Eurographics Italian Chapter Conference (2008)
V. Scarano, R. De Chiara, and U. Erra (Editors)

Combining Segmentations for Understanding and Annotating 3D Objects

N. Carissimi and L. Papaleo and L. De Floriani

Department of Information and Computer Science, University of Genova, Genova - Italy

Abstract

In the last years, 3D objects have become widely available and are used in many application domains. Thus it is becoming fundamental to develop techniques for extracting and maintaining the knowledge embedded into 3D models. Such knowledge usually cannot be simply identified and strongly depends on the specific application domain. In this paper, we present our work in combining segmentation and merging procedures in order to segment a manifold object. We also provide an interface for browsing the obtained decomposition discovering, for each components, the associated properties. This activity has been carried out as part of our semantic web system called `be-SMART`*.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling: Curve, surface, solid and object representations

1. Introduction

In the last few years, the amount of digital audio-visual information has become huge and rapidly increasing. This data is available under digital libraries and information repositories in a number of different formats including pictures, video, audio and also three-dimensional models. 3D objects play an important role in application domains like computer-aided manufacturing, design, science and entertainment. Efficient and effective methods to manage this data are crucial in making optimal use of it.

The most common and simple representation for a 3D object consists of a mesh of triangles joining points belonging to the object boundary. 3D models have a richer information content with respect to 2D images, as for example geometrical and topological information. Beyond geometry and topology, there is no semantics associated with a 3D object by default. Therefore, 3D models must be analyzed and successively semantically annotated, in order to improve the expressiveness of their representations. Semantic annotation consists of information about which *entities* (or, more generally, semantic features) appear in a 3D model and where they appear. A fundamental step in annotating a triangulated model is to segment it into *meaningful portions*. As pointed out in [aim07, HWGS*06], at this stage of research evolution, there is a general request of tools capable of extracting semantics from objects (e.g., automatic or semi-automatic annotation tools), and to enhance digital representations with context-dependent metadata. In Computer-Aided Design (CAD), for example, *form features* extraction (e.g. through holes or handles) is the basic step in order to recover semantic information from an object model [CLG07]. In such application, we also need to deal with objects having a complex shape, like non-manifold and non-uniformly dimensional objects. These are often the result of an idealization process applied to manifold objects performed in order to reduce their size. So it is necessary to develop a system for understanding and annotating models with a very complex shape, non-manifold singularities and parts of different dimensions.

For the above reasons, we designed a system, called `be-SMART` (*BEyond Shape Modeling for understAnding Real world represenTations*), for inspecting models representing manifold and non-manifold 3D objects (extracting geometrical and topological information) and for structuring and annotating them using ontology-driven metadata [DHP*07]. Here, we focus on the segmentation functionalities of the system.

We have developed a module which is able to combine segmentation and merging techniques for the analysis of manifold objects. It uses two partitioning techniques which have been proven to be suitable for CAD models, namely Variational Shape Approximation [CSAD04] and Face Clustering [She01]. Additionally, since a fully automatic segmentation process rarely produces *optimal* results, the module provides a simple manual editing functionality that, combined with the automatic process allows the user to produce meaningful results. The segmented model is structured in a graph-based representation, that we called a *segmentation graph*, in which the nodes represents the identified portions of the model and the arcs encodes the adjacencies among these portions. The system interface provides functionalities to visualize both the segmented model and the segmentation graph by zooming on a specific node and by browsing the graph discovering geometrical properties of each region of the model. For non-manifold models, this module can be used after the decomposition of these models into manifold parts joined by non-manifold elements as described in [DHP*07]. In such case, the segmentation module will work on the identified manifold components, allowing further decomposition of the model working on these manifold parts.

The reminder of this paper is organized as follows. Section 2 provides an overview of segmentation methods existing in literature. Section 3 describes the general structure of `be-SMART`, while the segmentation methods we used are described in Section 4. Section 5 discusses our approach providing implementation details. In Section 6 we show results of our implementation and in Section 7 we describe the user interaction with the system. Finally, in Section 8 some concluding remarks are drawn.

2. Related Work

Mesh segmentation is arguably the most ubiquitous and difficult technical problem in 3D model understanding. The problem of partitioning a mesh model into *meaningful pieces* or, alternatively, extracting regions of interest, has been proven to be difficult as other computational problems that attempt to mimic the capabilities of human intelligence, or perception. The ongoing difficulty of shape segmentation is not from a lack of attention; thousands of papers and theses describe a wide variety of approaches. Mesh segmentation assists parametrization, texture mapping, shape matching, morphing, multiresolution modeling, mesh editing, compression, animation and more. Moreover, shape understanding and semantic-based object representation rely on feature extraction [aim07].

