A Mesh Meaningful Segmentation Algorithm Using Skeleton and Minima-Rule

Zhi-Quan Cheng, Kai Xu, Bao Li, Yan-Zhen Wang, Gang Dang, and Shi-Yao Jin

DL Laboratory, National University of Defense Technology,
Changsha City, Hunan Province, P.R. China (410073)

Abstract. In this paper, a hierarchical shape decomposition algorithm is proposed, which integrates the advantages of skeleton-based and minima-rule-based meaningful segmentation algorithms. The method makes use of new geometrical and topological functions of skeleton to define initial cutting critical points, and then employs salient contours with negative minimal principal curvature values to determine natural final boundary curves among parts. And sufficient experiments have been carried out on many meshes, and shown that our framework can provide more reasonable perceptual results than single skeleton-based [8] or minima-rule-based [15] algorithm. In addition, our algorithm not only can divide a mesh of any genus into a collection of genus zero, but also partition level-of-detail meshes into similar parts.

1 Introduction

Mesh segmentation [1][2][3] refers to partitioning a mesh into a series of disjoint elements, and it has become a key ingredient in many mesh operation methods, including texture mapping [4], shape manipulation[5][6], simplification and compression, mesh editing, mesh deformation [7], collision detection [8], shape analysis and matching [9][10]. Especially, the process that decomposes a model into visually meaningful components is called part-type segmentation [1][2] (shape decomposition or meaningful segmentation). Now, more researches are seeking to propose automatic procedures, which can efficiently produce more natural results that are in keeping with human recognition and shape understanding. Especially, more advanced coherency issues should be addressed, such as pose invariance [11], handling more complex models e.g. David and Armadillo, extracting similar parts and shapes over similar objects and more.

1.1 Related Work

The basic segmentation problem can be viewed as clustering primitive mesh elements (vertices, edges and faces) into sub-meshes, and the techniques finishing the partition include hierarchical clustering [9], iterative clustering [4][5], spectral analysis [12], region growing [13], and other methods. The detail survey of mesh segmentation algorithms can be found in [1][2][3]. Among existed meaningful segmentation algorithms, two types of methods, developing at the same time, have caught more focus.

G. Bebis et al. (Eds.): ISVC 2007, Part II, LNCS 4842, pp. 671–680, 2007.

One type, including [13][14] [15], is guided by the minima rule [16], which states that human perception usually divides an object into parts along the concave discontinuity and negative minima of curvature. Enlightened by the minima rule, the mesh's concave features, identified as natural boundaries, are used for segmentation in the region growing watershed algorithm [13]. Due to the limitation of region growing, the technique cannot cut a part if the part boundary contains non-negative minimum curvatures. And then, Page et al. [14] have used the factors proposed in [17] to compute the salience of parts by indirect super-quadric model. To directly compute the part salience on a mesh part and avoid complex super quadric, Lee et al. [15] have experientially combined four functions (distance, normal, centricity and feature) to guide the cutting path in a reasonable way. So the nice segmentation results are completely dependent on the experimental values of the function parameters, as well as the structures of underlying manifold surface. Since directly targeting the cutting contours, the type algorithms can produce better visual effects, especially at the part borders. However, these algorithms are sensitive to surface noises and tending to incur over-segmentation problems (one instance is shown in Fig. 12.a) mostly due to local concavities.

The other, including [6][8][18], is driven by curve-skeleton/skeleton [19] that is 1D topologically equivalent to the mesh. The skeleton-type algorithms don't over-partition the mesh, but the part boundaries do not always follow natural visual perception on the surface. Consequently, it would be a better choice to combine the skeleton-based approach with the minima rule, since the minima rule alone doesn't give satisfying results, while skeleton-based methods don't guarantee a perceptually salient segmentation.

