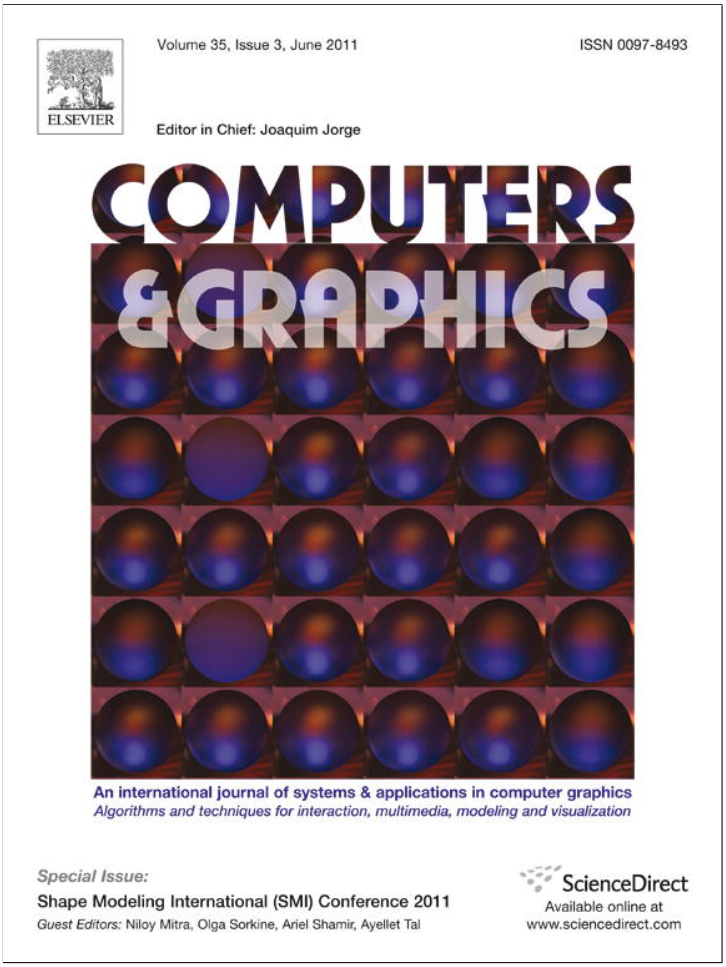

Computers & Graphics 35 (2011) 650–660

Contents lists available at ScienceDirect

Computers & Graphics

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

SMI 2011: Full Paper

A comparative evaluation of foreground/background sketch-based mesh segmentation algorithms

Min Meng [a,b], Lubin Fan [a,b], Ligang Liu [a,b,*]

[a] Department of Mathematics, Zhejiang University, China
[b] State Key Laboratory of CAD&CG, Zhejiang University, China

ARTICLE INFO

Article history:
Received 11 December 2010
Received in revised form
21 March 2011
Accepted 22 March 2011
Available online 8 April 2011

Keywords:
Interactive mesh segmentation
Sketch-based user interfaces
Evaluation

ABSTRACT

This paper presents an extensive comparative evaluation of five popular foreground/background sketch-based interactive mesh segmentation algorithms, addressing the quantitative assessment of the accuracy, efficiency, and stability of each algorithm. To facilitate the comparison, we have developed a complete framework with an intuitive and simple sketch-based interface to enable interactive mesh segmentation by marking strokes to specify the foreground and background with the mouse buttons, allowing us to quantify the algorithms in a unified manner. The evaluation has been performed via extensive user experiments in which each participant was assigned to segment models with the evaluated algorithms and the corresponding update of each segmentation was recorded as a new refinement when additional interactions were added. We then collected the segmentations from participants and evaluated them against the ground-truth corpus constructed from the Princeton segmentation database. To investigate how well the interactive segmentations match the ground-truth, five metrics were used to measure the boundary and region accuracy of segmentations. By studying the experimental results, we have analyzed the performance of the evaluated algorithms and provided valuable insights into their characteristics.

1. Introduction

The segmentation of 3D shapes into meaningful parts or semantic components is a key component in many graphics applications such as modeling [8], morphing [26,12], shape editing and deformation [14], texture mapping [16], shape retrieval [1,27]. The development of mesh segmentation algorithms has become a hot research topic, and numerous algorithms have been developed over the last decade [2,20].

Some applications, such as shape labeling and retrieval, require quick segmentation results and therefore require effective automatic segmentation algorithms. On the other hand, many other applications require accurate semantic segmented subparts. However, defining semantic subparts for shapes remains a challenging. It is time consuming to select an optimal automatic algorithm with well-tuned parameters for particular applications. Therefore, research on evaluating the quality of mesh segmentation algorithms has recently been recognized as important [6,3,4]. These methods are based on ground-truth evaluation benchmarks that consist of manual segmentations. The evaluation results have shown that no automatic segmentation algorithm is better than others.

Interactive segmentation algorithms can detect and extract semantic parts through a series of interactions where human operators provide useful high-level information to aid in this task. Due to their intuitive nature and simplicity, sketch-based user interfaces provide flexible interaction between computers and users and have been successfully used in segmenting 3D shapes into meaningful parts in an intuitive manner [13]. Specifically, the user simply draws rough, freehand sketches on the mesh surface to specify the foreground and background, and the algorithm updates the segmentation using the new information. By iteratively providing more interactions, the user can refine the segmentation. Fig. 1 illustrates an example of segmenting the head from a bunny model by specifying the sketches. In recent years, the study of sketch-based mesh segmentation has attracted much attention, and many techniques have been proposed [13,22,15,23,5,24,25].

