9

ORIGINAL PAPER

Evaluating contour segment descriptors

Cong Yang[1] · Oliver Tiebe[1] · Kimiaki Shirahama[1] · Ewa Łukasik[2] · Marcin Grzegorzek[1,3]

Received: 22 April 2016 / Revised: 24 November 2016 / Accepted: 4 January 2017 / Published online: 3 February 2017

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. 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 both CS lengths and matching algorithms affect the CS matching performance while matching algorithms have higher 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 individuals.

Keywords Contour segment · Contour segment descriptor · Open curve matching · Object retrieval

✉ Cong Yang
cong.yang@uni-siegen.de

Marcin Grzegorzek
marcin.grzegorzek@uni-siegen.de

[1] Research Group for Pattern Recognition, Institute for Vision and Graphics, University of Siegen, Hoelderlinstr. 3, 57076 Siegen, Germany

[2] Laboratory of Operational Research and Artificial Intelligence, Institute of Computing Science, Poznan University of Technology, Pl. Marii Skłodowskiej-Curie, 60-965 Poznan, Poland

[3] Faculty of Informatics and Communication, University of Economics in Katowice, Bogucicka 3, 40-226 Katowice, 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 Fig. 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 [24] and manual operations [55]. Thus, viewing CSs as local patches with any length provides more flexibility against boundary instabilities. Moreover, psychophysical studies [49] show that we can recognise objects using CSs alone. Therefore, CS is an important element for computer vision tasks [42,49].

Benefit from these properties, CS plays a key role in many different applications including object detection [45] and matching [58]. This is because an object shape cannot be ideally segmented from an image due to the background clutter [24] and object overlapping (occlusion) [35]. To overcome these, one typical method is to represent the object boundaries and background using CSs [9]. Moreover, CS is also commonly used for the application of sketch-based object retrieval [55] 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 [39,58]. Similar to CS, open curve is a fragment

Fig. 1 Shape boundary and a contour segment (the *red line*). **a** Shape, **b** shape boundary, **c** contour segment (colour figure online)

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 [6]. 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 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 point-wise matching algorithms like Hungarian [31], dynamic programming (DP) [7] and dynamic time warping (DTW) [1] are normally employed for searching the correspondences between sample points. For some CS descriptors [19,58], due to the structure of their feature vectors, vector-wise methods like correlation [63], χ^2-statistics [44], histogram intersection (HI) [46] and Hellinger [10] are used to calculate distances.

Despite the extensive usage of CS [16,39,45,50,55,58] and its extensions [17–20,38,55,57], most of existing works only introduce CS descriptors without any comparison. In addition, some survey papers focus on shape representation techniques [61,64], or only compare a handful of CS descriptors in specific applications [29]. 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. 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 described above, we draw the recommendation for different application scenarios by balancing the matching accuracy and speed.

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 three 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

As shown in Table 1, 27 CS descriptors are selected for the evaluation in this paper. They are put into three groups, "simple", "signature-based" and "rich", depending on structures of their feature vectors. Roughly speaking, a simple CS descriptor is a scalar representing a global feature of a CS, a signature-based descriptor is a vector characterising geometric relations among CS points, and a rich descriptor is a more complex and structured representation of a CS. The evaluated 27 CS descriptors are mainly selected from two sources: (1) directly designed for CS representation in the literature, including most rich CS descriptors ($\boldsymbol{f}_{19}$–$\boldsymbol{f}_{25}$, $\boldsymbol{f}_{27}$). (2) Most popular and originally designed for shape representation, but can be used or modified into CS description, including simple, signature-based and some rich CS descriptors ($\boldsymbol{f}_1$–$\boldsymbol{f}_{18}$, $\boldsymbol{f}_{26}$).

Before moving to detailed explanations, let us define and formulate a CS. We assume that it has the following restrictions: (1) with one pixel width, (2) with two endpoints, which only have one neighbour point and (3) with no intersection point: except two endpoints, all points only have two neighbour points. Formed on these restrictions, a CS C is represented as a sequence of CS points $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N$ 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

Table 1 Three types of CS descriptors and their symbols (Sym.)

Name	Sym.	Name	Sym.	Name	Sym.
Area [43,61]	f_1	Comcoor [61]	f_{10}	Point Triangle [36]	f_{19}
Circularity [43,61]	f_2	Cendistance [64]	f_{11}	Contour Context [65]	f_{20}
Eccentricity [43]	f_3	Tangent [64]	f_{12}	Beam Angle [42]	f_{21}
Bending [62,64]	f_4	Curvature [64]	f_{13}	Partial Contour [45]	f_{22}
Rectangularity [43]	f_5	Area Function [64]	f_{14}	Opt Partial Contour [29]	f_{23}
LineRatio [43,62]	f_6	Triangle Area [2]	f_{15}	Chord Distribution [20]	f_{24}
Convexity [43]	f_7	Chord Length [25]	f_{16}	Length Direction [37]	f_{25}
Solidity [43]	f_8	Turning Angle [14]	f_{17}	Line Segment [19]	f_{26}
Dislength [58]	f_9	Height Function [56]	f_{18}	Sub-Box [58]	f_{27}

f_1–f_9, simple CS descriptors; f_{10}–f_{17}, signature-based CS descriptors; f_{18}–f_{27}, rich CS descriptors

the same number of points. Then, an open curve matching task is accomplished by multiple CS matching tasks.

2.1 Simple CS descriptors

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 [43] 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 [61,64] and applications [43,62]. In general, these descriptors are intuitive and simple, but 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 Fig. 2b, the area descriptor [43,61] 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. In order to ensure the scale invariance, f_1 is normalised by the length of CS N, that is $f_1 = A_s/N$.

Circularity Circularity [43,61] f_2 illustrates how similar the CS C is to a circle. As shown in Fig. 2c, a circularity 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 [43] 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 the CS C (the red point in Fig. 2d), then eccentricity is calculated as $f_3 = l_1/l_2$ 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 [62,64] 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 $f_4 = \text{mean}(K(i)^2)$, where $K(i)$ denotes the curvature of point $\boldsymbol{p}_i$ (Fig. 2e).

Rectangularity Rectangularity [43] f_5 presents how rectangular a CS is, i.e. how much the CS fills its minimum bounding rectangle (Fig. 2f). Rectangularity is calculated as $f_5 = A_s/(w \cdot h)$ where w and h are the width and height of the CS minimum bounding rectangle.

LineRatio LineRatio [43,62] f_6 uses a straight line as a template and illustrates how similar a CS is to the straight line (Fig. 2g). LineRatio is calculated as $f_6 = h/N$ where h is the height of the CS minimum bounding rectangle.

Convexity Convexity [43] f_7 is defined as the ratio of the convex hull [3] over that of the CS length. Convexity captures the minimal convex covering of a CS. A straightforward measure for Convexity can be calculated as $f_7 = A_h/N$ where A_h denotes the CS convex hull area (Fig. 2h).

Solidity As shown in Fig. 2i, solidity [43] 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 [58] f_9 illustrates the skewness power of a CS (Fig. 2j). It is defined by the ratio between distance of endpoints l_3 and the CS length.

