Vis Comput
DOI 10.1007/s00371-009-0383-8

ORIGINAL ARTICLE

Protrusion-oriented 3D mesh segmentation

Alexander Agathos · Ioannis Pratikakis · Stavros Perantonis · Nickolas S. Sapidis

Abstract In this paper, we present a segmentation algorithm which partitions a mesh based on the premise that a 3D object consists of a core body and its constituent protrusible parts. Our approach is based on prominent feature extraction and core approximation and segments the mesh into perceptually meaningful components. Based upon the aforementioned premise, we present a methodology to compute the prominent features of the mesh, to approximate the core of the mesh and finally to trace the partitioning boundaries which will be further refined using a minimum cut algorithm. Although the proposed methodology is aligned with a general framework introduced by Lin et al. (IEEE Trans. Multimedia 9(1):46–57, 2007), new approaches have been introduced for the implementation of distinct stages of the framework leading to improved efficiency and robustness. The evaluation of the proposed algorithm is addressed in a consistent framework wherein a comparison with the state of the art is performed.

Keywords Mesh segmentation · Prominent feature extraction · Core approximation

A. Agathos (✉) · I. Pratikakis · S. Perantonis
Computational Intelligence Laboratory, Institute of Informatics and Telecommunications, NCSR 'Demokritos', Athens, Greece
e-mail: agalex@iit.demokritos.gr

I. Pratikakis
e-mail: ipratika@iit.demokritos.gr

S. Perantonis
e-mail: sper@iit.demokritos.gr

N.S. Sapidis · A. Agathos
Department of Product and Systems Design Engineering, University of the Aegean, Mytilene, Greece

N.S. Sapidis
e-mail: sapidis@aegean.gr

1 Introduction

3D mesh segmentation is the process which separates a mesh into disjoint connected components. It is an important process for many applications in Computer Graphics like metamorphosis [18], compression [17], 3D shape retrieval [24], 3D object recognition [19], texture mapping [11], etc.

As previously discussed in [1, 15] there are mainly two types of segmentation. In the first, namely, surface-based, the mesh is segmented into patches for which either basic primitives can be fitted or various criteria required by the application should be satisfied. In the second, namely part-based, the mesh is divided into volumetric parts that are perceptually meaningful.

In this paper, we will present a new part-based segmentation algorithm which is based on the premise that a 3D object consists of a core body and its constituent protrusible parts. For example, the proposed algorithm segments a 3D object representing a human figure into the main body along with the head, hands and legs (see Fig. 1).

The contribution of this paper is twofold:

i. A novel way to trace the partitioning boundaries of the 3D object using closed boundaries constructed with the aid of a distance function.
ii. A novel algorithm for the core approximation of the 3D object.

The proposed methodology leads to meaningful segmentation results that are proven after evaluation which is addressed in a consistent framework wherein a comparison with the state of the art is performed.

This paper is organized as follows. Section 2 discusses the related work. Section 3 is dedicated to the detailed description of the proposed methodology. In Sect. 4, the ex-

Published online: 13 August 2009

Fig. 1 Segmentation of a human figure into its meaningful constituent parts. Each part is shown with a different color

perimental results are discussed while in Sect. 5 conclusions are drawn.

2 Related work

As discussed in [1], 3D mesh segmentation algorithms can be categorized according to the core methodology used, namely: (i) Region growing; (ii) Watershed-based; (iii) Reeb graphs; (iv) Model-based; (v) Skeleton-based; (vi) Clustering; (vii) Spectral analysis; (viii) Explicit boundary extraction; (ix) Critical points-based; (x) Multiscale shape descriptors; (xi) Markov random fields, and (xii) Direct segmentation.

These categories consist of algorithms that produce segmentation in either a surface-based or a part-based fashion. In this paper we will focus on part-based algorithms which can be further classified based upon the underlying concept used. The most commonly used concepts comprise (i) the minima rule; (ii) the connection of the shape's volume to the mesh surface; (iii) hierarchical clustering; (iv) skeleton-driven segmentation; (v) salient point and core estimation.

The minima rule concept was introduced by Hoffman and Richards [5] and relies on the principle wherein an object is segmented by human perception at areas of concavity.

Zhang et al. [23] use the minima rule and construct a region growing algorithm. Seed vertices that do not belong on concavities are expanded creating regions that are surrounded by deep concavities. Their algorithm is able to segment meshes into volumetric parts provided that the boundaries of the part contain deep concavities.

Page et al. [14] try to conform to the minima rule with a algorithm. In this, a watershed-based algorithm a topographic surface, which is defined using a height function, is flooded. At the points where the flooded regions meet, watershed lines are created signifying the boundaries of distinct parts. The normal curvature of the surface of the object is used as the height function.

Wu and Levine [21] take also advantage of the minima rule and construct a model-based algorithm. Specifically, they create a surface model of the mesh and apply the theory of the electrical charge distribution across the surface and use its correlation with the minima rule, i.e. the density charge is very low at areas of deep concavities and high at the convexities of the surface. Thus, by tracing the local minima in the electrical charge density the boundaries of the parts can be traced.

The drawback of the above three algorithms is that the boundaries must contain concavities. Unfortunately, this is not often the case for many objects (e.g. the boundary between the arm and the main body in a human).

So far, from the aforementioned segmentation algorithms that use the minima rule it can be seen that all of them localize at the concavities of the mesh in order to segment without using any other information concerning the shape of the mesh.

Lee et al. [10] use the minimum curvature of the mesh in order to construct the boundaries of its parts. The main advantage of their approach is that they manage to construct closed boundaries of the parts even if they are not surrounded by deep concavities. This is achieved by an efficient shortest path algorithm, which drives the open boundary along the mesh so as to close in a perceptually meaningful manner. Although their algorithm manages to overcome the main problem that the other three aforementioned algorithms have, there exist cases in their algorithm where some extracted boundaries do not partition the mesh into perceptually meaningful parts.

Katz and Tal [7] use a combination of geodesic distance and concavity information in order to create clusters which contain perceptually coherent regions. Since concavity is considered in their clustering method, they also conform to the minima rule. However, the clustering dependence on the concavity information reduces robustness on a mesh with a lot of geometric texture.

Zhang and Liu [22] find two faces of the mesh that belong to two perceptually different parts and apply a Normalized Cut Algorithm like in [17] in order to segment the mesh at deep concavities into two parts, thus conforming to the minima rule. They apply their method recursively in order to acquire a full decomposition of the object. As in [7], their algorithm does not work well on a mesh with a lot of geometric texture. Also, in the recursive decomposition they apply, it is not possible to define a meaningful hierarchy as

small parts of the object might be segmented prior to bigger more salient parts.

Another concept which appears in mesh segmentation methodologies is based on the connection of the shape's volume to the mesh surface. Such a concept was presented by Kim et al. [9], who created a volumetric method to partition the object. Their aim is to extract the convex parts of the object. This is achieved by first doing a voxelization of the mesh and then maximizing the weighted convexity of the parts produced after a morphological opening on the voxelized grid with a sphere as a shape element.

Also, in Shapira et al. [16] a volumetric function is created which is called the Shape Diameter Function. This function is defined on the vertices of the mesh and equals to the average of the distances of a set of rays cast from the surface point in opposite directions from its normal. They use this function in a clustering algorithm in order to create a hierarchical characterization of the mesh which is later used by a k-way graph-cut algorithm to separate smoothly the distinct parts.

The idea behind hierarchical clustering lies upon edge contraction on the dual graph of the mesh [1]. With each edge contraction in the dual graph a new node on the hierarchical structure is generated, whose children are the nodes incident to the edge and the two areas that represent the children nodes are grouped into their parent common cluster. The edge contraction cost is application-driven.

Attene et al. [3] uses the hierarchical clustering concept to classify the mesh triangles into clusters that can be approximated by either a plane or sphere or cylinder primitive. The contraction cost of a dual edge is the minimum of the errors generated after fitting the plane, sphere and cylinder primitives to the triangles of the merged regions. The primitive corresponding to the minimum fitting error approximates the unified cluster.

The skeleton-driven segmentation concept uses the skeleton of the 3D object in order to extract the meaningful parts of the 3D object.

Li et al. [12] segment the mesh using the skeleton of the object. The skeleton of the object is constructed by performing simplification of the surface using the edge contraction method. The priority for contraction is the length of the edge to be contracted. Their simplification process leaves edges that do not have incident triangles unaltered; they call these edges skeletal. At the end of the contraction process only skeletal edges are left, whose union is the skeleton of the object. The partitioning boundaries of the object are extracted by using a plane which sweeps the mesh along the skeleton edges. The intersection of the plane and the mesh is one or more polygon contours. The segmentation areas are extracted by examining the way that these contours alter as the sweep plane moves. For this purpose, a parametric geometric and topological function is defined on these contours.

The concept that takes into account the salient points (prominent features residing at the extrema of the mesh) and core estimation aims to the extraction of other individual parts that do not belong to the main body (core) of the mesh. In Katz et al. [8], multidimensional scaling is used in order to create a pose invariant representation of the object. This representation aids in the computation of the salient points of the surface and then the salient points along with the representation are used in order to extract the core by spherical mirroring. Their algorithm requires to compute a coarse model (approximately 1000 faces) of the mesh using a simplification algorithm in order to run in a reasonable time and not to overflow the computer's memory.

In Valette et al. [20], the principle of the protrusion function [1] is used to express the degree of closeness of a vertex to a protrusible part. Specifically, the evolution of the protrusion function from its high towards its low values is used in order to extract the protrusions of the mesh and separate them from the main body. Although the protrusion function does not have pose sensitivity, their algorithm does (see Sect. 4). Furthermore, the segmentation boundaries are usually not extracted on the areas where human perception would naturally do.

Lin et al. [13] find also the salient points of the mesh and a core approximation. Each of the salient points create geodesic zones which are called locales. These locales are used to define a border function which identifies those locales containing the boundary of the protrusion. The main drawback with their approach is that some of their parameters are not set fixed and depend on the locales which in turn depend on the resolution of the mesh. Also, their core approximation relies on a fixed threshold which can significantly overestimate or underestimate the core of the mesh.