Although the foreground/background sketch-based user interfaces provide an intuitive method for mesh segmentation, these works use different mechanisms in their algorithms and thus have different results and performance levels, even with the same input sketches. However, there is as yet no work on the quantitative evaluation of how well these approaches perform. The goal of this paper is to evaluate the performance of these foreground/background sketch-based interactive segmentation algorithms in

* Corresponding author at: Department of Mathematics, Zhejiang University, China. Tel./fax: +86 571 87953668.
E-mail address: liangliu@zju.edu.cn (L. Liu).

0097-8493/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2011.03.038

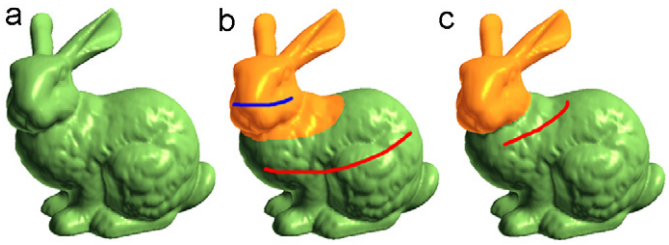

Fig. 1. Illustration of foreground/background sketch-based interactive mesh segmentation. Given a bunny model (a), the user marks a blue stroke on the area of the bunny's head (foreground) and a red stroke on the area of the bunny's body (background) and obtains a segmentation result in (b). The user then specifies an additional red stroke on the area of the bunny's body and the algorithm updates the segmentation using the new information, as shown in (c). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

a quantitative way via extensive user experiments. To this end, a complete framework for the performance evaluation of interactive segmentation algorithms, addressing the part-type components extraction from 3D models, is developed. We leverage the ideas of previous work in automatic comparison, and use related metrics in interactive segmentation to measure the similarities between the reference segmentation from the ground-truth (the Princeton segmentation benchmark [6]) and that obtained by interactive algorithms.

To our knowledge, our work is the first comprehensive evaluation of the state-of-the-art foreground/background sketch-based mesh segmentation algorithms. The contributions of this work can be summarized as follows:

- a software platform was implemented and designed for evaluating different foreground/background sketch-based interactive segmentation algorithms in a unified environment. We implemented five state-of-the-art interactive mesh segmentation algorithms in this platform for a fair and extensive comparison;
- we created a ground-truth segmentation dataset, which is well designed and selected from the Princeton segmentation benchmarks, for specifically evaluating the interactive segmentation algorithms. The dataset consists of 90 models with 18 different categories, and the segmented parts cover a wide range of types of shapes;
- extensive analysis and comparisons of the experimental results were demonstrated, and valuable insights into the performance and characteristics of all the algorithms have been provided.

2. Related work

In the processing of various media data, such as images, video, audio, and geometric shapes, the development of segmentation techniques has drawn extensive and consistent attention. Accordingly, a large number of studies [9,18,6,3] have been proposed for both computing and evaluating segmentations in these fields. Here we review only the most relevant studies, emphasizing the evaluation of mesh segmentation and sketch-based user interface techniques.

Research on the methods for evaluating the quality of mesh segmentation algorithms has recently been recognized as important. The state-of-the-art in 3D mesh segmentation evaluation was first proposed by Attene et al. [2]. Five segmentation algorithms were compared on 11 3D surface meshes in a qualitative way. The evaluations and comparisons were performed by showing side-by-side segmentations produced by different algorithms. Recently, two main works [6,3] for quantitatively evaluating *automatic* mesh segmentation were presented. Both works proposed systems and bechmarks which are based on a ground-truth corpus of human segmented 3D models. Recently, an extensive experimental comparison of existing metrics for the quality assessment problem of mesh segmentation was addressed by Benhabiles et al. [4]. All of the above studies were focused exclusively on evaluating automatic mesh segmentation. However, little attention has been dedicated to evaluating interactive mesh segmentation.

Table 1
Algorithms for sketch-based interactive mesh segmentation.

Method	Algorithms	Abbreviation
Region growing	Easy mesh cutting [13] A sketch-based interactive framework for real-time mesh segmentation [22]	EMC
Random walks	Fast mesh segmentation using random walks [15]	RWS
Bottom-up aggregation	Hierarchical aggregation for efficient shape extraction [23]	HAE
Graph-cut	Interactive part selection for mesh and point models using hierarchical graph-cut partitioning [5]	GCS
Harmonic field based	Sketching mesh segmentation based on feature preserving harmonic field [19] Mesh decomposition with cross-boundary brushes [25]	HFM

Sketch-based interfaces, which are simple and intuitive and help users easily express their intentions, have been successfully used in object selection in images [21], image segmentation using graph cuts [17], and mesh segmentation based on region growing [13]. All of these methods have developed sketch-based interfaces to provide flexible interactions between computers and users. Specifically, a series of foreground/background sketch-based methods based on multiple techniques (shown in Table 1), such as region growing [13,22], graph cuts [5], random walks [15,24], and hierarchical aggregation [23], have been proposed for interactive mesh segmentation. Recently, two other types of sketch-based user interfaces were proposed to segment meaningful parts of meshes: the *cross-boundary brushes* allow the user to draw strokes across a desired cutting boundary [25], and the *paint mesh cutting* allows the user to draw strokes on the foreground region only [7]. In this paper we focus on evaluating the performance of *foreground/background* sketch-based interactive mesh segmentation algorithms, as listed in Table 1.

3. Foreground/background sketch-based mesh segmentation algorithms

For the purpose of consistent and fair comparison, we only evaluate the interactive segmentation algorithms for extracting *meaningful parts* from 3D shapes. Table 1 lists all the foreground/background sketch-based mesh segmentation methods published in the literature so far. We classify these algorithms into five typical types. One representative algorithm is chosen from each type. Thus, we have five interactive algorithms (abbreviated EMC, RWS, HAE, GCS, and HFM, respectively) for evaluations, which provide a good coverage of the various approaches. We describe briefly each algorithm in this section. For further details, please refer to the original papers.