2.2 Signature-based CS descriptors

CS signature is the modified version of shape signature that represents a shape by a vector describing various spatial relations among shape boundary points [64]. Signature-based descriptors can capture the perceptual features of CSs and are often combined with some other feature extraction algorithms like Fourier descriptors [4,13,28,30,54] and wavelet descriptors [40,52,60]. 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. As those descriptors have both local and global features of a CS, both point-wise and vector-wise methods (Sect. 1) can be applied for CS matching and it is unclear which one

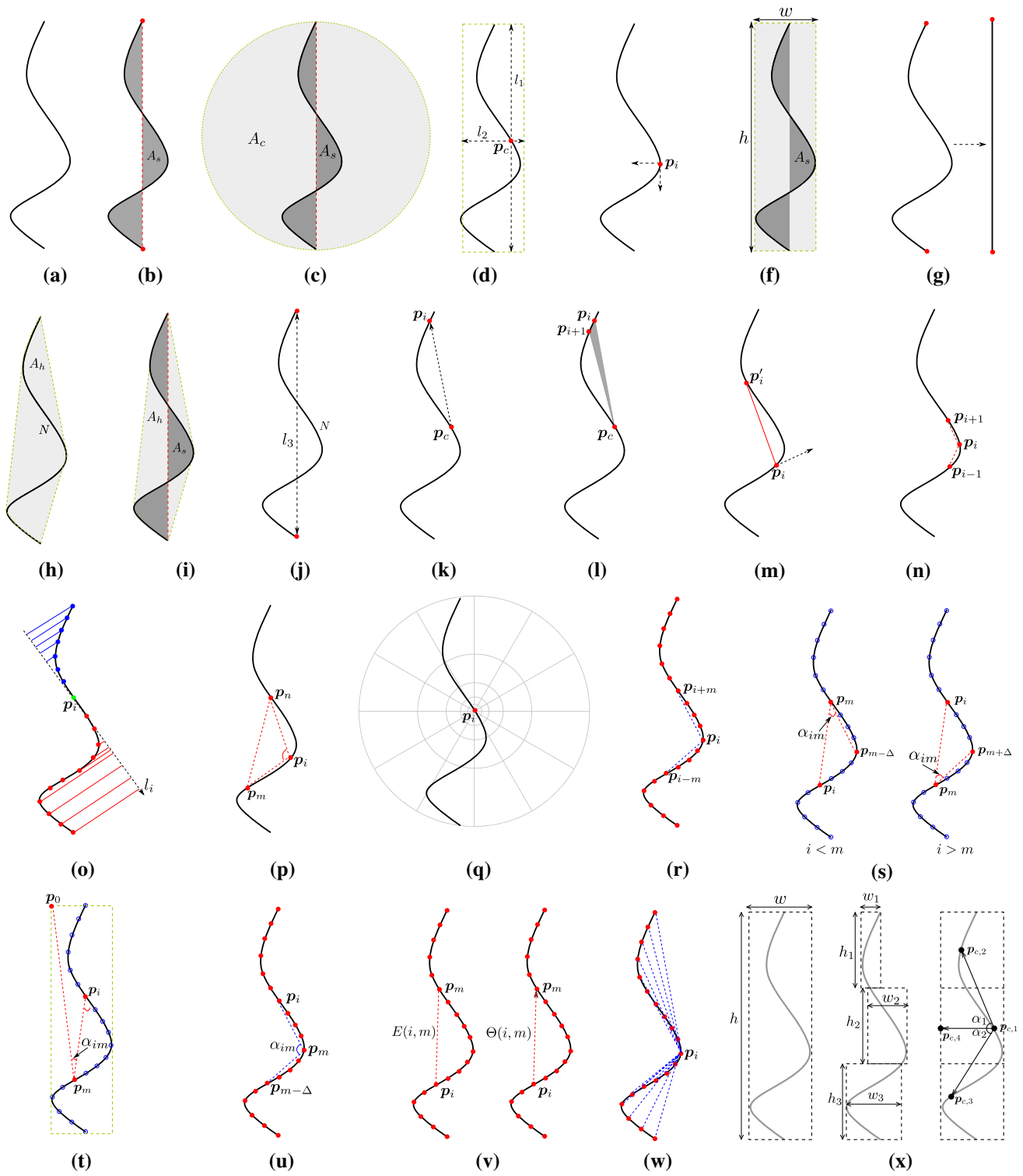

Fig. 2 Simple, signature-based and rich descriptors for a CS C. **a** CS C, **b** Area f_1, **c** Circularity f_2, **d** Eccentricity f_3, **e** Bending f_4, **f** Rectangularity f_5, **g** LineRatio f_6, **h** Convexity f_7, **i** Solidity f_8, **j** DisLength f_9, **k** Cendistance f_{11}, **l** AreaFunction f_{14}, **m** ChordLength f_{16}, **n** Turning Angle f_{17}, **o** Height Function f_{18}, **p** Point Triangle f_{19}, **q** Contour Context f_{20}, **r** Beam Angle f_{21}, **s** Partial Contour f_{22}, **t** Opt Partial Contour f_{23}, **u** Chord Distribution f_{24}, **v** Length Direction f_{25}, **w** Line Segment f_{26}, **x** Sub Box f_{27}

is more efficient. In such a case, we employ both matching types for signature-based CS descriptor evaluation in Sect. 4.

Comcoor Comcoor (Complex Coordinates) [61] descriptor 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 $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N$. Complex coordinates function is:

$$f_{10}^i = [x_i - x_c] + [y_i - y_c]. \tag{1}$$

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

Cendistance Cendistance (Centroid Distance Function) [64] 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 Fig. 2k, $\boldsymbol{f}_{11}$ is defined as $[f_{11}^1, \ldots, f_{11}^i, \ldots, f_{11}^N]$ where f_{11}^i is the angle between $\boldsymbol{p}_c$ and the ith CS point $\boldsymbol{p}_i$.

Tangent Tangent (Tangent Angle Function) [64] descriptor $\boldsymbol{f}_{12}$ is defined as follows:

$$f_{12}^i = \tan^{-1}\left(\frac{y_i - y_{i-w}}{x_i - x_{i-w}}\right), \tag{2}$$

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, $\boldsymbol{f}_{12} = [f_{12}^1, f_{12}^2, \ldots, f_{12}^N]$. 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 [64] descriptor $\boldsymbol{f}_{13}$ is very important for capturing salient perceptual characteristics. Superficially, curvature descriptor is calculated as $\boldsymbol{f}_{13} = [K'(1), \ldots, K'(N)]$ where $K(i)$ is the curvature of the ith 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 [64] 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 $\boldsymbol{p}_{i+1}$ (Fig. 2l). In other words, $\boldsymbol{f}_{14} = [f_{14}^1, \ldots, f_{14}^N]$ 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 the fix point $\boldsymbol{p}_c$.

Triangle Area Different from Area Function, triangle area [2] 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 point $\boldsymbol{p}_i$, the triangle area is formed by:

$$f_{15}^i = \frac{1}{2}\det\begin{pmatrix} x_{i-t_s} & y_{i-t_s} & 1 \\ x_i & y_i & 1 \\ x_{i+t_s} & y_{i+t_s} & 1 \end{pmatrix}. \tag{3}$$

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 [25] descriptor $\boldsymbol{f}_{16}$ is derived by the distance between a CS point and its reference point. As shown in Fig. 2m, 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, $\boldsymbol{f}_{16} = [f_{16}^1, \ldots, f_{16}^N]$ 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. 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 Fig. 2n, a Turning Angle [14] 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: $\boldsymbol{f}_{17} = [f_{17}^1, \ldots, f_{17}^N]$. 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 vector of rich descriptors has more dimensions and varieties. Therefore, rich CS descriptors carry more information of the original CS. However, considering the computational complexity, rich CS descriptors require more runtime for feature generation as the features are normally generated by considering the relationship between every CS sample points. Specifically, rich descriptors $\boldsymbol{f}_{18}$–$\boldsymbol{f}_{25}$ are usually generated based on feature vector F_i on each sample point $\boldsymbol{p}_i$ for a CS. By collecting F_i on every sample point $(F_1, F_2, \ldots, F_N)$, we can get the descriptor for representing the 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, so they are introduced independently.