In this paper, the proposed algorithm is based on the salient point and core estimation concept. It efficiently makes an approximation of the main body of the 3D object while the partitioning boundaries are found using sweeping closed boundaries which, first, do not depend on the resolution of the mesh as the locales do in [13] and, second, are far less susceptible to noise.

3 The proposed methodology

Our objective is to segment a 3D mesh into its meaningful components. To this end, we propose a methodology which consists of the consecutive steps in the following.

i. The salient points of the mesh are extracted. These points are the extrema of the mesh and characterize its protrusions.
ii. The salient points extracted in step (i) are grouped according to their geodesic proximity.

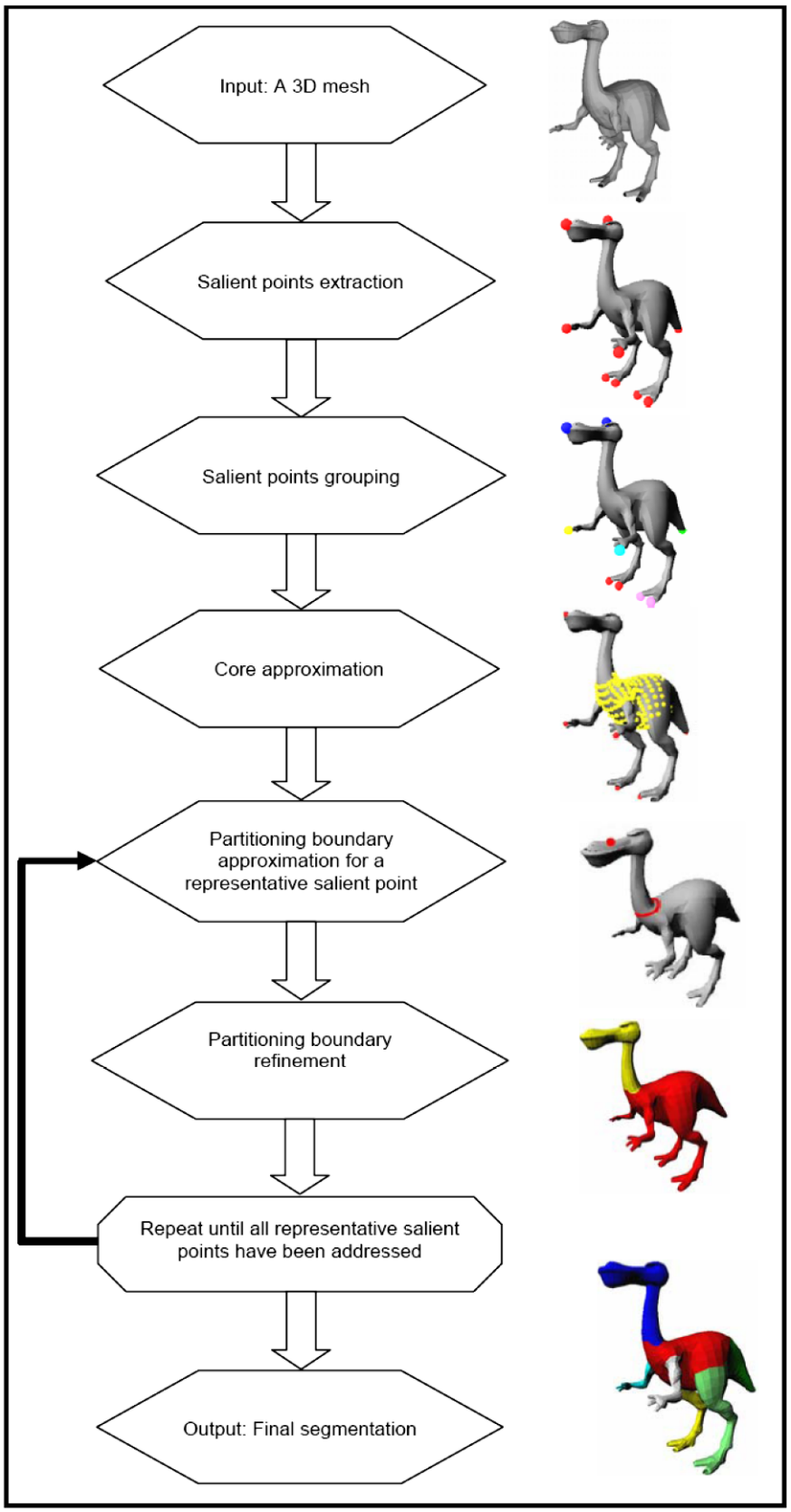


Fig. 2 The flowchart of the proposed 3D mesh segmentation methodology

iii. An approximation of the core (main body) of the mesh is found using the grouped salient points from step (ii).
iv. The partitioning boundaries are detected using closed boundaries along the protrusions constructed by a distance function.
v. The detected partitioning boundaries are refined.

The flowchart showing the steps which will be followed by the proposed methodology is shown in Fig. 2.

A detailed description of the aforementioned steps is given in the following sections.

3.1 Salient points—extraction stage

In this section, we describe the process of finding the salient points of the 3D object.

In order to realize this process, a function is going to be used which has the property of having low values at the center of the 3D object and high values at its protrusions [4]. This function is called in [1] *protrusion function, pf*().

Formally, for each point v of the surface S of a 3D object, the protrusion function is defined as:

$$pf(v) = \int_{p \in \mathrm{S}} g(v, p)\, \mathrm{dS} \tag{1}$$

where $g(v, p)$ denotes the geodesic distance between v and p [4]. Usually, this function is approximated on a 3D mesh by a tessellation of its surface in compact regions, such that (1) is transformed to:

$$pf(v) = \sum_i g(v, b_i)\, area(V_i) \tag{2}$$

where b_i denotes the center of the region V_i.

Also, another approximation of the protrusion function might alternatively be used as in [4]:

$$pf(v) = \sum_{v_i \in \mathrm{S}} g(v, v_i) \tag{3}$$

where v_i denotes the vertices of the mesh.

Since the protrusion function has the aforementioned property and salient points of the mesh reside at the extrema of the mesh, it is natural to search for them at the local maxima of the function, i.e. for each vertex $v \in \mathrm{S}$ a neighborhood of vertices N_v is considered, v is a salient point of the mesh if: $pf(v) > pf(v_i),\ \forall v_i \in \mathrm{N}_v$.

The neighborhood N_v can be either:

i. *A k-ring neighborhood* defined as the set of vertices within k edges away from vertex v.
ii. *A geodesic neighborhood* defined as the set of vertices for which the geodesic distance from vertex v is less than a threshold. This threshold is called the *radius* of the geodesic neighborhood.

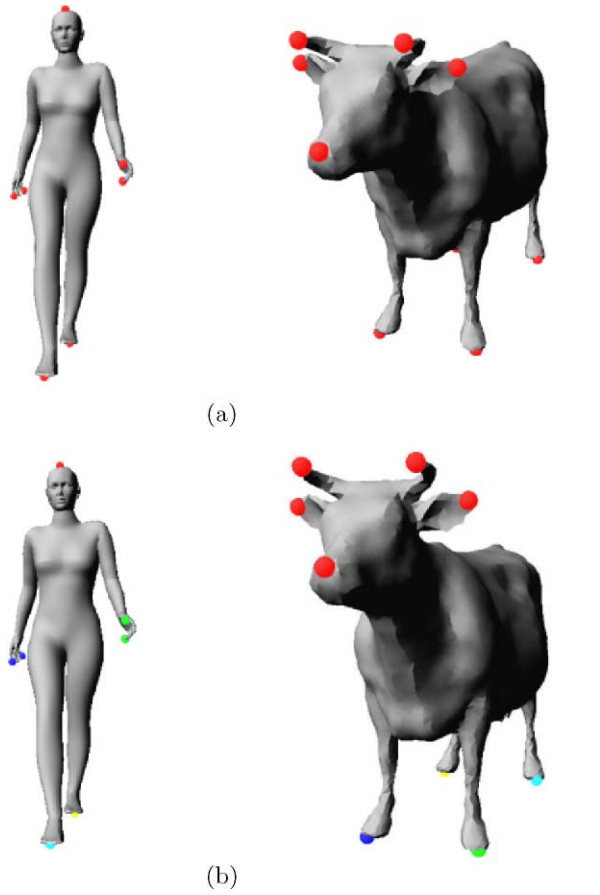


Fig. 3 Example of 3D meshes with their corresponding salient points at the (**a**) extraction stage (*red dots*) and (**b**) grouping stage—each color represents a different group

In Katz et al. [8], a 1-ring neighborhood is used. Since a 1-ring neighborhood can be susceptible to noise, their pose invariant representation is used in order to filter the candidate salient points. In Lin et al. [13], a geodesic neighborhood is used.

In our approach, similar to [13], a geodesic neighborhood is used with radius $\sqrt{5 \times 10^{-3} \times area(\mathrm{S})}$ as in [4]. In our implementation, the geodesic distances and neighborhoods are computed using the Dijkstra algorithm. The detected salient points are further filtered by requiring that their protrusion function should be higher than a threshold $\mathrm{t_{prot}}$. This step ensures that the salient points would be far from the center of the mesh. In our approach, we also detect whether two or more salient points have common geodesic neighborhoods and only one of them as a candidate salient point is chosen. The salient points computed by our approach on two example meshes are shown in Fig. 3(a).

3.2 Salient points grouping

Our mesh segmentation algorithm aims to extract the meaningful parts of the object. It often happens that the extracted

salient points belong to sub-parts of the objects. For example, in Fig. 3(a), there exist salient points that correspond to fingers in the 'human' model, as well as ears, horns and nose in the 'cow' model. Since our algorithm uses the salient points to extract the protrusions of the object, it is necessary to group them, each one of the groups representing a single part of the object. Thus, given the example of Fig. 3(a), the salient points in each of the hands of the 'human' model need to be gathered in one group in order to represent its arms and the salient points on the head of the 'cow' model should be gathered in one group in order to represent its head.

The salient points that are required to be grouped are those which are close to each other in terms of geodesic distance. In order to achieve this grouping, we use half of the mean of the geodesic distances between the salient points as a threshold T_S and group the salient points for which the geodesic distance is less than T_S.