3.1. Easy mesh cutting (EMC)

Easy mesh cutting (EMC), proposed by Ji et al. [13], was the first foreground/background sketch-based user interface for mesh segmentation. It uses a simple and efficient region growing algorithm based on an improved feature-aware isophotic metric. Due to its fast speed, it has gained popularity, even without any statistical or probabilistic mathematical foundation.

EMC starts with the seed vertices from the foreground and background sketches simultaneously and grows their corresponding regions incrementally. Each mesh vertex on the input sketches is labeled as "F" (foreground part) or "B" (background part). All the other vertices are labeled as "U" (unknown). At each step, a vertex with label "U", adjacent to the foreground or background seeds, is selected, relabeled and added to the corresponding set. The vertex is chosen to be the one with the minimum distance to the foreground or background seeds based on the improved isophotic metric.

Wu et al. [22] proposed a similar method for sketching mesh segmentation based on region growing by using a different metric. We also implemented their method in our system. From our experiments, both methods [13,22] achieved similar performance in both segmentation results, and running speed. We chose EMC [13] as the representative method of this type as it was published earlier.

3.2. Random walks segmentation (RWS)

The random walks segmentation (RWS), presented in [15], provides mesh segmentation according to the probability value computed by minimizing a Dirichlet energy, which is similar in spirit to the corresponding method for image segmentation [11].

RWS proceeds as follows: the user is assumed to pick n triangles as seeds, where n is the desired number of parts (we set $n=2$ in our evaluation). A probability is associated with each of the three edges of each non-seed triangle f_k, denoted by $p_{k,1},p_{k,2},p_{k,3}(\sum_{i=1}^{3} p_{k,i}=1)$, respectively. These correspond to the probabilities that a random walk will move across a particular edge to the corresponding neighbor. As the random steps increase, the following equation holds for each non-seed triangle f_k:

$$P^l(f_k)=\sum_{i=1}^{3} p_{k,i}P^l(f_{k,i}), \tag{1}$$

where the $f_{k,i}$ are the triangles neighboring f_k, $P^l(f_k)$ is the probability of a random walk starting from f_k arriving at seed s_l (foreground or background seed). The above formulation results in a sparse linear system. Finally, each non-seeded triangle is determined to belong to the foreground region if a random walk starting at that triangle has a higher probability of reaching foreground seeds than background seeds, and vice versa.

3.3. Bottom-up aggregation (HAE)

The bottom-up aggregation algorithm (HAE), described by Xiao et al. [23], employs a hierarchical method for extracting high-level features through local adaptive aggregation. First, it introduces a multi-scale geometric similarity measure between adjacent vertices, which indicates the difference in the local shapes. Second, a graph G weighted by the similarity measure is constructed according to the model and can be partitioned into two subgraphs by minimizing a normalized energy. Instead of directly minimizing the energy, an iterative process of adaptive vertex aggregation is adopted to reduce the computational cost. Statistics of curvature are important for recognizing patches of consistent geometry and are used to define the multi-scale similarity measure between neighboring aggregates to improve feature extraction results. Finally, the boundaries of features detected at coarse levels can be refined iteratively to the finest level. In order to extract features of interest interactively from the models, user-specified constraints are effectively incorporated into the bottom-up aggregation process.

3.4. Graph-cut segmentation (GCS)

The graph-cut segmentation (GCS), proposed by Brown et al. [5], utilizes the minimum graph-cut to determine optimal part boundaries. The core of this algorithm is the graph-cut formulation, minimizing a cost function that captures the hard constrains provided by the user interactions and the soft constraints indicating the relationships between adjacent vertices of the 3D model. Given the set of vertices $\mathcal{P}$, the unordered set of vertex pairs $\mathcal{N}$ representing mesh edges, and the binary partitioning vector $A=(A_1,\ldots,A_p,\ldots,A_{|\mathcal{P}|})$, the cost function $E(A)$ can be defined as

$$E(A)=\lambda\sum_{p\in\mathcal{P}} R_p(A_p)+\sum_{\{p,q\}\in\mathcal{N}} B_{p,q}(A_p,A_q), \tag{2}$$

where $R_p(\cdot)$ denotes the penalty cost for incorrectly labeling according to the user constraints, and $B_{p,q}(\cdot,\cdot)$ denotes the sum of the costs of edges along the partition boundary, aiming to encourage spatial coherence between similar neighboring vertices on the surface. For the graph-cut segmentation to be effective, the weight of the edge in $\mathcal{N}$ is set to $B_{p,q}=|n_p\cdot n_q|$, where n_p and n_q are the respective surface normals of adjacent vertices p and q along the edge. An optimal minimization of Eq. (2) can be obtained by using a min-cut/max-flow algorithm. To achieve interactive speed for larger models, a self-building octree hierarchy is used to represent the vertices of the model, with leaf nodes generally set to a maximum of 10 vertices per leaf cell. The representative superpoint for each leaf is formed by the vertices in the corresponding leaf, with a surface normal defined as the average of the vertices' normals, and the parent cell normals are represented by weighted averages of the child cell normals. Accordingly, the adjacency of octree cells is used as the graph adjacency, and the edge weights are calculated using the superpoint normals.

3.5. Harmonic field based method (HFM)

Meng et al. [19] proposed a harmonic field based method (HFM) for interactive mesh segmentation. It starts with the user's strokes, specifying a foreground seed set U and a background seed set V. By solving the following Poisson equation:

$$\Delta\Phi=0, \tag{3}$$

with boundary constraints $\Phi(x)=1,x\in U$ and $\Phi(x)=0,x\in V$, a harmonic field is generated on the surface. The harmonic field is smooth and can be viewed as a smooth interpolation between the constraints. The method then modifies the harmonic field to reflect the geometric features of the mesh. The graph-cut technique is then applied to produce the segmentation results that are consistent with the user intention and the surface features.

