Pattern Recognition 46 (2013) 539–550

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Shape clustering: Common structure discovery

Wei Shen [a], Yan Wang [a], Xiang Bai [a,*], Hongyuan Wang [a], Longin Jan Latecki [b]

[a] Huazhong University of Science and Technology, Wuhan, China
[b] Temple University, Philadelphia, USA

ARTICLE INFO

Article history:
Received 5 January 2012
Received in revised form
8 July 2012
Accepted 31 July 2012
Available online 10 August 2012

Keywords:
Shape
Shape clustering
Skeleton
Common structure
Hierarchical clustering

ABSTRACT

This paper aims to address the problem of shape clustering by discovering the common structure which captures the intrinsic structural information of shapes belonging to the same cluster. It is based on a skeleton graph, named common structure skeleton graph (CSSG), which expresses possible correspondences between nodes of the individual skeletons of the cluster. To construct the CSSG, we derive the correspondences by the optimal subsequence bijection (OSB). To cluster the shape data, we apply an agglomerative clustering scheme, in each iteration, the CSSGs are formed from each cluster and the two closest clusters are merged into one. The proposed agglomerative clustering algorithm has been evaluated on several shape data sets, including three articulated shape data sets, Torsello's data set, and a gesture data set. In all experiments, our method demonstrates effective performance compared to other algorithms.

1. Introduction

Shape clustering, the task of unsupervised grouping of shapes, is a fundamental problem in computer vision and cognitive perception. It is useful in many applications including speeding up the database retrieval and automatical labeling of objects presented in image collections.

Compared to general clustering, there are two main problems in shape clustering: (1) Shape data has a wide range of intra-class variations like deformation and articulation, which lead to the difficulty of designing robust descriptor to represent the shape data. (2) Most of the shape descriptors are not vectors and it is difficult to convert them to vectors, since they are usually represented by graphs, strings, or trees. Consequently, the similarity between shape descriptors cannot be measured by standard metrics, such as Euclidean distance, directly, and the popular clustering algorithm, such as k-means, cannot be directly used for shape data.

To capture the multiple variations of the shape, many effective descriptors have been introduced for shape representation [5,15,10,33,6]. All these shape descriptors are extracted from one single shape, so even the descriptors extracted from the shapes of the same class may be quite different due to the intra-class deformation. In addition, no common intra-class information is considered when computing the pairwise similarities. The performance of shape clustering suffers from these limitations. To solve these problems, we propose a hierarchical shape clustering approach. In statistics, hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. Algorithms for hierarchical clustering are generally either agglomerative, in which one starts at the leaves and successively merges clusters together; or divisive, in which one starts at the root and recursively splits the clusters [12]. The reason for applying the hierarchical clustering framework for the shape data is that it can find successive clusters using previously established clusters and the common intra-class information can be extracted from the current clusters to deliver more robust representation for the next iteration.

In this paper, we choose agglomerative strategy, which initializes each shape as a cluster at first, then iteratively merges two most similar clusters into a single cluster and produces one less cluster at the next iteration until the number of clusters is reduced to a desired number. Each cluster is represented by a common structure abstracted from the skeletons of the shapes in the cluster. Our motivation is that shapes in the same class have common structure encoded into their skeletons as shown in Fig. 1. The common structure is defined as a skeleton graph, called common structure skeleton graph (CSSG), in which each node consists of a set of matching nodes (instances) of the individual skeleton graphs of the same class and each edge consists of a set of edges (instances) in the individual skeleton graphs between the matching nodes. Nodes and edges in the CSSG have their weights related to the number of the instances, which is explicitly defined in Section 4.2. The common structure corresponding to multiple shape instances represents the intrinsic information of the class.

* Corresponding author. Tel.: +86 13297073017.
E-mail addresses: shenwei1231@gmail.com (W. Shen), wyanny.9@gmail.com (Y. Wang), xbai@hust.edu.cn, xiang.bai@gmail.com (X. Bai), hywang@mail.hust.edu.cn (H. Wang), latecki@temple.edu (L. Jan Latecki).

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.07.023

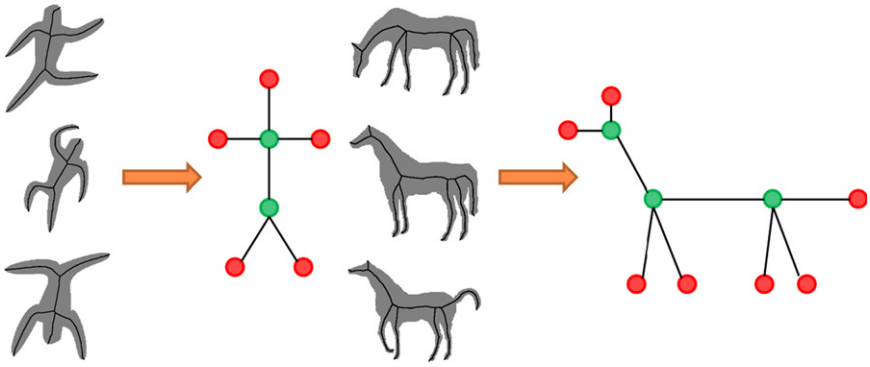

Fig. 1. The common structures of the shapes in the same classes.

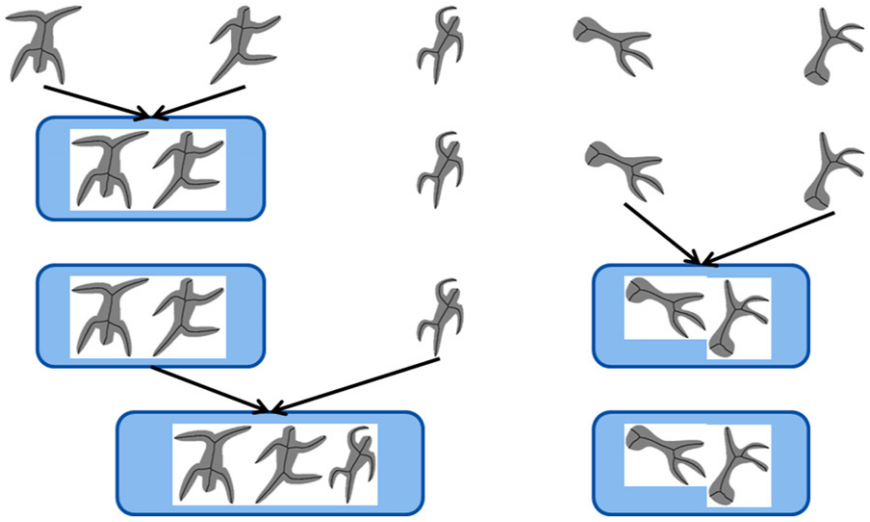

Fig. 2. The process of our hierarchical clustering. First, the five shapes are initialized as five clusters. In every step, two most similar clusters are merged into a new cluster. Finally, the five shapes are clustered into two clusters.

We utilize this information to improve the similarity measure between shapes. The outline of the proposed skeleton based hierarchical clustering is illustrated in Fig. 2. In the clustering process, once the shapes are merged into one cluster, they are taken as a whole and their CSSG is updated for the next iteration. The distance between clusters is calculated by a proposed measure between their CSSGs.

Our contribution can be divided into three aspects: (1) We represent a cluster of shapes by the CSSG and link the CSSG in our agglomerative shape clustering scheme. (2) We propose a distance measure between the skeletons of individual shapes which also can be used to match the nodes. (3) We extend the above-mentioned measure to computing the distance between clusters and locating the correspondence between the nodes of CSSGs.

The rest of this paper is organized as follows. In Section 2, we review the literature on shape clustering. Then we describe how to construct the common structure skeleton graph and propose our agglomerative shape clustering algorithm in Section 4. The experimental results on five shape data sets are presented in Section 5. Finally, we draw the conclusion in Section 6.

2. Related work

In this section, we review the related works, including shape descriptor extraction, skeleton graph matching and shape clustering.

Generally, the existing shape descriptors can be classified into two types: contour based and skeleton based. Shape context [5] is the most popular contour based shape descriptor, which describes the relative spatial distribution (Euclidean distance and orientation) of landmark points around feature points on the contour. Ling et al. [15] use inner distance, which is the length of the shortest path within the shape boundary, to replace the Euclidean distance in shape context. As an extension of shape context, inner distance is articulation insensitive. Felzenszwalb et al. [10] utilize a hierarchical model which represents the contour by the segments composed at multiple levels of resolution. This hierarchical representation can capture the important shape variations.

