Computer-Aided Design 44 (2012) 1253–1268

Contents lists available at SciVerse ScienceDirect

Computer-Aided Design

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

Robust shape normalization of 3D articulated volumetric models

Chao Wang [a,d], Yu-Shen Liu [a,b,c,*], Min Liu [e], Jun-Hai Yong [a,b,c], Jean-Claude Paul [a,f]

[a] School of Software, Tsinghua University, Beijing 100084, China
[b] Key Laboratory for Information System Security, Ministry of Education of China, China
[c] Tsinghua National Laboratory for Information Science and Technology, China
[d] Department of Computer Science and Technology, Tsinghua University, China
[e] Institute of Manufacturing Engineering, Tsinghua University, China
[f] INRIA, France

ARTICLE INFO

Article history:
Received 30 August 2011
Accepted 11 July 2012

Keywords:
Shape normalization
3D articulated models
Robust statistics
Implicit shape representation
Iteratively reweighted least squares

ABSTRACT

3D shape normalization is a common task in various computer graphics and pattern recognition applications. It aims to normalize different objects into a canonical coordinate frame with respect to rigid transformations containing translation, rotation and scaling in order to guarantee a unique representation. However, the conventional normalization approaches do not perform well when dealing with 3D articulated objects.

To address this issue, we introduce a new method for normalizing a 3D articulated object in the volumetric form. We use techniques from robust statistics to guide the classical normalization computation. The key idea is to estimate the initial normalization by using implicit shape representation, which produces a novel articulation insensitive weight function to reduce the influence of articulated deformation. We also propose and prove the articulation insensitivity of implicit shape representation. The final solution is found by means of iteratively reweighted least squares. Our method is robust to articulated deformation without any explicit shape decomposition. The experimental results and some applications are presented for demonstrating the effectiveness of our method.

1. Introduction

3D shape normalization is a common task in various computer graphics and pattern recognition applications. It aims to normalize different objects into a canonical coordinate frame in order to guarantee a unique representation [1]. One important premise about normalization is that only shapes with obvious orientation could be normalized easily and naturally because their principal axes could be easily obtained and accepted [2–4]. Many conventional normalization methods treat shapes as rigid bodies and usually use the centroid, principal axes, and the size of the bounding box for normalizing the localization, orientation, and scale of the shapes. However, a large number of 3D objects, such as humans and animals, are not rigid, but flexible to change their spatial poses. The conventional normalization methods do not handle shape deformation of flexible objects well. Thus, there is a growing need to develop new normalization algorithms for non-rigid shape analysis.

To address this issue, we introduce a new method for robust shape normalization of 3D articulated objects. The core idea is to combine robust statistical methods with a naturally defined shape representation—*implicit shape representation*. The presented method is robust with respect to articulated deformation and it avoids any explicit shape decomposition. In contrast to shape alignment/registration, which operates between the original object and reference ones, the normalization process presented in this paper only depends on the given individual object itself without referring to any other objects. For this reason, the proposed normalization can be computed off-line and this property is of great importance to many further applications that require fast and effective processing, such as shape retrieval and shape matching.

1.1. Related work

Although the main issue of this paper is shape normalization for 3D objects, many techniques are connected to 2D shape normalization. Therefore, reviewing the 2D case will provide a good understanding for the 3D situation. Note that shape normalization discussed in this paper is in terms of rigid transformations. Although a number of previous studies [5–7] are also named "*normalization*" in 2D, they are actually related to *affine transformation*, which mainly handle skew distortion. A review of the many

* Corresponding author at: School of Software, Tsinghua University, Beijing 100084, China. Tel.: +86 10 6279 0533, +86 159 1083 1178 (Mob.); fax: +86 10 6279 5460.
E-mail addresses: liuyushen@tsinghua.edu.cn, liuyushen00@gmail.com (Y.-S. Liu).

0010-4485/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2012.07.006

available methods for affine invariant normalization is beyond the scope of this paper. In this section, the work most related to ours will be reviewed.

1.1.1. Normalization of 2D shapes

The existing methods for 2D shape normalization can be roughly divided into two categories: boundary-based methods and region-based ones [4]. The former is dependent on the boundary points of shapes [3,4]. The latter takes into account all points that are not only on the boundary but also inside the boundary, and shape normalization is usually determined by the axis of the least second moment of inertia [1]. Many related studies focused on the shape orientation as the sub-problem of normalization. For instance, some methods determine the orientation of a shape using Feret's diameter (a line between the two points on the periphery that are farthest apart), the major axis of an ellipse fitted to the silhouette contour, or the angle of the axis of the least moment of inertia. It is worth mentioning that Žunić et al. proposed a series of methods for 2D shape orientation, such as boundary based shape orientation [4] and curvature weighted gradient based shape orientation [8]. Ref. [3] pointed out that, in some situations, the principal axis of normalization is debatable for the definition of shapes even without noise or deformation. However, most principal axis approaches are quite sensitive to shape deformations, which result in that the position of the center and the scaling size are also affected by deformations. The reader may consult [1–4,8] for a broader introduction of the relevant work.

The work most related to ours is Cortadellas et al.'s work [1] and Jiang et al.'s work [9]. Ref. [1] presented a 2D shape normalization method to cope with the silhouette deformations in the image domain. The orientation of a shape is computed based on solving a weighted least squares problem through a robust algorithm, called iteratively reweighted least squares (IRLS). The center of the shape is refined iteratively with respect to the updated weight function. Finally, the size of the shape is determined by the area of a silhouette. Although they illustrated the effectiveness of the algorithm with some experiments, there is no theoretical proof guarantee that the weight function used in [1] is robust to articulated deformations. Ref. [9] introduced a shape normalization method based on the implicit shape representation without any iterative process. Such representation was used for designing an intensity function to obtain a center and principal axis of a 2D shape. Although this method is robust to deformations, the final principal axis obtained by this method strongly relies on a good initial estimation, which may be deflected from the real position significantly if deformation exists. In this paper, we borrow some advantages from these two methods [1,9] and propose a new robust normalization method on 3D volumetric articulated shapes.

1.1.2. Normalization of 3D shapes

3D objects obtained by various modeling and scanning systems often have different coordinate frames, since the objects' shapes are always created in a particular coordinate system [10–12]. The coordinate frame normalization often needs to be accomplished at first for many further applications. For instance, many 3D shape retrieval techniques require *a priori* normalization for a given query object and the database's models, and then all of the normalized models are aligned into a common coordinate system before they are matched [13,14]. Another application is shape registration in which the most widely used technique is Iterative Closest Point (ICP). But ICP is only effective when the initial normalization of the input shapes is close to a correct shape alignment [15]. Furthermore, automatically selecting good viewpoints for 3D objects often seeks the shape normalization that can help determine the position of a camera for viewing an object in a natural way [10,16]. Analogously, normalization can help to easily generate the recognizable thumbnail images of objects, which is useful for the management of large 3D shape repositories. Some extra applications can be considered as sub-problems of shape normalization, such as generating 2D drawings from 3D models [17], principal axes determination [12], orientation and stable pose estimation [2–4,8,10].

One of the most popular approaches for normalizing 3D shapes with respect to rigid transformations involves computation of the center of gravity, the orientation of the principal axis, and the size of the bounding box [1,9–11]. Several early studies have been developed for shape normalization of volumetric models [18–20]. Galvez and Canton [19] presented an approach of shape normalization for recognition of 3D objects using the principal axes method. Bribiesca [18] normalized a volumetric shape based on the major axis defined by the line joining two voxels furthest away from each other, and this method was also applied to optimum transformation of 3D objects [20].

The simplest and most commonly used technique for this task is based on *principal component analysis* (PCA), in which the center of gravity is chosen as the origin, the principal axes are chosen as the canonical axes, and the size of the bounding box is set as the scale factor of the shape [21]. The main advantages of PCA is that it is simple, fast, and applicable for most 3D models. However, it is well known that PCA is not robust under shape deformation [13]. The normalization derived by PCA might be quite different for some similar shapes due to small local differences between them [13]. Ref. [11] made a detailed analysis of PCA's uncertainty in the coordinate direction normalization of meshes. Recently, Passalis et al. [14] improved PCA for normalization by considering the symmetry planes of a 3D mesh object. Their method relies on an assumption that most of real life objects are symmetrical with respect to a plane. However, it is a separate challenge to locate multiple symmetry planes for 3D objects [22]. Fu et al. [10] presented a solution to detect the object's upright orientation, but this method is not appropriate when dealing with deformable shapes. Lian et al. [23] introduced a new method to evaluate a 3D polygonal mesh based on rectilinearity measure, which is defined as the maximum ratio of the surface area to the sum of three orthogonal projected areas of the mesh. Their method could be used to normalize the pose of a 3D mesh by finding the fitted localization and orientation that maximizes its rectilinearity. The normalization method using rectilinearity measure is usually better than that using PCA. However, the method [23] does not deal with the articulated models either.

Recently, Liu and Ramani [12] proposed a robust statistical method for determining the principal axes of 3D point-based shapes. It is based on least median of squares (LMS) optimization for guiding the classical PCA computation. The method in [12] can automatically identify portions of a flexible shape as the major region or minor regions. Here the forward search technique is used for approximating the LMS optimization. The forward search technique starts from a small outlier-free subset robustly chosen as the initial major region. Then, the principal axes and the origin are computed using PCA to the chosen subset, and the point with the lowest residual in the remaining points is added to the subset. The step is repeated until the error is larger than a predefined threshold, and the three principal axes obtained by PCA during the last iteration are regarded as the final results. One main advantage of [12] is that it automatically disregards outliers and distinguishes the shape as the major and minor regions during the principal axes determination without any extra segmentation procedure. Nevertheless, Ref. [12] deals with the point-based shape with only its boundary points and it is sensitive to the density of sample points on the boundary surfaces. In addition, the method is time-consuming due to the forward search procedure and it has not been extended to the volumetric shapes in our current application.

1.2. Contributions

To address the issue of normalizing a 3D articulated shape, we propose a robust normalization algorithm for estimating the localization, orientation and scale of the shape. We first assume the shape representation to be an implicit function [24,25], which takes into account all points that belong to the shape. Meanwhile, we prove that such a representation is insensitive to articulated deformations. Our goal is to find a consistent normalization for a given articulated object. Our method can be considered as a variation of the conventional normalization methods by combining robust statistical techniques and implicit shape representation. The main contributions of our work can be summarized as follows.

- A new robust normalization algorithm is proposed for estimating the localization, orientation and scale of 3D articulated shapes based on solving a weighted least squares problem utilizing IRLS and implicit shape representation. A natural articulation insensitive weight function is proposed to reduce the influence of articulated deformation during the normalization procedure.
- The articulation insensitivity of the implicit shape representation is proved. We show that the relative change of such a representation for each object point is always bounded by and dependent on the maximal diameter of the junctions.
- We apply our algorithm to some shape analysis applications such as, shape alignment and major regions localization for articulated models.

Before we go on, we would like to point out that the shape alignment/registration between an original object and its reference objects would not be included in our normalization process. The remainder of this paper is organized as follows. Section 2 gives some preliminaries used in this paper. Section 3 describes the implicit shape representation for an articulated shape and proves the articulation insensitivity of this representation. Section 4 is devoted to the algorithm of our shape normalization based on the robust statistical techniques and the implicit shape representation. Section 5 presents our experimental results and Section 6 gives some discussions about our method including limitations. Finally, Section 7 concludes the paper.

2. Preliminaries

In this section, we first introduce the input 3D shape in a volumetric form. Then, we define some basic concepts for articulated shapes. Finally, the classical PCA is reviewed for deriving our method.

2.1. Volumetric models

Some techniques have been studied for the normalization problem of 3D shapes in variant surface forms, such as meshes [11,14] and point-sampled surfaces [12]. However, one known limitation is that it strongly depends on the sampling rule on boundary surfaces, which affects the result of normalization [11,12,14]. In fact, it is better to utilize not only the boundary surface but also the interior domain of 3D shapes. In this paper, we consider the input 3D shape as a digitized 3D object in the volumetric form, it encloses a volume properly and allows computation that is more robust to noise and perturbations. A volumetric object takes into account all of the voxels of the shape, which often make normalization better than those based on boundary samples because they are less influenced by the presence or absence of a single voxel around the periphery [1]. The volumetric models are also common to many applications in medical scanners, scientific simulations, articulated shape description and computational biology [26–29]. We consider a volumetric model as a uniform 3D lattice consisting of object points O and background points $\overline{O}$. Let $O = \{\mathbf{p}_i | i = 1, \ldots, N\}$, where each lattice point $\mathbf{p}_i = (x_i, y_i, z_i)^T$ is a 3D vector.