The more recent work of a different UI—cross-boundary brushes [25], also uses a harmonic field on the mesh for segmentation. The harmonic field is generated by the Poisson equation with constraints defined by the endpoints of the brush across the cutting boundary. Then, a multi-scale isoline selection scheme is designed to select the best isoline automatically as the final cutting boundary. However, the isoline does not reflect the local geometric features. Thus, multiple cross-boundary strokes are generally needed to generate the expected cutting results.

We combine the advantages of the above two methods in the HFM method for evaluation. Specifically, we use foreground/background sketches and use the modified geometric aware harmonic field as in [19]. We compute the cutting boundary based on isolines as in [25] and use its method for fast updating the harmonic field when the users specify additional sketches.

3.6. Optimization of segmentation boundaries

Most of these algorithms employ additional optimization methods for optimizing the segmentation boundaries. Various methods have been proposed, such as geometric snake, graph cut, and subdivision, to refine the cutting contours as a postprocessing. A more recent work [24] proposed a geodesic curvature flow based method for optimizing the cutting boundaries. Actually, most boundary optimization methods can be applied to each of the algorithms. For the purpose of fair and consistent comparisons of these algorithms, we do not use any boundary optimization in any of the algorithms in our evaluation.

4. Evaluation system and task assignments

In this section, we describe our design of the evaluation platform.

4.1. Ground-truth corpus

Our ground-truth corpus is constructed based on the Princeton segmentation database [6]. Considering the characteristics of interactive segmentation, we select 18 categories from the database with five models in different poses from each category, except tables owing to their strong symmetry. These models are chosen so that they have a variety of poses and tessellations that which might affect the segmentation algorithms.

Each model of the corpus has an average of 11 segmentations. We chose one part for each model that could be unambiguously described to the user for extraction. The parts were chosen to cover various shapes such as flat parts, sphere-like parts, and cylinder-like parts. For the clarity of the task description, we associated each part task with several images of the part segmentation from the model it belongs via different perspectives. Fig. 2 illustrates the models selected from the corpus, one from each category, with one segmentation per model. See Supplementary material for all the models and their selected parts.

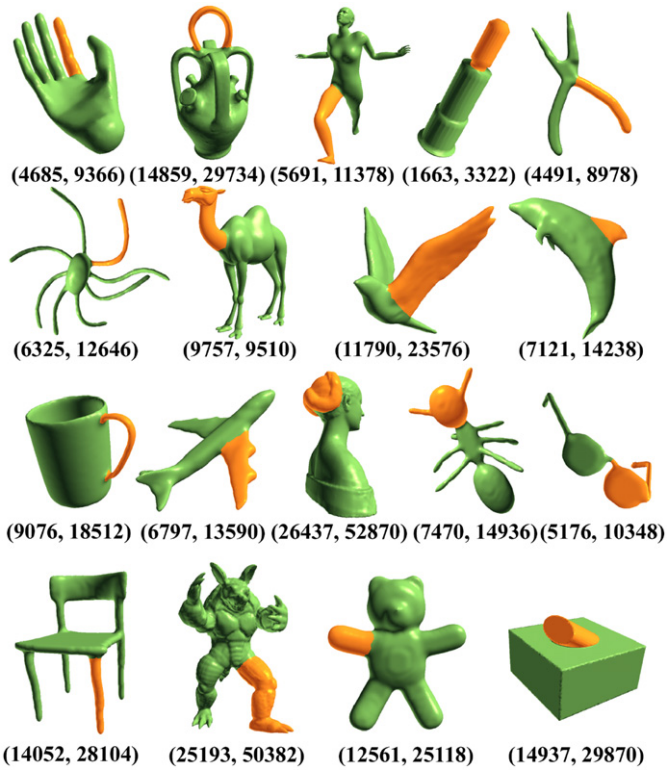

Fig. 2. Models in our ground-truth corpus, one segmentation per model of each category. Numbers in brackets denote vertices and triangles of models, respectively.

4.2. Evaluation system

To evaluate the various algorithms, we have implemented a system that uses sketch-based user interfaces with consistent capabilities for participants to segment the semantic parts from the models in a uniform way. All five algorithms mentioned in Section 3 have also been implemented and integrated in the system.

System overview: Fig. 3 shows a screen shot of our system. After loading a model into the system, several images that reflect the requirement of segmentation from different view points are displayed in the "*Evaluation panel*" located on the right of the system (shown in Fig. 3(b)). The user can drag the scroll bar at "*Change view*" to browse the images to see which part needs to be cut out from the model. Afterwards, the user can freely rotate and scale the model to an appropriate view direction. Then the user can draw strokes on the image plane (blue for foreground and red for background) by pressing the left and right mouse buttons, respectively, to specify the required parts on the model. After

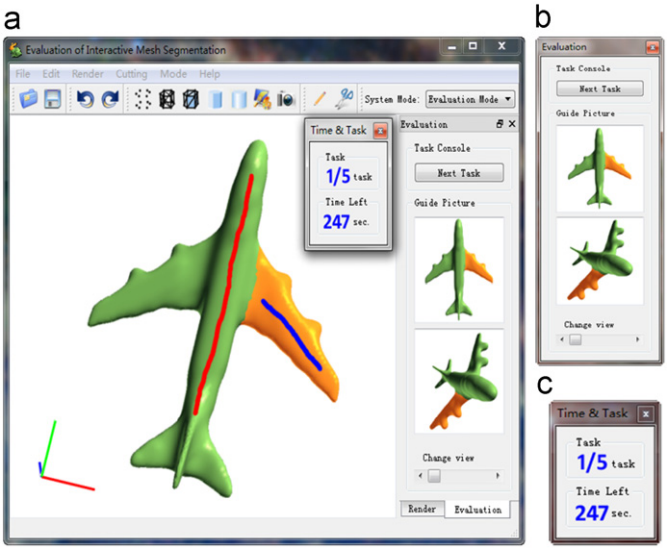

Fig. 3. A screen shot of our interactive segmentation system. (a) The interactive segmentation system in evaluation mode. (b) Task panel for evaluation. (c) Timer for restricting users to a maximum of 5 min per task. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

marking the strokes, the user can inspect the segmentation result on the screen and decide whether to specify new strokes to refine the segmentation. The corresponding segmentation result is updated when new strokes are marked. The system automatically records each update of the segmentation as a new refinement.