Algorithms developed for segmentation borrow techniques from related fields such as image segmentation, unsupervised machine learning and others. A complete survey on existing segmentation methods can be found in [Sha08]. The segmentation problem can be formulated as an *optimization problem*: given a mesh M and the set of elements $S = \{V, E, F\}$, vertexes, edges and faces of M respectively, the problem of segmenting M is equivalent to the problem of finding a disjoint partitioning S into $S_0, \ldots, S_{k-1}$ such that a given *criterion function* $J = J(S_0, \ldots, S_{k-1})$ is minimized (or maximized) under a set of specific constraints C. Shamir [Sha08] provides an effective classification of the existing segmentation techniques into *part-type segmentations*, where the goal is to segment the object represented by the mesh into meaningful, mostly volumetric, parts, and *surface-type segmentations*, where the objective is to partition the surface mesh into patches under some specific criteria.

In particular, focusing on CAD applications, surface-type segmentation is used in remeshing and simplification as, for example, in [EDD*95, KT96, She01, BM03, CSAD04]. In most of these works, each patch is replaced either by one or by a set of planar polygons. Hence planarity is the desired property of the patches. More recently, other types of patches have been used, e.g. spherical, cylindrical or rolling ball blends [WK05, AFS06]. Part-type segmentation objective is rooted in the study of human perception. Examining human image understanding many works indicate that recognition and object understanding are based on structural decomposition of the object shape into smaller parts.

Segmentation methods can be further classified based on their segmentation strategies into: region-growing techniques (see [KT96, LDB05]), multiple source region grow methods [SPP02, ZH04], hierarchical clustering [GWH01, She01], iterative clustering [CSAD04, GG04], spectral analysis [KG00, LZ07] and implicit methods [LLS*05, LKA06].

Also, several manual or user-guided segmentation techniques have been proposed in the literature as for example [FKS*04, JLCW06]. We are currently implementing one of these techniques as part of our system.

In [ARSF07], the concept of *multi-segmentation* is introduced and used in a system, which performs a part-based annotation of 3D models guided by domain-specific ontologies. In order to identify surface features, this system provides a set of segmentation algorithms and allows the user to select regions from the various obtained segmentations.

In the module presented in this work, we focus more in details on CAD representations (and needs) and we use two specific surface-type segmentation methods suitable for such application domain. Additionally, the module provides a simple manual editing functionality, since fully automatic segmentation still remains an unreached goal when segmentation is used as the basis for attaching semantic information.

3. Be-SMART: General Description

Extending the idea of `PhotoStuff` [HWSG*05], `be-SMART` is designed to be a modular and platform-independent system for geometric-topological inspection and semantic annotation and structuring of 3D models. It relies on the general idea of automatically extracting information about features and regions of interest and of providing an intuitive interface to researchers in order to easily understand digital models [DHP*07]. This means that the different modules act as a team generating the final annotated model following specific *ontology-driven processes* [Pap06]. `be-SMART` consists of the following modules:

1. *Geometry and Topology Analyzer (GTA)*: it analyses the input model and extracts geometrical/topological information such as the number of vertexes, faces, the number of connected components.
2. *Topological Decomposer (TD)*: starting from the information extracted with the *GTA* the *TD* produces a graph-based representation of the model, the *decomposition graph*, which groups geometric components (namely, vertexes, edges and triangles) and associates context-independent semantic meaning to each group.
3. *Manual Segmentation module (MS)*: it offers advanced editing functionalities allowing the user to select portions of the mesh and to annotate them
4. *Automatic Segmentation module (AS)*: it offers the possibility to apply different segmentation algorithms for decomposing the model into meaningful parts.
5. *Semantic Annotator (SA)*: it offers the capabilities of associating specific metadata values to portions of the model according to preloaded ontologies. This means that the tool associates metadata to nodes of the *decomposition graph* representation of the model.

The first two modules, described in [DHP*07], transform a non-manifold model into manifold parts connected together by singularities (vertexes or edges) allowing to segment a complex non-manifold 3D model into context-independent meaningful portions. For reasoning and annotating the initial model, these modules uses the *decomposition graph*. This graph has the nodes corresponding to the components of the decomposition, and the arcs capturing the structure of the connectivity among these components (see also [DH07]). In the following sections we will concentrate on the techniques we have developed for the manual and automatic segmentation modules.