In this paper we present a new method of normalization for volumetric objects, but we do not restrict ourselves to the volumetric form. Our method only utilizes a set of discrete points (e.g. voxels) and consequently, many discrete shape representations should be suited too. Another argument for using the volumetric form is that there are a variety of good algorithms for volumetric approximations of surface forms, such as *PolyMender* [30] and *binvox* [31], which produce a binary 3D voxel grid that approximates the original surface. Volumetric data can be generated by placing a 3D shape into a 3D cubic grid (such as $128 \times 128 \times 128$), compactly fitting the shape to the grid. Each lattice point is assigned either 1 or 0; 1 for object points O and 0 for background points $\overline{O}$.

2.2. Articulated shapes

Non-rigid shapes are ubiquitous in the world and, due to their physical properties, can undergo deformations [32]. Non-rigid shape analysis has been receiving growing attention in many applications in pattern recognition [24,32–35]. One simplified strategy to non-rigid shape analysis is based on the articulated shape model, which assumes that the non-rigid shape is composed of rigid parts, each of which has a certain freedom to move. In our work, we deal with the input 3D shape as a model of an *articulated object* defined roughly as follows based on Refs. [9,34,35].

Definition 1 (*Articulated Shape*)**.** An articulated shape O consisting of K disjoint rigid parts $R_1, \ldots, R_K$ and L flexible junctions $J_1, \ldots, J_L$, such that

$$O = (R_1 \cup \cdots \cup R_K) \cup (J_1 \cup \cdots \cup J_L). \tag{1}$$

Intuitively, O is an articulated object if it satisfies the following conditions:

(1) O can be decomposed into several rigid parts connected by junctions.
(2) The junctions between parts are very small compared to the rigid parts they connect.
(3) Let Φ be a transformation that changes the pose of an object O. Φ is roughly considered as an articulated transformation if the transformation of any part of O is rigid (rotation and translation only) and the transformation of junctions can be non-rigid or flexible.
(4) The new shape O' achieved from articulation of O is again an articulated object and can articulate back to O.

Besides the above definition, the articulated shapes discussed in our work are specially motivated by some classes of shapes, such as some human-like shapes or animal-like shapes. Within each of these classes, an articulated shape is usually composed of both a "main body" (e.g. the trunk or torso of a human being or an animal) and "branches" (e.g. head, limbs and tails). The similar assumption was also presented in Ref. [9]. Observing that the main body is usually "fat" and near the natural principal axis, while the branches are often "thin" and far away from the principal axis. Another fact needs to be emphasized that only the shapes with obvious orientation could be normalized easily and naturally because their principal axes could be explicitly obtained and accepted [2–4]. The shapes with unnatural orientations, such as rotationally symmetric shapes and other irregular shapes, cannot be normalized because their principal axes are confusing in some sense.

From Definition 1, the articulated transformation from an articulated shape O to the corresponding shape O' is considered as a one-to-one reversible mapping $\Phi(O) = \{\Phi(\mathbf{p}) : \mathbf{p} \in O\}$, where

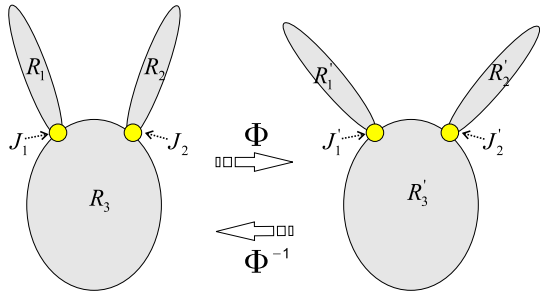

Fig. 1. Illustration of articulated shapes. (a) An articulated shape $O = (R_1 \cup R_2 \cup R_3) \cup (J_1 \cup J_2)$ with three rigid parts and two junctions. (b) The corresponding articulated shape O' of O after one articulation deformation Φ by rotating R_1 and R_2 with flexible deformation of J_1 and J_2. The reversible mapping is Φ^{-1}. Note that R_3 of rigid parts is the main body while R_1 and R_2 belong to the branches of O.

Φ^{-1} denotes an articulated transformation that maps O' to O. Then we have

$$O' = \Phi(O) \quad \text{and} \quad O = \Phi^{-1}(O').$$

Meanwhile, Φ can also be utilized on the rigid parts and junctions, i.e.

$$R_i' = \Phi(R_i) \quad \text{and} \quad J_j' = \Phi(J_j),$$

where $i = 1, 2, \ldots, K$, and $j = 1, 2, \ldots, L$. The same happens on each single object point, i.e. $\mathbf{p}' = \Phi(\mathbf{p}), \forall \mathbf{p} \in O$. Note that R_i' and J_j' are still rigid parts and junctions in O', respectively. This preserves the topology between the articulated shapes after articulated transformation (see Fig. 1).

The four intuitions in Definition 1 are fundamental and very important to our analysis next. Especially, the junctions between parts are very small compared to the parts they connect, which means that each junction J_j ($j = 1, 2, \ldots, L$) of an articulated shape O satisfies the following constraint condition about its size [35]

$$\text{diam}(J_j) \leq \varepsilon, \tag{2}$$

where $\text{diam}(J_j)$ is the diameter of a junction J_j, and it can be computed as the length of maximal distance between all pairs of points in J_j [35]. Eq. (2) means that the diameter $\text{diam}(J_j)$ of a junction J_j lies within a constant bound $\varepsilon \geq 0$, where ε is very small compared to the size of rigid parts. A special case is $\varepsilon = 0$, which means that all the junctions degenerate to the single points and O can be called an ideal articulated shape. In particular, since the deformable junctions stand flexible during articulated deformation, they still have the similarly constrained condition about their sizes, i.e.

$$\text{diam}(J_j') \leq \varepsilon', \tag{3}$$

where the diameter of J_j' lies within a constant bound ε', which is also very small compared to the size of rigid parts.

2.3. Review of PCA

One of most widely used methods in the principal axis based normalization applications is the classical PCA. As a least squares optimization process, PCA is widely used to compute the principal axes at a fixed center. Considering a 3D shape $O = \{\mathbf{p}_i | i = 1, \ldots, N\}$, PCA wants to find a orientation vector $\mathbf{e}$ through a specified center $\mathbf{o}$ such that the sum of the squared Euclidean distances between various points $\mathbf{p}_i \in O$ and the corresponding projection points $\mathbf{p}_i^*$ onto $\mathbf{e}$ are as small as possible, as described by the following minimization problem [21]:

$$\min_{\mathbf{e}} \sum_{i=1}^{N} \|\mathbf{p}_i - \mathbf{p}_i^*\|^2. \tag{4}$$

By adding the constraint $\|\mathbf{e}\| = 1$ for finding the best direction $\mathbf{e}$, the solution to this problem involves the covariance matrix defined by

$$\mathbf{A} = \sum_{i=1}^{N} (\mathbf{p}_i - \mathbf{o})(\mathbf{p}_i - \mathbf{o})^T. \tag{5}$$

The eigenvector corresponding to the largest eigenvalue of the covariance matrix $\mathbf{A}$ is the first principal axis $\mathbf{e}$. PCA is a linear transformation for 3D point sets. In fact, PCA is similar to the linear regression in a sense of least squares. However, a single sample with a large error can change the principal axis arbitrarily. This results in that the derived principal axis might be quite different between some similar shapes [13]. In other words, PCA is quite sensitive to the silhouette deformations if the principal axis is used to orientate the articulated models. Therefore, a robust method is needed to estimate the orientation of articulated shapes.

3. Implicit shape representation (IS-rep)

Shape representation plays a central role in shape analysis and its task is the selection of an appropriate representation for the shapes of interest. Point clouds, parametric curves/surfaces, and medial axes representations are often considered in various applications [24,25,36]. Although these representations are powerful, they usually require a large number of parameters to deal with shape deformations. For our purpose, the *implicit shape representation* (IS-rep) described by Paragios et al. [24,25] is chosen. The IS-rep uses the Euclidean distance maps to represent the shapes. It provides an excellent means to characterize shape variations under articulated deformation. Next we will first provide a mathematical definition of the IS-rep, and then prove the articulation insensitivity of such a representation.

Let f be a function of the Euclidean distance for a given shape O with respect to its boundary surface S. The shape O defines a partition of the space: the boundary S, the inside region $[O-S]$, and the background region $\overline{O}$. The IS-rep could be defined as follows.

Definition 2 (*Implicit Shape Representation*)**.** The implicit shape representation of O is defined by a signed distance function:

$$f(\mathbf{p}) = \begin{cases} 0 & \mathbf{p} \in S, \\ +\mathcal{D}(\mathbf{p}, S) > 0 & \mathbf{p} \in [O - S], \\ -\mathcal{D}(\mathbf{p}, S) < 0 & \mathbf{p} \in \overline{O}, \end{cases} \tag{6}$$

where $\mathcal{D}(\mathbf{p}, S)$ refers to the minimum Euclidean distance from the point $\mathbf{p}$ to the boundary S, i.e.

$$\mathcal{D}(\mathbf{p}, S) = \min_{\mathbf{q} \in S} \{\|\mathbf{p} - \mathbf{q}\|\}. \tag{7}$$

The $f(\mathbf{p})$ in Definition 2 is called the implicit intensity value of each point $\mathbf{p}$. It is essentially a level set representation of shapes [24,37]. The fast marching algorithm or other advanced techniques can be used for the construction of such implicit shape representation [24,25,37]. Note that it is unnecessary to discuss the implicit functions of the points outside O because we only concern the points inside O and on the boundary S in our application.

There are two appealing features of implicit representation when dealing with 3D shape normalization. Both features are directly derived from Definitions 2 and 1.

Proposition 1. *The implicit shape representation in Eq. (6) is invariant to translation and rotation. When a shape undergoes scale variation, the intensity values of its associated distance map scale accordingly.*

The proof of Proposition 1 has been given in [24,25]. What's more, the implicit shape representation is also insensitive to

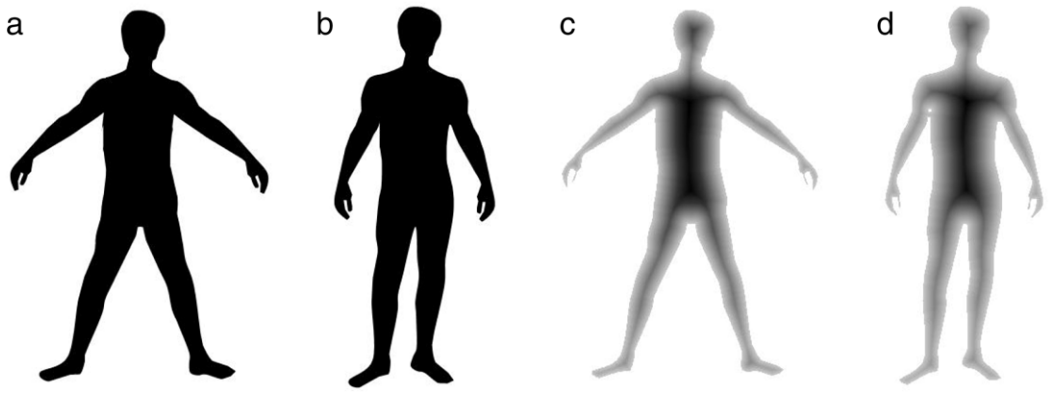

Fig. 2. Illustration of the IS-rep. (a) A 2D shape of a human model. (b) Another shape with different pose of the same human model in (a). (c) and (d) are the implicit distance maps of (a) and (b), respectively. The darker colors denote the higher implicit intensity values of points and the lighter colors denote the smaller ones.

articulated transformation. Intuitively, this is true since any articulated shape can be decomposed into rigid parts and small junctions connecting them. Here we will formally prove the articulation insensitivity feature of the IS-rep.

Proposition 2. *Let O be an articulated object and Φ be an articulation transformation of O as defined in Definition 1 such that $O' = \Phi(O)$. $\forall \mathbf{p} \in O$, and suppose $\mathbf{p}' = \Phi(\mathbf{p}) \in O'$. Let f be the implicit shape representation defined in Definition 2, then*

$$|f(\mathbf{p}') - f(\mathbf{p})| \leq \max\{\varepsilon, \varepsilon'\}, \tag{8}$$

where ε and ε' are the upper bounds of junction sizes of O and O', as defined in Eqs. (2) and (3), respectively.