1.2 Overview

In the paper, we want to develop a robust meaningful segmentation paradigm, aiming at integrating advantages of minima-rule-based and skeleton-based approaches. Besides this basic aim, the testing models, applied to our algorithm, would be more sophisticated models, e.g., Armadillo and David beside common Cow and Dinopet. And the new approach would also guarantee to divide an arbitrary genus mesh into a collection of patches of genus zero. Especially, the algorithm would be resolution-independent, which means that the same segmentation is achieved at different levels of detail (i.e., Fig. 1, two Armadillo meshes in different fidelity are decomposed into similar components, although segmented separately). Consequently, we simultaneously use the skeleton and surface convex regions to perform shape decomposition. In a nutshell, the partitioning algorithm can be roughly described in two stages. First, the hierarchical skeleton (Fig. 1.e) of a mesh is computed by using a repulsive force field (Fig. 1.d) over the discretization (Fig. 1.c) of a 3D mesh (Fig. 1.a and 1.b). And for every skeleton level, the cutting critical points (the larger points in Fig. 1.e) are preliminarily identified by geometric and topological functions. Second, near each critical point, corresponding final boundary is obtained using local feature contours in valley regions. As a result, our algorithm can automatically partition a mesh into meaningful components with natural boundary among parts (Fig. 1.f and 1.g).

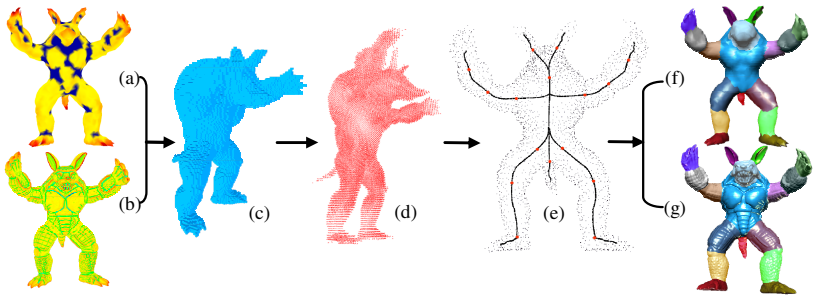

Fig. 1. Main steps of our mesh segmentation algorithm. (a)low level-of-detail Armadillo with 2,704 vertices (b)high level-of-detail Armadillo with 172,974 vertices (c)voxelized volume representation in 96^3 grids (d)the corresponding repulsive force field (e)core skeleton with cutting critical points (the larger red points) (f)corresponding segmentation of low level-of-detail Armadillo (g)corresponding segmentation of high level-of-detail Armadillo.

The rest of the paper is structured as follows. Cutting critical points are located in the section 3 based on the core skeleton extracted in section 2, and then the cutting path completion mechanism is illustrated in section 4. Section 5 demonstrates some results and compares them with related works. Finally, section 6 makes a conclusion and gives some future researching directions.

2 Core Skeleton and Branch Priority

2.1 Core Skeleton

In our approach, the skeleton of a mesh is generated by directly adapting a generalized potential field method [18], which works on the discrete volumetric representation [20] of the mesh. In [18], core skeleton is discovered using a force following algorithm on the underlying vector field, starting at each of the identified seed points. At seed points, where the force vanishes, the initial directions are determined by evaluating the eigen-values and eigen-vectors of the Jacobian. The force following process evaluates the vector (force) value at the current point and moves in the direction of the vector with a small pre-defined step (value σ, set as 0.2). Consequently, the obtained core skeleton consists of a set of points sampled by above process.

Once the core skeleton of the mesh extracted, a smoothing procedure, described in detail in [21], is applied to the point-skeleton to alleviate the problem of noise. Basically, this procedure expands the fluctuant skeleton to a narrow sleeve, defined by each point's bounding sphere with radius in the certain threshold value σ, then fines the shortest polygonal path lying in the 2σ wide sleeve. The procedure gives a polygonal approximation of the skeleton, and may be imagined as stretching a loose rubber band within a narrow sleeve. Subsequently, the position of each point in the original skeleton is fine-tuned by translating to the nearest point on the path.