4. Two Segmentation Methods

Here we describe the two segmentation and clustering methods we have used in `be-SMART`, namely the Variational Shape approximation (VSA) presented in [CSAD04] and the Face Clustering method [She01]. Variants and improvements of the method in [CSAD04] exist in the literature, as for example [WK05]. These two methods relies on the fact that they have been proved to be suitable for recognizing meaningful parts in CAD models. We use the VSA code freely available, while we fully implemented the approach presented in [She01] adapting it to our framework.

4.1. VSA: Variational Shape Approximation

The VSA method presented in [CSAD04] is an *iterative clustering* segmentation algorithm. Given a detailed 3D mesh as input, the algorithm partitions the mesh in planar regions that captures the geometry of the surface and approximates these regions with polygons of various shapes.

Given a partition R of a mesh, the key idea is that every region $R_i \in R$ can be represented by a pair $P_i = (X_i, N_i)$, where X_i is the average center of R_i and N_i is the average normal. P_i is called a *shape proxy* of the region R_i. Thus, for any given partition of a surface in k regions, there is a set $P = \{P_i\}, i = 1 \dots k$, of shape proxies that approximate the whole geometry. Partition R defines a *dual meta-mesh* [CSAD04] of the original: every shape proxy is a meta-face and the connectivity of the regions R_i defines the topology of the new mesh. In order to measure the quality of the approximation, a new metric $L_{2,1}$ is introduced, as an alternative to the standard L_2 error metric (see [CSAD04] for details). The partition algorithm is basically an extension of Lloyd's method and it consists of the following macro-steps:

- *seeding*: create the set of seeds from which the region growing process will start.
- *partitioning*: a region R_i is grown on the basis of specific conditions from every seed s_i, starting with the insertion in R_i of the three adjacent triangles for s_i. The algorithm uses a priority queue Q, where the triangle T_i priority is equal to a specific distortion error E_{T_i} When Q is empty, the partition R is created.
- *fitting*: For every region $R_i \in R$ the corresponding proxies $P_i = (X_i, N_i)$ are updated in order to minimize the error $E(R_i, P_i)$; X_i is the barycenter of R_i, while N_i depends on the used metric: for the L_2 metric, N_i is the direction indicated by the eigenvector associated with the smallest eigenvalue of the covariance matrix of the region; for the $L_{2,1}$ metric the proxy normal is the average of the triangles' normals of the region weighted by means of the area.

The strengths of the VSA method are twofold. Locally, the iterative technique is very sensitive to anisotropic bending of the surface which leads to an almost perfect alignment of the surface partition to the principal curvature directions. Globally, the technique detects and merges planar regions which can be approximated by a single flat proxy.

4.2. Face Clustering

The Face Clustering technique presented in [She01] is a *hierarchical clustering* method. In this case the focus is on CAD models. A CAD model is usually composed by discrete and regular elements (faces), but usually the number of these faces is more than necessary (over-segmentation problem), so a simplification procedure is necessary.

Given a CAD model of a manifold object, the algorithm initially considers every face of the model as a cluster C_i and at every step merges pairs of clusters (C_i, C_j) creating a new cluster C_{ij}. The choice of the clusters to be merged is done according to specific *weights* assigned to the boundary of each cluster. These weights are computed combining geometric properties of the edges and faces sharing the boundaries (see [She01] for details). Once the initial clustering phase is concluded, the method enters in the *face collapse* phase verifying if there exist *collassable* clusters, on the basis again of the weights of the cluster boundaries.

The weight W_a related to the boundary of a region is computed as combination of specific geometric criteria. In particular, one criterion prevents the union of clusters with *sharp corners* between them; another guarantees that the dimension of the clusters is sufficiently big compared to the element size of the model (avoiding small regions). Another criterion promotes the union between clusters that share long boundaries as the union between clusters when the angle at the extreme vertexes of their shared border is obtuse. Also the measure of the curvature's change between the clusters is considered as a criterion as the fact that clusters cannot be merged if they form a non manifold structure.

The algorithm checks all the boundaries and inserts them in a priority queue with the associated weights. It removes a boundary from the queue, merges the related regions if specific conditions are satisfied, until the queue is empty.