Training mode: Our system provides two operation modes for users: *training mode* and *evaluation mode*. In the *training mode*, users are asked to get familiar with the system by playing with

some sample models provided by the system. During the training process, users need to first know which part is required to be cut out from the loaded model by checking the images shown on the "*evaluation panel*" associated with the model. The users then specify the foreground and background strokes on the model. An algorithm is randomly chosen to do the segmentation. The users then decide whether more strokes are needed to refine the segmentation.

Evaluation mode: After being trained, users can switch the system to the *Evaluation mode* (shown in Fig. 3(a)). The users are required to load a model into the system and familiarize themselves with the segmentation task by viewing the images associated with the model. Once they are ready, they can click the "*Begin*" button and start to extract the required part from the model by drawing the strokes over the mesh, using different algorithms in an unknown order.

Timer: To preventing participants from spending too much time on refining their final results, it is necessary to impose a reasonable time limit on each task (shown in Fig. 3(c)). Each task is restricted to a maximum of 5 min, and users are allowed to proceed to the next task earlier when they have finished their current segmentation.

4.3. Task assignments

Over 100 individuals participated in our experiment, of which 30 participants had experience in geometry processing, 40 participants were familiar with human–computer interaction, and the rest needed to be trained for the task. In all, there were 79 males and 26 females. The ages ranged from 21 to 29 years, with an average of 24. Most of the participants were computer science graduates, and none of them has any particular prior experience in interactive mesh segmentation. A user guide and sufficient time were given to each participant, to help him/her get familiar with the system.

Corpus division: Our corpus contains 90 models, 18 categories with five models in each category. Each model is associated with several images describing the required part. For the purpose of acquiring segmentations from participants for each model in the corpus, we divided the ground-truth set randomly into 18 sets, ensuring that each set contains five models from different categories and each set has different types of part shapes.

Task for each participant: Each participant was assigned to segment the models of one set, each model with five segmentations using five interactive algorithms, respectively. To minimize the bias, five interactive algorithms were distributed to the participant in an unknown order. After loading a mesh model, the participant was asked to cut the required part out of the mesh using different algorithms successively, without knowing the order of the algorithms.

Questionnaire: Each participant was assigned to fill out a short questionnaire after completing the assigned tasks (extracting the required part from mesh model using all five algorithms). The questionnaire included a series of questions related to the evaluation of the algorithms. For instance, users were asked to rate the following questions on a scale of 1–5: how easily the users specified the segmentation, how fast they carried out their initial segmentations, how fast they refined their segmentations, and how accurate they considered their final segmentations, where 1 denotes the lowest score and 5 denotes the highest score. After answering all the questions, participants were asked to rate the interactive algorithms again on a scale of 1–5, indicating how effective they perceived these algorithms to be. Each question was answered in the order of the algorithms used, which was unknown to the users so as minimize bias. See the details in Supplementary materials.

4.4. Collected experiments

The above tasks were carried out over a period of two weeks. We had collected 2625 segmentations, of which 2310 were accepted and 315 were discarded as the segmentation conflicted with the requirements of the task. By distributing the model sets to participants equally, each model was segmented an average of five times by each algorithm. Thus, the stability of the interactive algorithms can be evaluated effectively based on these segmentations.

5. Accuracy measurement

The significant difference between automatic and interactive segmentation algorithms is that interactive segmentation algorithms require attention from the user. The interactions provided by the users usually have a profound effect on the resulting segmentations. To evaluate the interactive mesh segmentation algorithms effectively, three criteria need to be considered [18]:

- *Accuracy*: the degree to which the extracted part corresponds to the ground-truth.
- *Efficiency*: the amount of time or effort required to perform the desired segmentation.
- *Stability*: the extent to which the same result would be produced over different segmentation sessions when the user has the same intention.

It is noteworthy that these criteria are highly related for interactive segmentation algorithms. In particular, accuracy and efficiency are interdependent: the more time the users spend, the more accurate segmentations they obtain. We only discuss the issues of accuracy measurement in this section, the other two will be discussed in the next section.

Following prior work in computer vision, the existing measures used to evaluate mesh segmentation can be classified into three categories [4]: boundary matching, region difference, and non-parametric tests. Corresponding to the problem of our concern, we describe several metrics to evaluate how well interactive segmentation matches the ground-truth.

5.1. Boundary measure

The boundary measure is defined to compute the matching degree between the cut boundaries of two interactive segmentations in our study.

Chen et al. [6] described a measure called the *cut discrepancy* to measure the distances between cuts for evaluating automatic segmentations. Let S_1 and S_2 be two segmentations of surface mesh, and their respective sets of points on the cut boundaries C_1 and C_2. Let DCD be the directional function defined as $DCD(S_1,S_2)=mean\{d_G(p_1,C_2),\forall p_1\in C_1\}$, where $d_G(p_1,C_2)=\min\{d_G(p_1,p_2),\forall p_2\in C_1\}$ is the geodesic distance from a point $p_1\in C_1$ to cuts C_2. Based on considerations of symmetry and scaling effects of the metric, the cut discrepancy between S_1 and S_2 is defined as

$$CD(S_1,S_2)=\frac{DCD(S_1,S_2)+DCD(S_2,S_1)}{avgRadius}, \tag{4}$$

where *avgRadius* is the average Euclidean distance from a point on the surface to the centroid of the mesh.