An alternative to contour based representation is to use a structural abstraction, in the form of a shape skeleton. The skeleton, also known as medial axis, is defined as the set of centers of all maximally inscribed disks (disks that are contained inside the object but not contained in any other such disk) [6]. The skeleton is a very useful shape descriptor [22], and it can better capture articulation of shapes than contour [21]. Since skeletons contain the structural information of shapes, it is natural to organize them into attribute-relation graphs (ARG), which can be used to measure the similarities between skeletons. Shock graph is a kind of ARG proposed by Siddiqi et al. [25–27], which is obtained through specialized Shock Grammar [24]. In shock graph, shocks are the branch points, end-points, and those skeleton segments, which contains both topology and geometry information of the shape [20,19]. Bone graph is an extension of shock graph, which only retains the non-ligature structures of the shock graph and offers improved stability [16]. Matching ARG is an NPC problem, so several algorithms were proposed to obtain

approximate solutions. In [20], Sebastian et al. define the edit distance between shock graphs, however the computational cost is expensive due to the complex edit operations. Methods such as [8,18] could obtain the correspondence between nodes by converting the skeleton graph to a skeleton tree. However, such a conversion requires selecting one node as the root, which needs heuristic rules as in [8,18]. As pointed out in [20], a relatively small change in the shape causes the root to change, leading to a significant change in the topology of the tree representation. In addition, the conversion to a tree may result in loss of important structural information, and consequently, negatively influence the matching result [3]. Baseski et al. [4] use rooted-depth-1 tree expressed from Aslan skeleton [1] to represent shape data structure which benefits from the robustness of the disconnection locations of Aslan skeletons with respect to articulations. The dissimilarity between two skeletons is measured by the tree-edit distance involved in the influence of the category contexts. Han et al. [11] apply an EM algorithm to learn both the structure of the supergraph and the correspondences between the nodes of the sample graphs. They mainly focus on the topological structure and the general graph matching problem. Bai et al. [3] represent each end point by the skeletal shortest paths emanating from it and address skeleton graph matching by matching the sequence of end points.

Our method employs the framework of agglomerative clustering. Agglomerative clustering iteratively merges two closest clusters into a single one, therefore, a distance measure between two clusters (groups of instances), also called linkage criteria, must be defined [12]. The linkage criterion is always a function of the pairwise distances between instances. Several linkage criteria have been designed for determining the distance between groups of instances, such as single linkage, complete linkage [29], and average linkage [28]. However, the shape skeleton data has its own intrinsic properties. Therefore, we will design a special distance measure for the shape data.

In recent years, several methods have been proposed for shape clustering. For example, Lakäemper et al. [13] design a distance measure between a single shape and a group of shapes. Then a soft k-means like framework is applied for shape clustering as the basis of the new distance measure. Srivastava et al. [30] use the geodesic paths constructed between shape boundaries to measure the distance between shapes and perform clustering by using a minimum variance type criterion and a Markov process. In [35], shape is converted to a 1D time series represented by the distance from the centroid of the shape to the contour points. Then a nonlinear projection algorithm is used to group together similar shapes. In [17], the elastic properties of the shape boundaries are encoded in Riemannian metrics and the clustering is applied based on the elastic geodesic distance with DP alignment between shapes.

All the above clustering methods are contour based. Unlike these methods, we represent shapes in terms of their skeletons, which can better capture natural deformations of shapes. Another advantage of our method is that the common structure can be abstracted from skeletons which is useful for clustering. Learning a skeletal shape abstraction from a set of exemplars has been used for shape categorization. Demirci et al. [7] construct the class skeletal prototype based on a many-to-many correspondence between the nodes of skeletons obtained by the method in [8]. However, their prototype is still an exemplar, which is obtained by averaging all the exemplars in the class. Consequently, their prototype may not capture the intra-class structural variation well. Besides, the construction process for their prototype requires converting the skeleton graph to a skeleton tree, which needs heuristic rules as described above. Torsello et al. [32] attempted to find a mixture of tree unions that best accounts for the observed samples using a minimum encoding criterion. Our method is closely related to Torsello's, however, there are two differences. First, Torsello's method concentrated on trees, while we construct graphs for clusters. Second, in Torsello's tree union, only node weight are considered, while in our CSSG, both node weight and edge weight are defined. Erdem and Torsello [9] also proposed a skeleton-based shape clustering method aiming at simultaneously extracting shape classes and learning class-specific shape similarities. While they bias the similarity measure between a shape and a shape cluster by making use of some statistical values of the skeletal attributes in the cluster, we improve the similarity measure between the shape clusters based on the common structures abstracted from them.

3. Basic skeleton concepts

To better describe our approach, we give some basic skeleton concepts first. We follow the definitions of **end point**, **junction point**, **connection point**, **skeleton graph**, **skeleton path** and **path distance** in [3]: The skeleton point with only one adjacent point is an end point; the skeleton point with more than two adjacent points is a junction point. If a skeleton point is not an end point or a junction point, it is called a connection point. (For skeletons in digital images, we assume that the curves of the skeleton are one-pixel wide.) The end/junction points and the sequences of connecting points between two end/junction points form the nodes and the edges of the skeleton graph respectively. An end point and a junction point in a skeleton graph are called end node and junction node respectively. A skeleton path $P(u,v)$ is the shortest path between a pair of nodes u,v in a skeleton graph. Given are two skeleton paths $P(u_x,v_x)$, $P(u_y,v_y)$ represented by two vectors of the radii of their maximal disks centered at the M sample points: $(r_{xi}; i=1,\ldots,M)$ and $(r_{yi}; i=1,\ldots,M)$, respectively. The path distance between $P(u_x,v_x)$ and $P(u_y,v_y)$ is

$$pd(P(u_x,v_x),P(u_y,v_y)) = \sum_{i=1}^{M} \frac{(r_{xi}-r_{yi})^2}{r_{xi}+r_{yi}} + \alpha \frac{(l_x-l_y)^2}{l_x+l_y}, \quad (1)$$

where α is the weight factor and l_x and l_y are the lengths of $P(u_x,v_x)$ and $P(u_y,v_y)$, respectively. Both the radii and the length are normalized to ensure that the path distance is scale invariant.

Besides the above definitions, we also give some others here: The skeleton path between a pair of end nodes is called an end path. The skeleton path between a junction node and an end node is called a junction path. For two junction nodes $j_1 j_2$, if there are no other junction nodes in the skeleton path between them, j_1 and j_2 are connected directly, denoted by $j_1 \frown j_2$.

4. Agglomerative shape clustering

In this section, we introduce our agglomerative shape clustering method. First we propose our distance measure between shapes, then extend it to the distance measure between clusters. The main idea of our approach is to define distance measures between skeletons by extending the notion of the path distance introduced in [3] which allows for the treatment of endpoints first, followed by junction points, building up to the concept of a CSSG for agglomerative clustering.

We utilize junction paths to represent shapes, since junction points contain the structural information of shapes [34], as shown in Fig. 3. However, junction points are not stable [3], they may suffer from ligature-induced instability [16]. For example, in Fig. 4, there are actually five junction points in the first horse's skeleton. We observe that if junction points j_2 and j_3 are merged together as well as j_4 and j_5, then the skeleton graph structure will resemble the structure of the second horse. The instability of

Fig. 3. A junction point and its junction paths.

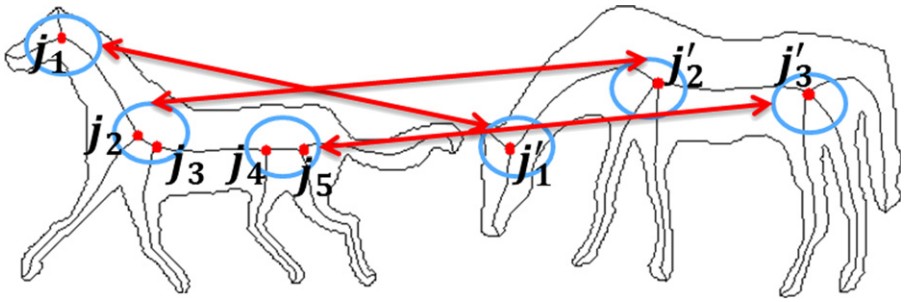

Fig. 4. The junction points on each shape are merged into three clusters (marked by the blue circles) to match well those on the other shape. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

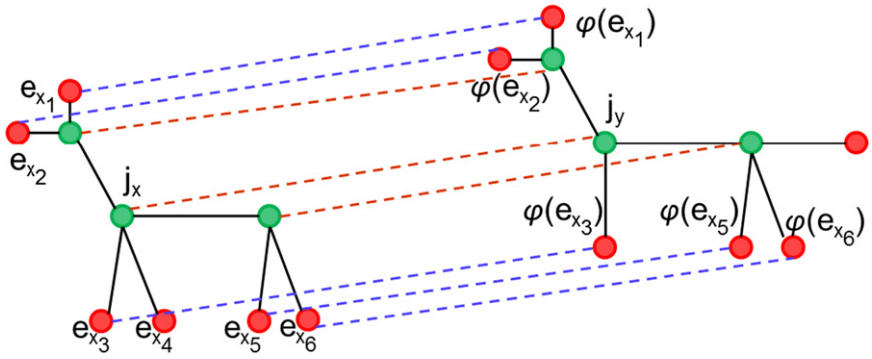

Fig. 5. Computing the distance between junction nodes j_x and j_y based on the matched of end nodes. There are $\kappa=5$ matched pairs of end nodes and $\epsilon=2$ unmatched end nodes.

shape skeletons has been studied in the literature [1,36,16,3,34]. In [16], the instable ligatures are removed which leads to a more stable skeletal representation. Here, we modify the merging strategy in [34] as a preprocessing step for our approach. The details of the preprocessing step are described in the Appendix.