5. Our Approach

In this section we present our approach for combining the two methods described above that we have implemented in `be-SMART`. We present the workflow the user can perform, the data structures and the clustering algorithm implemented.

For the sake of simplicity, we restrict our attention to the case in which the input model is a manifold mesh without boundary. So the system runs the *GTA* module and passes directly to the Automatic segmentation module (without considering the TD which works for non-manifold meshes, Section 3). Note that, in case of non-manifold models, the Automatic segmentation module will work on *portions* of the initial model decomposed into manifold parts by the TD module. This means that the segmentation modules (automatic and manual) will work on the nodes (manifold components) of the *decomposition graph* built by the TD module. The general workflow is the following:

- the system uses the VSA method as a *service* producing and visualizing a segmented mesh according to the planarity criteria as explained in Section 4.1.
- the system initializes the segmentation graph $G = (N,A)$ related to the obejct encoding in its nodes N the segmentation clusters and their adjacency relations in A.
- then, the clustering algorithm is initialized by computing, for every arc $a \in A$ the relative weight W_a related to the cluster boundaries and by creating a priority queue Q containing all the arcs.
- then, one or more steps of the clustering algorithm can be executed until Q is empty.
- finally, the user can manually refine the segmentation through the implemented editing functionalities by splitting and merging clusters, until the final segmentation is obtained.

The input model is encoded in the standard indexed data structure with adjacencies, while the graph $G = (N,A)$ is represented using a standard adjacency list with additional information necessary for the clustering algorithm. Recall that in G, a node n_i represents a cluster C of the model and an arc $a = (n_i, n_j)$ represents the adjacency relationship between the clusters C_i and C_j and, thus, their shared boundary. In particular:

- Each node $n_i \in N$ contains an integer i corresponding to the index of the associated segmented region R_i (and the region R_i is encoded again using the indexed data structure with adjacencies) plus additional information.
- Each arc $a_j \in A$ contains the list of edges forming a_j, two pointers to its extreme nodes n_i, n_k, the weight W_a associated to the arc and a Boolean U for determining if the arc is contractable or not, namely if the regions sharing the border a_j can be merged.

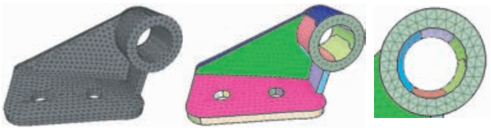
The weight W_a is computed as the combination of the geometric criteria presented in section 4.2, and here encoded by weights.

Once the segmentation graph $G = (N,A)$ has been created and all the arcs have the associated weights, the system applies the clustering algorithm simply performing a *sequence of contractions* of the arcs in the graph G. The merging of a pair of clusters C_i and C_j is done by contracting the corresponding arc $a = (n_i, n_j)$ in the graph G. This operation removes arc (n_i, n_j) from G, merges nodes n_i, n_j into a new node n_{ij} and transform the arcs incident in n_i and in n_j into arcs incident in node n_{ij}, inheriting the conditions and weights of the arcs and nodes involved.

6. Results

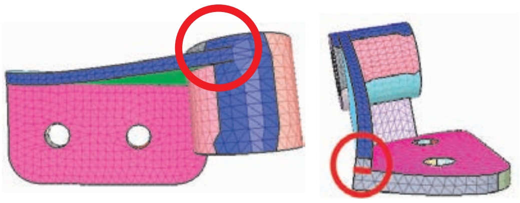
In this Section we present some results obtained by combining the segmentation and clustering methods presented in Section 4. Figure 1(a)-(b) shows the various steps of the segmentation of a hinge model: in (a) the original model is shown, in (b) the segmentation obtained with the VSA algorithm is depicted. Here flat surfaces are segmented with a single region, while cylindrical surfaces are segmented as several striped regions (see Figure 1(c)). As we can see in Figure 2, using few proxies can lead to a segmentation, where a region connects a part of a cylindrical surface to a flat surface. This happens because a set of triangles that belong to the cylindrical surface and a set of triangles that belong to the flat surface are assigned to the same proxy.