Formally, let assume that $S = \{s_i, i = 1, \ldots, N_S\}$ be the set of the salient points of the mesh, then the threshold T_S is defined as:

$$T_S = \frac{\sum_{i=1}^{N_S-1} \sum_{j=i+1}^{N_S} g(s_i, s_j)}{N_S(N_S - 1)} \tag{4}$$

where N_S denotes the number of the salient points and $g(s_i, s_j)$ denotes the geodesic distance between the salient points s_i and s_j.

A group C of salient points is defined as:

$$C = \left\{ s_i \in S : \forall s_k \in C, g(s_i, s_j) < T_S \right\} \tag{5}$$

We also choose as a representative of each group C the salient point with the highest protrusion value, i.e.

$$Rep(C) = \left\{ s_i \in C : pf(s_i) > pf(s_k), \forall s_k \in C \right\}$$

The efficiency of our grouping method in representing the main parts of the 'human' and 'cow' models is shown in Fig. 3(b).

3.3 Core approximation

In this section, we will present our core approximation algorithm which is a crucial step in our approach. By core approximation, we mean the approximation of the main body of an object. An algorithm which approximates the main body of the object is the one that can acquire all the elements (vertices or faces) of the mesh except those that belong to the protrusions of the mesh. These elements should separate all the protrusions from each other.

In Katz et al. [8], this is achieved by spherical mirroring of the pose invariant representation of the mesh. When they cannot achieve this entirely, they proceed in core extension until all features are separated. In Lin et al. [13], a simple threshold of the protrusion function is used, i.e. they define that the core of the mesh is the points of the mesh whose value is lower than a predefined threshold. However, a fixed threshold value is not reliable since it might lead to a significant overestimation or underestimation of the main body.

In our algorithm, the core approximation is addressed by using the minimum cost paths between the representative salient points. Let assume that $\hat{S} = \{\hat{s}_i, i = 1, \ldots, N_C\}$ be the set of representative salient points, where N_C denotes the number of clusters found in Sect. 3.2 and $\hat{s}_i$ the representative of the ith group. Also, let $P = \{P_{ij}, i, j \in \{1, \ldots, N_C\}\}$ be the set of all minimum cost paths of the points of $\hat{S}$, where P_{ij} denotes the minimum cost path between $\hat{s}_i$ and $\hat{s}_j$. The key idea of our core approximation is to expand a set of vertices in ascending order of protrusion function value until the expanded set touches a certain percentage of all elements of P. The motivation of this approach stems from the fact that the minimum cost paths cover a significant amount of the protrusible parts, thus by expanding a set of vertices by this way gives a guarantee that it will reach the protrusible parts and cover also an area inside them. The pseudo-code for our core approximation algorithm is shown in Fig. 4.

First, the vertices of the mesh M are inserted in a priority queue *PFHeap* in which the vertices with the minimum $pf()$ are inserted first. The vertices of the core approximation are stored in a list *CoreList*. The algorithm proceeds by extracting points from the priority queue, which incrementally expand the *CoreList*. Every point extracted is examined whether it belongs in P. A path P_{ij} in P remains *active* if the ratio of the number of vertices in the path P_{ij}, which have been visited during expansion over the total number of vertices, is less than t_c. This threshold denotes the aforementioned percentage of the points of the minimum cost path that the core approximation can touch. Using this threshold it is expected that the core approximation will cover even slightly the protrusible parts that are traversed by the minimum cost path traces. A salient point $\hat{s}_i \in \hat{S}$ remains *active* if $\exists P_{ij}$ for some $j \in \{1, \ldots, N_C\} \neq i : P_{ij}$ is active. A vertex v of the Mesh *CanBeAdded* in *CoreList* if the nearest salient point in $\hat{S}$ is active. *StopGrowing* becomes 'TRUE' when all salient points become non-active.

Our core approximation method has two main advantages over [8]:

i. There is no need to do multidimensional scaling, which is a time-consuming process, in order to extract the core. Instead, only the minimum cost paths are used in order to check whether the core has expanded sufficiently. This implies far less complexity.
ii. We have introduced a percentage of minimum cost path traces that should be covered for the termination of core expansion. Those traces span the protrusible parts at most. Thus, the selection of a percentage of the traces provides a high confidence that the core points will cover

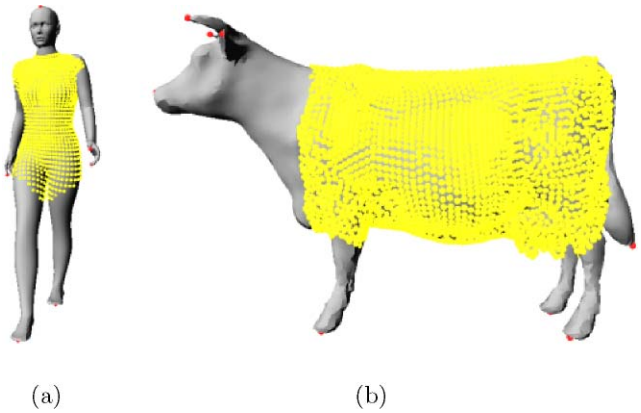
```
for all vertices v ∈ M do
    insert v in PFHeap with priority pf(v)
end for
StopGrowing = false
while !StopGrowing do
    pop a vertex v from PFHeap
    if vCanBeAdded then
        CoreList.add(v)
    end if
    for all P_ij ∈ P do
        if P_ij.active then
            if v ∈ P_ij then
                increment P_ij.counter
                if P_ij.counter / P_ij.SizeOfPath ≥ t_c then
                    P_ij.active = false
                end if
            end if
        end if
    end for
    for all ŝ_i ∈ Ŝ do
        if ŝ_i.active then
            ŝ_i.active = false
            for all ŝ_j ∈ Ŝ − ŝ_i do
                if P_ij.active then
                    ŝ_i.active = true
                end if
            end for
        end if
    end for
    //StopGrowing becomes true if all ŝ_i become non-active
end while
```

Fig. 4 The pseudo-code of the proposed core approximation algorithm

areas of the protrusible parts or being very close to the neighboring areas in which the real boundary is situated. Several examples are given in Fig. 13.

Figure 5 illustrates our core approximation on two meshes. As it can be observed, our algorithm produces consistent approximation of the core and its boundaries are near the actual boundaries identifying the initial approximation of the partitioning boundaries.

In general, it cannot be guaranteed that the core approximation overlaps exactly the partitioning boundaries. This happens because the minimum cost paths reside mostly in the protrusions and sometimes the core expands inside them in order to achieve the percentage of the points of the minimum cost paths it should touch. Thus, a further step is required that can detect the partitioning boundaries.



(a) (b)

Fig. 5 Examples of core approximation for the 3D models 'human' and 'cow.' The vertices representing the core are colored in *yellow*

3.4 Partitioning boundary detection

In this section, we will detail the stage of our algorithm that detects the partitioning boundary, that is the boundary between a protrusion and the main body of the mesh. We consider that in the area that is enclosed by the desired boundary between the protrusion and the main body, an abrupt change in the volume of the 3D object should occur; thus, our goal is to detect this change. To accomplish this, we construct closed boundaries which are defined by a distance function D associated with a salient representative of the group which represents the protrusion. The abrupt change of volume is detected by examining the closed boundaries' perimeter.

For each representative salient point $\hat{s}$, the distance function D is defined for each vertex v of the mesh as the shortest distance between v and $\hat{s}$. The shortest distance is computed using the Dijkstra algorithm with source $\hat{s}$ and cost for each edge (u, v) of the mesh denoted as:

$$\text{cost}(u, v) = \delta \frac{\text{length}(u, v)}{\text{avg\_length}} + (1 - \delta) \frac{\text{prot}(u, v)}{\text{avg\_prot}} \tag{6}$$

where $\text{prot}(u, v) = |pf(u) - pf(v)|$ and avg_length, avg_p-rot denote the average values of the length and protrusion difference of the edges of the mesh, respectively. This distance function was introduced in [13]. A proper balance between the two terms of (6) create a closed boundary whose approximating plane is nearly perpendicular to the principal direction of the corresponding protrusion. In our implementation, we set δ equal to 0.4.

The closed boundaries which are used in our algorithm are constructed by interpolating on the edges of the mesh the iso-contour generated by setting a constant value D_c on D.

The iso-contour C_{D_c} intersects the edge (u, v) of the mesh if $(D(u) - D_c)(D_c - D(v)) > 0$ and the intersection point is $v_{\text{int}} = (1 - \lambda)u + \lambda v$, where $\lambda = \frac{D_c - D(u)}{D(v) - D(u)}$. By

tracing all the triangles which are incident to all the intersecting edges, an approximation of this iso-contour can be constructed, which in most of the cases is a single closed boundary. In the case that more than one closed boundary is generated, we choose the one with the largest perimeter. In Fig. 6, an illustration of a closed boundary constructed by the method described above is shown.

As already mentioned in Sect. 3.3, our core approximation has its boundaries near the actual boundaries of the distinct parts of the model. Taking advantage of this, we set a predefined number l_{per} of closed boundaries in the area defined by the interval $[(1 - \text{d}_1)\text{D}_{\text{coremin}}, (1 + \text{d}_2)\ \text{D}_{\text{coremin}}]$ wherein the real protrusion is situated. $\text{D}_{\text{coremin}}$ denotes the value of the distance function between the nearest point of the core approximation and the representative $\hat{s}$; d_1, d_2 denote the extent of the interval ($0 < \text{d}_1 < 1$, $\text{d}_2 > 0$). For the approximation of the partitioning boundary, the area of the aforementioned interval is swept in fixed steps equal to $e = \frac{(\text{d}_1+\text{d}_2)\text{D}_{\text{coremin}}}{l_{\text{per}}}$. Sweeping of the area will be terminated when the change of the perimeter between successive closed boundaries will be greater than a threshold.