4.1. Pairwise distance between shapes

Our basic idea for a distance measure between shapes is to find the optimal correspondence between end nodes first, then using the optimal correspondence between end nodes to determine the one between junction nodes and the distance between shapes. The reason why we do not match the junction nodes antecedent to matching end nodes is that finding the optimal correspondence between end nodes benefits from the prior that they have the order along the shape contour. Since junction nodes do not have a natural order, they suffer from instability [3], therefore it is much harder to match junction nodes directly.

For two skeletons S_x with nx end nodes and mx junction nodes and S_y with ny end nodes and my junction nodes, their end nodes are $E_x=\{e_{x_1},\ldots,e_{x_{nx}}\}$ and $E_y=\{e_{y_1},\ldots,e_{y_{ny}}\}$, respectively. The goal is to obtain an optimal correspondence $\varphi:\{e_{x_1},\ldots,e_{x_{nx}}\}\rightarrow\{e_{y_1},\ldots,e_{y_{ny}},\phi\}$, where $e_x\in E_x$ is mapped to $\varphi(e_x)\in E_y$, and we allow a many-to-one mapping to ϕ. This is a sequence matching problem, so we compute all distances between the end nodes in E_x and those in E_y and obtain a $nx\times ny$ matrix $\Phi(E_x,E_y)$:

$$\Phi(E_x,E_y)=\begin{pmatrix} ed(e_{x_1},e_{y_1}) & ed(e_{x_1},e_{y_2}) & \ldots & ed(e_{x_1},e_{y_{ny}}) \\ ed(e_{x_2},e_{y_1}) & ed(e_{x_2},e_{y_2}) & \ldots & ed(e_{x_2},e_{y_{ny}}) \\ \vdots & \vdots & \ddots & \vdots \\ ed(e_{x_{nx}},e_{y_1}) & ed(e_{x_{nx}},e_{y_2}) & \ldots & ed(e_{x_{nx}},e_{y_{ny}}) \end{pmatrix}, \quad (2)$$

where $ed(\cdot,\cdot)$ is the distance between two end nodes computed by applying Optimal Subsequence Bijection (OSB) [14] to the matrix whose entry is the path distance between any two end paths emanating from the two end nodes, see [3] for more details.

In [3], the Hungarian algorithm is applied to $\Phi(E_x,E_y)$ to obtained the optimal correspondence between the end nodes in E_x and those in E_y. Here, we make a small change, since the order of the sequence is not considered in the Hungarian algorithm, we apply OSB to the matrix $\Phi(E_x,E_y)$ to obtain the optimal correspondence, which ensures the consistency of the orders of the pair of end nodes along the contours.

Now we introduce an optimal correspondence between junction nodes and a pairwise distance measure between shapes. For two skeletons S_x and S_y, their junction nodes are $J_x=\{j_{x_1},\ldots,j_{x_{mx}}\}$ and $J_y=\{j_{y_1},\ldots,j_{y_{my}}\}$, respectively. The optimal correspondence φ between end points has been obtained by the approach described above. Assuming that κ pairs of end nodes are matched and ϵ end nodes are not matched, as illustrated in Fig. 5. The distance between

two junction nodes $j_x \in J_x$ and $j_y \in J_y$ is

$$jd(j_x,j_y) = \frac{1}{\kappa}\left(\sum_{e \in E_m} pd(P(j_x,e),P(j_y,\varphi(e))) + \epsilon\zeta\right), \quad (3)$$

where $E_m \subseteq E_x$ is the matched end nodes set ($E_m = \{e_{x_1},e_{x_2},e_{x_3},e_{x_5},e_{x_6}\}$ in Fig. 5) and ζ is the penalty factor for the unmatched end nodes which is defined as

$$\zeta = \text{mean}_{j_x \in J_x, e_x \in E_x}\left(\min_{j_y \in J_y, e_y \in E_y} pd(P(j_x,e_x),P(j_y,e_y))\right) + \text{std}_{j_x \in J_x, e_x \in E_x}\left(\min_{j_y \in J_y, e_y \in E_y} pd(P(j_x,e_x),P(j_y,e_y))\right). \quad (4)$$

We define the penalty factor as the mean plus one standard deviation (std) of the distances between the closest skeleton paths is to ensure that the distance between the outlier and its closest skeleton path would be larger than the penalty factor, so that the outliers would be excluded from the matching with a relatively small penalty. This so defined penalty factor is inspired by the definition of jump cost in the sequence matching algorithm proposed by Latecki et al. [14]. Note that, ζ defined in Eq. (4) is not symmetric with respect to S_x and S_y, so we switch the roles of S_x and S_y, compute the value of ζ twice. Then the final penalty factor ζ is the average of the two values.

Then we compute all distances between the junction nodes in J_x and those in J_y and obtain a $mx \times my$ matrix $\Phi(J_xJ_y)$:

$$\Phi(J_x,J_y) = \begin{pmatrix} jd(j_{x_1},j_{y_1}) & jd(j_{x_1},j_{y_2}) & \cdots & jd(j_{x_1},j_{y_{ny}}) \\ jd(j_{x_2},j_{y_1}) & jd(j_{x_2},j_{y_2}) & \cdots & jd(j_{x_2},j_{y_{ny}}) \\ \vdots & \vdots & \ddots & \vdots \\ jd(j_{x_{nx}},j_{y_1}) & jd(j_{x_{nx}},j_{y_2}) & \cdots & jd(j_{x_{nx}},j_{y_{ny}}) \end{pmatrix}. \quad (5)$$

Finally, we apply the Hungarian algorithm to the matrix $\Phi(J_xJ_y)$ to obtain the optimal correspondence $\psi : \{j_{x_1},\ldots,e_{j_{mx}}\} \to \{j_{y_1},\ldots j_{y_{my}},\phi\}$ between the junction nodes and the distance between the pair of shapes S_x,S_y, since junction nodes do not have any natural order:

$$jpd(S_x,S_y) = \mathcal{H}(\Phi(J_x,J_y)), \quad (6)$$

where $\mathcal{H}(\cdot)$ is the Hungarian function. We refer to this distance measure as JPD, for junction path distance.

4.2. Common structure skeleton graph construction

The CSSG is constructed from a cluster of shapes. As shown in Fig. 6, each node in the CSSG consists of a set of matching nodes (instances) of the skeletons in this cluster. The number of the instances that belong to the node in the CSSG gives us a significant information: A node that consists of few instances is more likely an outlier, while a node which consists of many instances is an important node. Thus, we can assign different weights to different nodes according to the number of their instances. As the numbers shown in Fig. 6, for a node υ in the CSSG constructed by λ shapes, its weight is $w_N(\upsilon) = |\upsilon|/\lambda$, where $|\cdot|$ is the cardinality of a set.

As for the edge weight in the CSSG, it is computed by simply counting the edges in the original skeleton graphs between nodes that belong to two nodes in the CSSG. For an edge $P(\upsilon_x, \upsilon_y)$ between two nodes υ_x and υ_y in the CSSG, it consists of a set of edges that belong to the individual skeletons, whose element is $P(v_{x_i},v_{y_i})$, where $v_{x_i} \in \upsilon_x, v_{y_i} \in \upsilon_y$ and the same subscript of "v_x" and "v_y" means the nodes are in the same skeleton. The weight of edge $P(\upsilon_x, \upsilon_y)$ is simply its cardinality: $W_E(P(\upsilon_x, \upsilon_y)) = |P(\upsilon_x, \upsilon_y)|$. Fig. 6 shows an example of computing the edge weight: The edge $P(uj_x,ue_x)$ between two nodes $uj_x = \{j_{x_2},j_{x_3}\}$ and $ue_x = \{e_{x_1},e_{x_2},e_{x_3}\}$, then $P(uj_x,ue_x) = \{P(j_{x_2},e_{x_2}), P(j_{x_3},e_{x_3})\}$, thus, the weight $w_E(P(uj_x,ue_x)) = 2$. Similarly, the edge $P(uj_y,ue_x) = \{P(j_{y_1},e_{x_1}),P(j_{y_2},e_{x_2}),P(j_{y_3},e_{x_3})\}$ and its weight $w_E(P(uj_y,ue_x)) = 3$.

4.3. Distance measure between common structure skeleton graphs

Now we extend the junction path distance to measure the distance between CSSGs. The CSSG is also a skeleton graph, therefore, as computing junction path distance, we obtain the optimal correspondence between end nodes of the CSSGs first, then based on it, determine the optimal correspondence between junction nodes and the distance between CSSGs.