Figure 3 shows the refined segmentation with the clustering algorithm implemented in our application: cylindrical surfaces are partitioned with fewer regions, typically three



(a) (b) (c)

Figure 1: *Segmentation of a hinge model using the VSA method. (a) the original model, (b) the segmentation using* $L_{2,1}$ *metric, (c) flat regions segmented in a single region.*



(a) (b)

Figure 2: *Two results of the application of the VSA method. (a) using few proxies region connects part of a cylindrical surface to the flat surface. (b) a region goes out from the optimal border.*

or four (see Figure 3(b)), and some sides of the object's base are connected (see Figure 3(c)-(d)). We also prevented the algorithm to merge the three regions shown in Figure 3(e) via the interface. Figure 4 shows a final segmentation obtained through manual refinement: cylindrical surfaces are now segmented as a single region (see Figure 4(b)); the blue region shown in Figure 2(a) has been split in two parts (see Figure 4(c)), as the purple region in Figure 3(d) (see Figure 4(d)).

7. User Interaction

In this section, we briefly describe the interface of the segmentation module (see Figure 5), focusing on the way in which the user can interact with the clustering algorithm and can perform the manual segmentation.

The interface area that gives control to the user on the clustering algorithm is shown in Figure 5(top-right): in the top area we can find the buttons that, bootstrap the algorithm and execute a single step. In the text area labeled `fit-time(s)` the user can select the number of steps to be executed. The rest of the area is used by the user to assign parameters that will be applied in the computation of the weights associated to the arcs of the graph G. Figure 6 shows the merging of two clusters after the execution of a step of the algorithm.

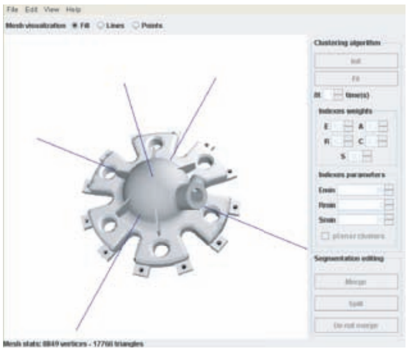


Figure 5: *The GUI of the application with a zoom on the buttons related to the face clustering algorithm (top-right) and those related to the manual editing operations (bottom-right).*

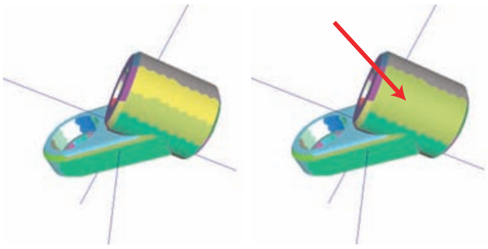


Figure 6: *From left to right: execution of a step of the clustering algorithm and merging of two clusters.*

In Figure 5(bottom-right) the interface area for the manual refinement operations is shown. The `Merge` button forces the merging of two selected clusters (Figure 7-first row). The user selects the first cluster C_1, then the second C_2 and merges them. The `Split` button splits a cluster C into two parts, along a line selected by the user (Figure 7-second row). The user selects C and presses the button: to make the operation easier, all the other clusters are made invisible, the user selects the vertices on C for the construction of the boundary which will be shared by the two new clusters C_1 and C_2, then then the new clusters are created. Finally, the `Do not merge` button prevents two clusters to be merged by the clustering algorithm. The execution steps for this operation are similar to those for the merge operation.

The interface we developed offers also a visual tool for the topological analysis of the obtained segmentation. Once a region has been selected, it is possible to visualize a graphical representation of the corresponding node in the segmentation graph (see Figure 8(a)). The user can select one of the arcs incident in the central node and visualize related data, such as the associated geometric index (see Figure 8(c)), or navigate the graph by selecting adjacent nodes (see Figure 8(c)-(d)).

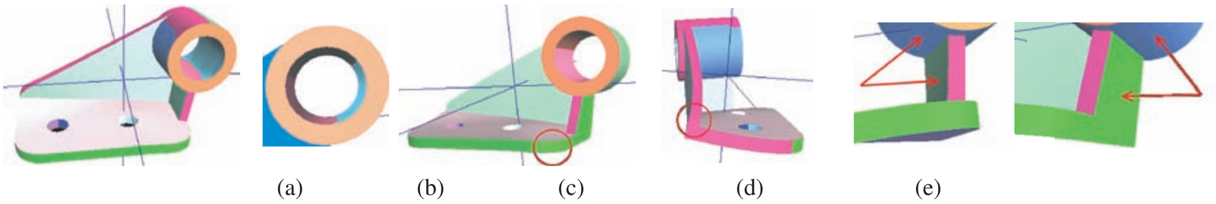

(a) (b) (c) (d) (e)