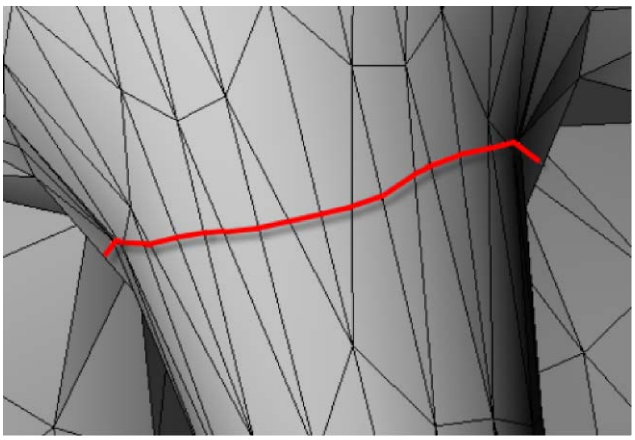


Fig. 6 The closed boundary approximating an iso-contour of the distance function D

Specifically, let per_i be the perimeter of the ith closed boundary. We define the ratios:

$$r_i = \begin{cases} \frac{per_{i+1}}{per_i} & \text{if } per_{i+1} > per_i, \\ \frac{per_i}{per_{i+1}} & \text{if } per_{i+1} \leq per_i, \end{cases} \qquad i = 1, \ldots, l_{\text{per}} \tag{7}$$

These ratios represent the change of the perimeter between successive closed boundaries. Our algorithm searches for r_k which is the first ratio that is greater than a threshold $r_{\max}$ that has been defined experimentally and sets the kth closed boundary as the partitioning boundary. If such a ratio cannot be found, the algorithm finds the maximum ratio $r_{k\max}$ and sets the $k_{\max}$-th closed boundary as the partitioning boundary. In this way, we detect the aforementioned abrupt change in the volume of the object at the boundary between its main body and the protrusion.

Initially, in our algorithm we have chosen the simple solution to select the representative of the group, defined in Sect. 3.2, as the source point from which the distance function D is computed. This choice may lead to the creation of skewed closed boundaries near the real boundary of the protrusion. This has the consequence that the algorithm may not be able to trace the real boundary properly as it will largely deviate from it. See Fig. 7(b) for a demonstration of this problematic case. In the sequel, we present how to find a proper *refined representative* $\hat{s}_r$ in order to be used instead of the representative of the group. The proper refined representative should create closed boundaries that are parallel to the protrusion boundary like the one presented in Fig. 7(c).

In order to find the proper refined representative in a group of salient points, we proceed as follows.

First, we find the vertex s_{min} of the mesh, for which the distance to all of the salient points in the group is minimal. Then, we find the point c_{min} of our core approximation with the minimum geodesic distance from s_{min}. The geodesic distance of s_{min} and c_{min} is denoted as d_{min}. Afterwards, we find the point p_{thres} which is the first point on the minimum cost path from s_{min} to c_{min}, for which the geodesic distance

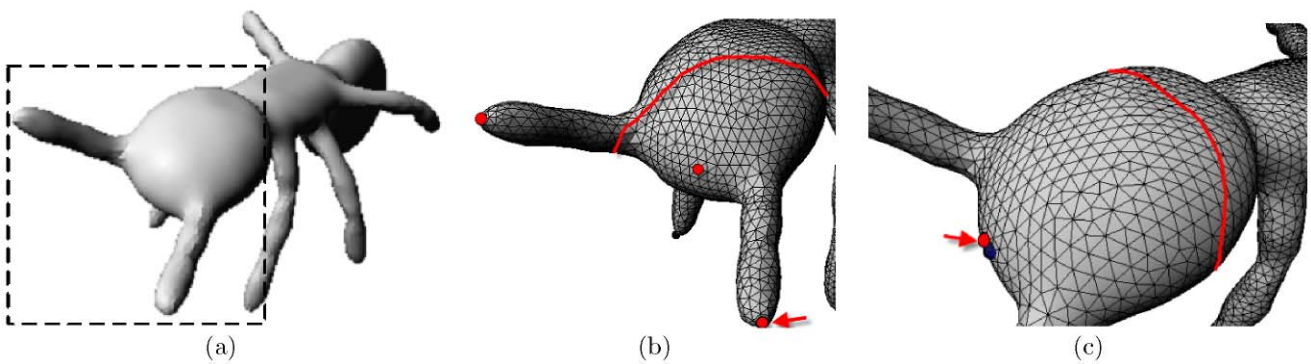


Fig. 7 (**a**) The original 3D model 'ant'; (**b**) skewed closed boundary generated by the salient representative, indicated by an *arrow*, of the group denoted by the *red spheres*; (**c**) the closed boundary generated by the indicated refined representative

from s_{min} is greater than $0.3d_{min}$. Next, we consider the iso-contour $C_{p_{thres}}$ generated by the protrusion function by setting it to the constant value $pf(p_{thres})$. The part of the iso-contour which belongs to the protrusion of the object being examined is close to the salient points of the group and in most of the cases its best fit approximating plane is nearly perpendicular to the principal direction of the protrusion. We approximate this contour using the same interpolating technique as discussed above.

We then take into account only the part of the mesh which contains s_{min} and is constrained by $C_{p_{thres}}$, for which we compute the protrusion function. The *refined representative* $\hat{s}_r$ will be the point of the constrained mesh with the minimum value of the protrusion function.

In Fig. 8 we illustrate this process. The salient points that belong to the group are the red spheres. The yellow sphere represents s_{min} and the red line approximates the iso-contour $C_{p_{thres}}$ (the red line is generated by the triangles that $C_{p_{thres}}$ intersects). The *refined representative* $\hat{s}_r$ is the point of the constrained mesh (colored in blue) with the minimum protrusion value and is illustrated by the green sphere.

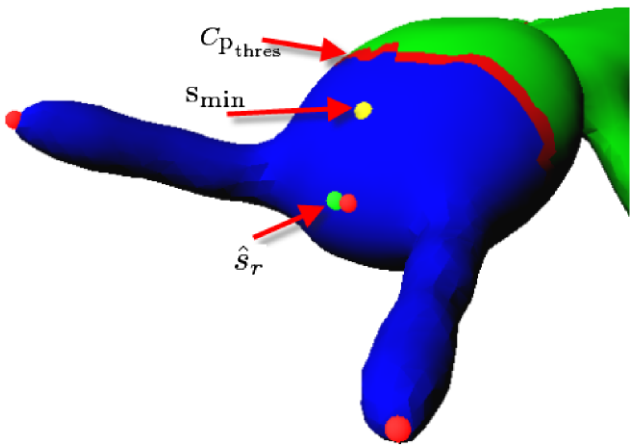


Fig. 8 Refined representative selection (*green sphere*)

3.5 Partitioning boundary refinement

The partitioning boundary detected in Sect. 3.4 is a closed boundary approximating an iso-contour of the distance function D. In most of the cases this boundary is a rough approximation of the true protrusion boundary of the object, thus it has to be refined and placed according to Hoffman and Richards [5] at the concavities of the object.

In order to achieve this, we use the minimum-cut methodology, as in Katz and Tal [7]. Specifically, we construct a flow network graph using the dual graph of the mesh. In order to construct the network, three regions should be defined. Specifically, we define a region **A** containing the triangles of the protrusion of the mesh, a region **C** which is going to contain the partitioning boundary and a region **B** containing the faces of the rest of the mesh.

Region **C** is constructed as follows. First we find the average geodesic distance, denoted as AvgGeodDist, between the boundary extracted in Sect. 3.4 and the refined representative calculated also in the same section. Then, region **C** is defined as the triangles of the mesh, the geodesic distance vertices of which from the refined representative all lie in the interval [0.9·AvgGeodDist, 1.1·AvgGeodDist]. This interval denotes a small area around the estimated partitioning boundary where it is expected that the true partitioning boundary should reside.

The set **A** is constructed by performing at breadth first search starting from the refined representative of the protrusion until region **C** is reached.

Let $V_{\mathbf{C}}$, $E_{\mathbf{C}}$ be the nodes and edges of the graph, representing **C**. Let $V_{\mathbf{CA}}$ be the nodes of the graph, which represent the triangles of **A** that share a common edge with the

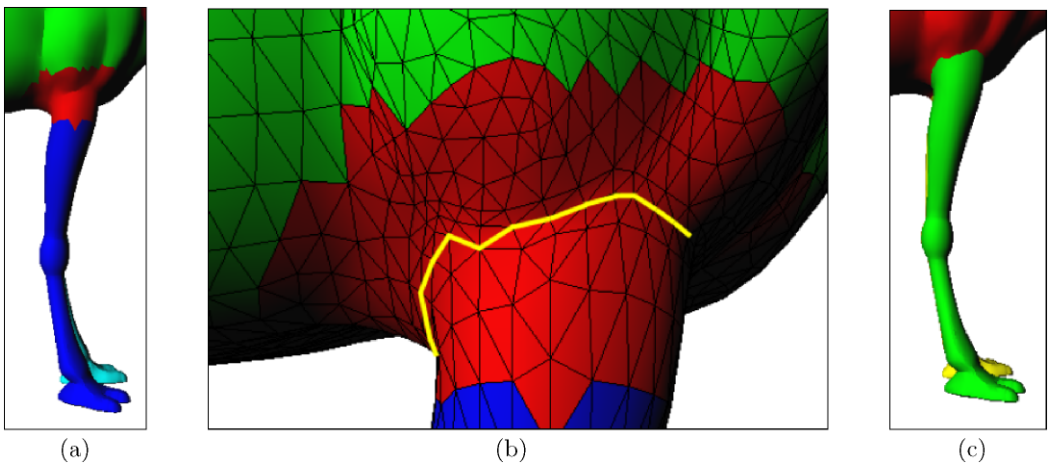


Fig. 9 (**a**) Region **A** is shown in *blue*, region **B** is shown in *green* and region **C** is shown in *red*; (**b**) region **C** enhanced, the *yellow line* denotes the minimum-cut in this region; (**c**) the final segmentation of the protrusion from its main body

triangles of **C**; and V_{CB} be the nodes of the graph, which represent the triangles of **B** that share a common edge with the triangles of **C**.