Each node in the CSSG consists of a set of matched nodes of the skeletons in the cluster. Therefore, in order to measure the distance between CSSGs, we need to measure the distance between sets of corresponding nodes. For two CSSGs CS_x with nx end nodes and mx junction nodes and CS_y with ny end nodes and my junction nodes, their end nodes are $UE_x = \{ue_{x_1},\ldots,ue_{x_{nx}}\}$ and $UE_y = \{ue_{y_1},\ldots,ue_{y_{ny}}\}$, respectively, and their junction nodes are $UJ_x = \{uj_{x_1},\ldots,uj_{x_{mx}}\}$ and $UJ_y = \{uj_{y_1},\ldots,uj_{y_{my}}\}$, respectively. Consider two end nodes $ue_x = \{e_{x_1},\ldots,e_{x_{lx}}\} \subset UE_x$ and $ue_y = \{e_{y_1},\ldots,e_{y_{ly}}\} \subset UE_y$, where lx and ly are the cardinalities of the sets. Intuitively, for two CSSGs formed by the shapes of the same class, an important node in the one is less likely to match an outlier in the other. Therefore, we define the distance between ue_x and ue_y as

$$ued(ue_x,ue_y) = \frac{\sum_i\sum_k ed(e_{x_i},e_{y_k})}{lx \cdot ly}\left(1 + \frac{\|w_N(ue_x) - w_N(ue_y)\|}{w_N(ue_x) + w_N(ue_y)}\right), \quad (7)$$

where $\|\cdot\|$ denotes the absolute value. Note that, by introducing the penalty factor $(\|w_N(ue_x) - w_N(ue_y)\|)/(w_N(ue_x) + w_N(ue_y))$, two important nodes are more likely to match each other and an outlier is less likely to match an important node. If both ue_x and ue_y are important nodes, then both $w_N(ue_x)$ and $w_N(ue_y)$ are large, thus $\|w_N(ue_x) - w_N(ue_y)\|$ and $w_N(ue_x) + w_N(ue_y)$ are small and large respectively. Consequently, the penalty factor $(\|w_N(ue_x) - w_N(ue_y)\|)/(w_N(ue_x) + w_N(ue_y))$ is very small. While, without loss of generality, if ue_x and ue_y are an important node and an outlier respectively, then

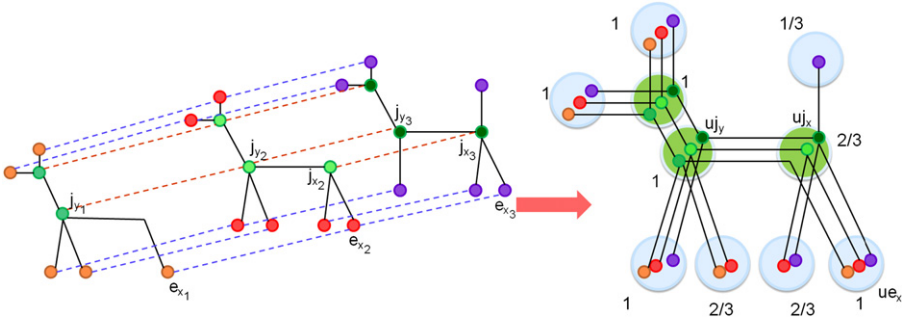

Fig. 6. Common structure skeleton graph construction. Colors indicate the correspondences between the nodes in the CSSG and those in the individual skeletons. The instances belong to ue_x , uj_x and uj_y are $\{e_{x_1},e_{x_2},e_{x_3}\}$, $\{j_{x_2}j_{x_3}\}$ and $\{j_{y_1}j_{y_2}j_{y_3}\}$, respectively. Hence, the junction path $P(uj_x,ue_x) = \{P(j_{x_2},e_{x_2}),P(j_{x_3},e_{x_3})\}$ and $P(uj_y,ue_x) = \{P(j_{y_1},e_{x_1}),P(j_{y_2},e_{x_2}),P(j_{y_3},e_{x_3})\}$. The numbers next to the nodes in the CSSG are their weights.

$\|w_N(ue_x)-w_N(ue_y)\|$ and $w_N(ue_x)+w_N(ue_y)$ are lager and smaller respectively. Consequently, the penalty factor $(\|w_N(ue_x)-w_N(ue_y)\|)/(w_N(ue_x)+w_N(ue_y))$ is much larger than the one computed in the former case. If both ue_x and ue_y are outliers, then both $\|w_N(ue_x)-w_N(ue_y)\|$ and $w_N(ue_x)+w_N(ue_y)$ are small, since there is no hypothesis that the outliers are prone to match each other. Then we compute a $nx \times ny$ matrix of the distances of end nodes as we did in Section 4.1:

$$\Phi(UE_x,UE_y)=\begin{pmatrix} ued(ue_{x_1},ue_{y_1}) & ued(ue_{x_1},ue_{y_2}) & \dots & ued(ue_{x_1},ue_{y_{ny}}) \\ ued(ue_{x_2},ue_{y_1}) & ued(ue_{x_2},ue_{y_2}) & \dots & ued(ue_{x_2},ue_{y_{ny}}) \\ \vdots & \vdots & \ddots & \vdots \\ ued(ue_{x_{nx}},ue_{y_1}) & ued(ue_{x_{nx}},ue_{y_2}) & \dots & ued(ue_{x_{nx}},ue_{y_{ny}}) \end{pmatrix}. \quad (8)$$

Similarly, we apply OSB to the matrix $\Phi(UE_x,UE_y)$ to obtain the optimal correspondence $\varphi:\{ue_{x_1},\dots,ue_{x_{nx}}\}\to\{ue_{y_1},\dots,ue_{y_{ny}},\phi\}$ between end nodes of the two CSSGs.

As for the optimal correspondence between junction nodes and the pairwise distance between CSSGs, we apply the algorithm with similar form as introduced in Section 4.1. Each edge in a CSSG corresponds to multiple skeleton paths, so path distance defined in [3] can be adapted to compute the distance between the edges in CSSGs. For two junction nodes $uj_x=\{j_{x_1},\dots,j_{x_{ix}}\}\subset UJ_x$ and $uj_y=\{j_{y_1},\dots,j_{y_{iy}}\}\subset UJ_y$, where ix and iy are the cardinalities of the sets. For two junction paths $P(uj_x,ue_x)$ and $P(uj_y,ue_y)$, the distance between them is

$$upd(P(uj_x,ue_x),P(uj_y,ue_y))=\frac{\sum_{P(j_{x_i},e_{x_i})\in P(uj_x,ue_x)}\sum_{P(j_{y_k},e_{y_k})\in P(uj_y,ue_y)}pd(P(j_{x_i},e_{x_i}),P(j_{y_k},e_{y_k}))}{w_E(P(uj_x,ue_x))\cdot w_E(P(uj_y,ue_y))}, \quad (9)$$

where the same subscript of "j" and "e" means that the junction node and the end node are in the same skeleton graph.

For the two CSSGs CS_x and CS_y, assuming that κ pairs of end nodes are matched and ϵ end nodes UE_ϵ are skipped (not matched). Skipping an outlier is trivial to the matching, while skipping an important node increases the dissimilarity between the two CSSGs. Therefore, we would like to assign a penalty to each skipped node according to their weights. As shown in Fig. 7, the distance between uj_x and uj_y is

$$ujd(uj_x,uj_y)=\frac{1}{\kappa}\left(\sum_{ue\subset UE_m}upd(P(uj_x,ue),P(uj_y,\varphi(ue)))+\zeta\sum_{ue\subset UE_\epsilon}w_N(ue)\right)\times\left(1+\frac{\|w_N(uj_x)-w_N(uj_y)\|}{w_N(uj_x)+w_N(uj_y)}\right), \quad (10)$$

where $UE_m\subseteq UE_x$ is matched end nodes set ($UE_m=\{ue_{x_1},ue_{x_2},ue_{x_3},ue_{x_5},ue_{x_6}\}$ in Fig. 7) and ζ is the penalty factor for the unmatched end nodes and defined as:

$$\zeta=\mathrm{mean}_{uj_x\in UJ_x,ue_x\in UE_x}\left(\min_{uj_y\in UJ_y,ue_y\in UE_y}upd(P(uj_x,ue_x),P(uj_y,ue_y))\right)$$
$$\mathrm{std}_{uj_x\in UJ_x,ue_x\in UE_x}\left(\min_{uj_y\in UJ_y,ue_y\in UE_y}upd(P(uj_x,ue_x),P(uj_y,ue_y))\right). \quad (11)$$

We also switch the roles of CS_x and CS_y, compute the value of ζ twice, and use the average as the final penalty factor ζ. Notice that Eq. (3) is the special case of Eq. (10) when the common structure only corresponds to a single instance.

Then we compute all distances between the junction nodes in UJ_x and those in UJ_y and obtain an $mx \times my$ matrix $\Phi(UJ_x,UJ_y)$:

$$\Phi(UJ_x,UJ_y)=\begin{pmatrix} ujd(uj_{x_1},uj_{y_1}) & ujd(uj_{x_1},uj_{y_2}) & \cdots & ujd(uj_{x_1},uj_{y_{ny}}) \\ ujd(uj_{x_2},uj_{y_1}) & ujd(uj_{x_2},uj_{y_2}) & \cdots & ujd(uj_{x_2},uj_{y_{ny}}) \\ \vdots & \vdots & \ddots & \vdots \\ ujd(uj_{x_{nx}},uj_{y_1}) & ujd(uj_{x_{nx}},uj_{y_2}) & \cdots & ujd(uj_{x_{nx}},uj_{y_{ny}}) \end{pmatrix}. \quad (12)$$

Finally, we apply the Hungarian algorithm to the matrix $\Phi(UJ_x,UJ_y)$ to obtain the optimal correspondence $\psi:\{uj_{x_1},\dots,uj_{x_{mx}}\}\to\{uj_{y_1},\dots,uj_{y_{my}},\phi\}$ between the junction nodes and the distance between the pair of CSSGs CS_x, CS_y:

$$csd(CS_x,CS_y)=\mathcal{H}(\Phi(UJ_x,UJ_y)). \quad (13)$$

We refer to this distance measure between clusters as CSD, for common structure distance.