Proof. According to Definition 1, an articulated object O can be decomposed into a set of rigid parts $\{R_i\}$ and small junctions $\{J_j\}$, $i = 1, 2, \ldots, K$, and $j = 1, 2, \ldots, L$. According to Proposition 1, it is obvious that the value of the implicit function f for each point $\mathbf{p} \in \{R_i\}$ stands constant during articulation transformation (rotation and translation only), i.e. $f(\mathbf{p}') = f(\mathbf{p}), \forall \mathbf{p}' = \Phi(\mathbf{p})$. That means that the implicit function of rigid parts $\{R_i\}$ is unchangeable during articulation transformation.

For each point $\mathbf{p}$ from junctions, since the junctions are flexible and may undergo a small deformation, the implicit intensity value $f(\mathbf{p}')$ might change. However, according to the constraint conditions about the original junctions and deformed junctions, i.e. Eqs. (2) and (3), the change of the implicit function of each point $\mathbf{p} \in \{J_j\}$ can be represented as

$$\begin{cases} 0 \leq f(\mathbf{p}) \leq \operatorname{diam}(J_j) \leq \varepsilon & \forall \mathbf{p} \in \{J_j\}, \\ 0 \leq f(\mathbf{p}') \leq \operatorname{diam}(J_j') \leq \varepsilon' & \mathbf{p}' = \Phi(\mathbf{p}), \end{cases}$$

where ε and ε' are the upper bounds of the diameters of $\{J_j\}$ and $\{J_j'\}$, respectively. Then, the relationship between the implicit functions of $\mathbf{p}$ and $\mathbf{p}'$ is as follows:

$$|f(\mathbf{p}') - f(\mathbf{p})| \leq |\varepsilon - \varepsilon'| \leq \max\{\varepsilon, \varepsilon'\}, \quad \forall \mathbf{p} \in \{J_j\}.$$

Combining the above discussion of parts and junctions, this implies Eq. (8). □

Eq. (8) proves that the relative change from $f(\mathbf{p})$ to $f(\mathbf{p}')$ is always bounded and depends on the maximal diameter of junctions. When ε and ε' are very small compared to the size of the rigid parts, the change is tiny such that the two implicit function values can be roughly regarded as equivalent. This consequently means that the implicit function of junctions is almost invariant during articulated transformation. Fig. 2 illustrates the IS-reps for a 2D articulated shape and its deformation. The comparison of the IS-reps of the two shapes shown in Fig. 2(c) and (d) indicates that the implicit intensity values of points, such as the ones of the head, trunk and the four limbs, stay roughly consistent after articulated transformation.

We further clarify several issues. The above proof depends on the size limitation of junctions. The assumption is that a junction should have a relatively smaller size compared to parts; otherwise, it is more like a part itself. A more precise part–junction definition may provide a tighter upper bound but sacrifice some generality [35]. The definition also captures our intuition about what distinguishes articulation from other types of deformation. In fact, one advantage of using the implicit shape function is that it implicitly captures the part structure of an articulated shape.

4. Robust shape normalization

The approaches of shape normalization generally involve the determination of the localization, orientation and size of the shape. In this section, we will first introduce the procedure of our new normalization method and describe each step of the method in detail. Finally, the implementation details are given and the computational complexity of our algorithm is discussed.

During the normalization process, the key step is to compute the center and three principal axes to present the localization and orientation of the shape. Of the three principal axes, the first one presents the orientation of the shape and the other two could be computed based on the center and the first one. The center and the first principal axis obtained by solving the sum of squares minimization problem shown in Eq. (4) by PCA are sensitive to articulation deformations. One possible solution to reduce the sensitivity is to decrease the influence of deformation of the shape during computation. A general strategy is to weight the contribution of each object point during the principal axis computation. By advancing the minimization problem by PCA, we present the robust orientation problem as follows.

Definition 3 (*The Robust Orientation Problem*)**.** Given a center $\mathbf{o}$ and weight function ω_i of each point $\mathbf{p}_i$ from a shape O, the problem wants to find the orientation vector $\mathbf{e}$ through $\mathbf{o}$ such that minimization of the sum of weighted squared Euclidean distances described below is achieved [21]:

$$\min_{\mathbf{e}} \sum_{i=1}^{N} \omega_i \|\mathbf{p}_i - \mathbf{p}_i^*\|^2, \tag{9}$$

where $\mathbf{p}_i^*$ is the projection point of $\mathbf{p}_i$ onto the line through $\mathbf{o}$ in the direction of $\mathbf{e}$.

The problem shown in Eq. (9) cannot be solved linearly due to the diversity of weight function ω_i. One commonly used method to solve the problem is the iteratively reweighted least

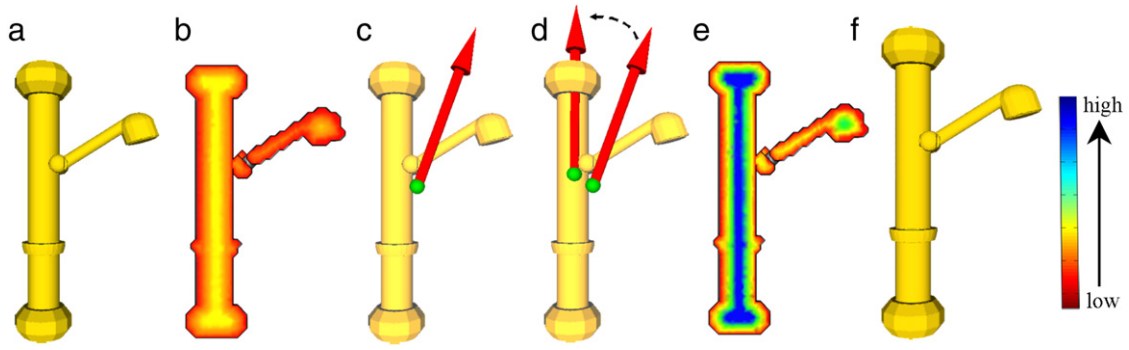

Fig. 3. Illustration of the procedure of the robust shape normalization. (a) An original shower-like shape. (b) The cross-section of the shape presenting the IS-rep. (c) Computing the initial center (green ball) and first principal axis (red arrow). (d) Computing the center and first principal axis iteratively (the dashed line denotes the moving direction of the center and axis during the iterative process). (e) The cross-section of the shape presenting the final weights after iterative computation. (f) Final shape after size, position and orientation adjustment. The rightmost color bar maps the variation of weight function values. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

squares (IRLS) [1]. Its main idea is to iteratively compute the center and the principal axis through re-weighting a shape dependent weight function. Its main advantage is that the initial localization and orientation of shapes could be refined gradually during the iterative process to a satisfactory result. Here the weight function ω_i plays an important role in solving the robust orientation problem using IRLS. Cortadellas et al. [1] presented a weight function in their normalization algorithm to obtain the robust results to deformations, where the weight function mainly balances the contribution of distance between each silhouette pixel to its principal axis. In this paper, we propose a new articulation insensitive weight function by combining the weight function in [1] and the IS-rep.

Based on the classical procedure of normalization and IRLS algorithm, the main procedure of our shape normalization method on a volumetric articulated shape is given as follows:

(1) Calculate the IS-rep of the shape.
(2) Estimate the initial center and first principal axis, and compute the initial weight functions of all object points of the shape.
(3) Compute the center and three principal axes iteratively until the termination condition is satisfied.
(4) Adjust the size, position and orientation of the shape in the canonical coordinate frame by scaling, translation and rotation.

Fig. 3 illustrates the procedure of the robust shape normalization on an example. The details of these four steps will be described from Sections 4.1–4.4, and the implementation details will be given in Section 4.5.

4.1. Calculation of the IS-rep

The first step of our normalization is calculating the IS-rep, which will be utilized in the next steps for estimating the weight functions and initial center and axes. According to the definition of the IS-rep, the implicit intensity value $f(\mathbf{p}_i)$ of point $\mathbf{p}_i$ is the minimum Euclidean distance from $\mathbf{p}_i$ to the boundary points set S of the shape. Therefore, the IS-rep can be obtained by computing the distances between $\mathbf{p}_i$ and all boundary points in S, and then choosing the smallest one as the intensity value $f(\mathbf{p}_i)$. Alternatively, the fast marching algorithm or other advanced techniques can be used for speeding up the computation [24,25]. The reader may consult [9,24,25,36,38,39] for the details of computing the IS-rep.

4.2. Initial estimation

During the second step of our normalization, some variables including the center, first principal axis and weights of all points are initialized and will be used for the following iterative process. In this section, we focus on the computation of the initial center and the first principal axis. Note that the weights are closely related to the iterative process, and the method to obtain them will be introduced in Section 4.3.

The initial center and the first principal axis play an important role in the normalization process. An undesirable initial estimation will make the final center and axes worse. The traditional methods including PCA usually use the center of gravity (barycenter) to define the shape position. However, the localization of the barycenter is easily affected by the shape deformation, which has a negative influence in the performance of traditional normalization methods [1,14]. Recently, Rustamov et al. [40] introduced a new geometric property "barycentroid" capturing the notion of semantic center of surface meshes. Although this kind of center is claimed to be insensitive to articulation deformations, it is defined on surface meshes with minimizing the average squared interior distances to the set elements, which cannot be directly used in the volumetric models. It is also interesting to extend the barycentroid computation from surface meshes to volumetric models in the future, which may be an alternative center in our initial estimation.

Considering the feature of the IS-rep, we define a new form of center called the *implicit center*:

$$\mathbf{o} = \frac{\sum_{i=1}^{N} (f(\mathbf{p}_i) \cdot \mathbf{p}_i)}{\sum_{i=1}^{N} f(\mathbf{p}_i)}, \tag{10}$$

where $f(\mathbf{p}_i)$ is the implicit intensity value of $\mathbf{p}_i \in O$ described in Eq. (6). The implicit center represents the weighted average of all points in the shape. It is more closer to the center of the "fat" main body of the shape than the traditional barycenter that is easily influenced by "thin" branches away from the initial first principal axis. The first principal axis $\mathbf{e}_1$ through the implicit center $\mathbf{o}$, obtained by the classical PCA, is used for our initial principal axis.

4.3. Computing the center and principal axes iteratively

After initial estimation, the center and principal axes will be computed iteratively using the IRLS strategy. The weight function ω_i is critical during the iterative process to obtain the desirable final center and axes. In this section, we will first review the weight function used in [1] and then introduce our new articulation insensitive weight function. At last, the iterative process of our algorithm is described based on the new weight function.

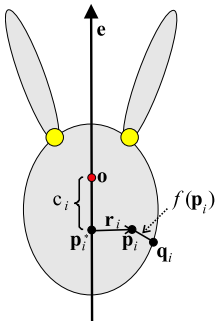

Fig. 4. Illustration of relative parameters. The desirable weight function ω_i should take not only the projection distance $\|\mathbf{r}_i\|$ but also the implicit intensity value $f(\mathbf{p}_i)$ into consideration.

4.3.1. Principal axis dependent weight function

Cortadellas et al. [1] presented a weight function in their normalization algorithm, where the weight function mainly balances the contribution of distance between each silhouette pixel to its principal axis. In this paper, we call the weight function in Ref. [1] the *principal axis dependent weight function*. This weight function in the case of a 3D shape is described below.

Let $\mathbf{p}_i$ be an arbitrary object point of a 3D volumetric model O. Assume that $\mathbf{o}$ and $\mathbf{e}_1$ are the center and first principal axis, respectively. Let $\mathbf{p}_i^*$ be the point by projecting $\mathbf{p}_i$ onto the line through $\mathbf{o}$ in the direction of $\mathbf{e}_1$. Then, the length of the projection line is written by

$$c_i = \|\overrightarrow{\mathbf{op}_i^*}\| = \frac{\mathbf{e}_1{}^T \cdot (\mathbf{p}_i - \mathbf{o})}{\|\mathbf{e}_1\|}. \tag{11}$$

Based on Eq. (11), the residual vector between $\mathbf{p}_i$ and its projection point $\mathbf{p}_i^*$ can be denoted by $\mathbf{r}_i = \mathbf{p}_i - \mathbf{p}_i^* = \mathbf{p}_i - \mathbf{o} - c_i\mathbf{e}_1$. Fig. 4 illustrates the residual vector $\mathbf{r}_i$ in a simple case.