The flow network graph G = (V, E) is constructed by adding also two more nodes—s, t—and V, E which are defined as:

$$\begin{aligned} \mathrm{V} &= \mathrm{V_C} \cup \mathrm{V_{CA}} \cup \mathrm{V_{CB}} \cup \{\mathrm{s}, \mathrm{t}\} \\ \mathrm{E} &= \mathrm{E_C} \cup \left\{(\mathrm{s}, v), \forall v \in \mathrm{V_{CA}}\right\} \cup \left\{(\mathrm{t}, v), \forall v \in \mathrm{V_{CB}}\right\} \\ &\quad \cup \left\{e_{uv} \in \mathrm{E} : u \in \mathrm{V_C}, v \in \{\mathrm{V_{CA}} \cup \mathrm{V_{CB}}\}\right\} \end{aligned} \tag{8}$$

A capacity function on E is defined as:

$$\mathrm{Cap}(u, v) = \begin{cases} \frac{1}{1+\frac{Ang\_Dist(\alpha_{uv})}{\mathrm{avg}(Ang\_Dist)}} & \text{if } u, v \neq \mathrm{s}, \mathrm{t} \\ \infty & \text{otherwise} \end{cases} \tag{9}$$

$$Ang\_Dist(\alpha_{uv}) = n(1 - \cos a_{uv}) \tag{10}$$

where α_{uv} denotes the dihedral angle between the two faces which share the edge (u, v), and avg(*Ang_Dist*) is the average angular distance. Note that $n = 1$ for concave angles and $n \ll 1$ for convex angles.

Using this capacity function, the minimum cut algorithm applied on this network passes through concave angles. Figure 9 illustrates the partitioning boundary refinement process.

4 Experimental results

In this section, we present experimental results for twenty 3D models segmented with our algorithm which are further compared with the results produced by three other popular segmentation algorithms whose segmentation results are in our disposal, namely Lin et al.'s (LIN) [13], Valette et al.'s (VALETTE) [20] and Kim et al.'s [9] (KIM). The results of LIN algorithm were produced by the original authors, VALETTE is our own implementation, while KIM is the original implementation which is used in the MPEG-7 experimentation model.

In our comparison, we are going to use the evaluation criteria set in [2], namely, (i) Type of segmentation; (ii) Extracting the "correct" segments; (iii) Boundaries; (iv) Hierarchical/multi-scale segmentation; (v) Sensitivity to pose; (vi) Asymptotic complexity; (vii) Control parameters. Furthermore, we are also going to show the behavior of our algorithm in the presence of noise.

i. **Type of segmentation.** All of the algorithms considered are part-based and are designed to segment the mesh into components that are meaningful in human perception.

ii. **Extracting the "correct" segments.** In this criterion, the "correct" segmentation depends on the application, the viewer's perspective and knowledge of the world which can only be judged qualitatively by looking at the images of the output of the segmentation algorithms. The application that we opt to couple with the proposed mesh segmentation scheme is 3D model retrieval. Judging from the overall segmentation results presented in Fig. 10 it can be seen that the proposed approach and LIN perform best, i.e. both algorithms segment the meshes in a perceptually correct way and it can be observed that meshes that belong in the same category, for example all humans, are segmented consistently, which is very important in the shape retrieval context. VALETTE manages to extract also perceptually meaningful results, but as there are many cases where the segmentation among meshes of the same category are not consistent (e.g. models (12)–(13) in Fig. 10), this has a negative effect in retrieval. KIM manages also to extract perceptually meaningful components, but as in VALETTE, there are cases where the segmentation among meshes of the same category are non-consistent (e.g. models (7)–(8), (9)–(10), (12)–(13), (14)–(19) in Fig. 10).

iii. **Boundaries.** Here the "correctness" of the boundaries will be judged upon two geometric properties: (a) the smoothness of the boundary and (b) its location along concave features. From Fig. 10 it can be seen that our algorithm and LIN produce the most smooth results. This is expected because both algorithms use a minimum-cut algorithm based on the concave features of the mesh in order to refine the boundary. In respect to the location of the boundaries along concave features, it can be seen that our algorithm performs best.

iv. **Hierarchical/multi-scale segmentation.** Among all tested algorithms only KIM produces hierarchical results.

v. **Sensitivity to pose.** Models (15)–(18) in Fig. 10 illustrate the pose sensitivity of the four algorithms. These models represent the same human in different poses: running, jumping, sitting and walking. It can be seen that our approach and LIN algorithms manage to remain invariant throughout all of the pose changes of the mesh. VALETTE and KIM though show sensitivity to the pose of the object.

vi. **Complexity.** Our algorithm: Let N be the number of points and N_S the number of salient points of the mesh.
   a. Protrusion function computation: In our implementation the geodesic distances are computed using the Dijkstra algorithm. In order for the computation to be more accurate, we use midpoint subdivision to increase the resolution of the mesh. In most of the cases

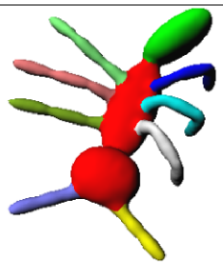
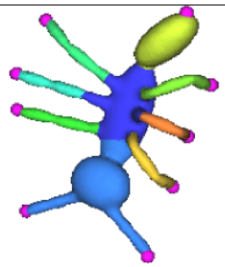
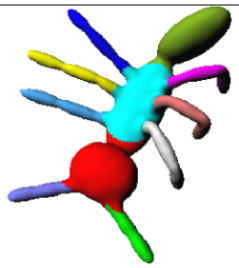
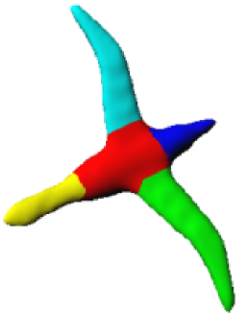
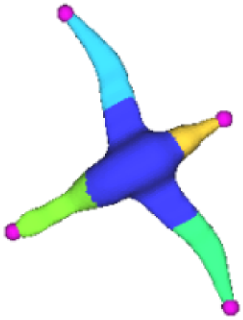
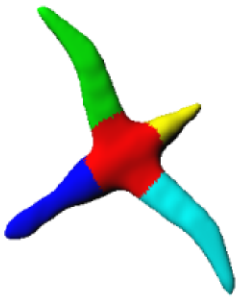
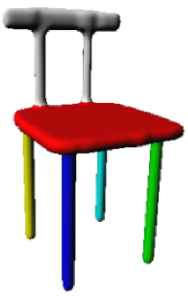
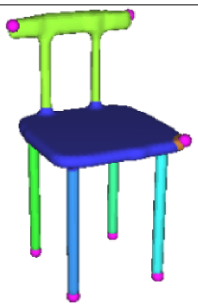
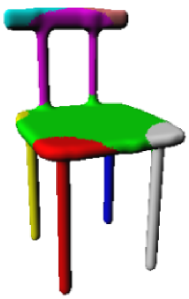
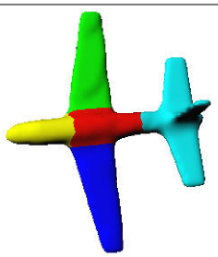
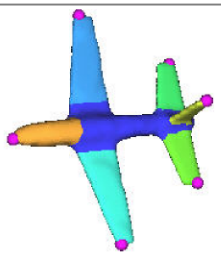
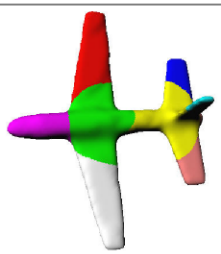
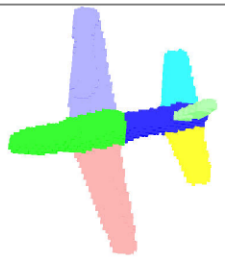
3D Model N_r	Proposed algorithm	LIN	VALETTE	KIM
(1)				
(2)				
(3)				
(4)				
(5)				

Fig. 10 Experimental results

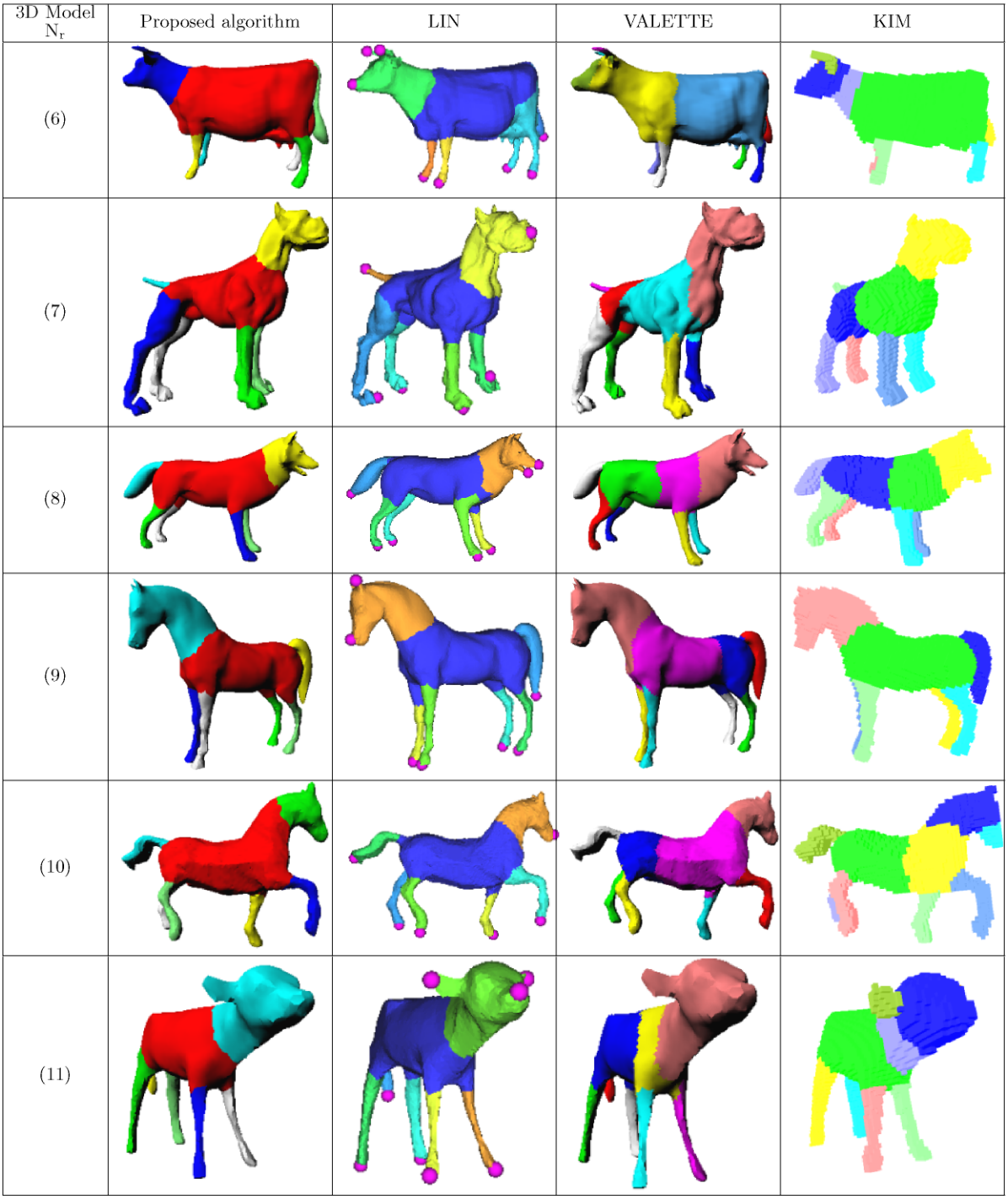


Fig. 10 (*Continued*)

the subdivision is done twice, so the number of points of the mesh increase at the magnitude of $8N$. Let us assume that the number of compact regions are M. The complexity of the protrusion function calculation is $O(8MN\log(N))$.