4.4. Clustering scheme

Now we demonstrate our clustering scheme. Given are an unlabeled shape set $X=\{x_1,x_2,\dots,x_N\}$ to be clustered into Θ clusters $\mathcal{C}=\{C_1,C_2,\dots,C_\Theta\}$, where N is the number of shapes, and their skeletons are $SS=\{s_1,s_2,\dots,s_N\}$. The agglomerative shape clustering scheme is shown in Fig. 8.

We refer to the proposed shape clustering method as CSD+AHC, for agglomerative hierarchical clustering based on common structure distance.

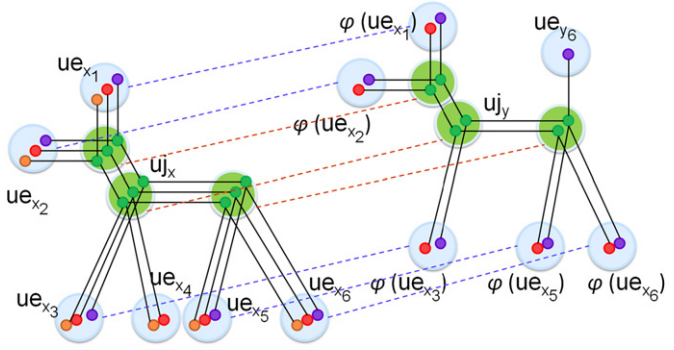

Fig. 7. Computing the distance between junction nodes uj_x and uj_y based on the matched of end nodes. There are $\kappa=5$ matched pairs of end nodes and $\epsilon=2$ unmatched end nodes $UE_\epsilon=\{ue_{x_4},ue_{y_6}\}$.

Procedure Clustering(Input SS, Θ, Output $\mathcal{C}$)
1. Initialize $\mathcal{C}=\{C_1,C_2,\dots,C_N\}$ and $\theta=N$, where $C_i=\{s_i\}_{i=1}^N$.
2. **While** $\theta>\Theta$
3. Let CS_i be the CSSG of $C_i(i=1,2,\dots,\theta)$, compute its node weights and edge weights.
4. Find the closet two common structures $CS_{\hat{i}}$ and $CS_{\hat{k}}$, $(\hat{i},\hat{k}=1,2,\dots,\theta,\hat{i}\neq\hat{k})$.
5. Merge $C_{\hat{i}}$ and $C_{\hat{k}}$ into one: $\hat{C}=C_{\hat{i}}\bigcup C_{\hat{k}}$.
6. Update $\mathcal{C}$: $\mathcal{C}=\mathcal{C}\backslash C_{\hat{i}}$, $\mathcal{C}=\mathcal{C}\backslash C_{\hat{k}}$, $\mathcal{C}=\mathcal{C}\bigcup\hat{C}$, $\theta=\theta-1$.
7. **End**

Fig. 8. The agglomerative shape clustering algorithm.

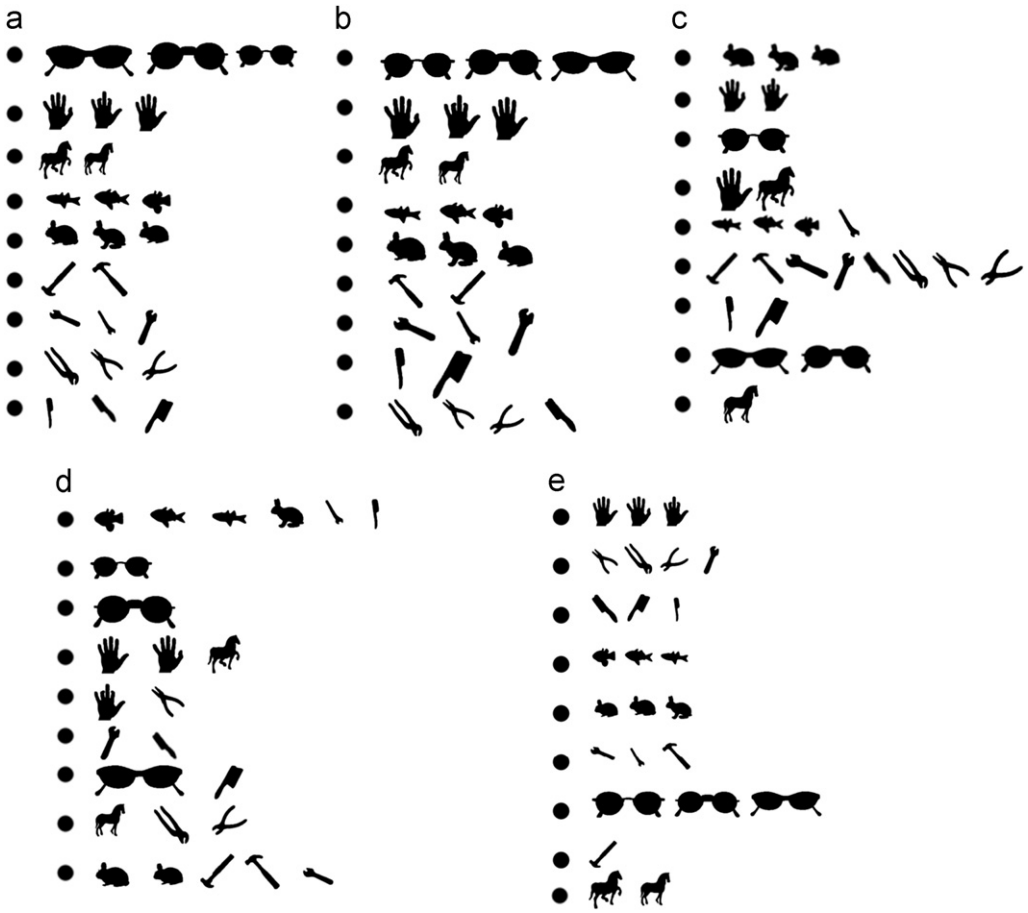

Fig. 9. Comparison between different clustering methods on Torsello's data set [32]. (a) CSD+AHC, (b) JPD+AHC, (c) JPD+Ncuts, (d) IDSC+Ncuts and (e) Union of attributed trees [32].

5. Experimental results

To assess the quality of the proposed clustering method, we evaluate it on five standard data sets: Torsello's data set [32], "50 hands" data set [17], and three Aslan and Tari data sets [2,1,4]. The following clustering methods are considered for comparison in all our experiments: The pairwise distance between shapes by inner distance [15] with normalized cuts [23] used to cluster shapes, denoted by IDSC+Ncuts; Junction path distance with normalized cuts, denoted by JPD+Ncuts; JPD but with normalized cuts replaced by agglomerative hierarchical clustering with average linkage, denoted by JPD+AHC; the proposed CSD+AHC. Our experiments are divided into two parts. We commence by illustrating qualitative examples of the clusters obtained by different clustering methods. Then we give quantitative comparison between the clustering methods. All results are obtained by tuning the desired number of clusters to be equal to the natural number of classes of the data set. To apply Ncuts algorithm, the distances between shapes must be converted to similarities first. This usually can be done by using a Gaussian kernel. Assuming that the distance between two shapes S and S' is $D(S,S')$, then the similarity between them is obtained by $W(S,S') = \exp(-D^2(S,S')/\sigma(2)$. In our experiments, we set $\sigma = 10.0$ empirically.

5.1. Clustering examples

We begin with the illustration of clusters on a small data set provided by Torsello [32] that contains 25 shapes grouped in 9 classes. We set $\alpha = 150$ for this dataset. We compare the proposed CSD+AHC to the union of attributed trees which obtains the best clustering result in [32], IDSC+Ncuts, JPD+AHC and JPD+Ncuts. Fig. 9 shows the result of comparison, we set the number of clusters $\Theta = 9$ for all the clustering methods. As shown in Fig. 9, comparing with JPD+Ncuts, JPD+AHC and IDSC+AHC, CSD+AHC achieves the perfect result which shows the advantage of common structure.

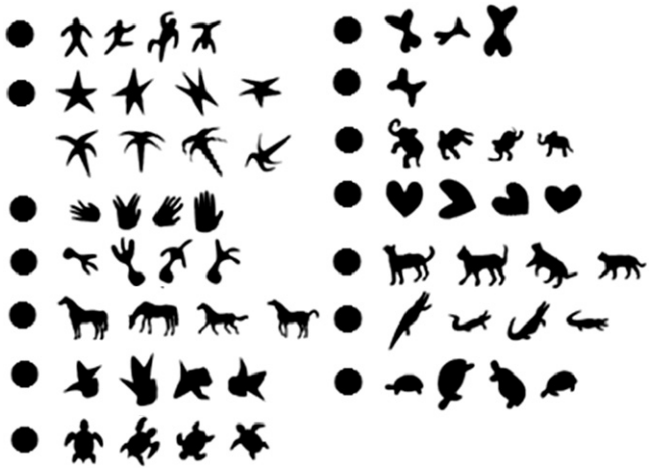

Fig. 10. Clusters of the Aslan and Tari data set with 56 shapes by CSD+AHC.