Height Function As shown in Fig. 2o, a height function [56] F_i is defined based on the distances of the other sample points to its tangent line, $i = 1, \ldots, N$. 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 [36] descriptor $\boldsymbol{f}_{19}$ is inspired by Carlsson [51] 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 Fig. 2p, a Point Triangle descriptor F_i is a histogram of all triangles spanned by $\boldsymbol{p}_i$ and all pairs of points $\boldsymbol{p}_m$, $\boldsymbol{p}_n$. In particular, F_i is the concatenation of triplets each of which represents a triangle and consists of 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 $\boldsymbol{f}_{20}$ [65] 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 [8], in order to capture the geometrical information of $\boldsymbol{p}_i$, a log-polar histogram is defined (Fig. 2q) 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 [42] 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 $(\boldsymbol{p}_i, \boldsymbol{p}_{i-m})$ (Fig. 2r), where $m = 1, \ldots, N'$ and N' is determined experimentally. Therefore, $\boldsymbol{p}_i$ is represented by N' weighted angles.

Partial Contour Partial Contour [45] descriptor $\boldsymbol{f}_{22}$ is calculated by the relative orientations between lines that connect the CS sampled points. As shown in Fig. 2s, for a CS point p_i, its feature vector F_i is formed by multiple angles α_{im}. In particular, each angle is constructed 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 [29] descriptor $\boldsymbol{f}_{23}$ analyses angles defined by lines connecting a reference point (CS-dependent) 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 Fig. 2t). 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 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 [15]. 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 Fig. 2u, 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 Fig. 2v, for a CS point $\boldsymbol{p}_i$, Length Direction descriptor $\boldsymbol{f}_{25}$ [37] 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 [19] descriptor $\boldsymbol{f}_{26}$ is generated based on straight-line segment statistics. As shown in Fig. 2w, for each CS point, the descriptor considers a continuous portion of the CS with the length equal to a pre-defined percentage of the CS size. Then, the length of the straight-line segment between a reference point and another 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 [58] $\boldsymbol{f}_{27}$ preserves the medium-grained CS features by using the full- and sub-bounding boxes. As shown in Fig. 2x, in order to generate sub-box descriptor, a CS bounding box is partitioned into three sub-boxes with the same height ($h_1 = h_2 = h_3$). Then a CS C is represented by an 11-dimensional feature vector including the CS length N, endpoint distance, area, angles and height-to-width ratios of the full and the sub-boxes.

3 Contour segment matching

Here, we introduce the methods for calculating the distance $D(C_1, C_2)$ between two CSs C_1 and C_2. After that, we introduce the open curve matching approach for our experiments. For each CS descriptor, the matching method is assigned based on the CS descriptor types. In particular, as each simple descriptor is actually a single value, distances between them can be directly computed. For a signature-based descriptor, on the one hand, the feature values are computed from each sample point. On the other hand, an N-dimensional feature vector is generated by collecting all feature values. Thus, we can use the point-wise methods with respect to sample points and the vector-wise methods with respect to feature vectors. A rich descriptor consists of "non-uniform" feature vectors in the sense that, each dimension represents a completely different information, that is, it has a significantly different value range. Thus, we cannot directly use the aforementioned matching methods. In such a case, we employ their original distance methods to express the best ability of CS descriptors. After that, three matching approaches are employed for sample point matching.

3.1 CS matching: simple descriptors

Since the simple descriptors $\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_9$ are scalar, we calculate the distance $D(C_1, C_2)$ between two CSs, $D(C_1$ and C_2:

$$D(C_1, C_2) = \frac{|\boldsymbol{f}_{1,j} - \boldsymbol{f}_{2,j}|}{\boldsymbol{f}_{1,j} + \boldsymbol{f}_{2,j}}, \tag{4}$$

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

With the point-wise methods, we first calculate the differences $d(f^m_{1,j}, f^n_{2,j})$ between $f^m_{1,j}$ in $\boldsymbol{f}_{1,j}$ and $f^n_{2,j}$ in $\boldsymbol{f}_{2,j}$ $(m, n = 1, 2, \ldots, N)$. 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 [1], DP [7] and Hungarian [31] on the matrix. Specifically, in order to avoid the brute-force approach [7] of the standard DTW algorithm, we employ the FastDTW [47] for CS point matching. For the DP, we employ the solution proposed by Sellers [48] to reduce the time complexity of traditional approach [7]. Hungarian algorithm [31] 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 $s_1, \ldots, s_N$ and the similarity between C_1 and C_2 is calculated as the mean value.

With the vector-wise methods, 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 [63], (2) histogram intersection (HI) [46], (3) χ^2-statistics [44] and (4) Hellinger (or Bhattacharyya Coefficient) [10]. These distance measures are selected based on their simpleness and applicability evaluated in [27]. Since there are many other existing distance measures, in Sect. 6, we will discuss their adoption as our future work.

3.3 CS matching: rich descriptors

Since rich descriptors $\boldsymbol{f}_{18}, \ldots, \boldsymbol{f}_{27}$ consist of non-uniform feature vectors, they have their own ways 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 [56]. Then, a cost matrix is generated. Here, high weights are put on CS points near to the centre 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 which the distance and orientation matrices are fused to get an affinity matrix that represents the similarities of corresponding point pairs [37]. 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.

It should be noted that for some rich descriptors, e.g. $\boldsymbol{f}_{18}$–$\boldsymbol{f}_{25}$, we can employ other vector-wise methods to calculate the distance between CS points. Taking $\boldsymbol{f}_{19}$ as an

example, instead of the histogram intersection method, the χ^2-statistics [44] method could also be used. However, we will extend and assess this idea as a future work (Sect. 6) for the following three reasons: (1) since each rich descriptor has their correlated distance method in the original paper, there is a high probability that we do not fully express the best ability of those descriptors if we directly employ other methods and ignore the original one. (2) Based on a cost matrix, our target is to find an optimum match between CS points. Using different vector-based methods only change the range of dissimilarity values within the cost matrix. This means that this matrix is still representative and the CS point correspondences mostly remain the same [12]. (3) Theoretically, the overall similarity between rich descriptors is the accumulation of elementary similarities computed by a vector-wise method, so even if some of elementary similarities are not the most accurate one, such errors could be mitigated in the overall similarity [32]. So, we believe that even if a more accurate vector-wise method is used, the overall similarity between rich descriptors is not so affected.

3.4 Open curve matching

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 $D(C_1, C_2)$, $m \leq N_1$, $n \leq N_2$, $N \leq \min(N_1, N_2)$ (Fig. 3a). Therefore, the problem is reduced to search 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 [11, 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 Fig. 3b, we define two quantities: the partiality $\lambda(C^t_1, C^t_2)$ which describes the length of the CS (the higher the value the smaller the CS), and the distance $D(C^t_1, C^t_2)$ which measures the dissimilarity. A Pareto optimum is defined by $\Phi(C^t_1, C^t_2)$ which is a pair of partiality and dissimilarity values that fulfil 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.