The principal axis dependent weight function [1] can be defined as

$$\omega_i = \frac{\sigma^2}{(\mathbf{r}_i^T \cdot \mathbf{r}_i + \sigma^2)^2}, \tag{12}$$

where σ is a non-zero scale parameter related to $\mathbf{r}_i$. In Ref. [1], σ is typically selected as $\sigma = 1.4826 \cdot med_i\{|(\|\mathbf{r}_i\| - med_j(\|\mathbf{r}_j\|))|\}$, where '*med*' is the median operator. Eq. (12) means that the principal axis dependent weight function ω_i is inversely proportional to the distance between $\mathbf{p}_i$ and $\mathbf{e}_1$.

4.3.2. New articulation insensitive weight function

The weight function in Eq. (12) only considers the distance between $\mathbf{p}_i$ and its corresponding projection point $\mathbf{p}_i^*$ onto the principal axis $\mathbf{e}_1$. This is simple and desirable in the general normalization; however, for the articulated shape, it cannot penalize the objects' points on the periphery region which are also near to the estimated first principal axis. Considering the feature of the IS-rep, the implicit intensity value of a periphery point is very possibly smaller than that of a central region point. Hence, the IS-rep could improve the definition of the weight function.

By combining the IS-rep and the principal axis dependent weight function in Eq. (12), we present a new *articulation insensitive weight function* defined as

$$\omega_i' = (f(\mathbf{p}_i))^\lambda \cdot (\omega_i)^\gamma, \tag{13}$$

where $f(\mathbf{p}_i)$ is the implicit intensity value in Eq. (6), ω_i is the principal axis dependent weight function in Eq. (12), and the parameters λ ($\lambda = 1, 2, \ldots$) and γ ($\gamma = 1, 2, \ldots$) both are positive integer constants to balance the effects of $f(\mathbf{p}_i)$ and ω_i. The larger λ is, the more sensitive the shape normalization is with respect to the IS-rep. The larger γ is, the more sensitive the shape normalization is with respect to the projection distances from object points to the axis. Intuitively, the new weight function in Eq. (13) takes not only the projection distance of an arbitrary point (i.e. the distance from $\mathbf{p}_i$ to the first principal axis) but also the implicit intensity value $f(\mathbf{p}_i)$ (i.e. the distance from $\mathbf{p}_i$ to the boundary of the model) into consideration.

Comparison of two weight functions. The following will compare our weight function in Eq. (13) with the principal axis dependent weight function in Eq. (12) in term of analyzing the effectiveness of two weight functions to the final center and principal axis. During normalizing an articulated shape, the center and first principal axis, computed based on a desirable weight function, should be able to present the localization and orientation of the shape's main body rather than its branches. To reach this goal, the desirable weight function should be able to assign the larger weights to the object points on the main body than ones on the branches. Here, we will give a proposition to show that our weight function Eq. (13) is more desirable than Eq. (12) in weight assignment.

Proposition 3. *Let O be an articulated shape and O' be the shape after articulation transformation from O. Suppose that a point $\mathbf{p}$ is on the main body of O and another point $\mathbf{q}$ is on a branch of O, and then their corresponding points $\mathbf{p}'$ and $\mathbf{q}'$ after articulation transformation are both on the main body and branches of O', respectively. If the minimum distance from $\mathbf{p}$ to the boundary is larger than that from $\mathbf{q}$ to the boundary, then*

$$\frac{\omega_{\mathbf{p}}'}{\omega_{\mathbf{q}}'} > \frac{\omega_{\mathbf{p}}}{\omega_{\mathbf{q}}}, \qquad \frac{\omega'_{\mathbf{p}'}}{\omega'_{\mathbf{q}'}} > \frac{\omega_{\mathbf{p}'}}{\omega_{\mathbf{q}'}}, \tag{14}$$

where ω and ω' are the principal axis dependent weight function and our new articulation insensitive weight function, as defined in Eqs. (12) and (13), respectively.

Proof. According to Eq. (13), the new articulation insensitive weights of $\mathbf{p}$ and $\mathbf{q}$ are

$$\omega_{\mathbf{p}}' = (f(\mathbf{p}))^\lambda \cdot (\omega_{\mathbf{p}})^\gamma, \qquad \omega_{\mathbf{q}}' = (f(\mathbf{q}))^\lambda \cdot (\omega_{\mathbf{q}})^\gamma.$$

Note that the minimum distance from $\mathbf{p}$ to the boundary is larger than the one from $\mathbf{q}$ to the boundary. Then, due to the definition of the IS-rep in Eq. (6), we have $f(\mathbf{p}) > f(\mathbf{q})$. Hence,

$$\frac{\omega_{\mathbf{p}}'}{\omega_{\mathbf{q}}'} = \left(\frac{f(\mathbf{p})}{f(\mathbf{q})}\right)^\lambda \cdot \left(\frac{\omega_{\mathbf{p}}}{\omega_{\mathbf{q}}}\right)^\gamma > \frac{\omega_{\mathbf{p}}}{\omega_{\mathbf{q}}}.$$

After articulation transformation of O, the new weights of $\mathbf{p}'$ and $\mathbf{q}'$ are

$$\omega_{\mathbf{p}'}' = (f(\mathbf{p}'))^\lambda \cdot (\omega_{\mathbf{p}'})^\gamma, \qquad \omega_{\mathbf{q}'}' = (f(\mathbf{q}'))^\lambda \cdot (\omega_{\mathbf{q}'})^\gamma.$$

According to Proposition 2 demonstrated in Section 3, the IS-rep is insensitive to articulation transformation and the implicit intensity value of an object point can be regarded as nearly constant after articulation transformation. Then we have $f(\mathbf{p}') = f(\mathbf{p}) > f(\mathbf{q}) = f(\mathbf{q}')$. Therefore,

$$\frac{\omega_{\mathbf{p}'}'}{\omega_{\mathbf{q}'}'} = \left(\frac{f(\mathbf{p}')}{f(\mathbf{q}')}\right)^\lambda \cdot \left(\frac{\omega_{\mathbf{p}'}}{\omega_{\mathbf{q}'}}\right)^\gamma > \frac{\omega_{\mathbf{p}'}}{\omega_{\mathbf{q}'}}.$$

For an articulated shape and its deformation, Eq. (14) implies that our weight function can assign a greater proportion of weights to the points on the main body than on the branches, in contrast with the principal axis dependent weight function in Eq. (12). This will distinguish the weights of points on the main body with ones on branches more clearly without requiring any prior shape decomposition. □

Fig. 5 compares the final weight distributions produced by Eq. (12) and our weight function with respect to two different poses of a 3D articulated shape. This comparison shows that our weight function keeps almost consistent weight distributions for the two poses, in which the weight values on the main body are significantly larger than the ones on four limbs.

4.3.3. Computing the center and principal axes based on new weight function

By combining our new weight function and the IRLS strategy, the final center and the first principal axis could be iteratively computed. At the k-th iteration, the equations to obtain the center $\mathbf{o}(k+1)$ and the first principal axis $\mathbf{e}_1(k+1)$ are as follows [1]:

$$\mathbf{o}(k+1) = \frac{\sum_{i=1}^{N} \omega_i'(k) \cdot (\mathbf{p}_i - c_i(k) \cdot \mathbf{e}_1(k))}{\sum_{i=1}^{N} \omega_i'(k)}, \tag{15}$$

$$\mathbf{e}_1(k+1) = \frac{\sum_{i=1}^{N} \omega_i'(k) \cdot c_i(k) \cdot (\mathbf{p}_i - \mathbf{o}(k))}{\sum_{i=1}^{N} \omega_i'(k) \cdot c_i^2(k)}, \tag{16}$$

where $c_i(k)$ and $\omega_i'(k)$ are computed using Eqs. (11) and (13), respectively. The iterative process ends if the difference in orientation between $\mathbf{e}_1(k+1)$ and $\mathbf{e}_1(k)$ is less than an angle error and $\|\mathbf{o}(k+1) - \mathbf{o}(k)\|$ is less than a distance error.

Direction ambiguity. The step after the iterative process is to determine the direction of the first principal axis. The vector $\mathbf{e}_1$ obtained after the iterative process is still a two-direction line with a 180° ambiguity, which means that if we rotate the model by 180° about the line, $\mathbf{e}_1$ still stands for the first principal axis of the model. An intuitive criterion of judging the direction of the principal axis is that the vector $\mathbf{e}_1$ should turn towards the half area holding the larger weights. For $\forall \mathbf{p}_i \in O$, let $\mathrm{sgn}(c_i)$ be the sign function of c_i, where c_i is the length of the projection line in Eq. (11). It can be easily found that $\sum_{i=1}^{N} \frac{1}{2}(\mathrm{sgn}(c_i) + 1)$ indicates the number of voxels that lie in the half space pointed by $\mathbf{e}_1$. Then the value z can be computed as follows [1]:

$$z = \frac{\sum_{i=1}^{N} (\mathrm{sgn}(c_i) + 1) \cdot \omega_i'}{\sum_{i=1}^{N} (-\mathrm{sgn}(c_i) + 1) \cdot \omega_i'}, \tag{17}$$

where ω_i' is the final articulation insensitive weight function of $\mathbf{p}_i$ in Eq. (13) after the iterative process. If $z > 1$, $\mathbf{e}_1$ actually holds the same direction of the principal axis; otherwise, the vector must be reversed in order to adapt the reasonable orientation.

After the center and first principal axis are obtained and the direction is determined, the other two axes of the shape can be computed as follows. We first project O onto the plane through $\mathbf{o}$ and perpendicular to $\mathbf{e}_1$, and then perform a 2D weighted PCA for getting the second and third principal axes $\mathbf{e}_2$ and $\mathbf{e}_3$ in a way similar to Ref. [12].

4.4. Size, position and orientation adjustment

The last step in our normalization process is to adjust the size, position and orientation of the shape, of which the size adjustment should be done first by scaling. In the 2D image domain framework, the scale factor is generally achieved by resizing the silhouette of a shape into a fixed size defined by its bounding box. However, the bounding box is very sensitive to shape deformations. During our normalization method, we use the scale factor defined by [9]:

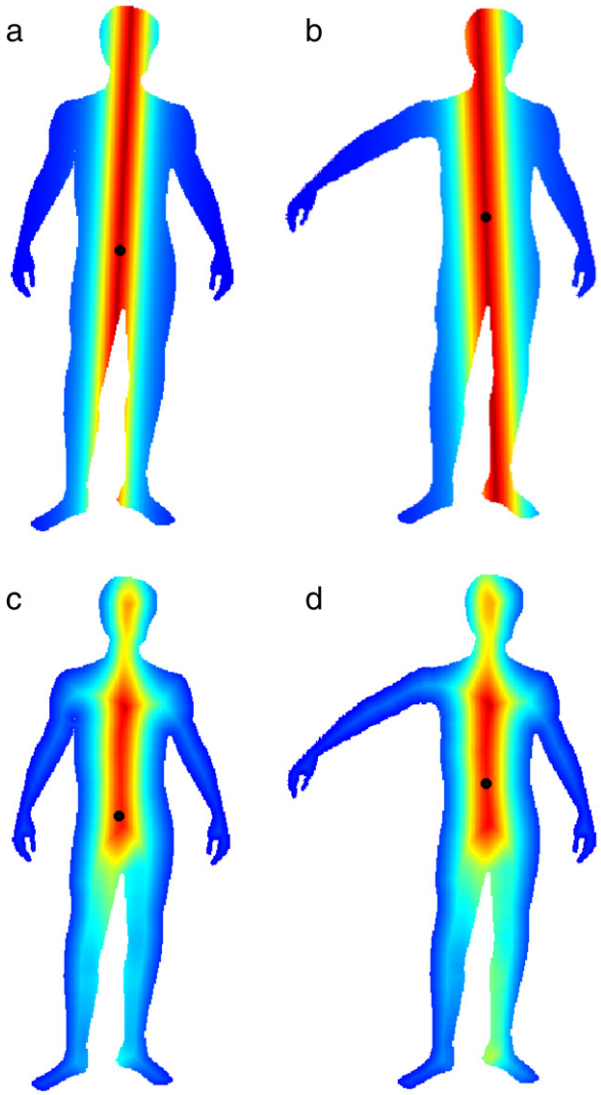

Fig. 5. The final weight distributions for two different poses of a 3D articulated shape. (a) and (b) are the final weight distributions of the two poses using the weight function in Eq. (12). (c) and (d) are the final weight distributions of the two poses using our weight function. The final centers are depicted as the black points. In contrast to the results in (a) and (b), (c) and (d) show that our weight function keeps the almost consistent weight distributions for the two poses, in which the weight values on the main body are significantly larger than the ones on four limbs.

$$s = \sqrt[3]{\sum_{i=1}^{N} f(\mathbf{p}_i)}, \tag{18}$$

where $f(\mathbf{p}_i)$ is the implicit intensity value of $\mathbf{p}_i$ in Eq. (6). The scale factor s is based on the whole volume of the model, and it is effective in normalizing the size of 3D shapes, especially with articulated deformations on the periphery.