We also test our method on the Aslan and Tari data set 56 shapes [2], which includes 14 classes of articulated shapes with 4 shapes in each class. We set $\alpha = 100$ for this dataset and set the number of clusters $\Theta = 14$ for all the clustering methods. Fig. 10

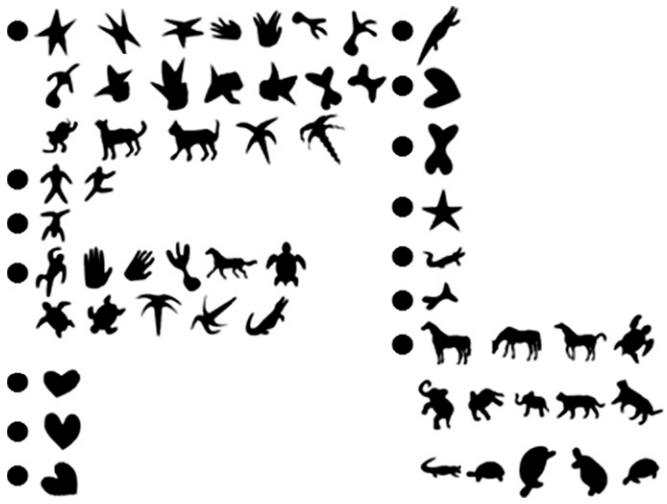

Fig. 11. Clusters of the Aslan and Tari data set with 56 shapes by IDSC+Ncuts.

Fig. 12. Clusters of the Aslan and Tari data set with 56 shapes by JPD+Ncuts.

shows the clustering result obtained by CSD+AHC. There are only two errors: four windmills are clustered with pentagrams and a bone is clustered into a new cluster. However, the windmill is very similar to the pentagrams so that the error seems acceptable. The cluster results obtained by IDSC+Ncuts, JPD+Ncuts, JPD+AHC are shown in Figs. 11–13, respectively. Clearly, CSD+AHC achieves the best result. The results on the Aslan and Tari data set with 56 shapes depict the advantage of JPD for measuring the distance between articulated shapes and how CSD+AHC works in clustering. Fig. 14 illustrates several results of common structures obtained by CSD+AHC. The left column shows the common structure of clusters and the other columns show skeleton graphs from shapes which belong to the cluster. Note that, the end nodes and the junction nodes marked without number are the outliers detected by the common structure, as the one in the skeleton of the fourth turtle. Fig. 15 depicts the hierarchical property of CSD+AHC. We show the clustering results obtained by turning different desired numbers of clusters. Note that, if the clustering is allowed to be continued a bit further (i.e. $\Theta = 13$), cat and horse shapes will be grouped together, since they are two visually similar shape classes.

5.2. Quantitative analysis

To quantitatively analyze the clustering results, we use the normalized mutual information (NMI) as a measure of clustering

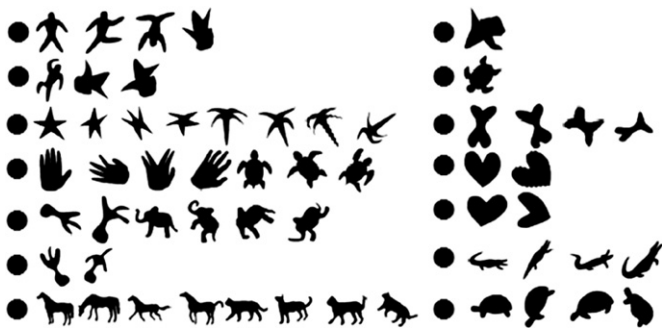

Fig. 13. Clusters of the Aslan and Tari data set with 56 shapes by JPD+AHC.

results. The ground-truth class partition Γ and the returned cluster partition Δ define a confusion matrix with each entry $n_i^{(j)}$ being the number of data samples in cluster i and class j, and n is the total number of samples. Then NMI is computed as follows:

$$\frac{2\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{n_i^{(j)}}{n}\log\frac{n_i^{(j)}n}{\sum_{k=1}^{I}n_k^{(j)}\sum_{k=1}^{J}n_i^{(k)}}}{H(\Gamma)+H(\Delta)}$$

where I is the number of clusters and J is the number of classes. $H(\Gamma) = -\sum_{i=1}^{I}(n_i/n)\log n_i/n$ and $H(\Delta) = -\sum_{j=1}^{J}(n^{(j)}/n)\log n^{(j)}/n$ are the entropies of partition Γ and Δ, respectively. A high value of NMI indicates that Γ and Δ match well.

We list the NMIs of the clustering results on Torsello's data set and the Aslan and Tari data set with 56 shapes in Tables 1 and 2 , respectively.

The "50 hands" data set [17] consists of 50 boundary shapes of hands in different postures. The natural number of clusters in this data set is (depending on subjective interpretation) 8 or 9, with cluster sizes between 2 and 8 elements. The groundtruth clusters of these two cases are illustrated in Fig. 16. We set $\alpha = 100$ for this dataset. Several clustering methods are compared on this data set, including the method in [17], the method in [13], IDSC+Ncuts, JPD+Ncuts, JPD+AHC and our CSD+AHC. As shown in Table 3, whether the desired number of clusters $\Theta = 9$ or $\Theta = 8$, CSD+AHC achieves the perfect result.

We also test cluster methods on a larger data set, the Aslan and Tari data set with 180 shapes, which includes 30 classes of articulated shapes with 6 shapes in each class. We set $\alpha = 100$ for this data set and set the number of clusters $\Theta = 30$ for all the clustering methods. Table 4 shows the clustering results obtained by IDSC+Ncuts, JPD+Ncuts, JPD+AHC and CSD+AHC. Our result is much better than the results of other methods.

Finally, we test cluster methods on a much larger data set, the Aslan and Tari data set with 1000 shapes, which includes 50 classes of articulated shapes with 20 shapes in each class. We set $\alpha = 100$ for this data set and set the number of clusters $\Theta = 50$ for all the clustering methods. Table 5 shows the clustering results obtained by IDSC+Ncuts, JPD+Ncuts, JPD+AHC, CSD+AHC and Foreground Focus [37]. The result of Foreground Focus is quoted from [9]. Other results in [9] are not obtained by tuning the desired number of clusters to be equal to the natural number of classes of the data set, so we do not quote them here. In Table 5, our result is still the best, which illustrates the effectiveness of the common structure on large data set.

To assess the ability of the proposed clustering algorithm to classify the shape classes, we perform experiments on an increasing number of shapes in the two Aslan and Tari data sets. As for the Aslan and Tari data set with 56 shapes, we commence with the 8 shapes from the first 2 classes and then increase the number of shape classes under consideration until the full set of 56 shapes

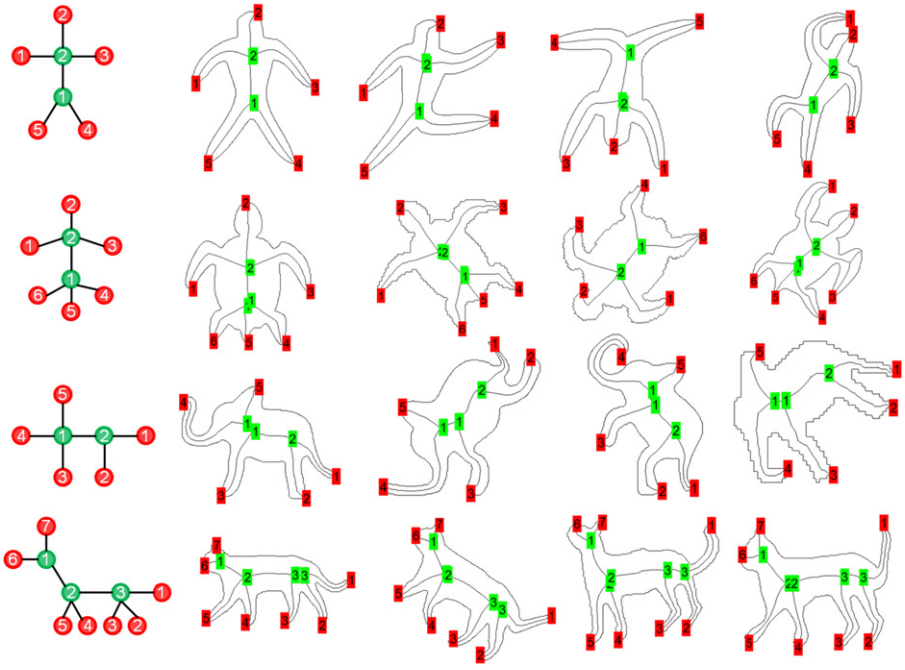

Fig. 14. The common structure of several shape classes in the Aslan and Tari data set 56 shapes [2]. The end nodes (in red) and junction nodes (in green) are marked with the numbers. The same number indicates the correspondences between nodes in the sample shapes and nodes in the common structure. The junction node and the end node in the skeleton of the fourth turtle marked without numbers have low weight, therefore, we do not show them in the common structure. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

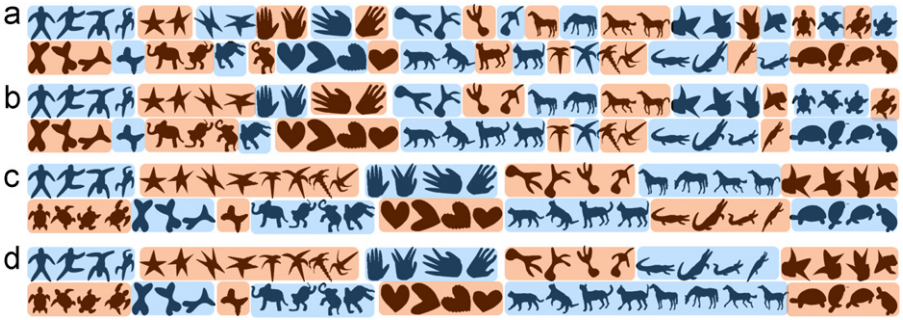

Fig. 15. The hierarchical property of CSD+AHC shown on the Aslan and Tari data set 56 shapes [2]. The shapes in the same rectangular region are grouped into a cluster by CSD+AHC. The rectangular regions are marked by two colors alternately to make them visible. From top to bottom, the desired number of clusters is decreased: (a) $\Theta=36$. (b) $\Theta=24$. (c) $\Theta=14$. (d) $\Theta=13$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

Table 1
The NMIs of the clustering results on Torsello's data set [32].