2.2 Skeleton Branch Selection

According to the number of neighboring points, every point of the skeleton can be classified into three kinds: terminal nodes (one neighbor), common nodes (two neighbors) and branch nodes (more than two neighbors). In the paper, terminal points and branch points would be viewed as feature points, and any subset of the skeleton, bounded by the feature points, is called a skeleton branch. In the following sweeping process, all branches would be tested. It's important to determine the order of the branches, since our approach would like to detect the accurate cutting critical points by measuring related geometric and topological properties and the separated parts would not be taken account into subsequent computation of critical points. Basically, the ordering should allow small but significant components to be extracted first, so that they are not divided and absorbed into larger components in an improper way. We use three criteria to find the best branch: the type, its length and centricity.

- The type of a branch is determined by the classification of its two end points. The type weight of the branch with two branch nodes is low (value is 0), that with one terminal and one branch node is medium (value is 1), and that with two terminal nodes is high (value is 2).
- The centricity of a point t is defined as the average hops from t to all the points of the mesh's skeleton. In a mesh, let $maxH$ represent the maximum average hopping numbers among all the points, $maxH = max_t(avgH(t))$. We normalize the centricity value of vertex t as $C(t) = avgH(t)/maxH$.

For each branch b, which is a set of points, we define its priority $P(b)$ as its type value adding the product of the reciprocal length and sum of all normalized centricity of its points, since we treat the total number of points as its length.

$$P(b) = Type(b) + \frac{1}{Length(b)} \sum_{t \in b} C(t)$$

After a mesh has been partitioned based on the cutting critical points in the selected branch, the current centricity values of points are no longer valid in the following segmentation. Hence, we should re-compute the centricity values after each partitioning when we want to select another branch.

3 Locating Cutting Critical Points

Just as the principle observed by Li et al. [8], the geometrical and topological properties are two important characteristics, which distinguish one part from the others in mesh segmentation. We adapt the space sweep method (Fig. 2), used in computational geometry, to sweep a given mesh perpendicular to the skeleton branches (represented by points). Our approach prefers to disjoin parts along concave region. In the following, we will define a new function to measure the variety of the geometrical properties and find candidate critical points to identify the corresponding salient changes, rather than adapting the function defined in [8].

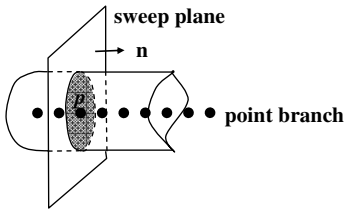

Fig. 2. The schematic of space sweep [8]

Let b be a selected branch. If b is a medium type branch, we will sweep it from its terminal point p_{start} to the other branch node p_{end}. And some points nearby the branch node are excluded from the scan, due that no effective cutting critical points lie in the neighbouring region. In the paper, the nearby region is a sphere, whose centre is the current branch node and the radius is the minimal distance from the point to the nearest vertex on the surface. Under other conditions, b is a high or low type branch, start to sweep from the end point p_{start} with larger cross-section area.

For the selected branch b, we compute the area of cross-section at each point p on the sweeping path from p_{start} to p_{end-1}, and then define our geometric function as following:

$$G(p) = \frac{AreaCS(p+1) - AreaCS(p)}{AreaCS(p)}, p \in [p_{start}, p_{end-1}]$$

To accelerate the computation of cross-section area at each point, we approximately calculate it by summing up the number of the voxels that are intersected by the perpendicular sweeping plane of the current point.