Figure 3: *(a) A refined segmentation with the face clustering algorithm implemented. (b) Cylindrical surfaces are partitioned with fewer regions. (c) and (d) some sides of the object's base are connected. (e) our application prevented the algorithm to merge three regions via the interface.*

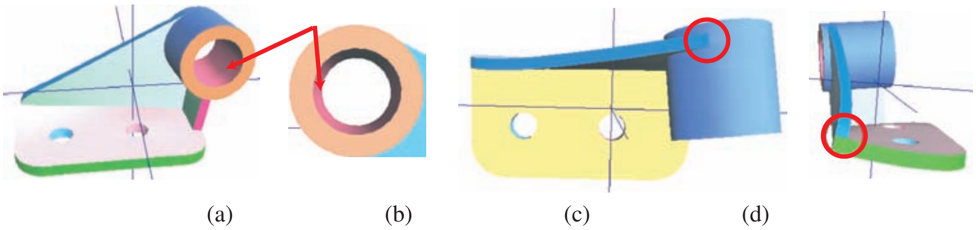

(a) (b) (c) (d)

Figure 4: *A possible final segmentation obtained through a manual refinement (a). (b) cylindrical surfaces are segmented; (c) the blue region in (a) has been split in two parts, as the fuchsia region (d).*

8. Concluding Remarks

We have presented a Semantic Web environment, `be-SMART`, for inspecting 3D models and for structuring and annotating them according to ontology-driven metadata. The system is at an initial step of development. Since segmentation is the basis for semantic annotation, our contribution here is inherent to the step of segmentation of a manifold mesh without boundary. We present also how our approach is integrated in `be-SMART`.

In particular, we focused on results for the automatic and manual segmentation modules of `be-SMART`. We have shown how we are able to combine a segmentation and a clustering algorithm, namely the Variational Shape Approximation [CSAD04] and the Face Clustering [She01]. We are currently working in order to integrate other segmentation techniques in the module to allow multiple combinations. We have been recently focusing on applications to CAD models and thus we are investigating techniques which provide a segmentation of a manifold object by identifying handles and tunnels. In [DLS07], for example, handle and tunnel loops are extracted thus providing a segmentation of a triangulated model into form features. Moreover, since an *optimal* segmentation depends on context-specific application constraints (as for example for CAD application) we have implemented a first version of the manual segmentation module which allows the user to merge or split selected portions of the model. This functionality is simple and we are currently enhancing the editing capability of the module through user-guided partitioning techniques as the ones described in [FKS*04, JLCW06].

Finally, the graph visualization toolkit we have implemented allows the user to show and browse the segmentation graph providing a good support in understanding the structure of the segmented model. This tool will be useful also in the semantic annotation procedure. We started working on innovative human-computer interaction techniques to improve the visualization graph toolkit functionalities with the final goal to allow the visualization of both the identified component and the attached semantic information in a multilevel/multi-facet way.

Acknowledgments

This work has been partially supported by Project FIRB SHALOM funded by the Italian Ministry of Education, Research and University (MIUR) contract n.$RBIN04HWR8$ and partially by the Science and Technological Park of Liguria $POS.N.5 - Avv.1/2006$.

References

[AFS06] ATTENE M., FALCIDIENO B., SPAGNUOLO M.: Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer 22*, 3 (2006), 181–193.

[aim07] The European Network of Excellence AIM@SHAPE - contract number 506766. http://www.aimatshape.net, 2004-2007.

[ARSF07] ATTENE M., ROBBIANO F., SPAGNUOLO M., FALCIDIENO B.: Semantic annotation of 3d surface meshes based on feature characterization. In *Proc. of the International Conference on Semantic and Digital Media*