Method	CSD+AHC	JPD+AHC	JPD+Ncuts	IDSC+Ncuts	[32]
NMI	**1.0000**	0.9618	0.7778	0.6431	0.9313

Table 2
The NMIs of the clustering results on Aslan and Tari data set with 56 shapes [2].

Method	CSD+AHC	JPD+AHC	JPD+Ncuts	IDSC+Ncuts
NMI	**0.9734**	0.8674	0.6174	0.5660

was included. We plot the curves of NMI as the number of shape classes is increased in Fig. 17. Similarly, we plot the curves obtained by all the cluster methods on the Aslan and Tari data set with 180 shapes in Fig. 18. We commence with 30 shapes from the first 5 classes and then increase the number of shape classes until all 180 shapes were included. In both Figs. 17 and 18, CSD+AHC appears as the curve marked with green "square" and clearly outperforms other clustering methods.

5.3. Parameter discussion

There is a parameter introduced in this paper, the threshold T_m for merging junction points. Bigger threshold means that more junction points would be merged together. T_m is a constant in all our experiments: $T_m=5$. There is another parameter introduced in [3], the weight factor α. In Fig. 19, we show that the clustering result is not sensitive to the choice for α.

6. Conclusion

In this paper, we present an agglomerative hierarchical algorithm for shape clustering based on a common structure formed by the shapes belonging to the same clusters. Unlike the general clustering methods whose results only depend on the pairwise similarities, the proposed clustering method extracts the common

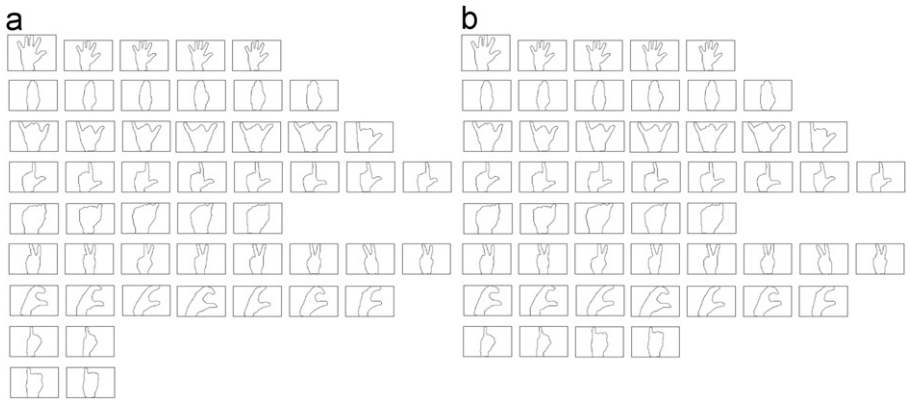

Fig. 16. The groundtruth clusters of data set '50 hands', taken from Mio et al. [17]. The natural numbers of clusters are 9 and 8 in (a) and (b) respectively.

Table 3
The NMIs of the clustering results on "50 hands" data set [17].

Method	CSD+AHC	[17]	[13]	IDSC+Ncuts	JPD+Ncuts	JPD+AHC
NMI ($\Theta=9$)	**1.0000**	1.0000	1.0000	0.6055	0.8403	0.9726
NMI ($\Theta=8$)	**1.0000**	–	1.0000	0.6732	0.8568	0.9331

Table 4
The NMIs of the clustering results on Aslan and Tari data set with 180 shapes [1].

Method	CSD+AHC	IDSC+Ncuts	JPD+Ncuts	JPD+AHC
NMI	**0.9694**	0.5423	0.5785	0.8793

Table 5
The NMIs of the clustering results on Aslan and Tari data set with 1000 shapes [4].

Method	CSD+AHC	IDSC+Ncuts	JPD+Ncuts	JPD+AHC	Foreground Focus [37]
NMI	**0.8096**	0.5433	0.4549	0.7693	0.7329

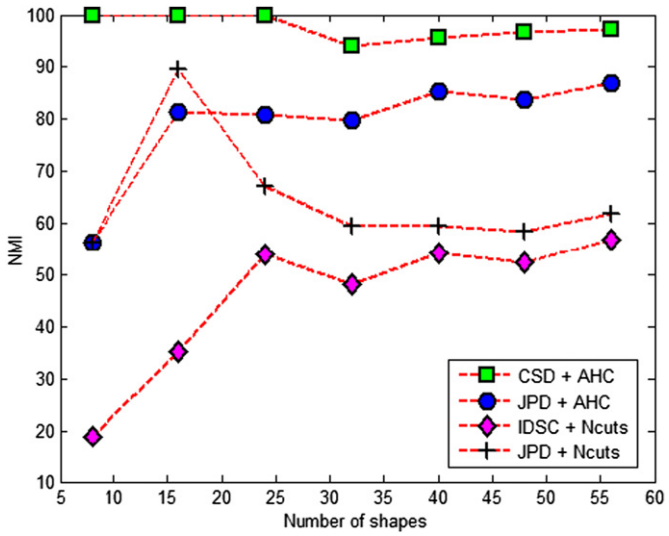

Fig. 17. The curves of NMI as the number of shape classes is increased on the Aslan and Tari data set with 56 shapes [2].

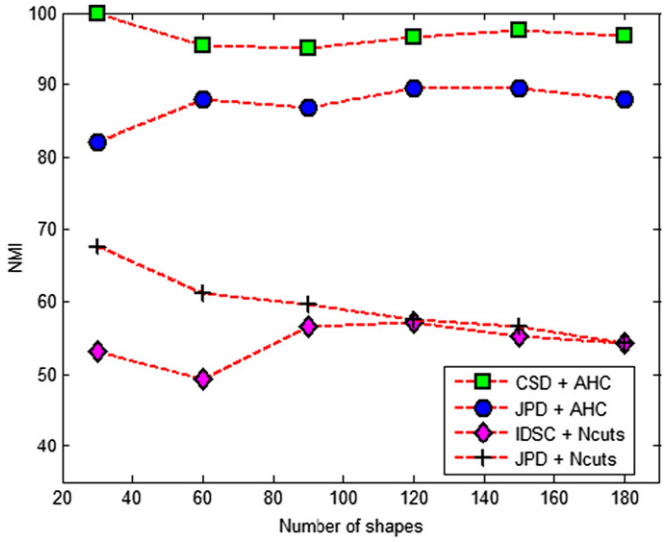

Fig. 18. The curves of NMI as the number of shape classes are increased on the Aslan and Tari data set with 180 shapes [1].

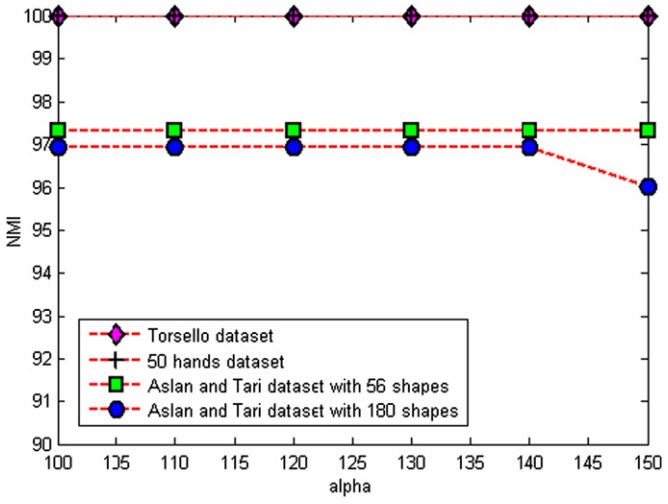

Fig. 19. The curves of NMI as the weight factor α are varied on the four data sets.

structure which captures the intrinsic intra-class structural information of the cluster of shapes. Consequently, it can be used to deliver more robust distance measure between clusters. The presented experimental results demonstrate that our shape clustering algorithm significantly outperforms other state-of-the-art methods.

The high time complexity is the limitation of our method, since the overall time required for a basic agglomerative hierarchical

clustering is $O(N^2 \log(N))$ [31], where N is the number of data points. However, we can speed up the clustering by merging more than two closest clusters in each iteration, which will be the topic of our future work.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities' HUST 2011TS110 and National Natural Science Foundation of China #60903096.