By lining the dots of *G(p)*, we get a connected polygonal curve. The *G(p)* curve has one general property: it fluctuates in the way that there are a few outburst pulses on the straight line very close to zero. Three kinds of dots would be filtered out as accurate as possible. Fig. 3 shows the three kind sample profile of *AreaCS(p)* and *G(p)*. Based on the *AreaCS(p)*, it's obvious that Fig. 3.a denotes a salient concavity, Fig. 5.b and 5.c respectively mean the fact that how a thin part is connecting with another thick part. In the *G(p)* curve, if the rising edge of one pulse goes through the *p* axis and its trough is less than -0.15 (relative to Fig. 3.a or 3.c), the cross point *t* would be certainly selected. In addition, the peak of positive pulse (relative to Fig. 3.b), whose value is more than a threshold (0.4), would be obtained. The points located in the skeleton, corresponding to dots *t* as indicated by dashed lines, are marked as cutting critical points to divide the original mesh.

The segmentation based on *G(p)* can handle L-shaped object (Fig 4.a), which is just the ambiguity of the minima rule theory. However, it is not practical to directly treat all selected points found by before checking procedure (Fig. 4.b), lying in the turning space region, as real critical points, since the straight absorption would lead to over-segmentation for the type object, as shown in Fig. 4.c. Hence, we should avoid some over-parts by excluding useless candidate points from the set of critical points.

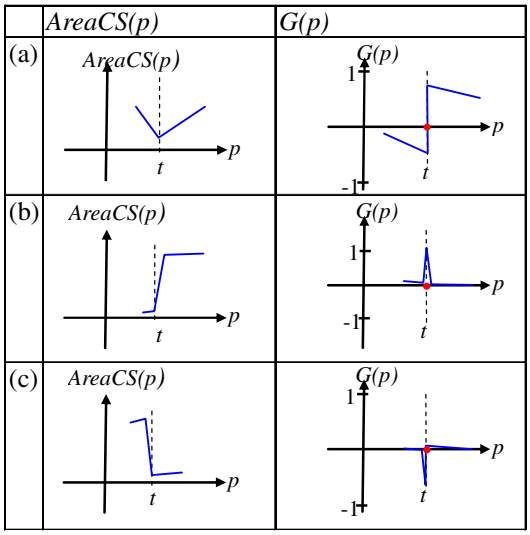

Fig. 3. Three kind sample profiles of *AreaCS(p)* and *G(p)*

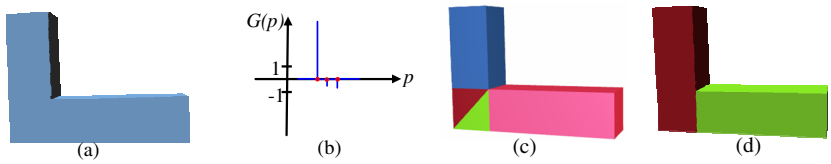

Fig. 4. The segmentation of L-shaped object. (a)the object can't be partitioned by watershed algorithm [13] based on minima rule (b)the corresponding profile of *G(p)* with candidate critical dots in red color (c)the over-segmentation problem (d)the final result of our algorithm.

The exclusion is performed by checking whether three nearby candidates are located in a same space. The neighbourhood, defined as earlier, a detecting sphere whose centre is the current testing node and radius is the minimal distance from the point to the nearest vertex on the surface. And if the nearby phenomenon happens, the middle point is definitely discarded and the one of two side point (preferring to Fig. 4.c type) is preserved as a real candidate in the paper. Therefore, the over-segmentation problem would be effectively resolved, and the natural result is gotten as Fig. 4.d.

4 Cutting Path Completion

Common methods, such as [8], scissor a mesh by the cutting planes, which is perpendicular to the orientation of related critical points. However, the critical points can only be able to capture coarse characteristics and may not have an exact and smooth boundary between different parts. Hence, we use the cutting critical point to find

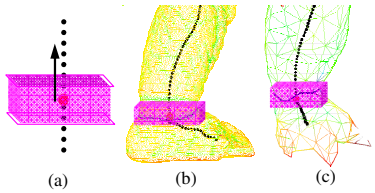

Fig. 5. The operational principle sketch of our segmentation. (a)the restricting zone, built on a larger red critical point, is formed by two parallel planes, which are perpendicular to the direction of the critical point and distance to the point in a threshold value. (b)(c)the feature contour, located at the armadillo's ankle in various resolution meshes, is used to refine the boundary.

primary position, and the ultimately boundary is refined based on the feature contours of the underlying valley surface. Therefore, the skeleton-based and minima-rule-based algorithms are consistently embodied by the paper, whose operational principle is sketched out in Fig. 5.