Finally, the articulated shape is normalized into a canonical coordinate frame by achieving translation, rotation and scaling with respect to the final center, principal axes and scale factor.

4.5. The algorithm implementation

The outline of the robust shape normalization algorithm, named *RobustNormalization*, is described in Algorithm 1. Algorithm 1 calls

Algorithm 2 *GetCenterAxes* to compute the center and three principal axes. Algorithm 1 takes the original shape O as input, which results in the output including: the final center $\mathbf{o}$, the three principal axes $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$, as well as the weight set $\mathbf{W}$ of all points and the final shape point set O'. Firstly, the IS-rep of O is calculated, and then the initial values of some parameters are estimated. After that, Algorithm 2 is called to iteratively compute the center $\mathbf{o}$ and the first principal axis $\mathbf{e}_1$, on which the computation of the other two principal axes $\mathbf{e}_2$ and $\mathbf{e}_3$ are based. Finally, the size of the shape is adjusted through scaling, while the position and orientation of the shape in the coordinate frame are modified to ensure $\mathbf{o}$ as the origin and $\mathbf{e}_1$ as the vertical axis of the frame through translation and rotation.

Algorithm 1 : **RobustNormalization** (O, $\mathbf{o}$, $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$, $\mathbf{W}$, O')

Input:
$O \in \mathbb{R}^{3\times N}$: the input point set of the shape with N points
Output:
$\mathbf{o}$: the final center
$\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$: the final first, second and third principal axis
$\mathbf{W}$: the final weight set
O': the final point set of the shape
Local variables:
$f(O)$: the IS-rep of $O, f(O) = \{f(\mathbf{p}_i)|i = 1, 2, \ldots, N\}$
$\mathbf{o}_0$: the initial center
$\mathbf{e}_0$: the initial first principal axis
$\mathbf{W}_0$: the initial weight set, $\mathbf{W}_0 = \{\omega_i'(0)|i = 1, 2, \ldots, N\}$
Begin
1: Compute the IS-rep $f(O)$;
2: Compute $\mathbf{o}_0$, $\mathbf{e}_0$ and $\mathbf{W}_0$ using Eq. (10), PCA and Eq. (13), respectively;
3: **GetCenterAxes**($O, f(O), \mathbf{o}_0, \mathbf{e}_0, \mathbf{W}_0, \mathbf{o}, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{W}$);
4: Adjust the size of the shape based on the scale factor in Eq. (18) and modify the position and orientation of the shape in the coordinate frame by putting $\mathbf{o}$ at the origin of the frame through translation and $\mathbf{e}_1$ as the vertical axis of the frame through rotation.
End

The iterative process of IRLS is included in Algorithm 2, where the core equations are Eqs. (15) and (16) to compute the center and first principal axis. The algorithm uses the initial values obtained after the first two steps of Algorithm 1 as input and outputs the final center, three principal axes and weight set. The iterative process in Algorithm 2 ends when the errors between two adjacent iterations are less than the pre-defined thresholds. After that, the other two principal axes are computed based on the center and the first principal axis computed. The iterative computation process in Algorithm 2 is derived from the standard IRLS algorithm and its convergence has been shown in [41–43].

Computational complexity. We assume that the input volumetric model O consists of N object points, where its boundary surface S contains M boundary points ($M \leq N$). An upper bound of the running time to compute the IS-rep of O is $O(MN)$. In Algorithm 1, the computational complexity of the second step (i.e. initial estimation) is $O(N)$, since PCA is a linear transformation for 3D point sets. The exact cost of the third step, i.e. Algorithm 2, is hard to determine because the terminal iteration number is variant with respect to different shapes. Assuming that the terminal iteration number is k ($k <$ MAX_ITERS), it takes $O(kN)$ for computing the final center and axes in Algorithm 2. The complexity of the last step in Algorithm 1 for adjusting the size, position and orientation of the shape is linear. In summary, the total cost of our normalization algorithm is approximately $O(MN + kN)$. For our normalization algorithm, a great deal of the running time is spent on computing the IS-rep.

Algorithm 2 : **GetCenterAxes**($O, f(O), \mathbf{o}_0, \mathbf{e}_0, \mathbf{W}_0, \mathbf{o}, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{W}$)

Input:
$O \in \mathbb{R}^{3\times N}$: the input point set of the shape with N points
$f(O)$: the IS-rep of $O, f(O) = \{f(\mathbf{p}_i)|i = 1, 2, \ldots, N\}$
$\mathbf{o}_0$: the initial center
$\mathbf{e}_0$: the initial first principal axis
$\mathbf{W}_0$: the initial weight set, $\mathbf{W}_0 = \{\omega_i'(0)|i = 1, 2, \ldots, N\}$
Output:
$\mathbf{o}$: the final center
$\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$: the final first, second and third principal axis
$\mathbf{W}$: the final weight set
Local variables:
$\mathbf{o}(k)$, $\mathbf{e}_1(k)$: the center and first principal axis in the iteration (k)
$\omega_i(k)$: the weight of $\mathbf{p}_i$ in the iteration (k)
ε_o, ε_θ: the pre-defined distance threshold and angle threshold
MAX_ITERS: the maximum number of iterative process
begin
1: $k \Leftarrow 0$, $\mathbf{o}(0) \Leftarrow \mathbf{o}_0$, $\mathbf{e}_1(0) \Leftarrow \mathbf{e}_0$;
2: **while** ($k <$ MAX_ITERS) **do**
3: Compute $\mathbf{o}(k+1)$ and $\mathbf{e}_1(k+1)$ via Eq. (15) and Eq. (16);
4: **for** ($i = 1$ to N) **do**
5: Compute $\omega_i'(k+1)$ via Eq. (13);
6: **end for**
7: **if** ($\|\mathbf{o}(k+1) - \mathbf{o}(k)\| < \varepsilon_o$ and the angle between $\mathbf{e}_1(k+1)$ and $\mathbf{e}_1(k)$ is smaller than ε_θ) **then**
8: **return**
9: **end if**
10: $k \Leftarrow k+1$;
11: **end while**
12: Determine the direction of $\mathbf{e}_1(k+1)$ via Eq. (17);
13: $\mathbf{W} \Leftarrow \{\omega_i'(k+1)|i = 1, \ldots, N\}$, $\mathbf{o} \Leftarrow \mathbf{o}(k+1)$, $\mathbf{e}_1 \Leftarrow \mathbf{e}_1(k+1)$;
14: Project O onto the plane through $\mathbf{o}$ and perpendicular to $\mathbf{e}_1$ and then perform a 2D case of this algorithm on these projection points to achieve the second and third principal axis $\mathbf{e}_2$ and $\mathbf{e}_3$.
end

5. Experimental results and applications

In this section we give some experimental results obtained by our normalization algorithm described in the previous section. Our algorithm is implemented in C++ on a Pentium Dual-Core 2.60 GHz processor with 3.0 GB RAM while the execution time is given in seconds excluding the time of loading the volumetric objects.

For the experimentation in this section, the parameters of the weight function presented in Eq. (13) are typically selected as $\lambda = \gamma = 1$. We also give an algorithm for determining adaptively the values of λ and γ with respect to the given shape variation in Section 6.1. The distance threshold and angle threshold in the termination condition in Algorithm 2 are defined as small values (e.g. $\varepsilon_e = 0.05$ and $\varepsilon_o = 0.01$), while the maximum iterative time MAX_ITERS $= 1000$ is used here to guarantee enough iterative time. Fig. 6 gives the normalization results of several 3D articulated shapes. It shows that our method performs better than the PCA based method.

5.1. Shape alignment

Shape alignment is usually a prerequisite step for some shape retrieval applications [12–14]. The intention is to put different shapes into a canonical coordinate frame. The shape alignment method based on PCA could easily cause similar local features to be misaligned due to its sensitivity to non-rigid transformations. We first normalize the 3D shapes to their corresponding frames

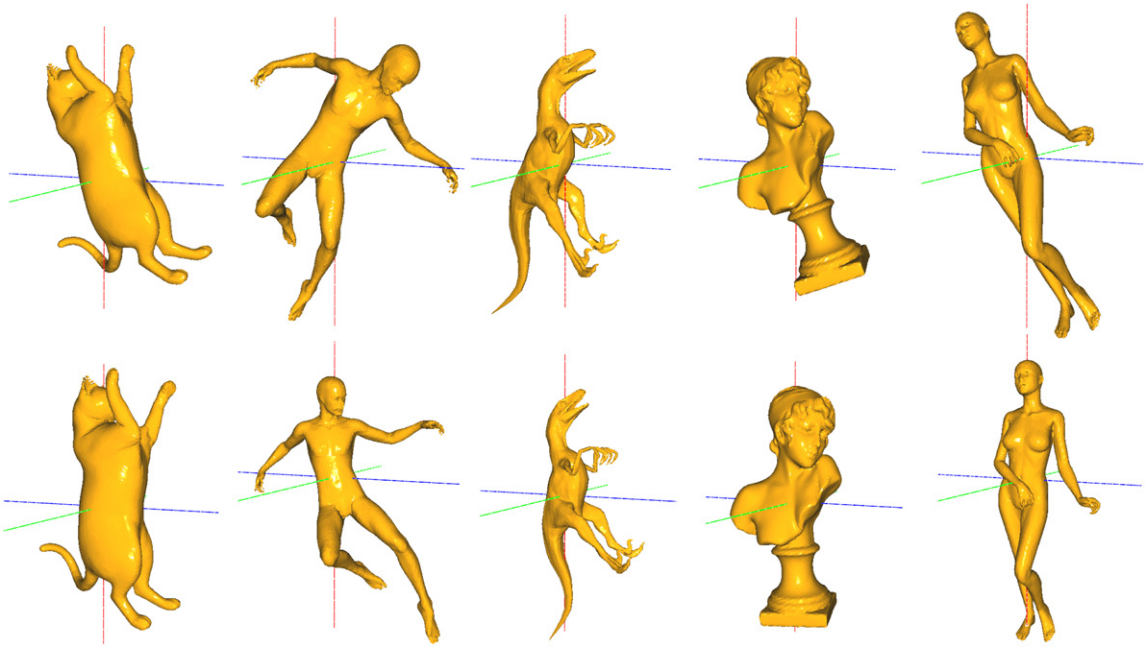

Fig. 6. The iso-surface display of normalization using PCA (first row) and our method (second row) for several 3D volumetric models. The red, green and blue axes are the first, second and third principal axes of the models, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

utilizing the robust normalization method and then compute the transformation between the two reference frames of shapes to align them together.

Fig. 7 shows the shape alignment results of six pairs of 3D models by PCA and our normalization method, where the models in the first column are the original ones and the models in the second column are similar to their respective original ones in the left but with articulation or some different partial details. The models in the third and forth columns give the shape alignment results by PCA and our normalization method, respectively. We can see that PCA cannot align the original pairs of models effectively. For example, the shape alignment result of two seahorses by PCA is not satisfactory in two aspects. One aspect is that the final localization and orientation of the red seahorse is quite different from that of the yellow seahorse. Another aspect is that the adjusted size of the red seahorse shape is obviously larger than the yellow seahorse, where the size adjustment of PCA is based on a shape's bounding box. Similar results also happen in the two examples of men and women models. The results in Fig. 7 suggest that our method is more effective than PCA in shape alignment applications.