Appendix A

Preprocessing. A merging strategy given in [34] says that any pair of junction points are merged when the distance between them are less than a threshold. However, this strategy is not described clearly. For example, both the distance between junction points j_1 and j_2 and the one between junction points j_1 and j_2 are less than the threshold, however the distance between junction points j_2 and j_3 is larger than the threshold. Should j_2 and j_3 be merged together? Here, we propose to cluster the junction points by agglomerative hierarchical clustering with single linkage. At first, each junction point starts in its own cluster, then in each iteration, two closest clusters of junctions are merged as one moves up the hierarchy. The distance measure between clusters is always based on the distance measure between pairs of closest instances.

For two junction points j_1 and j_2 on the skeleton S with n end points $E=\{e_1,\ldots,e_n\}$, the distance between them is

$$d(j_1,j_(2)=\begin{cases}\frac{1}{n}\sum_{i=1}^{n} pd(P(j_1,e_i),P(j_2,e_i)) & \text{if } j_1 \frown j_2,\\ \infty & \text{otherwise.}\end{cases} \tag{14}$$

To cluster the junction points on skeleton S, we should give the distance measure between clusters. For two cluster junction points JP,JQ on skeleton S, the distance between them determined by the single linkage criterion is

$$cd(JP,JQ)=\begin{cases}\min_{j_p \in JP, j_q \in JQ} d(j_p,j_q) & \text{if } JP \frown JQ,\\ \infty & \text{otherwise,}\end{cases} \tag{15}$$

where $JP \frown JQ$ means JP and JQ are connected directly, i.e., there are no junction points of other clusters in any paths between the junction points of JP and JQ.

Suppose that all junctions had been merged into m clusters $\{JP_i\}_{i=1}^{m}$, the stopping criterion of the merge process is

$$\min_{i,k} cd(JP_i,JP_k) \geq T_m, (i,k=1,2,\ldots,m, i\neq k), \tag{16}$$

where T_m is a threshold. The larger threshold means less clusters, i.e. more junction points would be merged together. Since the path distance is scale invariant, the threshold T_m is scale invariant too. We set $T_m=5$ empirically in all our experiments.

The junction points in one shape merged into the same cluster are considered as a whole for the next computation. This means a junction node in a skeleton graph corresponds a cluster of junction points. To compute the distance between two junction nodes JP,JQ, actually, we choose $jp^* \in JP$ and $jq^* \in JQ$ to represent JP and JQ, respectively, if satisfying

$$(jp^*,jq^*)=\arg \min_{j_p \in JP, j_q \in JQ} jd(j_p,j_q). \tag{17}$$

Then the distance between JP and JQ is $jd(jp^*,jq^*)$. Other distances related to junction nodes are computed by the similar way.

References

[1] C. Aslan, A. Erdem, E. Erdem, S. Tari, Disconnected skeleton: shape at its absolute scale, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008) 2188–2203.
[2] C. Aslan, S. Tari, An axis based representation for recognition, in: Proceedings of the ICCV, 2005.
[3] X. Bai, L. Latecki, Path similarity skeleton graph matching, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008) 1282–1292.
[4] E. Baseski, A. Erdem, S. Tari, Dissimilarity between two skeletal trees in a context, Pattern Recognition 42 (2009) 370–385.
[5] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 509–522.
[6] H. Blum, Biological shape and visual science, Journal of Theoretical Biology 38 (1973) 205–287.
[7] M. Demirci, A. Shokoufandeh, S. Dickinson, Skeletal shape abstraction from examples, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2009) 944–952.
[8] M. Demirci, A. Shokoufandeh, Y. Keselman, L. Bretzner, S. Dickinson, Object recognition as many-to-many feature matching, International Journal of Computer Vision 69 (2006) 203–222.
[9] A. Erdem, A. Torsello, A game theoretic approach to learning shape categories and contextual similarities, in: Proceedings of the SSPR, 2010.
[10] P.F. Felzenszwalb, J. Schwartz, Hierarchical matching of deformable shapes, in: Proceedings of the CVPR, 2007.
[11] L. Han, R.C. Wilson, E.R. Hancock, A supergraph-based generative model, in: Proceedings of the ICPR, 2010.
[12] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer, New York, 2009.
[13] R. Lakäemper, J. Zeng, A context dependent distance measure for shape clustering, in: Proceedings of the ISVC, 2008.
[14] L. Latecki, V. Megalooikonomou, Q. Wang, R. Lakämper, C. Ratanamahatana, E. Keogh, Partial elastic matching of time series, in: Proceedings of the ICDM, 2005.
[15] H. Lin, D.W. Jacobs, Shape classification using the inner-distance, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (2007) 286–299.
[16] D. Macrini, K. Siddiqi, S. Dickinson, From skeletons to bone graphs: medial abstraction for object recognition, in: Proceedings of the CVPR, 2008.
[17] W. Mio, A. Srivastava, S. Joshi, On shape of plane elastic curves, International Journal of Computer Vision 73 (2007) 307–324.
[18] M. Pelillo, K. Siddiqi, S. Zucker, Matching hierarchical structures using association graphs, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1999) 1105–1120.
[19] T. Sebastian, P. Klein, B. Kimia, Recognition of shapes by editing shock graphs, in: Proceedings of the ICCV, 2001.
[20] T. Sebastian, P. Klein, B. Kimia, Recognition of shapes by editing their shock graphs, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 550–571.
[21] T.B. Sebastian, B.B. Kimia, Curves vs skeletons in object recognition, in: Proceedings of the ICIP, 2001.
[22] W. Shen, X. Bai, R. Hu, H. Wang, L. Latecki, Skeleton growing and pruning with bending potential ratio, Pattern Recognition 44 (2011) 196–209.
[23] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 888–905.
[24] K. Siddiqi, B. Kimia, A shock grammar for recognition, in: Proceedings of the CVPR, 1996.
[25] K. Siddiqi, B. Kimia, A. Tannenbaum, S. Zucker, Shocks, shapes, and wiggles, Image and Vision Computing 17 (1999) 365–373.
[26] K. Siddiqi, A. Shkoufandeh, S. Dickinson, S. Zucker, Shock graphs and shape matching, in: Proceedings of the ICCV, 1998.
[27] K. Siddiqi, A. Shokoufandeh, S. Dickinson, S. Zucker, Shock graphs and shape matching, International Journal of Computer Vision 35 (1999) 13–32.
[28] R. Sokal, C. Michener, A statistical method for evaluating systematic relationships, University of Kansas Science Bulletin 38 (1958) 1409–1438.
[29] H. Späth, Cluster Analysis Algorithms, Ellis Horwood, Chichester, 1980.
[30] A. Srivastava, S. Joshi, W. Mio, X. Liu, Statistical shape analysis: clustering, learning, and testing, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 590–602.
[31] P.N. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, Addison Wesley, 2005.
[32] A. Torsello, E. Hancock, Learning shape-classes using a mixture of tree-unions, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 954–967.
[33] B. Wang, X. Bai, X. Wang, W. Liu, Z. Tu, Object recognition using junctions, in: Proceedings of the ECCV, 2010.
[34] Y. Xu, B. Wang, W. Liu, X. Bai, Graph matching based on critical points using path similarity, in: Proceedings of the ACCV, 2009.
[35] D. Yankov, E. Keogh, Manifold clustering of shapes, in: Proceedings of the ICDM, 2006.
[36] S. Zhu, A.L. Yuille, Forms: a flexible object recognition and modeling system, International Journal of Computer Vision 20 (1996) 187–212.
[37] Y.J. Zhu, K. Grauman, Foreground focus: unsupervised learning from partially matching images, International Journal of Computer Vision 85 (2009) 143–166.

Wei Shen received both his B.S. and Ph.D. degree in Electronics and Information Engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2007 and 2012. From April 2011 to November 2011, he worked in Microsoft Research Asia as an intern. Now he is a faculty of School of Communication And Information Engineering, Shanghai University.

Yan Wang received her B.S. degree in Electronics and Information Engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2011. Currently, she is a Ph.D. student at Nanyang Technological University, Singapore.

Xiang Bai received both his B.S. and M.S. degree both in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003 and 2005, respectively. He obtained his Ph.D. degree from HUST in 2010. From January 2006 to May 2007, he worked in the Department of Computer Science and Information, Temple University. From October 2007 to October 2008, he worked in the University of California, Los Angeles as a joint Ph.D. student. Now he is a faculty of EI Deptartment, HUST. His research interests include computer graphics, computer vision, and pattern recognition.

Hongyuan Wang is a professor in the Department of Electronics and Information Engineering at the Huazhong University of Science and Technology. From 1984 to 1985, he worked in the University of Oklahoma as a visiting scholar. His current research areas include digital video communication and digital signal processing.

Longin Jan Latecki is a professor in the Department of Computer and Information Sciences at the Temple University in Philadelphia. He is the winner of the 25th Pattern Recognition Society Award together with Azriel Rosenfeld for the best paper published in the journal Pattern Recognition in 1998. He received the main annual award from the German Society for Pattern Recognition (DAGM), the 2000 Olympus Prize. He is a member of the Editor Board of Pattern Recognition and chairs the IS&T/SPIE annual conference series on Vision Geometry. He has published and edited over 150 research articles and books. His main research areas are shape representation and shape similarity, object recognition, robot mapping, video analysis, data mining, and digital geometry.