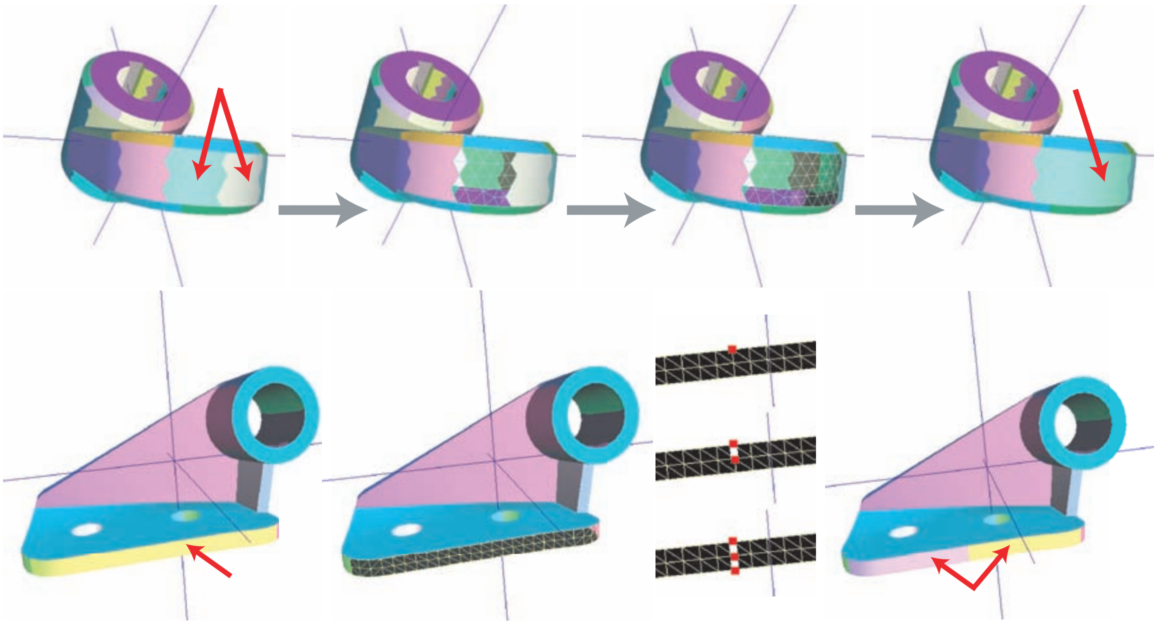
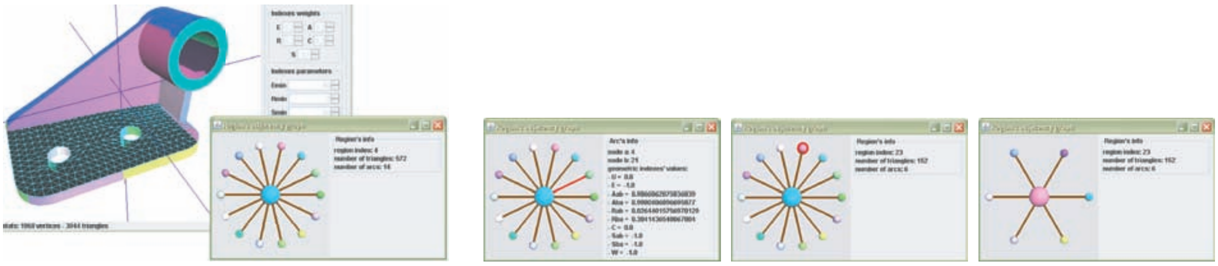


Figure 7: *First row (left to right): execution of the manual merging operation: segmented model, selection of the first cluster, selection of the second cluster, resulting segmentation. Second row (left to right): execution for the manual split operation: segmented model, cluster selection, construction of the border between the two new clusters and resulting segmentation.*



(a) (b) (c) (d)

Figure 8: *(a) Selected region in the main window, corresponding graph in the secondary window. (b)-(d) Visualization of data related to an arc incident to the central node; selection of an adjacent node, visualization of the associated information and visualization of its adjacence graph.*

Technologies (SAMT 2007) (2007), vol. 4816 of *Lecture Notes in Computer Science*, Springer, pp. 126–139.

[BM03] BOIER-MARTIN I. M.: Domain decomposition for multiresolution analysis. In *Proc. of the Eurographics/ACM SIGGRAPH symposium on Geometry Processing* (2003), Eurographics Association, pp. 31–40.

[CLG07] CROVETTO C., LEILADE FLORIANI, GIANNINI F.: Form Features in Non-manifold Shapes: A First Classification and Analysis . Eurographics Association, pp. 95–101.

[CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Trans. Graph. 23*, 3 (2004), 905–914.

[DH07] DE FLORIANI L., HUI A.: A two-level topological decomposition for non-manifold simplicial shapes. In *Proceedings of Solid and Physical Modeling Symposium* (Beijing - China, June 2007), ACM Press.

[DHP*07] DE FLORIANI L., HUI A., PAPALEO L., HUANG M., HENDLER J. A.: A semantic web environment for digital shapes understanding. In *Proc. of the International Conf. on Semantic and Digital Media Technologies (SAMT 2007)* (2007), Lecture Notes in Computer Science, Springer, pp. 226–239.