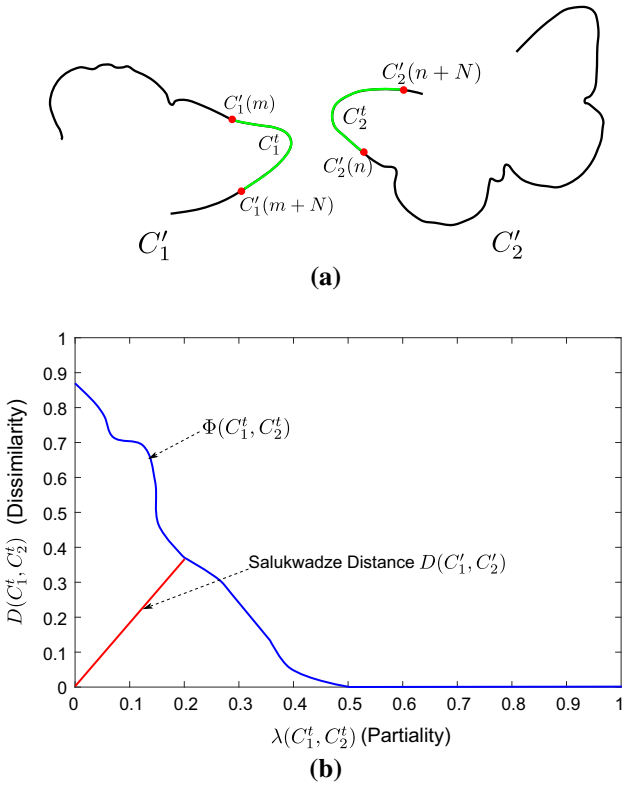

Fig. 3 Open curve matching and the open curve similarity method. **a** Open curve matching using CSs (*green lines*), **b** Salukwadze distance (the *red line*) (colour figure online)

4 Experimental design

This section outlines our experimental design and setup to evaluate CS descriptors. As described in Sect. 1, our evaluation is applied by taking four factors into account: invariant properties, computational complexity (theoretical), matching performance and runtime. With these factors, we can make fair comparisons regarding 27 CS descriptors, their combinations and matching algorithms. In general, this evaluation has the following two advantages: (1) it is possible for us to recommend the best combinations of CS descriptors and matching algorithms for different scenarios. (2) It is easier for us to explore a proper way how to improve the CS matching performance. Both advantages are verified in Sect. 5.4.

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 [14,29,36, 42,45,56,65], researchers propose CS descriptors only for their own specific scenarios. Moreover, we cannot directly employ the existing datasets [19,20,37,58] for CS evaluation since they contain only images or shapes rather than CS.

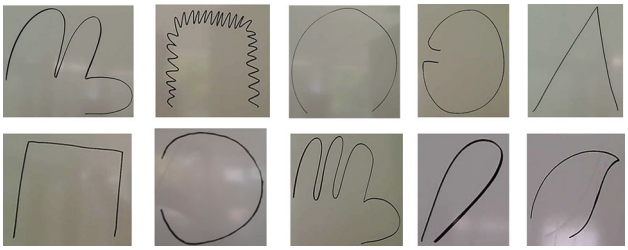

Fig. 4 Sample sketches in the sketching CS dataset

Thus, we have designed and collected three datasets for our experiments.

MPEG7 CS MPEG7 [33] 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, 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 [23] 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.

Sketching CS This database is collected by ourselves including 18 open curves of sketches which are commonly used for the application of sketch-based object retrieval [21]. We first collect images showing sketches drawn on a white board (see Fig. 4). After that, all sketches are segmented and processed into open curves using the methods introduced in [5,41]. 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. Since each CS descriptor and matching algorithm are different and have different inherent difficulty to apply, it is reasonable to evaluate both computational complexity (theoretical and real runtime) and matching performance for the purpose of recommendation. 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

As illustrated in Table 2, 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 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 [33] 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 Sect. 3.4. Specifically, we use each of 271 curves as a query and retrieve the 100 most similar curves from the whole dataset. The final evaluation is based on F1 score which is averaged over curves in each class [22]. For example, since there are 44 open curves in the Apple logo class, the F1 score for this class is calculated by taking the average of F1 scores 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 first collect the ground truth by carrying out an online questionnaire for

Table 2 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

Descriptors	Distance functions		Matching	Lengths
Simple (f_1–f_9)	Difference value		NULL	L1, L2 L3, L4
Signature (f_{10}–f_{17})	Difference value		(1) DTW (2) DP (3) Hungarian	L1, L2 L3, L4
	(1) Correlation, (2) HI, (3) χ^2-statistics, (4) Hellinger		NULL	
Rich (f_{18}–f_{27})	f_{18}–f_{25}	Proposed distances	(1) DTW (2) DP (3) Hungarian	L1, L2 L3, L4
	f_{26}, f_{27}	(1) Correlation, (2) HI, (3) χ^2-statistics, (4) Hellinger	NULL	

selecting the most similar sketches to each 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: computational 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 [26]. The real runtime is evaluated based on the full retrieval time of 1400 queries on the MPEG7 CS dataset.

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 2.2 GHz CPU, 8.00 GB 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 Sect. 4.

Table 3 Invariance properties of different CS descriptors (Des.)

Simple CS				Signature-based CS				Rich CS			
Des.	SI	RI	TI	Des.	SI	RI	TI	Des.	SI	RI	TI
f_1	+	+	+	f_{10}	−	−	−	f_{18}	+	+	+
f_2	+	+	+	f_{11}	−	+	+	f_{19}	+	+	+
f_3	+	+	+	f_{12}	+	+	+	f_{20}	+	−	+
f_4	+	+	+	f_{13}	+	+	+	f_{21}	+	+	+
f_5	+	+	+	f_{14}	−	+*	+	f_{22}	+	+	+
f_6	+	+	+	f_{15}	−	+	+	f_{23}	+	+	+
f_7	+	+	+	f_{16}	+	+	+	f_{24}	+	+	+
f_8	+	+	+	f_{17}	+	+	+	f_{25}	+	+	+
f_9	+	+	+					f_{26}	+	+	+
								f_{27}	+	+	+

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 f_{14}, if the sampling is dense enough, then it is rotation invariant, and vice versa

5.1 Experiment 1: invariance properties

Table 3 illustrates the theoretical invariance properties of different CS descriptors. Firstly, all the CS descriptors are invariant to translation of CSs, except for Comcoor (f_{10}) which is represented by coordinates of CS points (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 (f_{14}) and Triangle Area (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 (f_{20}), all the rich descriptors comply with the three invariance properties. In practice, the rotation property of Contour Context descriptor can be preserved by the preprocessing method introduced in [58].

5.2 Experiment 2: matching performance

In this section, we report the detailed performance of each combination on three datasets.

5.2.1 Evaluation on MPEG7 CS dataset

Table 4 illustrates the CS retrieval results on MPEG7 CS dataset using simple CS descriptors. We can see that Eccentricity (f_3), Bending (f_4) and Rectangularity (f_5) outperform the other descriptors in which Bending (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.

For the signature-based CS descriptors on MPEG7 CS dataset, in Table 5, Comcoor (f_{10}) with Hungarian matching method achieves the best bulls-eye score. In Table 6, both Comcoor (f_{10}) and Cendistance (f_{11}) with Hellinger distance method obtain the best score. Among all results, Comcoor (f_{10}) with Hungarian matching method achieves the best performance (64.4% bulls-eye score). 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 4 CS matching results (%) using Simple CS descriptors (Des.) on MPEG7 CS dataset

Des.	L1	L2	L3	L4
f_1	3.7	3.7	3.7	3.7
f_2	3.7	3.7	3.7	3.7
f_3	**22.3**	**22.8**	**22.8**	**22.8**
f_4	**22.9**	**24.0**	**24.1**	**23.4**
f_5	**22.3**	**22.8**	**22.8**	**22.8**
f_6	18.7	18.9	18.9	18.9
f_7	16.6	15.9	15.7	16.0
f_8	3.7	3.7	3.7	3.7
f_9	18.7	18.9	18.9	18.9

Bold values indicate the results with outstanding matching performance

Table 5 CS matching results (%) with signature CS descriptors (Des.) using point matching methods on MPEG7 CS Dataset

Des.	DTW				DP				Hungarian			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
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**
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
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
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
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
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
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
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

Bold values indicate the results with outstanding matching performance

Table 6 CS matching results (%) with signature CS descriptors (Des.) using vector distance methods on MPEG7 CS dataset

Des.	Correlation				HI				χ^2-statistics				Hellinger			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
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**
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**
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
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
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
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
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
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

Bold values indicate the results with outstanding matching performance

Table 7 CS matching results (%) using rich CS descriptors (Des.) and point matching methods on MPEG7 CS dataset

Des.	DTW				DP				Hungarian			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
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
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
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
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**
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
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
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
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

Bold values indicate the results with outstanding matching performance

Table 8 CS matching results (%) using rich descriptors (Des.) and vector distance methods on MPEG7 CS dataset

Des.	Correlation				HI				χ^2-Statistics				Hellinger			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
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
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

Tables 7 and 8 illustrate the CS retrieval performance using rich CS descriptors. Among all rich descriptors, Beam Angle (f_{21}) achieves promising results in all three matching methods, in which Hungarian method yields the best score (79.6% bulls-eye). For most rich descriptors (f_{18}–f_{26}), they are relatively robust to the CS length and matching algorithm change. However, the performance of Sub-Box (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 an 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. As Sub-Box descriptor only captures the medium-grained features, its matching accuracies are worse than Line Segment (f_{26}) descriptor which preserves 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.