5.2. Testing the robustness of shape normalization

Referring to the criteria of testing robustness of shape normalization carried out in [1,9], two kinds of parameters are computed in our experiments for the six pairs of models shown in Fig. 7. In Table 1, $C_{dif\,1}$ and $C_{dif\,2}$ represent the differences of distances between the two centers of shapes using PCA and our method, respectively. $\theta_{dif\,1}$ and $\theta_{dif\,2}$ denote the differences between the orientation estimations of two shapes, i.e. the angle between the two principal axes obtained by PCA and our method, respectively. From the experimental data comparisons in Table 1, we can see that our normalization method is effective and robust to deformations and performs much better than PCA.

The robustness of our method can also be recognized by its insensitivity to the noise which may exist in many scanned models.

Table 1
Results of PCA based method ($C_{dif\,1}$, $\theta_{dif\,1}$) and our normalization method ($C_{dif\,2}$, $\theta_{dif\,2}$) on the six pairs of 3D models in the first two columns in Fig. 7.

Models	Parameters			
	$C_{dif\,1}$	$C_{dif\,2}$	$\theta_{dif\,1}$	$\theta_{dif\,2}$
Candles	8.2789	2.7919	4.3701	1.0283
Cats	5.7227	1.6537	5.0653	1.2430
Guns	3.6882	0.1714	2.2324	0.5178
Men	12.6406	3.8904	5.5884	1.1314
Seahorses	10.8892	2.4551	6.4264	0.1695
Women	11.0924	2.9913	5.3174	1.1165

To test the ability of our method to handle the noise, we add the uniformly distributed random noise (along the normals with increasing variances of the diagonal of the bounding box of the seahorse model). Fig. 8 shows the normalization of the models with the different level noise. Fig. 9 illustrates the angle differences of the first principal axes between the original model and the noisy models. The results show that the first principal axis of the original model stands almost consistent with respect to the different level noise when using our method.

5.3. Computational time

The computational time of Algorithm 1 on the six 3D articulated volumetric models in the first column in Fig. 7 is given in Table 2, where "N" is the number of object points of the volumetric models, "T_1" is the time of computing the IS-rep at the start of Algorithm 1, "T_2" is the time of computing the center and principal axes in Algorithm 2, and "#iter" is the iterative number in Algorithm 2. All of the times are counted in seconds.

Although the employed threshold values $\varepsilon_e = 0.05$ and $\varepsilon_o = 0.01$ are small, the convergence of the algorithm is fast; in most cases, three to five iterations are required. The results in Table 2 show that the computational time increases when the number of object points of the volumetric models grows. Moreover, when the proportion of the number of inside points to the total number

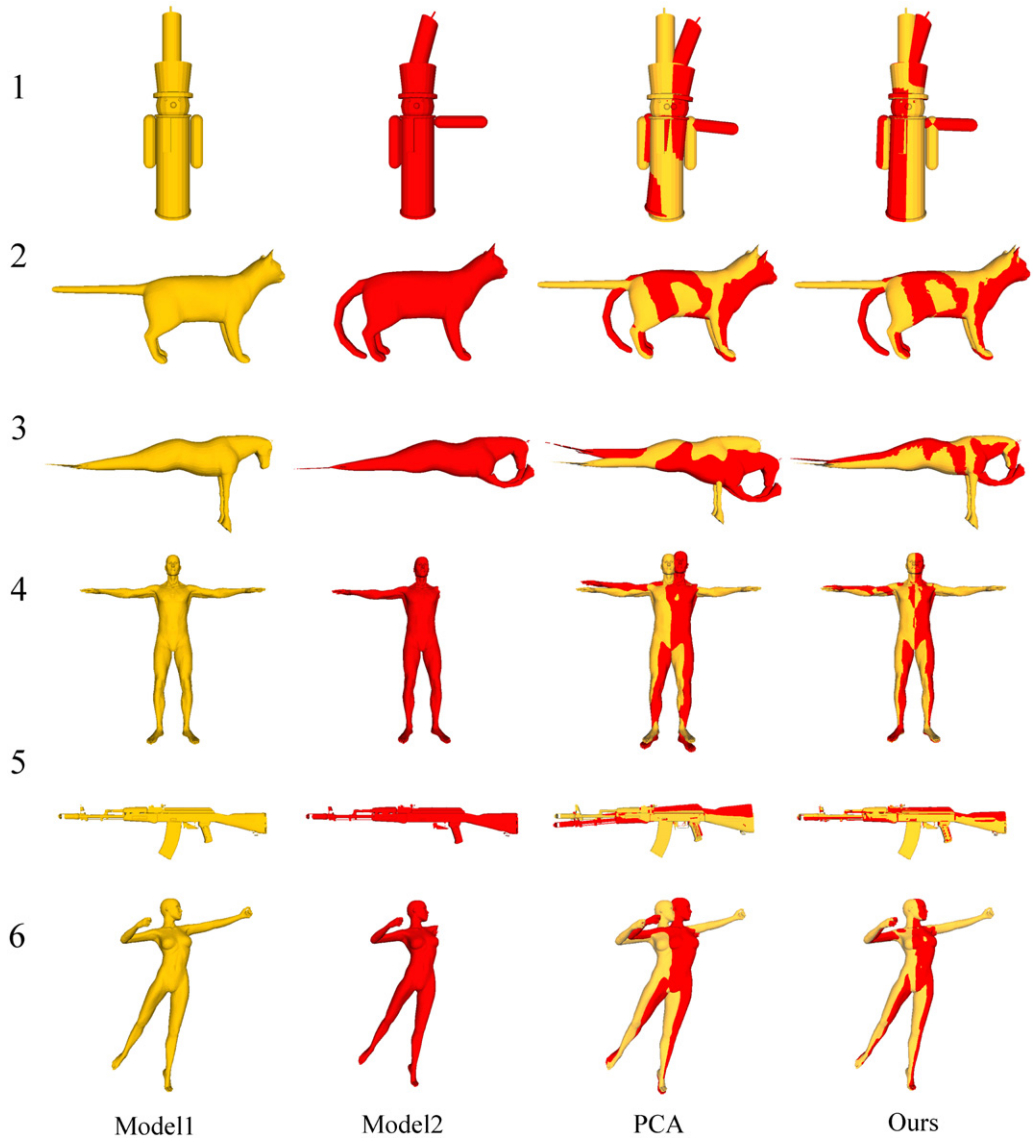

Fig. 7. The iso-surface models of shape alignment results by PCA and our normalization method by aligning six pairs of 3D models with articulated deformation or missing information.

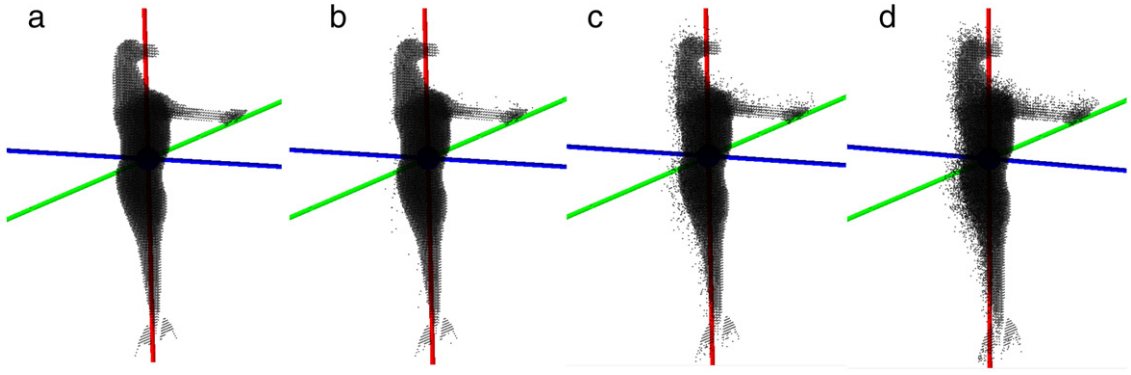

Fig. 8. Comparing the normalization of the models added with the noise along the normals with different variances. (a) shows normalization of the original seahorse model without adding the noise. (b), (c) and (d) show the normalization of the models added with the noise along the normals with 10%, 30% and 60% variances, respectively.

of points grows, the computational time of normalization also increases due to the time of the IS-rep calculation increasing quickly. The results in Table 2 suggest that the computational time is dependent on the number of object points of the models.

6. Discussions

This section first discusses the strategy of determining two representative parameters used in our method, and then gives an

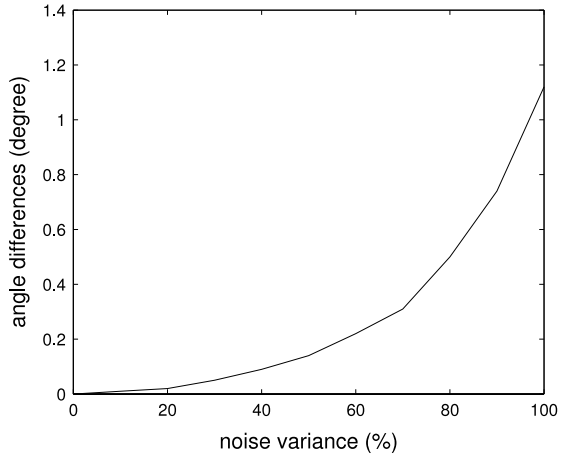

Fig. 9. The angle difference between the principal axes of the original model and the model with noise with respect to noise with increasing variances.

Table 2
The computational time of our normalization method on the six 3D volumetric models in the second column in Fig. 7.

Models	N[a]	T_1[b](s)	T_2[c](s)	#iter[d]
Candle	28 818	234.9509	0.6649	3
Cat	52 577	689.3381	1.6376	4
Gun	2 410	0.8691	0.0764	4
Man	18 231	102.0179	0.5704	4
Seahorse	12 977	50.2021	0.2971	3
Woman	13 533	56.0939	0.5339	5

[a] N: the number of object points of volumetric models.
[b] T_1: time to compute the IS-rep.
[c] T_2: time of the iterative process to compute the center and axes.
[d] #iter: the iterative number in Algorithm 2.

application for finding major regions of articulated shapes. Finally, we compare our method with some previous methods and discuss some limitations of our method.

6.1. Parameters

Five parameters (i.e. λ, γ, ε_o, ε_θ and MAX_ITERS) are defined in Algorithm 1 and Algorithm 2, and they play an important role in the normalization process. In the above five parameters, ε_o, ε_θ and MAX_ITERS in Algorithm 2 are to control the terminal condition of the iterative process, and their values are usually pre-defined, as described in Section 5. In contrast, the remaining two parameters λ and γ in Eq. (13) are relative to the shape variation, and their goal is to balance the contribution of the articulation insensitive weight function on all object points, which will affect computation of the center and principal axes during the iterative process. In this section, we mainly focus on the strategy of adaptively determining λ and γ as well as their effect to the final normalization results.

In Eq. (13), λ dominates the contribution of the implicit intensity value, while γ decides the contribution of the principal axis dependent weight function in Eq. (12). The former contribution could be approximately measured by counting the number of object points with large implicit intensity values, while the latter contribution could be approximately measured by counting the number of object points with the large weights computed initially in Eq. (12). Consequently, the central role of λ and γ is induced to balance the above two numbers counted, which are actually relative to the shape variation. Based on comparing the two numbers counted, we present a simple algorithm for adaptively determining the candidate values of two parameters (λ and γ) in Eq. (13). Algorithm 3, named "*ParaDetermination*", lists the pseudo-code of the algorithm.

Algorithm 3 : **ParaDetermination**($f(O)$, $\mathbf{W}$, λ, γ)

Input:
$f(O)$: the IS-rep of O, $f(O) = \{f(\mathbf{p}_i)|i = 1, 2, \ldots, N\}$
$\mathbf{W}_1$: the set of weights computed in Eq. (12), i.e. $\mathbf{W}_1 = \{\omega_i|i = 1, 2, \ldots, N\}$
Output:
λ, γ: the values of two parameters in Eq. (13)
Local variables:
$f_{\max}$, the maximum value of $f(O)$, i.e. $f_{\max} = \max_i\{f(\mathbf{p}_i)\}$
ω_{max}: the maximum value of $\mathbf{W}_{axis}$, i.e. $\omega_{\max} = \max_i\{\omega_i\}$
N_1: the number of points with weights within $[\frac{\omega_{\max}}{2}, \omega_{\max}]$
N_2: the number of points with implicit intensity values within $[\frac{f_{\max}}{2}, f_{\max}]$
Begin
1: Compute $f_{\max}$ and $\omega_{\max}$;
2: Compute N_1 based on $\mathbf{W}_1$ and $\omega_{\max}$, and compute N_2 based on $f(O)$ and $f_{\max}$;
3: **if** $N_2 > N_1$ **then**
4: $\lambda \Leftarrow 1, \gamma \Leftarrow \lfloor \frac{N_2}{N_1} \rfloor$;
5: **else**
6: $\gamma \Leftarrow 1, \lambda \Leftarrow \lfloor \frac{N_1}{N_2} \rfloor$;
7: **end if**
End

Algorithm 3 takes the IS-rep and the initial weight set $\mathbf{W}_1$ using Eq. (12) as input. The algorithm first computes the maximum value $f_{\max}$ of all the implicit intensity values and the maximum weight $\omega_{\max}$ of $\mathbf{W}_1$. Then, Algorithm 3 computes N_1 and N_2, which respectively denote the number of points with the weights that are larger than half of $\omega_{\max}$, and the number of points with implicit values that are larger than half of $f_{\max}$. By comparing the values of N_1 and N_2, Algorithm 3 finally gives a choice of λ and γ.