[DLS07] DEY T. K., LI K., SUN J.: On computing handle and tunnel loops. *International Conference on Cyberworlds (CW'07) 0* (2007), 357–366.

[EDD*95] ECK M., DEROSE T., DUCHAMP T., HOPPE H., LOUNSBERY M., STUETZLE W.: Multiresolution analysis of arbitrary meshes. *Computer Graphics 29*, Annual Conference Series (1995), 173–182.

[FKS*04] FUNKHOUSER T., KAZHDAN M., SHILANE P., MIN P., KIEFER W., TAL A., RUSINKIEWICZ S., DOBKIN D.: Modeling by example. *ACM Transactions on Graphics (Proc. SIGGRAPH)* (August 2004).

[GG04] GELFAND N., GUIBAS L. J.: Shape segmentation using local slippage analysis. In *Proc. of the Eurographics/ACM SIGGRAPH symposium on Geometry Processing* (2004), ACM, pp. 214–223.

[GWH01] GARLAND M., WILLMOTT A., HECKBERT P. S.: Hierarchical face clustering on polygonal surfaces. In *I3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics* (New York, NY, USA, 2001), ACM, pp. 49–58.

[HWGS*06] HALASCHEK-WIENER C., GOLBECK J., SCHAIN A., GROVE M., PARSIA B., HENDLER J.: Annotation and provenance tracking in semantic web photo libraries. In *International provenance and annotation workshop (IPAW)* (2006).

[HWSG*05] HALASCHEK-WIENER C., SCHAIN A., GOLBECK J., GROVE M., PARSIA B., HENDLER J.: A flexible approach for managing digital images on the semantic web. In *5th International Workshop on Knowledge Markup and Semantic Annotation* (Galway - Ireland, 2005).

[JLCW06] JI Z., LIU L., CHEN Z., WANG G.: Easy mesh cutting. *Computer Graphics Forum 25*, 3 (Sept. 2006), 283–292.

[KG00] KARNI Z., GOTSMAN C.: Spectral compression of mesh geometry. In *Siggraph Computer Graphics Proceedings* (2000), Akeley K., (Ed.), ACM Press SIGGRAPH - Addison Wesley Longman, pp. 279–286.

[KT96] KALVIN A. D., TAYLOR R. H.: Superfaces: Polygonal mesh simplification with bounded error. *IEEE Comp. Graphics and Applications 16*, 3 (1996), 64–77.

[LDB05] LAVOUÉ G., DUPONT F., BASKURT A.: A new CAD mesh segmentation method, based on curvature tensor analysis. *Computer Aided Design 37*, 10 (2005), 975–987.

[LKA06] LIEN J.-M., KEYSER J., AMATO N. M.: Simultaneous shape decomposition and skeletonization. In *Proc. of the ACM symposium on Solid and Physical Modeling* (2006), ACM, pp. 219–228.

[LLS*05] 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 (2005), 444–465.

[LZ07] LIU R., ZHANG H.: Mesh segmentation via spectral embedding and contour analysis. *Computer Graphics Forum (Special Issue of Eurographics 2007) 26* (2007), 385–394.

[Pap06] PAPALEO L.: *A Semantic Web System for Understanding and Reasoning on Digital Shapes*. Tech. Rep. DISI-TR-06-09, Department of Computer Science - University of Genova, May 2006.

[Sha08] SHAMIR A.: A survey on mesh segmentation techniques. *Computer Graphics Forum* (2008).

[She01] SHEFFER A.: Model simplification for meshing using face clustering. *Computer-Aided Design 33*, 13 (Nov. 2001), 925–934.

[SPP02] SUN Y., PAGE D., PAIK J.: Triangle mesh-based edge detection and its application to surface segmentation and adaptive surface smoothing. In *IEEE International Conference on Image Processing* (2002), pp. 825–828.

[WK05] WU J., KOBBELT L.: Structure recovery via hybrid variational surface approximation. *Computer Graphics Forum 24*, 3 (2005), 277–284.

[ZH04] ZHOU Y., HUANG Z.: Decomposing polygon meshes by means of critical points. In *MMM '04: Proceedings of the 10th International Multimedia Modelling Conference* (Washington, DC, USA, 2004), IEEE Computer Society, p. 187.