5.2.2 *Evaluation on ETHZ CS dataset*

Table 9 shows matching results (F1 score) of simple CS descriptors. As seen from this table, Eccentricity (f_3) and Rectangularity (f_5) outperform the other descriptors (68.4). Moreover, considering the performance on different lengths, like MPEG7 CS dataset, all simple CS descriptors are robust to CS length change.

Table 9 Open curve matching results (F1 score) using simple CS descriptors (Des.) on ETHZ CS dataset

Des.	L1	L2	L3	L4
f_1	42.5	42.6	42.7	42.6
f_2	54.3	54.6	54.5	54.5
f_3	**66.3**	**66.4**	**66.4**	**66.4**
f_4	60.3	61.1	61.2	61.7
f_5	**66.3**	**66.4**	**66.4**	**66.4**
f_6	65.7	65.7	65.7	65.7
f_7	46.2	46.1	44.7	45.2
f_8	48.2	48.2	48.3	48.3
f_9	65.7	65.7	65.7	65.7

Bold values indicate the results with outstanding matching performance

For the signature-based CS descriptors, as shown in Tables 10 and 11, we can clearly observe that for both point matching and vector distance strategies, matching performances are highly related to the matching methods. For two strategies, point matching methods perform better than the vector distance methods. Specifically, Tangent (f_{12}) with both DTW and DP and Comcoor (f_{10}) with χ^2-statistics achieve the best performance in two strategies, respectively. However, considering their best F1 scores, f_{12} and f_{10} are very close to each other though f_{12} is slightly better.

For the rich CS descriptors, similar to signature-based descriptors, matching algorithms have big influences on F1 scores (Tables 12, 13). Compared to other CS rich descrip-

Table 10 Open curve matching results (F1 score) with signature CS descriptors (Des.) using point matching on ETHZ CS dataset

Des.	DTW				DP				Hungarian			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
f_{10}	65.3	65.5	65.5	65.5	43.5	43.3	43.4	43.3	59.3	61.6	61.6	61.6
f_{11}	40.3	40.3	40.4	40.4	40.1	40.1	40.2	40.3	37.9	37.6	37.8	37.6
f_{12}	**66.1**	**67.5**	**66.8**	**67.4**	**66.1**	**66.5**	**66.2**	**66.4**	65.5	65.2	65.9	66.5
f_{13}	59.8	61.9	58.6	55.5	58.2	56.1	53.9	51.7	58.1	55.3	40.8	50.6
f_{14}	41.1	40.0	41.9	39.4	38.8	38.1	40.1	38.0	40.6	40.9	38.3	40.2
f_{15}	47.8	47.5	47.5	47.1	40.8	40.4	40.4	40.0	43.0	43.1	43.2	43.2
f_{16}	28.2	27.6	25.4	25.5	28.1	27.6	25.5	25.5	28.1	27.5	25.4	25.5
f_{17}	30.2	27.5	29.0	28.3	35.9	33.2	33.5	34.6	40.1	36.6	37.5	38.5

Bold values indicate the results with outstanding matching performance

Table 11 Open curve matching results (F1 score) with signature CS descriptors (Des.) using vector distance on ETHZ CS dataset

Des.	Correlation				HI				χ^2-Statistics				Hellinger			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
f_{10}	61.8	62.5	62.3	62.6	37.0	34.5	35.4	33.7	**67.1**	**67.1**	**66.2**	**66.1**	62.3	62.2	62.1	61.6
f_{11}	7.3	7.6	7.6	7.6	21.1	21.4	21.2	21.3	43.6	43.4	43.5	43.5	62.3	62.3	62.1	61.6
f_{12}	14.6	17.4	16.3	16.2	5.4	6.8	7.0	7.4	63.9	62.3	62.2	62.2	54.9	49.9	49.8	48.7
f_{13}	6.0	6.6	8.5	9.8	57.0	37.5	15.4	13.1	58.8	51.9	42.5	37.9	66.3	68.1	64.4	61.9
f_{14}	5.5	5.8	4.4	6.9	17.4	17.4	17.5	17.8	47.5	47.3	47.0	46.6	61.7	59.8	59.6	59.5
f_{15}	11.3	11.7	11.3	10.6	34.9	34.1	34.4	33.7	50.1	49.1	49.1	48.9	54.6	55.0	55.0	56.1
f_{16}	28.0	27.1	25.4	25.6	24.6	24.6	24.7	24.7	27.6	27.5	25.4	25.5	23.9	24.5	24.8	24.8
f_{17}	43.1	33.5	33.6	38.0	45.3	37.8	39.2	39.4	25.3	20.0	20.5	27.5	43.6	35.1	32.3	32.0

Bold values indicate the results with outstanding matching performance

Table 12 Open curve matching results (F1 score) with rich CS descriptors (Des.) and point matching methods on ETHZ CS dataset

Des.	DTW				DP				Hungarian			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
f_{18}	53.5	52.9	52.4	51.6	60.3	61.0	61.0	60.7	61.2	61.7	61.7	61.7
f_{19}	56.8	54.5	54.3	54.2	**66.6**	**65.7**	**65.4**	**65.3**	66.2	65.4	65.2	65.3
f_{20}	54.4	54.3	54.3	54.3	65.3	65.3	64.9	64.9	64.2	64.2	64.2	64.2
f_{21}	59.4	54.2	49.1	46.4	53.9	48.7	44.8	42.6	57.1	52.6	49.0	46.3
f_{22}	53.5	52.4	52.1	51.6	60.4	60.7	60.6	60.5	58.2	57.0	56.4	56.2
f_{23}	63.3	63.2	63.2	63.3	40.0	37.9	37.3	37.2	40.9	38.7	38.1	37.9
f_{24}	54.0	52.8	52.2	51.8	61.2	61.5	61.6	61.6	60.7	61.0	61.2	61.3
f_{25}	53.0	53.1	53.1	53.0	63.4	63.4	63.4	63.4	52.8	52.8	52.9	52.8

Bold values indicate the results with outstanding matching performance

tors, Point Triangle (f_{19}), Contour Context (f_{20}) and Length Direction (f_{25}) are stable for CS length change and also have good matching performance (more than 60 F1 score). Among all these descriptors, Point Triangle (f_{19}) with dynamic programming achieves the best performance (66.6 F1 score). For Line Segment (f_{26}) and Sub-Box (f_{27}) descriptors (Table 13) which are using the vector distance methods, both of their matching performances are not as good as Point Triangle (f_{19}) with dynamic programming.