b. Salient points extraction computation: Let us assume that the maximum number of points in a geodesic neighborhood is $N_{\max}$, then the complexity of the salient points extraction computation is $O(NN_{\max}\log(N_{\max}))$.
c. Salient points grouping: The complexity of the salient point grouping is $O(N_S^2 N\log(N))$.
d. The complexity of the core approximation is $O(NN_S^2)$.

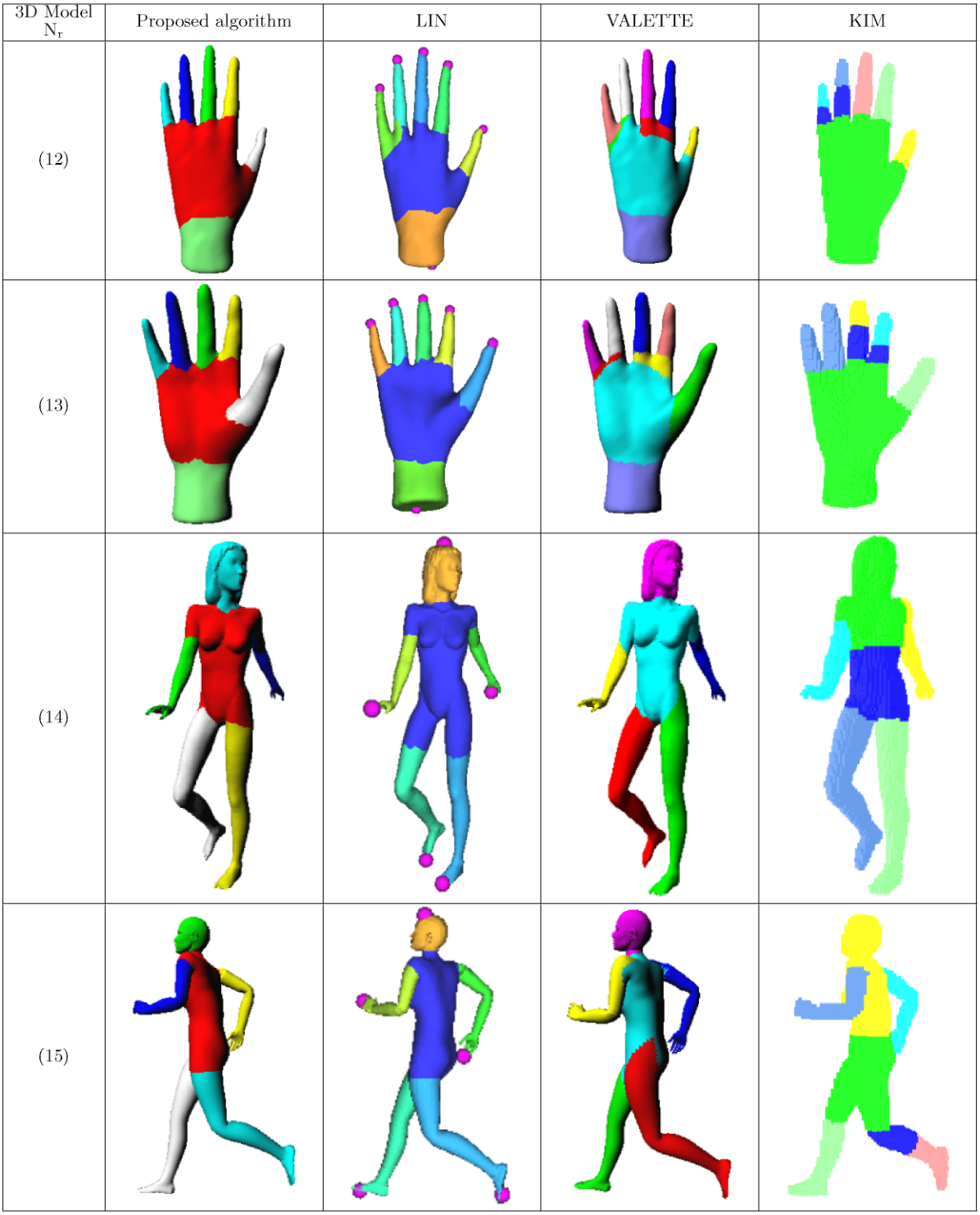


Fig. 10 (*Continued*)

e. The complexity of the protrusion boundary detection is $O(kl_{\text{per}}N + kN\log + 8kmn\log n)$, where m is the number of compact regions, n is the number of points of the part of the mesh where the refined representative is calculated, k is the number of the representative salient points, and l_{per} the number of the closed boundaries.
f. The complexity of the protrusion boundary refinement is $O(N_{\text{ng}}^2 \log(N_{\text{ng}}))$, where N_{ng} is the number of nodes in the network graph.

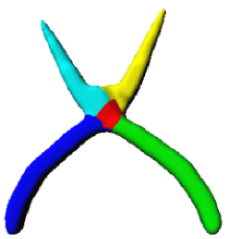
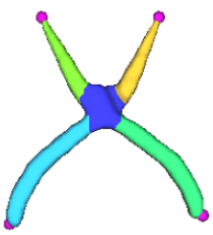
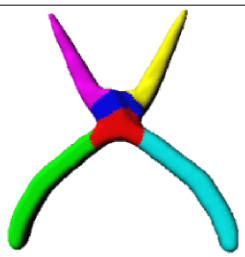
3D Model N_r	Proposed algorithm	LIN	VALETTE	KIM
(16)				
(17)				
(18)				
(19)				
(20)				

Fig. 10 (*Continued*)

Overall, the algorithm dominant complexity is $O(8MN \times \log(N) + NN_{\max}\log(N_{\max}) + 8kmn\log n)$.

LIN: The total complexity of LIN is $O(MF\log(F) + F^2\log(F))$, where F the total number of faces of the mesh and M the number of compact regions used in the calculation of the protrusion function.

VALETTE: The total complexity of VALETTE is $O(N\log(N))$, where N the total number of vertices of the mesh.

vii. **Control parameters.** The results of our algorithm in Fig. 10 are obtained by setting $t_{prot} = 0.45$ (the protrusion function is scaled between [0,1]), $t_c = 0.15$, $d_1 = 0.1$, $d_2 = 0.4$, $l_{per} = 12$, $r_{max} = 1.3$. All of them remain fixed and user interactivity is not required.

Table 1 Computation time for the segmentation of Fig. 10 3D models

Model No.	No. Vertices	No. Triangles	s
(1)	8504	17004	60
(2)	3478	6952	35
(3)	12326	24652	68
(4)	6689	13374	31
(5)	9761	19518	60
(6)	7255	14506	38
(7)	9492	18980	45
(8)	4712	9420	55
(9)	7268	14532	35
(10)	11312	22620	61
(11)	3703	7402	44
(12)	7242	14480	34
(13)	6607	13210	29
(14)	11016	22028	59
(15)	5775	11546	52
(16)	5766	11528	72
(17)	5772	11540	54
(18)	5769	11534	55
(19)	9509	19014	55
(20)	6104	12204	24

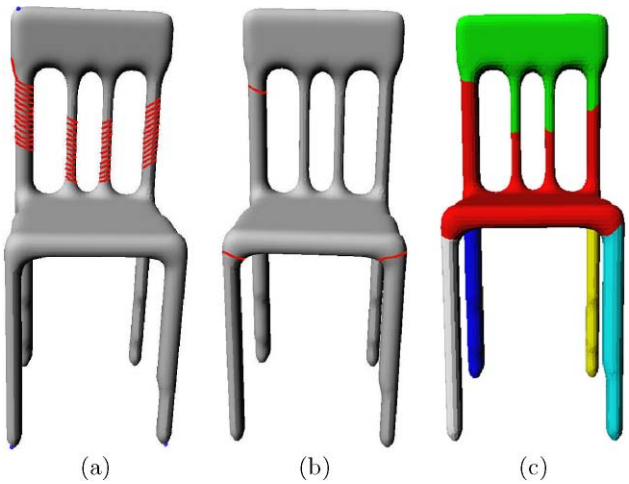


Fig. 11 (**a**) The closed boundaries spanning the area $[(1-d_1)D_{coremin}, (1+d_2)D_{coremin}]$; (**b**) the approximation of the partitioning boundary; (**c**) final segmentation