On one hand, we employ local medial axis, when partitioning a mesh. Fortunately, we have computed its skeleton, and found the cutting position marked by the critical points. Therefore, we can determine the segmentation regions on the mesh's surface, enclosed by restricting zones. For example, the ankle of the armadillo is enclosed by a restricting zone, shown as Fig. 5.b and 5.c. One zone (Fig 5.a) is sliced by two parallel planes, whose normal is identical to the direction of the corresponding critical point, and both planes keep a same distance to the critical point in a threshold d value.

$$d = \begin{cases} 2 * \sigma, & if \ \sigma > LNG_{edge} \\ 2 * LNG_{edge}, & else \end{cases}$$

where, value σ, defined in section 2, is the distance between the adjacent skeleton points, and LNG_{edge} is the average edge length of the mesh.

On the other hand, we prefer to divide a given mesh into disjoint parts along the concave region. Therefore, if one concave region lies in the previous restricting zone, we are inclined to extract the cutting boundary from the region. For instances, the Fig 5.b and 5.c demonstrate that the dark blue contour, located in the restricting pink zone, is used to get natural perceptual boundary between foot and leg of the armadillo model. Similar to the [22], we use proper normalization to unify the minima curvature value of each mesh's vertex, obtain the concave feature regions by filtering out the vertices with higher normalized value, extract contour curves from the graph structures of the regions (e.g. the blue regions in Fig. 1.a and 1.b) and complete best curve path going over the mesh in the shortest way. We refer readers to [22] for details regarding the feature contour extraction and completeness. For every feature contour, we compute its main direction based on principle component analysis of its vertices. But only the feature contour, whose main direction is approximate to the orientation of the corresponding critical point, is treated as one boundary curve. The approximation is measured by the angle between them. And if the separation angle is less than $\pi/4$ in radian, we say that they are approximate. Note that, if there is none concaving

contour locating in a restricting zone, the corresponding cutting critical point can be removed and no partitioning action happens.

5 Results and Discussion

5.1 Results

Fig. 6 demonstrates the final decomposition of two-level David and Buddha in 256^3 grids. As shown, our algorithm respects the more likely segmentation that a human observer would choose for the scenes, and it is resolution-independent. The voxelized resolution is an important external factor affecting the skeleton, since it defines the precision of the repulsive force fields and determines the computing time and memory requirement. It is evident that a 10^3 grid will yield a less accurate result than a 100^3 grid. However, it doesn't mean that the finer resolution is the better, in view of the application request and algorithm complexity. Especially, for different level-of-detail of a mesh, the volume representation is always similar, if the voxelized resolution is lower than 256^3.