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 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) Considering CS descriptors with the best performance in Tables 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, we can observe that the F1 scores 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

Table 13 Open curve matching results (F1 score) with Rich CS descriptors (Des.) and vector distance methods on ETHZ CS dataset

Des.	Correlation				HI				χ^2-statistics				Hellinger			
	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4	L1	L2	L3	L4
f_{26}	51.3	51.0	51.3	51.2	43.2	43.2	43.5	43.2	38.2	38.2	38.3	38.2	56.7	56.4	56.7	56.5
f_{27}	7.2	5.0	4.5	4.6	3.5	3.0	3.0	3.1	63.7	66.1	66.5	66.0	64.5	65.8	66.0	65.8

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. All in all, 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 illustrates the comparison of all accuracy values among three types of CS descriptors. Each mark in this figure represents a mean accuracy value of four different lengths with a combination of CS descriptors and matching algorithms. Note that the horizontal axis represents each value's ID which is used only for visualisation. In order to save space of x-axis, we annotate the ID of accuracy values independently based CS descriptor types. For instance, since there are nine combinations for the simple CS descriptors, the ID is in the range [1, 9]. 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 employed 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%).

Figure 6 illustrates three examples of incorrect matching between CSs. Particularly, In Fig. 6a, we use the simple descriptor Area (f_1) and its correlated matching algorithm

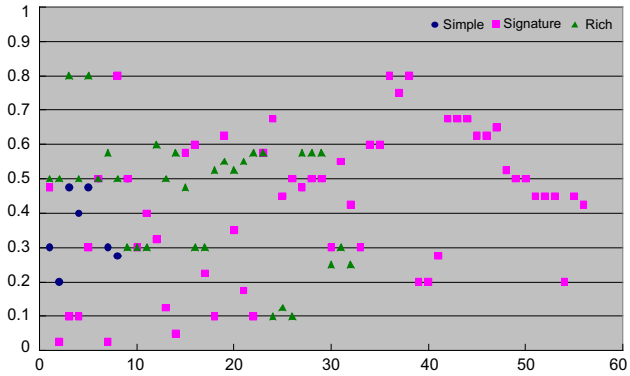

Fig. 5 Comparison of accuracy values between three types of CS descriptors. *Each mark* represents a accuracy value with a different combination of a descriptor and matching algorithm

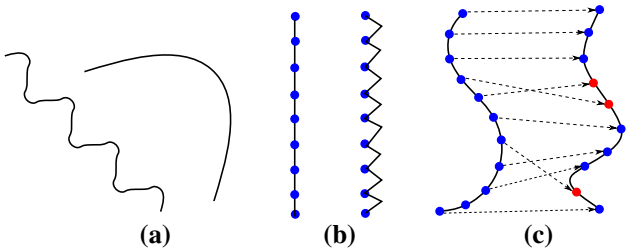

Fig. 6 Examples of incorrect matching. Sample points are marked with the *blue colour*. *Red points* are the incorrect correspondences (colour figure online)

in Sect. 3.1. Figure 6b shows an incorrect matching using the signature-based descriptor Area Function (f_{14}). This phenomenon normally appears within both point-wise and vector-wise matching methods, like signature-based and rich (f_{18}–f_{25}) descriptors. Figure 6c illustrates the incorrect matching (red points) using the rich descriptor Contour Context (f_{20}) and the Hungarian algorithm. With these examples, readers can better visualise the wrong cases of different matching approaches.

5.3 Experiment 3: computational complexity

Table 14 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

Table 14 Theoretical analysis of computational complexity for each CS descriptor in terms of feature generation and matching

Name	Feature generation	Matching (point matching)			Matching (vector distance)
		DTW [1]	DP [7]	Hungarian [31]	
f_1–f_9	$O(N)$	Difference value $O(1)$			
f_{10}–f_{14}, f_{17}	$O(N)$	$O(N^2)$	$O(N^2)$	$O(N^3)$	$O(N)$
f_{15}, f_{16}	$O(N^2)$	$O(N^2)$	$O(N^2)$	$O(N^3)$	$O(N)$
f_{18}–f_{20} f_{22}–f_{25}	$O(N^2)$	$O(N^2)$	$O(N^2)$	$O(N^3)$	NULL
f_{21}	$O(N)$	$O(N^2)$	$O(N^2)$	$O(N^3)$	NULL
f_{26}	$O(N)$	NULL	NULL	NULL	$O(1)$
f_{27}	$O(N^2)$	NULL	NULL	NULL	$O(1)$

NULL means this part is not considered in our experiments

Table 15 Matching performance and runtime (hour) comparison between selected CS descriptors which have outstanding matching performances on ETHZ and MPEG7 CS datasets

Name	Notation	Method	FG	Matching	ETHZ CS[a]	MPEG7 CS[a]
Comcoor	f_{10}	χ^2-Stat.	0.0001	0.0542	66.6	56.2
Comcoor	f_{10}	Hungarian	0.0001	213.74	61.1	64.0
Comcoor	f_{10}	Hellinger	0.0001	0.0531	62.1	63.6
Cendistance	f_{11}	Hellinger	0.0001	0.0462	62.1	63.6
Tangent	f_{12}	DP	0.0001	3.86	67.0	57.8
Point Triangle	f_{19}	DP	6.82	121.8	65.8	69.7
Contour Context	f_{20}	DP	0.04	6.7	65.0	63.7
Beam Angle	f_{21}	Hungarian	0.01	147.1	51.3	77.6
Partial Contour	f_{22}	DP	0.18	5.3	60.5	74.1
Opt Par. Con.	f_{23}	DTW	0.17	22.1	63.3	60.0
Cho. Dis.	f_{24}	DP	0.18	5.4	61.5	74.6
Length Direction	f_{25}	DP	0.02	212.1	63.4	64.3

FG feature generation

[a] We take the mean value of four lengths for comparison

values, their distance can be computed by scalar subtraction. Thus, its computation cost is $O(1)$.

For signature-based descriptors, except Triangle Area (f_{15}) and Chord Length (f_{16}), most descriptors take $O(N)$ complexity for feature generation since each element in a feature vector is calculated only using one CS point. f_{15} and 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 [7]) to $O(N^2)$ with the method introduced in [48]. This is because the method in [48] 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 [34].

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 (f_{21}) is calculated by the angles between CS points and its neighbouring points which can be captured directly. Line Segment (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 is determined by the number of CS points. Therefore, the complexity of f_{21} and f_{26} are $O(N)$. For CS matching, the point matching strategy is applied to descriptors ranging from f_{18} to f_{25}, in which the matching algorithms have the same complexity as signature-based descriptors. For the vector distance strategy, Line Segment (f_{26}) and Sub-Box (f_{27}) have the complexity $O(n)$ where n

Table 16 Retrieval results on two datasets using the fused descriptors

Descriptors	Matching algorithm	ETHZ CS dataset	MPEG7 CS dataset
Eccentricity (f_3)	D-value	68.4	22.7
Point Triangle (f_{19})	DP	65.8	69.7
Fused	DP + D-value	**71.2**	**74.6**

Bold values indicate the results with outstanding matching performance

Table 17 Recommended choices of CS descriptors

Best accuracy	Promising accuracy and less runtime	Fast speed and relatively promising accuracy
Point Triangle (f_{19}) + DP	Partial Contour (f_{22}) + DP	Comcoor (f_{10}) + Hellinger
	Chord Distribution (f_{24}) + DP	Cendistance (f_{11}) + Hellinger

is the feature dimension. Since their feature dimensions are fixed and the distance between two vectors can be calculated in a constant time, their complexity is $O(1)$.

5.4 Runtime comparison and discussion

Table 15 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 (f_{21}) with Hungarian [31] achieves the best bulls-eye score (77.6%). Tangent (f_{12}) and Point Triangle (f_{19}) with DP [7] obtain the best F1 score. However, considering the runtime and retrieval performance, Partial Contour (f_{22}) and Chord Distribution (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 in which the time complexity is not considered as the primary importance, the best choice is Point Triangle (f_{19}) with Dynamic Programming [7] since it is scale, rotation and translation invariant (Table 3) and robust to CS length change (Tables 7, 12). Moreover, it achieves promising results on both ETHZ and MPEG7 CS datasets (Table 15). (ii) Partial Contour (f_{22}) and Chord Distribution (f_{24}) with DP are the best choices to obtain a stable and promising performance while taking less computational time, as shown in Table 15. (iii) If we want to apply a fast open curve matching and obtain a relatively promising results, Comcoor (f_{10}) and Cendistance (f_{11}) with Hellinger [10] 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.