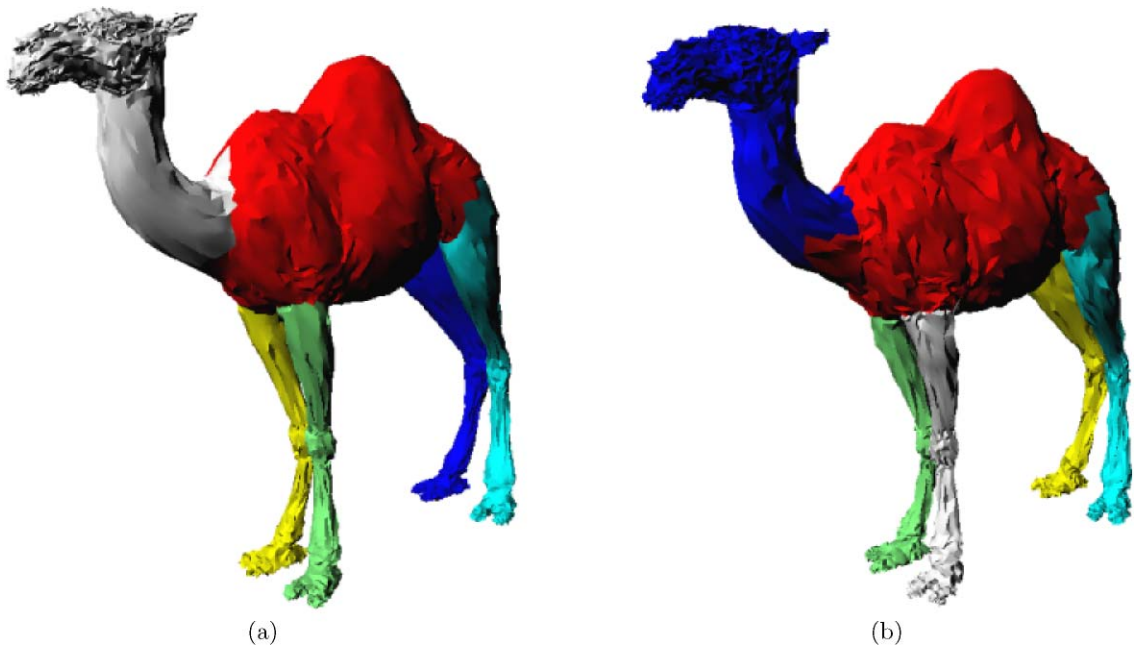


Fig. 12 Segmentation of the 'camel' model under different levels of noise: (**a**) SNR = 52 dB, (**b**) SNR = 50 dB

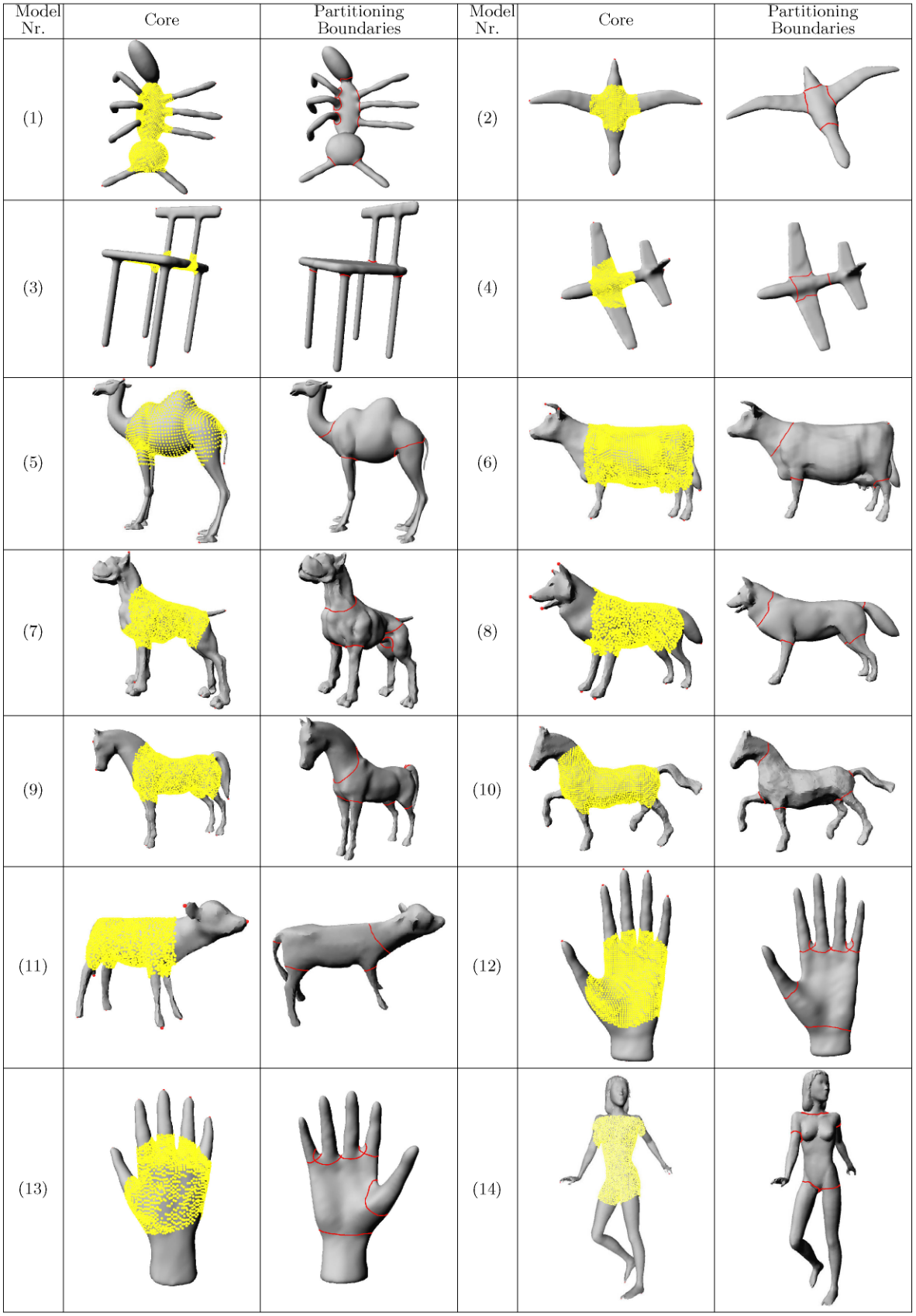


Fig. 13 Core approximation and partitioning boundaries of the models of Fig. 10

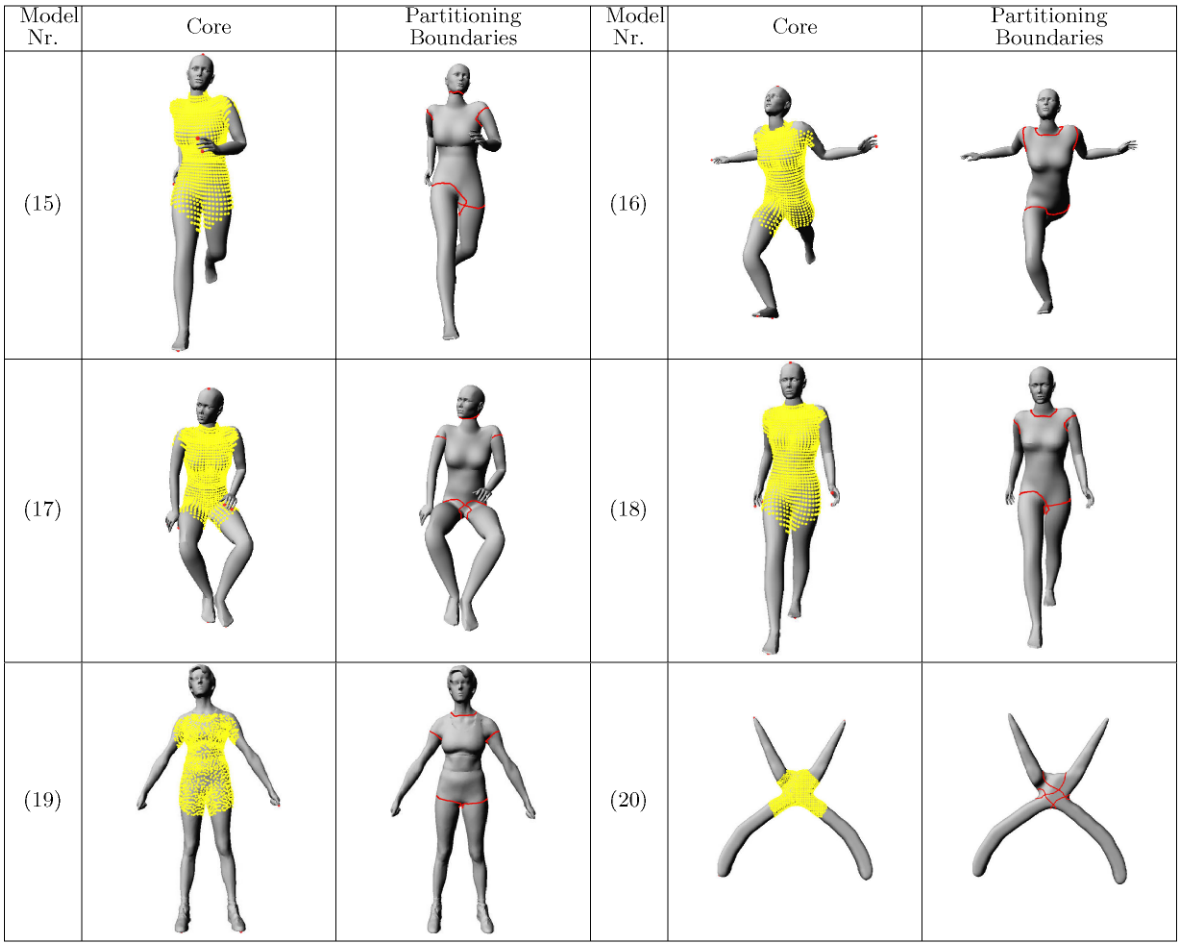


Fig. 13 (*Continued*)

In LIN algorithm, there exist three parameters that the user has to set manually and depend on the core approximation and the locales produced by their algorithm. Namely, the parameters that have to be set by the user are: (1) the parameter Δ_β which controls the extent that the locales overlap with the core of the object, (2) the parameters Δ_+, Δ_- which control the range of the locales that contain the protrusion's boundary.

It can be seen that our algorithm performs equally or better than their results in spite that our parameters remain fixed.