We adopt this boundary measure for evaluating the boundary matching degree of the interactive segmentation in our study. Specific to our problem, the cut boundary of the interactive segmentation is usually a single curve embedded in the surface relative to the models with genus zero.

5.2. Region measure

Region measures are defined to compute the consistency degree between the part of interest produced by interactive segmentations in our study.

Chen et al. [6] described three measures related to region differencing: the Hamming distance, the Rand index and the consistency error. We adopt these region-based measures to evaluate interactive segmentations in our study. In our case, we assume that $S_1=\{S_1^1, S_1^2\}$ and $S_2=\{S_2^1,S_2^2\}$ are two interactive segmentations.

5.2.1. Hamming distance

The Hamming distance measures the region difference between the respective sets of segmentations S_1 and S_2. The directional Hamming distance is defined as: $DH(S_1 \Rightarrow S_2)=\sum_i \|S_2^i \backslash S_1^{i_t}\|$ where the operator $\backslash$ denotes the set difference, $\|x\|$ denotes the cardinality of set x, and $i_t=\text{argmax}_k \|S_2^i \cap S_1^k\|$, which finds the closest segment in S_1 to the segment S_2^i in S_2. Considering S_2 as the ground-truth, the Hamming distance can be defined as the average of the missing rate and the false alarm rate:

$$HD(S_1,S_2)=\frac{M_r(S_1,S_2)+F_r(S_1,S_2)}{2}, \tag{5}$$

where the missing rate is $M_r(S_1,S_2)=DH(S_1 \Rightarrow S_2)/\|S\|$ and the false alarm rate is $F_r(S_1,S_2)=DH(S_2 \Rightarrow S_1)/\|S\|$.

5.2.2. Rand index

The Rand index is computed as the ratio of the number of pairs of mesh elements (e.g., vertices or triangles) with compatible label relationships in S_1 and S_2. Denote by s_i^1 and s_i^2 the segment IDs of triangle i in S_1 and S_2 and by N the number of triangles in the mesh; $C_{ij}=1$ if $s_i^1=s_j^1$ and $P_{ij}=1$ if $s_i^2=s_j^2$. The region measure based on the Rand index is defined as

$$RI(S_1,S_2)=\binom{n}{2}^{-1}\sum_{i,j,i<j}[C_{ij}P_{ij}+(1-C_{ij})(1-P_{ij})], \tag{6}$$

where $C_{ij}P_{ij}=1$ and $(1-C_{ij})(1-P_{ij})=1$ indicates that triangles i and j have the same and different id in both S_1 and S_2, respectively.

5.2.3. Consistency error

The consistency error measure is based on computing a local refinement error, which represents the ratio of the number of elements (e.g., vertices or triangles) not shared between the segments S_1 and S_2. Denote by $R(S, f_i)$ the segment in segmentation S that contains triangle f_i; the local refinement error is defined as

$$E(S_1,S_2,f_i)=\frac{\|R(S_1,f_i)\backslash R(S_2,f_i)\|}{\|R(S_1,f_i)\|}. \tag{7}$$

Given the local refinement error, the two region measures global consistency error (GCE) and local consistency error (LCE) are defined as:

$$GCE(S_1,S_2)=\frac{1}{n}\min\{E_{12}^g,E_{21}^g\}, \tag{8}$$

$$LCE(S_1,S_2)=\frac{1}{n}\sum_i \min\{E_{12}^l,E_{21}^l\}, \tag{9}$$

where $E_{12}^g=\sum_i E(S_1,S_2,f_i), E_{21}^g=\sum_i E(S_2,S_1,f_i), E_{12}^l=E(S_1,S_2,f_i)$, and $E_{21}^l=E(S_2,S_1,f_i)$.

5.2.4. Binary Jaccard index

McGuinness and O'Connor [18] also described a region measure based on the binary Jaccard index to compare the object accuracies of interactive image segmentations. We extend it to measure the region accuracies between two parts in mesh segmentation. Specifically, the region measure based on the binary Jaccard index is defined as

$$JI(S_1,S_2)=\frac{\|S_1 \cap S_2\|}{\|S_1 \cup S_2\|}. \tag{10}$$

5.2.5. Normalized measures

In order to quantify the similarity (the higher the number, the better the segmentation) between segmentations rather than the dissimilarity, we define the measure $NCD(S_1,S_2)=1-CD(S_1,S_2)$ as the normalized cut discrepancy to quantify the similarity of the cut boundary, $NHD(S_1,S_2)=1-HD(S_1,S_2)$ as the normalized Hamming distance, and $NLCE(S_1, S_2)=1-LCE(S_1, S_2)$ and $NGCE(S_1, S_2)=1-GCE(S_1, S_2)$ as the normalized local consistency error and global consistency error, respectively.

6. Analysis

Based on the experimental results, in this section we present our analysis and comparisons among the different foreground/background sketch-based interactive segmentation algorithms.

6.1. Accuracy

To study the accuracy of each algorithm, we computed:

- the average final accuracy: the average boundary and region accuracy measured when the participant had finished the task or the task timing was up, across all the models for each algorithm,
- the variance of the final accuracy: the variance of the boundary and region accuracy measured when the participant had finished the task or the task timing was up, across all the models for each algorithm

using the accuracy measures introduced in Section 5.

Boundary and region accuracy: Fig. 4 shows the evaluation of the average boundary and region accuracy, computed across all the models for each algorithm. Higher bars represent better accuracy. According to Fig. 4(a), the best performing algorithm, in terms of the measured boundary accuracy, is EMC. Furthermore, according to Fig. 4(b), the best performing algorithms, in terms of all the region accuracies, are EMC and RWS, which perform equally well. In both cases, GCS has the worst performance, perhaps due to its relatively simple edge weights of the graph associated with the mesh model. From Section 3, we know that both RWS and HFM depend on the global solution of the Poisson equation. However, HFM performs relatively poorly, probably because its cutting boundary is composed of an isoline of the harmonic field so it cannot effectively respect the geometric features of the models.