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 [53]. As discussed in [58], even adding simple 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 fusion weights, we first divide the dataset into two parts with the equal size, 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 (f_{19}) and Eccentricity (f_3) descriptors. Our experiments show that, compared to Point Triangle (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 16). Meanwhile, the computational complexity and runtime generally remain the same since Eccentricity (f_3) has low complexity for both feature generation and matching. 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, computational complexity and runtime 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 2). From the theoretical and experimental results, it can be derived that the selection of 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 fulfil different requirements in terms of speed and accuracy for CS matching. The overall recom-

mendations for choosing CS descriptors and their settings are illustrated in Table 17. In addition, a proper 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. Al-Naymat, G., Chawla, S., Taheri, J.: Sparsedtw: a novel approach to speed up dynamic time warping. In: Australasian Data Mining Conference, pp. 117–127 (2009)
2. Alajlan, N., Rube, I.E., Kamel, M.S., Freeman, G.: Shape retrieval using triangle-area representation and dynamic space warping. Pattern Recogn. **40**(7), 1911–1920 (2007)
3. Andrew, A.: Another efficient algorithm for convex hulls in two dimensions. Inf. Process. Lett. **9**(5), 216–219 (1979)
4. Arbter, K., Snyder, W.E., Burhardt, H., Hirzinger, G.: Application of affine-invariant fourier descriptors to recognition of 3-d objects. IEEE Trans. PAMI **12**(7), 640–647 (1990)
5. Bai, X., Latecki, L., Liu, W.: Skeleton pruning by contour partitioning with discrete curve evolution. IEEE Trans. PAMI **29**(3), 449–462 (2007)
6. Baust, M., Demaret, L., Storath, M., Navab, N., Weinmann, A.: Total variation regularization of shape signals. In: IEEE CVPR, pp. 2075–2083 (2015)
7. Bellman, R.: The theory of dynamic programming. Bull. Am. Math. Soc. **60**(6), 503–516 (1954)
8. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Trans. PAMI **24**(4), 509–522 (2002)
9. Bertasius, G., Shi, J., Torresani, L.: Deepedge: a multi-scale bifurcated deep network for top-down contour detection. In: IEEE CVPR, pp. 4380–4389 (2015)
10. Bhattacharyya, A.: On a measure of divergence between two multinomial populations. Indian J. Stat. **7**(4), 401–406 (1946)
11. Bronstein, A.M., Bronstein, M.M., Bruckstein, A.M., Kimmel, R.: Partial similarity of objects, or how to compare a centaur to a horse. IJCV **84**(2), 163–183 (2009)
12. Burkard, R.E., Dell'Amico, M., Martello, S.: Assignment Problems, Revised Reprint, pp. 93–99. SIAM (2009)
13. Chellappa, R., Bagdazian, R.: Fourier coding of image boundaries. IEEE Trans. PAMI **6**(1), 102–105 (1984)
14. Chen, L., Feris, R., Turk, M.: Efficient partial shape matching using smith-waterman algorithm. In: CVPR, pp. 1–6 (2008)
15. Cootes, T.F., Cooper, D., Taylor, C., Graham, J.: Trainable method of parametric shape description. Image Vis. Comput. **10**(5), 289–294 (1992)
16. Daliri, M.R., Torre, V.: Robust symbolic representation for shape recognition and retrieval. Pattern Recogn. **41**(5), 1782–1798 (2008)
17. Daliri, M.R., Torre, V.: Classification of silhouettes using contour fragments. Comput. Vis. Image Underst. **113**(9), 1017–1025 (2009)
18. Daliri, M.R., Torre, V.: Shape recognition based on kernel-edit distance. Comput. Vis. Image Underst. **114**(10), 1097–1103 (2010)
19. de Junior, Mesquita Sa J.J., Backes, A.R.: Shape classification using line segment statistics. Inf. Sci. **305**, 349–356 (2015)
20. Donoser, M., Riemenschneider, H., Bischof, H.: Efficient partial shape matching of outer contours. In: ACCV, pp. 281–292 (2010)
21. Eitz, M., Richter, R., Boubekeur, T., Hildebrand, K., Alexa, M.: Sketch-based shape retrieval. ACM Graph. **31**(4), 1–10 (2012)
22. Fawcett, T.: An introduction to roc analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006)
23. Ferrari, V., Tuytelaars, T., Gool, L.V.: Object detection by contour segment networks. In: ECCV, pp. 14–28 (2006)
24. Hariharan, B., Arbelaez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Mortensen, E., Fidler, S. (eds.) IEEE CVPR, pp. 447–456 (2015)
25. Harris, J.W., Stocker, H.: Segment of a circle. In: Stocker, H. (ed.) Handbook of Mathematics and Computational Science, pp. 92–93. Springer, New York, USA (1998)
26. Homer, S., Selman, A.: Introduction to complexity theory. In: Gries, D., Schneider, FB. (eds.) Computability and Complexity Theory. Texts in Computer Science, pp. 75–80. Springer, New York, USA (2011)
27. Karczmarek, P., Kiersztyn, A., Pedrycz, W., Rutka, P.: Chain code-based local descriptor for face recognition. In: CORES, pp. 10–20 (2015)
28. Kauppinen, H., Seppanen, T., Pietikainen, M.: An experimental comparison of autoregressive and fourier-based descriptors in 2d shape classification. IEEE Trans. PAMI **17**(2), 201–207 (1995)
29. Kontschieder, P., Riemenschneider, H., Donoser, M., Bischof, H.: Discriminative learning of contour fragments for object detection. In: BMVC, pp. 1–12 (2011)
30. Krzyzak, A., Leung, S., Suen, C.: Reconstruction of two-dimensional patterns from fourier descriptors. Mach. Vis. Appl. **2**(3), 123–140 (1989)
31. Kuhn, H.W.: The Hungarian method for the assignment problem. Nav. Res. Logist. Q. **2**, 83–97 (1955)
32. Kurtzberg, J.M.: On approximation methods for the assignment problem. J. ACM **9**(4), 419–439 (1962)
33. Latecki, L.J., Lakamper, R., Eckhardt, T.: Shape descriptors for non-rigid shapes with a single closed contour. In: Werner, B. (ed.) IEEE CVPR, pp. 424–429. IEEE Computer Society, Los Alamitos, CA, USA (2000)
34. Liu, L., Shell, D.: Assessing optimal assignment under uncertainty: an interval-based algorithm. In: Ayanian, N., Kuindersma, S. (eds.) Robotics: Science and Systems. The MIT Press, Cambridge, MA USA (2010)
35. Liu, Y., Gall, J., Stoll, C., Dai, Q., Seidel, H.P., Theobalt, C.: Markerless motion capture of multiple characters using multiview image segmentation. IEEE Trans. PAMI **35**(11), 2720–2735 (2013)
36. Lu, C., Latecki, L., Adluru, N., Yang, X., Ling, H.: Shape guided contour grouping with particle filters. In: IEEE ICCV, pp. 2288–2295 (2009)
37. Ma, T., Latecki, L.: From partial shape matching through local deformation to robust global shape similarity for object detection. In: IEEE CVPR, pp. 1441–1448 (2011)
38. Ma, T., Latecki, L.J.: From partial shape matching through local deformation to robust global shape similarity for object detection. In: IEEE CVPR, pp. 1441–1448 (2011)
39. Maheshwari, A., Sack, J.R., Shahbaz, K., Zarrabi-Zadeh, H.: Improved algorithms for partial curve matching. Algorithmica **69**(3), 641–657 (2014)
40. Ohm, J.R., Bunjamin, F., Liebsch, W., Makai, B., Mller, K., Smolic, A., Zier, D.: A set of visual feature descriptors and their combination in a low-level description scheme. Sig. Process. Image Commun. **16**(12), 157–179 (2000)
41. Otsu, N.: A threshold selection method from gray-level histograms. Automatica **11**(285–296), 23–27 (1975)
42. Payet, N., Todorovic, S.: From a set of shapes to object discovery. In: ECCV, pp. 57–70 (2010)