Fig. 10 illustrates the effectiveness of different combinations of λ and γ to the final weights. We can find that λ and γ are determined using Algorithm 3 giving a more desirable weight distribution (see Fig. 10(b)).

6.2. Application on finding major regions

One direct application of our normalization method is to find the major regions of articulated models using the final weights produced. The similar application also appeared in Ref. [12]. The major region of an articulated model is assumed to contain the points mainly nearby the first principal axis and holds the main body of the model (e.g. the trunk of a human model). Consequently, the minor regions contain the remaining points outside the major region, and they are relatively far from the first principal axis and usually prone to deformation (e.g. the arms or legs of a human model).

Algorithm 4, called *FindMajorRegion*, shows the outline implementation of this method. The algorithm takes the final weights of object points, obtained by our normalization method, as inputs, and outputs the major region point set. By sorting the weights of all object points, the procedure divides the points into n_{bins} equally spaced bins, where n_{bins} is a predefined positive integer. Then the points with the weight values in the smallest k bins are regarded as the minor region, while the other points with the larger weight values are regarded as the major region. The time complexity is $O(N \log N)$ because this algorithm only consists of a sort step and a loop to classify data.

Fig. 11 shows the results of finding the major regions (blue) and minor regions (yellow) on four different poses of a seahorse model using Algorithm 4. We typically use $n_{\text{bins}} = 10, k = 2$, $\lambda = 1$ and $\gamma = 3$ in this experiment. The corresponding ratios of

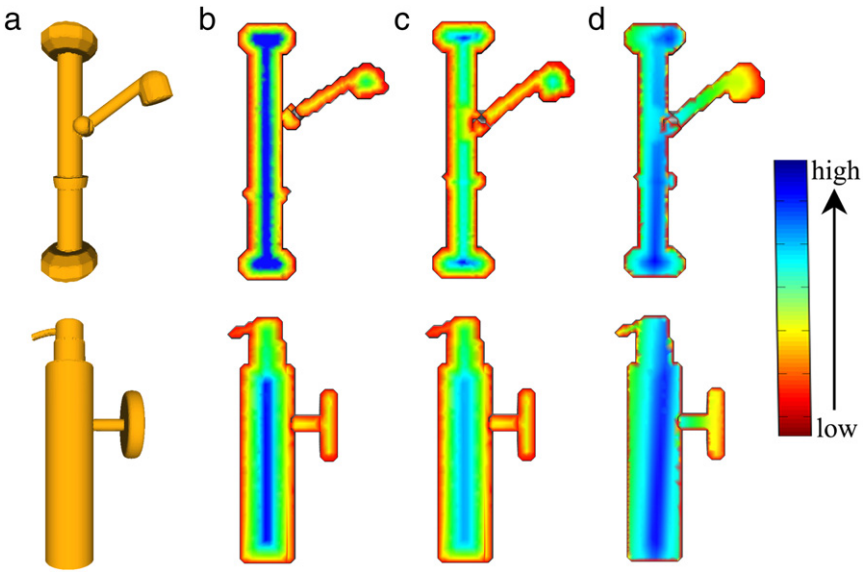

Fig. 10. Illustrating the effectiveness of different combinations of λ and γ to the final weights. (a) The original iso-surface models (a shower-like one and a sprayer-like one). (b) Cross-sections with $\lambda = 1$ and $\gamma = 1$. (c) $\lambda = 5$ and $\gamma = 1$. (d) $\lambda = 1$ and $\gamma = 5$. Note that the two parameters in (b) are adaptively determined using Algorithm 3, while the parameters in (c) and (d) are manually set to be compared with (b). All of the weights in this figure are computed using our weight function in Eq. (13).

Algorithm 4 : **FindMajorRegion**(O, **W**, n_{bins}, O_{major}, O_{minor})

Input:
O: the input 3D volumetric model
W: the weight set of all the final weight functions
n_{bins}: the number of all bins
k: the number of bins selected for minor regions
Output:
O_{major}: the point set of major region of O
O_{minor}: the point set of minor region of O
Local variables:
ω_{max}: the maximum value of **W**
ω_{min}: the minimum value of **W**
Begin
1: $O_{major} \Leftarrow \varnothing$, $O_{minor} \Leftarrow \varnothing$;
2: Sort the weight values of **W** and get ω_{max} and ω_{min};
3: **for** ($i = 1$ to N) **do**
4: **if** $\omega_i > \omega_{min} + k\frac{\omega_{max}-\omega_{min}}{n_{bins}}$ **then**
5: $O_{major} \Leftarrow O_{major} + \mathbf{p}_i$;
6: **else**
7: $O_{minor} \Leftarrow O_{minor} + \mathbf{p}_i$;
8: **end if**
9: **end for**
End

the number of major region points to the number of whole model points from Fig. 11(a)–(e) are 80.2%, 80.9%, 80.5%, 78.3% and 75.0%, respectively. Note that in Fig. 11(e), parts of the tail of the seahorse are divided into separate parts from the major region. Unlike the tails from Fig. 11(a)–(d), the tail in Fig. 11(e) is rotated away from the principal axis of the shape and is therefore considered as a separate part. Despite the situation in Fig. 11(e), the major regions in the five figures are almost consistent.

6.3. Comparisons with some previous works

This section compares our work with some previous works. Considering that shape alignment is one of the most important applications of normalization methods and is usually used to check the effectiveness of these methods, we will compare our method with some previous methods through shape alignment. Several different types of robust statistical methods, such as Iterative Closest Point (ICP) [15], Least Trimmed Squares (LTS) [43] and the normalization method by Jiang et al. [9], have improved the classical PCA, in which all methods could be utilized on shape alignment.

ICP is a widely used geometric alignment technique to match two similar models. It starts with two input shapes represented by point clouds. An initial guess is made for the relative rigid transformation, and ICP iteratively refines the transformation by repeatedly generating pairs of corresponding points on the shapes through minimizing an error metric. One limitation is that ICP is only effective on rigid shape alignment, it is not suitable for non-rigid shapes. Another limitation of ICP and its variants is that, as a local optimization method, it is not guaranteed to find the globally optimal alignment. Therefore, ICP is only effective when the initial position of the input rigid shapes is close to the correct shape alignment [15,44]. Our robust normalization method can be expected to obtain a better initial position and improve the robustness of the ICP step for articulated shape alignment.

Another choice for improving the PCA method is LTS [43] for the principal axis computation. LTS first fits the whole data using an ordinary least squares. Then LTS re-fits the remaining data and identifies those points with the largest residuals and discards them for the final principal axis computation. LTS has a similar disadvantage like LMS in that no consideration is taken for the effect of the periphery on the articulated shape.

Jiang et al. [9] introduced a shape normalization method based on the IS-rep without the iterative process. It uses the IS-rep to design an intensity function to obtain a center and principal axis of a 2D shape. Although the method is designed for deformable shapes, its robust-to-deformation feature is only reflected in the experimental results and has not been demonstrated rigorously. Moreover, the final principal axis obtained by this method counts on a good initial estimation, which may be deflected from the real position significantly with deformation exists. For the purpose of comparison, we implemented Jiang's method by extending their method from 2D images to 3D volumetric models.

Fig. 12 shows comparisons of the shape alignment results of two pairs of 3D models using four methods: ICP, LTS, Jiang's method and our method. Note that ICP is affected by the initial relative positions of the two deformable shapes and LTS is easily influenced by some

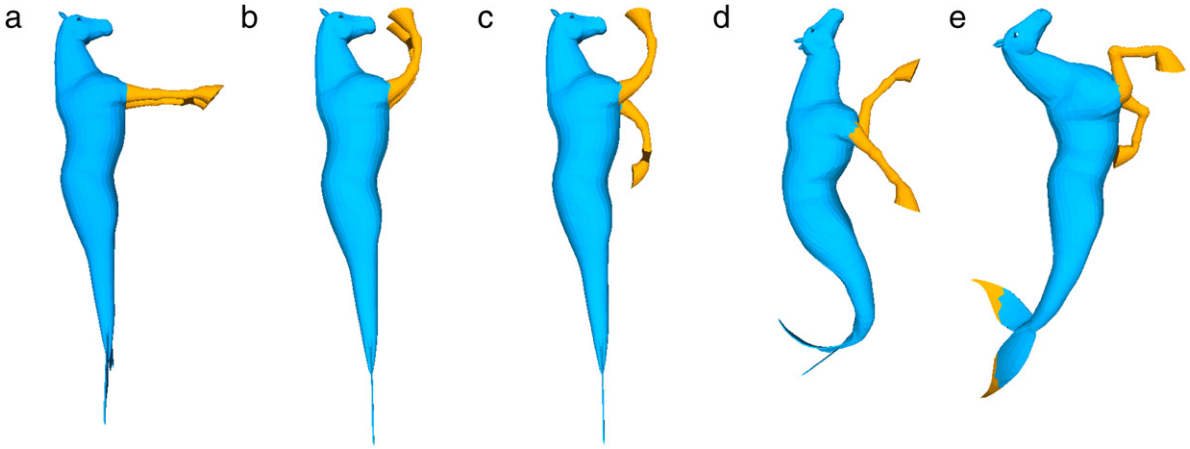

Fig. 11. The application of finding the major regions (blue) for several deformations of a seahorse model by Algorithm 4. The corresponding ratios of the number of major region points to the number of whole model points are as follows: (a) 80.2%. (b) 80.9%. (c) 80.5%. (d) 78.3%. (e) 75.0%. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

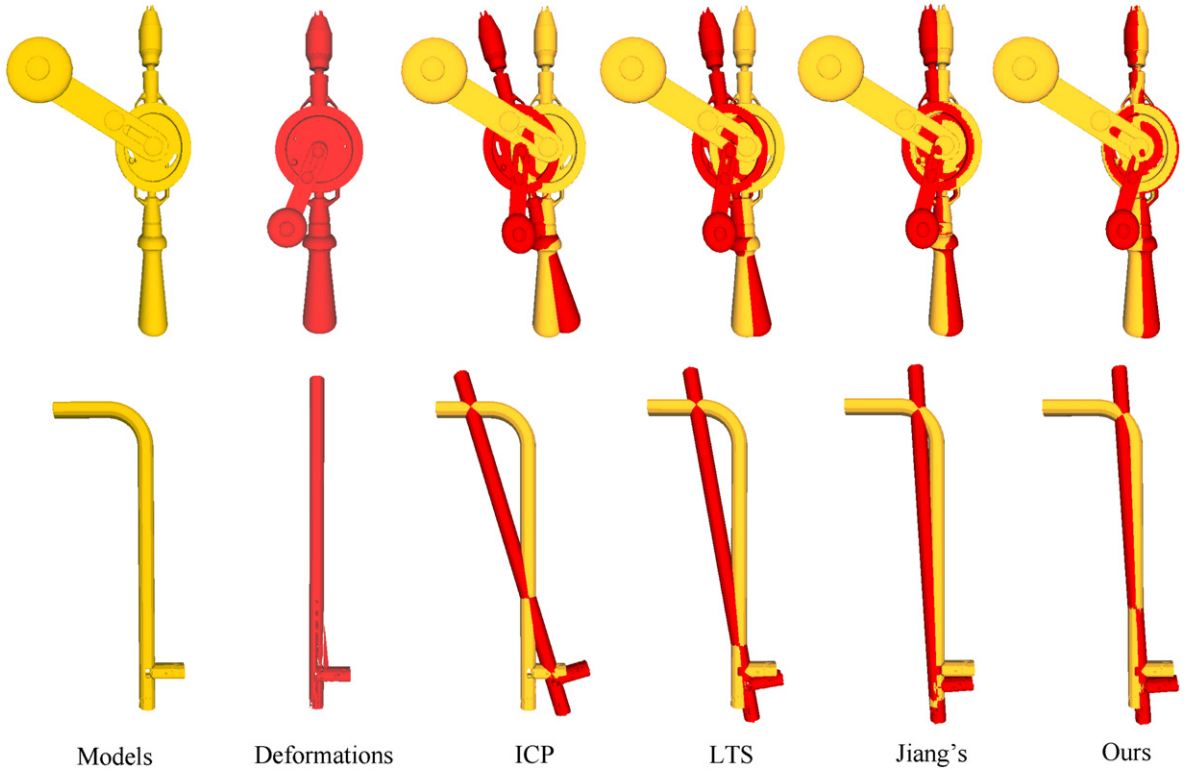

Fig. 12. The iso-surface models of comparisons between four shape normalization methods by aligning two pairs of deformable volumetric models.

deformable regions connecting the major rigid part, while Jiang's method is better than the former two methods. Meanwhile, our method is the best of the four and better than Jiang's method due to the step of iterative refinement.