In VALETTE algorithm, there exists only one parameter P_{ratio} which we have set after experimentation to 15% to all of the objects. We have found that this parameter value produces the best results in their algorithm.

In KIM algorithm, despite that they set the parameters fixed, the output is strongly dependent of the voxelization resolution. This means that there is a need for human inspection to correct the resolution when the output is not satisfactory.

In general, there is no particular geometrical or topological feature that limits the functionality of our segmentation methodology for the models used in our experimentation. However, the success of the proposed segmentation is bound to the interval $[(1 - \mathrm{d}_1)\mathrm{D}_{\text{coremin}}, (1 + \mathrm{d}_2)\mathrm{D}_{\text{coremin}}]$. For a meaningful segmentation, the aforementioned interval should produce a region that includes the partitioning boundary. If this is not the case, there will be a segmentation outcome as in the 3D model of Fig. 11.

In Table 1, the time required for segmenting the models in Fig. 10 is given in seconds. The computation time was calculated in a Pentium 4, 3 GHz PC with 1.5 GB memory.

Our algorithm has been tested also in terms of robustness with respect to noise, i.e., it produces the same segmentation results with different levels of noise. A representative example of the proposed mesh segmentation for different levels of Gaussian noise is shown in Fig. 12.

In Fig. 13, the core and partitioning boundary approximation of the models in Fig. 10 are shown. It can be observed that the core approximation either covers portions of the protrusible parts areas or is very close to the neighboring areas where the real boundary is situated. It can be observed

that the approximation of the partitioning boundary is also effective since it is very close to the real partitioning boundary.

5 Conclusion

In this paper, a novel 3D mesh segmentation algorithm has been presented.

The novelty of our approach is twofold:

i. A novel way to trace the boundary of the protrusion of the 3D object using closed boundaries is constructed with the aid of a distance function.
ii. A novel algorithm for the core approximation of the 3D object is introduced.

The proposed algorithm is capable of segmenting a 3D object into perceptually meaningful parts and is pose invariant. From the experimental results presented it has been shown that it can successfully segment a wide range of articulated 3D objects. The evaluation of the proposed algorithm is addressed in a consistent framework wherein a comparison with the state of the art is performed.

The algorithm is based on the basic idea that an object can be segmented into its parts if its main body and salient points are reliably approximated.

Our algorithm is capable of producing compatible segmentations; for instance, the human objects are always segmented into the head, body, arms and legs, thus it can be successfully applied to applications that require consistent segmentations like 3D shape retrieval based on graph based representations.

Acknowledgements This research was supported by the Greek Secretariat of Research and Technology (PENED "3D Graphics search and retrieval" 03 ED 520).

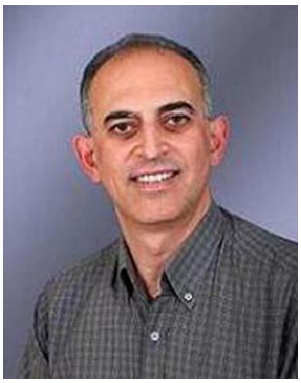
References

1. Agathos, A., Pratikakis, I., Perantonis, S., Sapidis, N., Azariadis, P.: 3D Mesh segmentation methodologies for CAD applications. Comput. Aided Des. Appl. **4**(6), 827–841 (2007)
2. Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., Tal, A.: Mesh segmentation—a comparative study. In: IEEE International Conference on Shape Modeling and Applications. IEEE, Matsushima (2006)
3. Attene, M., Falcidieno, B., Spagnuolo, M.: Hierarchical segmentation based on fitting primitives. Vis. Comput. **22**(3), 181–193 (2006)
4. Hilaga, M., Shinagawa, Y., Komura, T., Kunii, T.L.: Topology matching for full automatic similarity estimation of 3D. In: SIGGRAPH, pp. 203–212. ACM, Los Angeles (2001)
5. Hoffman, D., Richards, W.: Parts of recognition. Cognition **18**, 65–96 (1984)
6. Karni, Z., Gotsman, C.: Spectral compression of mesh geometry. In: SIGGRAPH, pp. 279–286. ACM, New Orleans (2000)
7. Katz, S., Tal, A.: Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Trans. Graph. **22**(3), 954–961 (2003)
8. Katz, S., Leifman, G., Tal, A.: Mesh segmentation using feature point and core extraction. Vis. Comput. **21**(8–10), 639–648 (2005)
9. Kim, D.H., Yun, I.D., Lee, S.U.: A new shape decomposition scheme for graph-based representation. Pattern Recognit. **38**(5), 673–689 (2005)
10. Lee, Y., Lee, S., Shamir, A., Cohen-Or, D., Seidel, H.-P.: Mesh scissoring with minima rule and part salience. Comput. Aided Geom. Des. **22**(5), 444–465 (2005)
11. Levy, B., Petitjean, S., Ray, N., Maillot, J.: Least squares conformal maps for automatic texture atlas generation. ACM Trans. Graph. **21**(3), 362–371 (2002)
12. Li, X., Toon, T., Tan, T., Huang, Z.: Decomposing polygon meshes for interactive applications. In: Proc. of the 2001 Symposium on Interactive 3D Graphics, pp. 35–42, NC, USA (2001)
13. Lin, H.S., Liao, H.M., Lin, J.: Visual salience-guided mesh decomposition. IEEE Trans. Multimedia **9**(1), 46–57 (2007)
14. Page, D., Koschan, A., Abidi, M.: Perception-based 3D triangle mesh segmentation using fast marching watersheds. In: Proc. Intl. Conf. on Computer Vision and Pattern Recognition, pp. 27–32, Wisconsin, USA (2003)
15. Shamir, A.: Segmentation and shape extraction of 3D boundary meshes. In: State-of-the-Art Report, Proceedings Eurographics (2006)
16. Shapira, L., Shamir, A., Cohen-Or, Lior Shapira, D.: Consistent mesh partitioning and skeletonisation using the shape diameter function. Vis. Comput. **24**(4), 249–259 (2008)
17. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
18. Shlafman, S., Tal, A., Katz, S.: Metamorphosis of polyhedral surfaces using decomposition. In: Eurographics, pp. 219–228. Eurographics Association, Saarbrücken (2002)
19. Sukumar, S.R., Page, D.L., Koschan, A.F., Gribok, A.V., Abidi, M.A.: Shape measure for identifying perceptually informative parts of 3D objects. In: 3D Data Processing, Visualization and Transmission, pp. 679–686. IEEE, Chapel Hill (2006)
20. Valette, S., Kompatsiaris, I., Strintzis, M.G.: A polygonal mesh partitioning algorithm based on protrusion conquest for perceptual 3D shape description. In: Workshop towards Semantic Virtual Environments, pp. 68–76. Villars, Switzerland (2005)
21. Wu, K., Levine, M.D.: 3D Part Segmentation Using Simulated Electrical Charge Distributions. Trans. Pattern Anal. Mach. Intell. **19**(11), 1223–1235 (1997)
22. Zhang, H., Liu, R.: Mesh segmentation via recursive and visually salient spectral cuts. In: Vision, Modeling, and Visualization, pp. 429–436. Erlangen (2005)
23. Zhang, Y., Paik, J., Koschan, A., Abidi, M.A.: A simple and efficient algorithm for part decomposition of 3D triangulated models based on curvature analysis. In: International Conference on Image Processing, pp. 273–276. IEEE, Rochester (2002)
24. Zuckerberger, E., Tal, A., Shlafman, S.: Polyhedral surface decomposition with applications. Comput. Graph. **5**, 733–743 (2002)

Alexander Agathos is currently a Ph.D. candidate at the Department of Product & Systems Design Engineering of the University of the Aegean and a Ph.D. candidate at the Institute of Informatics and Telecommunications, NCSR 'Demokritos'. He received the Diploma in Mathematics from the National & Kapodistrian University of Athens and the M.Sc. degree in Informatics from the National & Kapodistrian University of Athens. His main research interests are in the field of computer graphics and 3D computer vision, with special focus on 3D model segmentation and retrieval.

Ioannis Pratikakis received the Ph.D. degree in 3D Image Analysis from Vrije Universiteit Brussel, Belgium, in January 1999. From March 1999 to March 2000, he was at IRISA/ViSTA group, Rennes, France as an INRIA postdoctoral fellow. He is currently working as a Research Scientist at IIT/NCSR 'Demokritos,' Athens, Greece. His research interests include 3D image analysis, 3D computer vision and content-based multimedia search and retrieval with a particular focus on 3D objects. He has published more than 90 papers in journals, book chapters and conference proceedings in the above areas. He has participated in more than 15 national and international R&D projects. Finally, he has served as a Co-chair of the Eurographics Workshop on 3D Object Retrieval (3DOR) in 2008 and 2009.

Stavros Perantonis is the holder of a B.Sc. degree in Physics from the Department of Physics, University of Athens, an M.Sc. degree in Computer Science from the Department of Computer Science, University of Liverpool and a D.Phil. degree in Computational Physics from the Department of Physics, University of Oxford. Since 1992 he has been with the Institute of Informatics and Telecommunications, NCSR, where he currently holds the position of Senior Researcher and Head of the Computational Intelligence Laboratory. His current technical and research activities are in the areas of computational intelligence, pattern recognition and multimedia processing and graphics. Dr. Perantonis is the author or co-author of over 110 papers in international journals and conference proceedings. He has managed or participated in many EU or nationally funded projects.



Nickolas S. Sapidis is currently a Professor of Computer-Aided Product Design with the Department of Product and Systems Design Engineering of the University of the Aegean (Syros, Greece). He holds a Diploma in Naval Architecture & Marine Engineering (1985), an M.A. in Applied Mathematics (1987) and an M.Sc. in Mechanical Engineering (1988). He received his Ph.D. in Mechanical and Aerospace Sciences from the University of Rochester in 1993. N. Sapidis has been pursuing research in design of mechanical products, computer-aided design (CAD), computer-aided engineering (CAE), solid modeling, geometric modeling of surfaces, geometric algorithms, virtual engineering, and computer graphics. He is the author of more than 50 papers on the above subjects. His research has been implemented in industrial CAD/CAE-software systems by the Massachusetts Institute of Technology (MIT) and companies including General Motors (GM), Intergraph and Tribon Solutions.