Variance of accuracy: Fig. 5 shows the evaluation of variance of the boundary and region accuracy, computed across all the models for each algorithm. Lower bars represent better consistency. According to this figure, the variances of accuracy of all the proposed measures suggest that RWS has the most stable performance within the segmentations of all the models, followed by HAE, and the other three algorithms have comparable stability. RWS may perform more stably due to the global property of Poisson equation it solved, while EMC probably performs less stably due to the local property of region growing that it employs.

From the comparison, we can see that the proposed evaluation measures have significant consistency with one another.

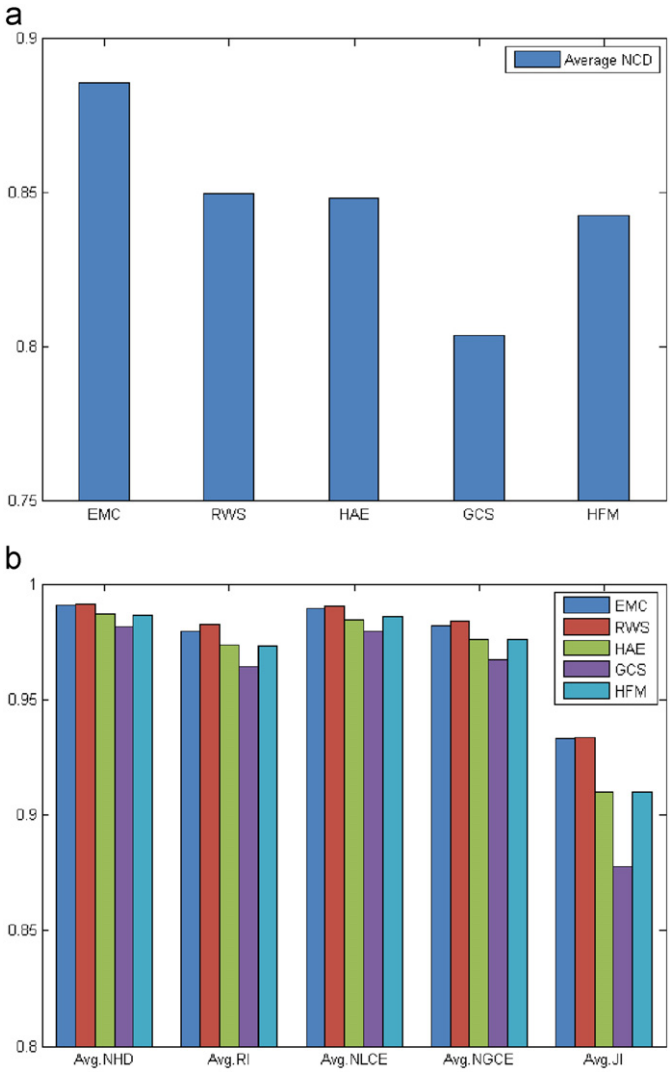

Fig. 4. Comparison of interactive segmentation algorithms by overall average boundary accuracy and region accuracy. (a) Average boundary accuracy; (b) average region accuracy.

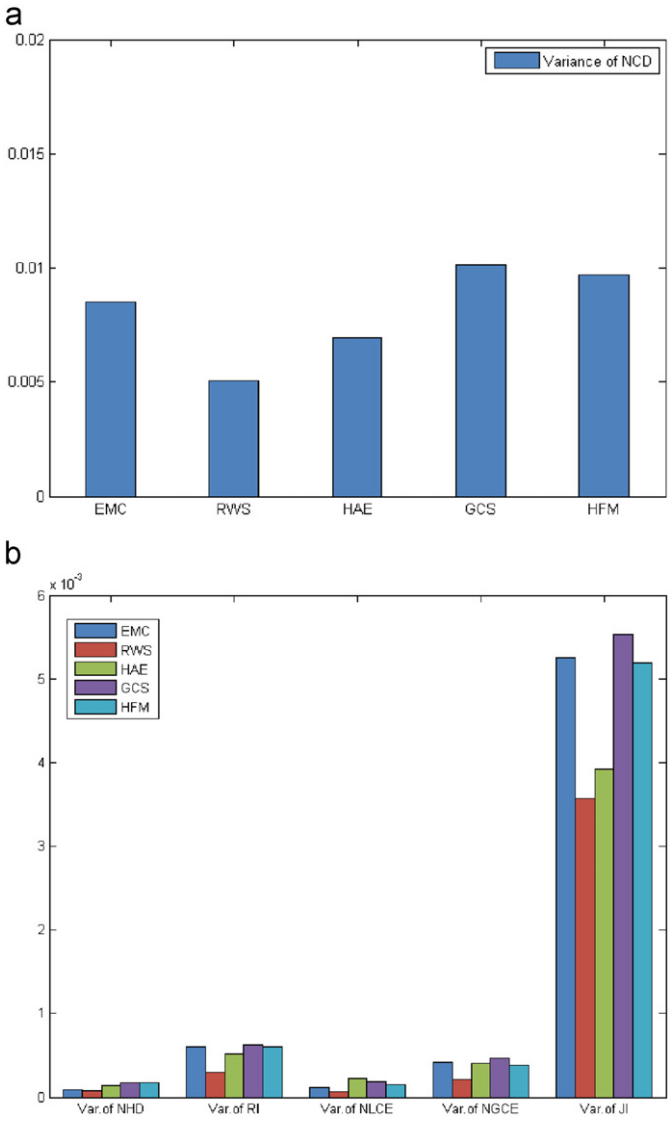

Fig. 5. Comparison of interactive segmentation algorithms by overall variance of boundary accuracy and region accuracy. (a) Variance of boundary accuracy; (b) variance of region accuracy.