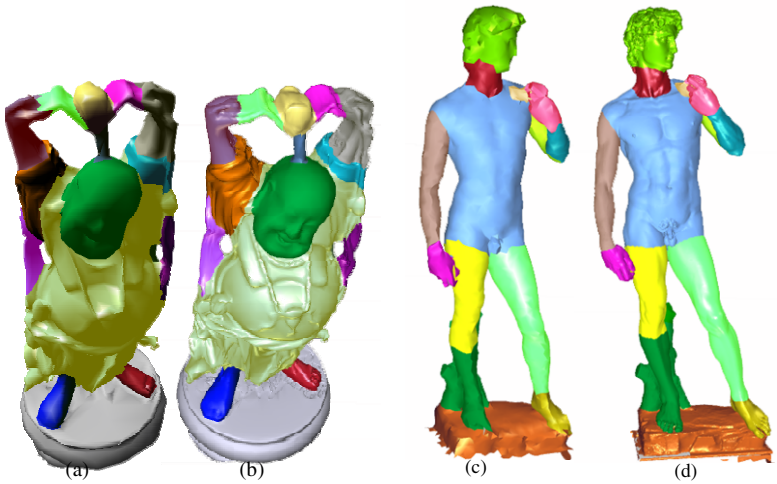

Fig. 6. Segmentation instances in 256^3 grids. (a)low level Buddha with 10,000 vertices (b) high level Buddha with 100,000 vertices (c)low level David with 4,068 vertices (d)high level David with 127,465 vertices.

5.2 Comparison and Discussion

Fig. 7 begins to give a comparison of the visual effect between the state-of-the-art minima-rule-based segmentation [15] and our algorithm using the core skeleton. For the Disonaur mesh, the over-segmentation would definitely happen in [15] and [22](Fig. 7.a), while the problem disappears in the paper. And then, we compare our results with the typical skeleton-based algorithm [8] by Dinopet and Hand meshes in Fig. 8. It's obvious that the cutting boundaries of all parts have been improved in the both models.

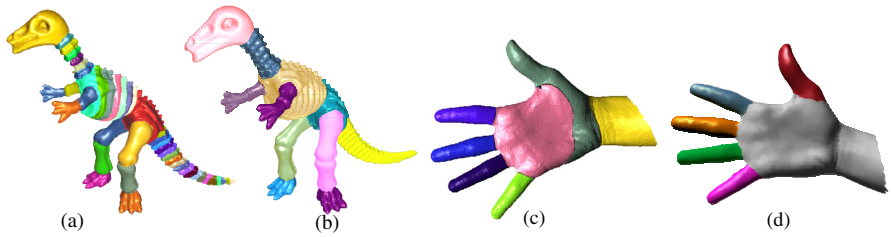

Fig. 7. The comparison with minima-rule-based algorithm [15]. (a)possible over-segmented Disonaur with 56,194 vertices generated by [15] and [22] (b)Disonaur with 56,194 vertices partitioned by our algorithm (c)Hand in [15] with 10,070 vertices (d)Hand with 5,023 vertices segmented by our approach.

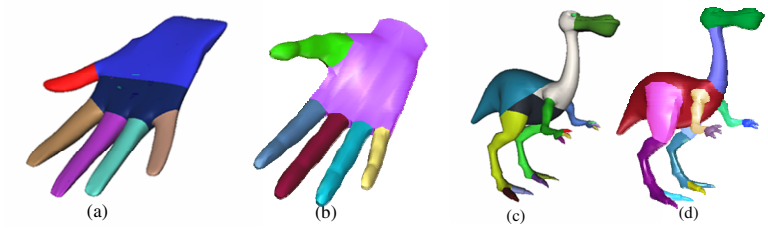

Fig. 8. The comparison with skeleton-based algorithm [8]. (a)Hand in [8] (b)likely Hand with 1,572 vertices segmented by our approach (c)Dinopet in [8] (d)Dinopet with 4,388 vertices partitioned by ours.

6 Conclusion

In this paper, we have developed an algorithm to decompose a mesh into meaningful parts, which integrates the advantages of skeleton-based and minima-rule based segmentation algorithms. In a nutshell, on one hand, our algorithm assures the cut would be smooth and follow natural concaving regions as much as possible, on the other hand it uses the more robust skeleton of the mesh and isn't sensitive to surface noises any more.