6.4. Limitations

The normalization method introduced in this paper could be used in shape alignment, finding the major regions and some other applications, such as shape retrieval, shape orientation and prior viewpoints [4,8,10,12]. However, some limitations also exist in our current implementation. Firstly, when the orientation of a shape is not obvious, or when a shape undergoes a very large extent of deformation, our method may produce undesirable results. Fig. 13 illustrates some undesirable normalization examples. The models without obvious orientations are very hard to be normalized because their principal axes do not exist or are very hard to determine [2–4]. In fact, due to the variety of shapes as well as the diversity of applications, there is probably no single method for computing shape orientation that could be efficiently and successfully applicable to all shapes [4]. Secondly, the values of two parameters λ and γ in Eq. (13) will not be suitable for some special shapes, which will in turn influence the final normalization results. For instance, when it comes to normalize a shape with some "fat" branches and "thin" main body, our method may bring in undesirable results because the values of λ and γ obtained by Algorithm 3 are likely to be inappropriate. This is one of our future works, improving the choice of parameters with respect to variant shapes.

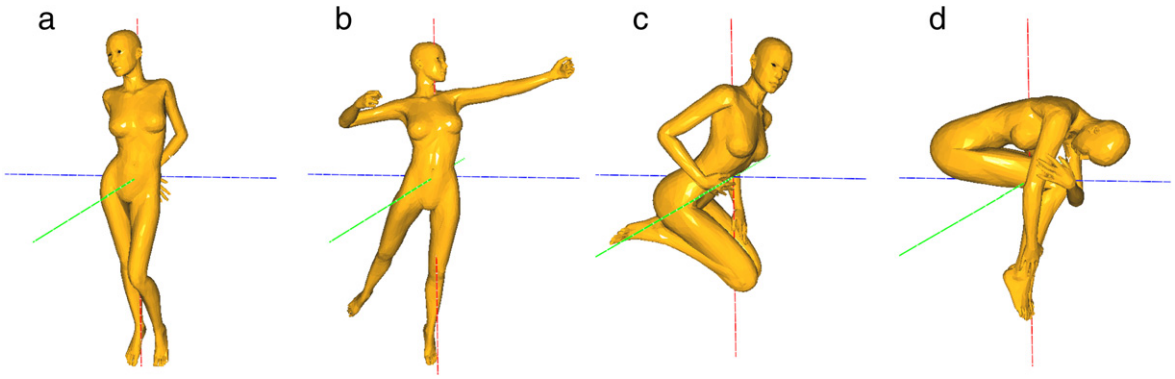

Fig. 13. The iso-surface display of normalization results of some women volumetric models by our method. Note that (c) and (d) are not normalized well because their orientations are not quite obvious.

7. Conclusions

We have developed a new robust shape normalization method for 3D articulated objects. Our framework is similar to Cortadellas et al.'s normalization [1], whereas we extend it to 3D articulated volumetric models. Our main contributions are to combine the implicit shape representation (IS-rep) with the framework and give a theoretical proof for guaranteeing that the IS-rep is robust to articulated deformations. The main advantage of the IS-rep is to produce a natural articulation insensitive weight function to reduce the influence of articulated deformation during the normalization computation. Our normalization method is robust with respect to articulated deformation without requiring any prior shape decomposition. The experimental results demonstrate that our method can normalize 3D volumetric shapes better than some previous works used to align two articulated shapes.

Some works can be considered for extending our current work as a future direction of research. Firstly, the current implementation of our method is only developed for 3D volumetric models, and we plan to extend our method on models with other formats such as 3D meshes in the future. Secondly, some other applications based on our method will be investigated, such as automatically selecting a good viewpoint or some relative applications in shape retrieval. The articulation-insensitivity feature of the IS-rep demonstrated in our work could be extended to construct a 3D shape descriptor used as an index in a database of shapes, which further enables fast query and retrieval.

Acknowledgments

The research was supported by the National Science Foundation of China (61003095), the National Technological Support Program for the 12th-Five-Year Plan of China (2012BAJ03B07), the Chinese 863 Program (2012AA040902), and the Chinese 973 Program (2010CB328001). J-C P was supported by the ANR-NSFC (60911130368).

References

[1] Cortadellas J, Amat J, de la Torre F. Robust normalization of silhouettes for recognition applications. Pattern Recognition Letters 2004;25(5):591–601.
[2] Žunić J, Rosin PL. An alternative approach to computing shape orientation with an application to compound shapes. International Journal of Computer Vision 2009;81(2):138–54.
[3] Žunić J, Rosin PL, Kopanja L. On the orientability of shapes. IEEE Transactions on Image Processing 2006;15(11):3478–87.
[4] Žunić J, Stojmenović M. Boundary based shape orientation. Pattern Recognition 2008;41(5):1768–81.
[5] Leu J-G. Shape normalization through compacting. Pattern Recognition Letters 1989;10(4):243–50.
[6] Pei SC, Lin CN. Image normalization for pattern recognition. Image and Vision Computing 1995;13(10):711–23.
[7] Shen D, Ip HHS. Generalized affine invariant image normalization. IEEE Transactions on Pattern Analysis and Machine Intelligence 1997;19(5):431–40.
[8] Martinez-Ortiz C, Žunić J. Curvature weighted gradient based shape orientation. Pattern Recognition 2010;43(9):3035–41.
[9] Jiang T, Tomasi C. Robust shape normalization based on implicit representations. In: International conference on pattern recognition. ICPR. 2008. p. 1–4.
[10] Fu H, Cohen-Or D, Dror G, Sheffer A. Upright orientation of man-made objects. ACM Transactions on Graphics 2008;27(3).
[11] Liu H, Yan J, Zhang D. What is wrong with mesh PCA in coordinate direction normalization. Pattern Recognition 2006;39(11):2244–7.
[12] Liu Y-S, Ramani K. Robust principal axes determination for point-based shapes using least median of squares. Computer-Aided Design 2009;41(4):293–305.
[13] Funkhouser T, Min P, Kazhdan M, Chen J, Halderman A, Dobkin D, Jacobs D. A search engine for 3D models. ACM Transactions on Graphics 2003;22(1):83–105.
[14] Passalis G, Theoharis T, Kakadiaris IA. PTK: a novel depth buffer-based shape descriptor for three-dimensional object retrieval. The Visual Computer 2007;23(1):5–14.
[15] Gelfand N, Mitra NJ, Guibas L, Helmut P. Robust global registration. In: Proceedings of eurographics symposium on geometry processing. SGP'05. 2005. p. 197–206.
[16] Saleem W, Wang D, Belyaev A, Seidel H-P. Automatic 2D shape orientation by example. In: International conference on shape modeling and applications. SMI'07. 2007. p. 221–5.
[17] Pu J, Ramani K. A 3D model retrieval method using 2D freehand sketches. Lecture Notes in Computer Science 2005;3515:343–7.
[18] Bribiesca E. Measuring 3-D shape similarity using progressive transformations. Pattern Recognition 1996;29(7):1117–29.
[19] Galvez JM, Canton M. Normalization and shape recognition of three-dimensional objects by 3D moments. Pattern Recognition 1993;26(5):667–81.
[20] Sánchez-Cruz H, Bribiesca E. A method of optimum transformation of 3D objects used as a measure of shape dissimilarity. Image and Vision Computing 2003;21(11):1027–36.
[21] Duda RO, Hart PE, Stork DG. Pattern classification. 2nd ed. John Wiley & Sons; 2001.
[22] Raviv D, Bronstein AM, Bronstein MM, Kimmel R. Full and partial symmetries of non-rigid shapes. International Journal of Computer Vision 2010;89(1):18–39.
[23] Lian Z, Rosin PL, Sun X. Rectilinearity of 3D meshes. International Journal of Computer Vision 2010;89(2–3):130–51.
[24] Paragios N, Rousson M, Ramesh V. Non-rigid registration using distance functions. Computer Vision and Image Understanding 2003;89:142–65.
[25] Paragios N, Taron M, Huang X, Rousson M, Metaxas D. On the representation of shapes using implicit functions. In: Statistics and analysis of shapes. Springer; 2006 [Chapter].
[26] Fang Y, Liu Y-S, Ramani K. Three dimensional shape comparison of flexible protein using the local-diameter descriptor. BMC Structural Biology 2009;9(29).
[27] Liu Y-S, Fang Y, Ramani K. IDSS: deformation invariant signatures for molecular shape comparison. BMC Bioinformatics 2009;10(157).
[28] Liu Y-S, Ramani K, Liu M. Computing the inner distances of volumetric models for articulated shape description with a visibility graph. IEEE Transactions on Pattern Analysis and Machine Intelligence 2011;23(12):2538–44.
[29] Liu Y-S, Yi J, Zhang H, Zheng G, Paul J-C. Surface area estimation of digitized 3D objects using quasi-Monte Carlo methods. Pattern Recognition 2010;43(11):3900–9.
[30] Ju T. Robust repair of polygonal models. ACM Transactions on Graphics 2004;23(3):888–95.

[31] Min Patrick. Binvox: 3D mesh voxelizer. http://www.cs.princeton.edu/~min/binvox/.
[32] Bronstein AM, Bronstein MM, Kimmel R. Numerical geometry of non-rigid shapes. Springer; 2007.
[33] Bronstein AM, Bronstein MM, Kimmel R, Mahmoudi M, Sapiro G. A Gromov–Hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. International Journal of Computer Vision 2010;89(2–3):266–86.
[34] Bronstein AM, Bronstein MM, Bruckstein AM, Kimmel R. Analysis of two-dimensional non-rigid shapes. International Journal of Computer Vision 2008; 78(1):67–88.
[35] Ling H, Jacobs D. Shape classification using the inner-distance. IEEE Transactions on Pattern Analysis and Machine Intelligence 2007;29(2): 286–99.
[36] Huang X, Paragios N, Metaxas D. Shape registration in implicit spaces using information theory and free form deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(8):1303–18.
[37] Sethian JA. Level set methods and fast marching methods. Cambridge University Press; 1999.
[38] Huang X, Metaxas D. Metamorphs: deformable shape and appearance models. IEEE Transactions on Pattern Analysis and Machine Intelligence 2008;30(8): 1444–59.
[39] Yatziv L, Bartesaghi A, Sapiro G. $O(n)$ implementation of the fast marching algorithm. Journal of Computational Physics 2006;212:393–9.
[40] Rustamov RM, Lipman Y, Funkhouser T. Interior distance using barycentric coordinates. Computer Graphics Forum 2009;28(5):1279–88.
[41] Coleman D, Holland P, Kaden N, Klema V, Peters SC. A system of subroutines for iteratively reweighted least squares computations. ACM Transactions on Mathematical Software 1980;6(3):327–36.
[42] O'Leary DP. Robust regression computation using iteratively reweighted least squares. SIAM Journal on Matrix Analysis and Applications 1990;11(3): 466–80.
[43] Pighin F, Lewis JP. Practical least-squares for computer graphics. In: ACM SIGGRAPH'07 courses. 2007. p. 1–57.
[44] Mitra NJ, Gelfand N, Pottmann H, Guibas L. Registration of point cloud data from a geometric optimization perspective. In: Proceedings of eurographics symposium on geometry processing. 2004. p. 23–32.