43. Peura, M., Iivarinen, J.: Efficiency of simple shape descriptors. In: Aspects of visual form, pp. 443–451 (1997)
44. Plackett, R.L.: Karl Pearson and the chi-squared test. Int. Stat. Rev. **51**(1), 5972 (1983)
45. Riemenschneider, H., Donoser, M., Bischof, H.: Using partial edge contour matches for efficient object category localization. In: ECCV, pp. 29–42 (2010)
46. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. IJCV **40**(2), 99–121 (2000)
47. Salvador, S., Chan, P.: Fastdtw: toward accurate dynamic time warping in linear time and space. In: KDD, pp. 70–80 (2004)
48. Sellers, P.H.: The theory and computation of evolutionary distances: pattern recognition. J. Algorithms **1**(4), 359–373 (1980)
49. Shotton, J., Blake, A., Cipolla, R.: Multiscale categorical object recognition using contour fragments. IEEE Trans. PAMI **30**(7), 1270–1281 (2008)
50. Shu, X., Wu, X.J.: A novel contour descriptor for 2d shape matching and its application to image retrieval. Image Vis. Comput. **29**(4), 286–294 (2011)
51. Thureson, J., Carlsson, S.: Appearance based qualitative image description for object class recognition. In: ECCV, pp. 518–529 (2004)
52. Tieng, Q.M., Boles, W.: Recognition of 2d object contours using the wavelet transform zero-crossing representation. IEEE Trans. PAMI **19**(8), 910–916 (1997)
53. van de Sande, K., Gevers, T., Snoek, C.: Evaluating color descriptors for object and scene recognition. IEEE Trans. PAMI **32**(9), 1582–1596 (2010)
54. Van Otterloo, P.J.: A Contour-Oriented Approach to Shape Analysis. Prentice Hall International Ltd., Hertfordshire (1991)
55. Wang, F., Kang, L., Li, Y.: Sketch-based 3d shape retrieval using convolutional neural networks. In: IEEE CVPR, pp. 1875–1883 (2015)
56. Wang, J., Bai, X., You, X., Liu, W., Latecki, L.J.: Shape matching and classification using height functions. PR Lett. **33**(2), 134–143 (2012)
57. Wang, X., Feng, B., Bai, X., Liu, W., Jan Latecki, L.: Bag of contour fragments for robust shape classification. Pattern Recogn. **47**(6), 2116–2125 (2014)
58. Yang, C., Tiebe, O., Pietsch, P., Feinen, C., Kelter, U., Grzegorzek, M.: Shape-based object retrieval by contour segment matching. In: IEEE ICIP, pp. 2202–2206 (2014)
59. Yang, C., Tiebe, O., Pietsch, P., Feinen, C., Kelter, U., Grzegorzek, M.: Shape-based object retrieval and classification with supervised optimisation. In: ICPRAM, pp. 204–211 (2015)
60. Yang, H.S., Lee, S.U., Lee, K.M.: Recognition of 2d object contours using starting-point-independent wavelet coefficient matching. VCIR **9**(2), 171–181 (1998)
61. Yang, M., Kpalma, K., Idiyo, R.J.: A survey of shape feature extraction techniques. In: Pattern Recognition, pp. 43–90 (2008)
62. Young, I.T., Walker, J.E., Bowie, J.E.: An analysis technique for biological shape. I. Inf. Control **25**(4), 357–370 (1974)
63. Yule, G., Kendall, M.: An Introduction to the Theory of Statistic, 14th edn. Griffin, London, UK (1968)
64. Zhang, D., Lu, G.: Review of shape representation and description techniques. Pattern Recogn. **37**(1), 1–19 (2004)
65. Zhu, Q., Wang, L., Wu, Y., Shi, J.: Contour context selection for object detection: a set-to-set contour matching approach. In: ECCV, pp. 774–787 (2008)

Cong Yang is a Postdoc researcher at the MAGRIT team in INRIA (France). His main research interests are computer vision, pattern recognition and their interdisciplinary applications. Cong earned his Ph.D. degree in computer vision and pattern recognition from the University of Siegen (Germany) in 2016. During Ph.D. study, he worked at the Research Group for Pattern Recognition, under the supervision of Prof. Marcin Grzegorzek. From 2014 to 2016, he was also the member of Research Training Group 1564 (GRK 1564), supported by the German Research Foundation (DFG). Before coming to Siegen, Cong received his Bachelor's degree of Software Engineering from Northeast Normal University (NENU) in 2010 and obtained Master's degree of Computer Software and Theory from the joint-supervision program between NENU and Chinese Academy of Sciences in 2012.

Oliver Tiebe started his Bachelor studies of Visual Computing in 2012. Since this time, he is working as a Student Assistant in the Research Group for Pattern Recognition at University of Siegen. Oliver is working in the area of Pattern Recognition and Computer Vision. He is supported by German Research Foundation within the Research Training Group 1564 (GRK 1564).

Kimiaki Shirahama received his B.E., M.E. and Ph.D. degrees in Engineering from Kobe University, Japan in 2003, 2005 and 2011, respectively. After working as an assistant professor in Muroran Institute of Technology, Japan, since 2013, he is working as a postdoctoral researcher at Pattern Recognition Group in University of Siegen, Germany. From 2013 to 2015, his research activity was supported by the Postdoctral Fellowship of Japan Society for the Promotion of Science (JSPS), and is now supported by German Federal Ministry of Education and Research within the project "Cognitive Village: Adaptively Learning Technical Support System for Elderly (Grant Number: 16SV7223K)". His research interests include multimedia data processing, machine learning, data mining and sensor-based human activity recognition. He is a member of ACM SIGKDD, ACM SIGMM, the Institute of Image Information and Television Engineers in Japan (ITE), Information Processing Society of Japan (IPSJ) and the Institute of Electronics, Information and Communication Engineering in Japan (IEICE).

Ewa Łukasik received MSc degree in Control Engineering and PhD degree in Electronics and Telecommunications from Poznan University of Technology (Poland) in 1992. In 1984 she joined the Institute of Control Engineering that through the years evolved to the Institute of Computing Science, where now she holds the position of Senior Lecturer in the Laboratory of Operational Research and Artificial Intelligence. Her main research interests are signal processing, data analysis, multimedia information retrieval and human computer interaction. She is a member of IEEE Signal Processing, Computer and Computational Intelligence Societies, Audio Engineering Society and Polish Information Processing Society. She has been a reviewer of multiple conferences and Journal papers, e.g. in IEEE Audio, Speech and Language Processing, Elsevier Information Systems, Archives of Acoustics and many others.

Marcin Grzegorzek is Professor for Pattern Recognition at the University of Siegen, Professor for Multimedia at the University of Economics in Katowice and Chairman of the Board of a company Data Understanding Lab. He studied Computer Science at the Silesian University of Technology, did his PhD at the Pattern Recognition Lab at the University of Erlangen-Nuremberg, worked scientifically as a Postdoc in the Multimedia and Vision Research Group at the Queen Mary University of London, did his habilitation at the AGH University of Science and Technology in Krakw. He published around 100 papers in pattern recognition, image processing, machine learning, and multimedia analysis. For the time being, he runs five externally funded research projects.