References

[1] Shamir, A.: A Formulation of Boundary Mesh Segmentation. In: Proceedings of 3DPVT, pp. 82–89 (2004)
[2] Shamir, A.: Segmentation Algorithms for 3D Boundary Meshes. In: Proceedings of EuroGraphics, Tutorial (2006)
[3] Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., Tal, A.: Mesh Segmentation - a Comparative Study. In: Proceedings of SMI, pp. 14–25 (2006)
[4] Zhang, E., Mischaikow, K., Turk, G.: Feature-Based Surface Parameterization and Texture Mapping. ACM Transactions on Graphics 24(1), 1–27 (2005)

[5] Katz, S., Tal, A.: Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Transactions on Graphics 22(3), 954–961 (2003)
[6] Lien, J.M., Keyser, J., Amato, N.M.: Simultaneous shape decomposition and skeletonization. In: Proceedings of the ACM SPM, pp. 219–228 (2006)
[7] Huang, J., et al.: Subspace gradient domain mesh deformation. ACM Transactions on Graphics (Special Issue: Proceedings SIGGRAPH) 25(3), 1126–1134 (2006)
[8] Li, X., Toon, T.W., Huang, Z.: Decomposing polygon meshes for interactive applications. In: Proceedings of ACM Symposium on Interactive 3D Graphics, pp. 35–42 (2001)
[9] Attene, M., Falcidieno, B., Spagnuolo, M.: Hierarchical Mesh Segmentation Based-on Fitting Primitives. The Visual Computer 22(3), 181–193 (2006)
[10] Podolak, J., Shilane, P., Golovinskiy, A., Rusinkiewicz, S., Funkhouser, T.: A Planar-Reflective Symmetry Transform for 3D Shapes. ACM Transactions on Graphics, 25(3), 549–559 (2006)
[11] Katz, S., Leifman, G., Tal, A.: Mesh Segmenation Using Feature Point and Core Extraction. The Visual Computer (Special Issus: Pacific Graphics) 21(8-10), 649–658 (2005)
[12] Liu, R., Zhang, H.: Segmentation of 3D Meshes Through Spectral Clustering. In: Proceedings of Pacific Graphics, pp. 298–305 (2004)
[13] Page, D.L., Koschan, A.F., Abidi, M.A.: Perception-based 3D triangle mesh segmentation using fast marching watersheds. In: Proceedings of IEEE Computer Vision and Pattern Recognition (CVPR), vol. II, pp. 27–32 (2003)
[14] Page, D.L., Abidi, M.A., Koschan, A.F., Zhang, Y.: Object representation using the minima rule and superquadrics for under vehicle inspection. In: Proceedings of the IEEE Latin American Conference on Robotics and Automation, pp. 91–97 (2003)
[15] Lee, Y., Lee, S., Shamir, A., Cohen-Or, D., Seidel, H.-P.: Mesh Scissoring with Minima Rule and Part Salience. Computer Aided Geometric Design 22, 444–465 (2005)
[16] Hoffman, D., Richards, W.A.: Parts of recognition. Cognition 18, 65–96 (1984)
[17] Hoffman, D., Signh, M.: Salience of visual parts. Cognition 63, 29–78 (1997)
[18] Cornea, D.N., Silver, D., Yuan, X.S., Balasubramanian, R.: Computing Hierarchical Curve-Skeletons of 3d Objects. The Visual Computer 21(11), 945–955 (2005)
[19] Cornea, D.N., Silver, D., Min, P.: Curve-Skeleton Applications. In: Proceedings of IEEE Visualization, pp. 23–28 (2005)
[20] Dachille, F., Kaufman, A.: Incremental Triangle voxelization. In: Proceedings of Graphics Interface, pp. 205–212 (2000)
[21] Kalvin, A., Schonberg, E., Schwartz, J.T., Sharir, M.: Two Dimensional Model Based Boundary Matching Using Footprints. International Journal of Robotics Research 5(4), 38–55 (1986)
[22] Cheng, Z.-Q., Liu, H.-F., Jin, S.-Y.: The Progressive Mesh Compression based-on meaningful segmentation. The Visual Computer 23(9-11), 651–660 (2007)