6.2. *Efficiency*

We evaluate the efficiency of these segmentation algorithms as follows.

Interaction time: Fig. 6 shows the time required for segmentation and user interaction for each algorithm, averaged over all the models. Several observations can be drawn from this figure. Users spent the least amount of time on EMC and the most with HAE. Furthermore, HFM slightly underperforms EMC and obviously outperforms the other three algorithms for the averaged measured efficiency. Although both HFM and RWS solve the Poisson equations, which can be reduced to the sparse linear equations, HFM evidently outperforms RWS because HFM employs some numerical factorization updating techniques to solve the linear system more efficiently when new sketches are added.

Updating time for new sketches: Obviously, accuracy and efficiency are interdependent: the more time users spend, the more accurate segmentations they obtain. To completely illustrate the performance of each algorithm, the relationship between the number of updates and accuracy for all algorithms is shown in Fig. 7. The accuracy referred to the figure is the region accuracy,

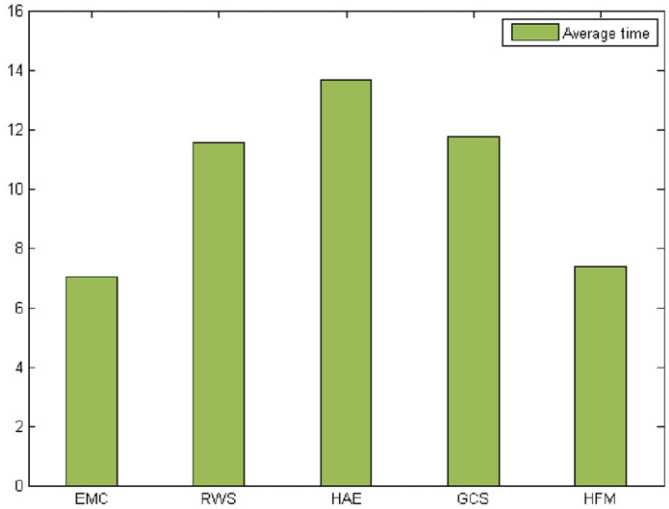

Fig. 6. Average time required for segmentation and user interaction with each algorithm.

measured using the binary Jaccard index. Observing the average measured accuracy over user interactions for each algorithm, we can see that EMC and RWS have comparable performance while EMC provides better final accuracy. HAE gives the best initial segmentation while producing small changes during the whole update process. Comparatively, despite its relatively poor initial segmentation, HFM also provides superior final accuracy. Furthermore, GCS gives the poorest initial segmentation and provides satisfactory final accuracy as additional sketches are added.

Number of interactions: We also compare the numbers of interactions required of the users. Fig. 8 shows the averaged number of interactions (drawing strokes) required for segmentation with each algorithm averaged over all the models. We can see that EMC requires the minimal amount of user interaction, while GCS requires the maximal amount of user interaction during the whole segmentation process. It is worth noting that EMC requires less user participation than RWS although they produce a comparable final accuracy of segmentation. This implies that RWS tends to need more user interactions to achieve a high accuracy via iterative improvement. Comparatively, GCS requires more user participation to achieve satisfactory final segmentation than the other four algorithms.

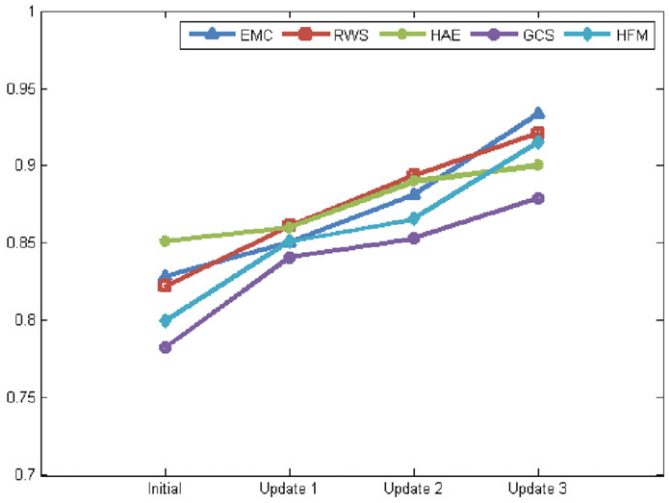

Fig. 7. The performance of each algorithm during the updating process.

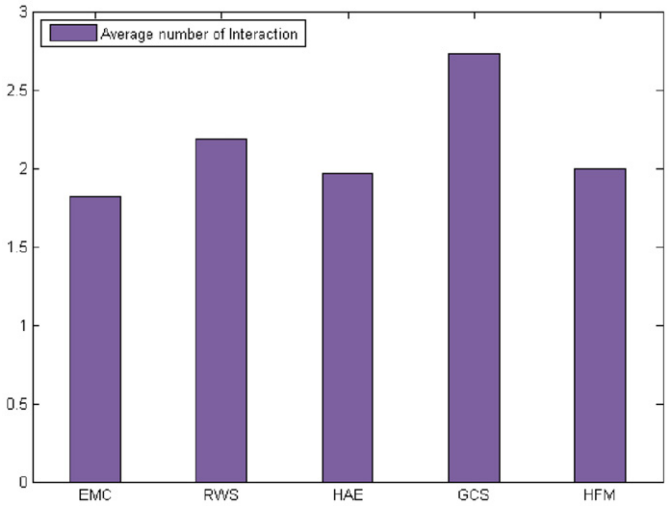

Fig. 8. Averaged number of interactions required for each algorithm.

6.3. Stability

To study the stability with respect to different user inputs, we computed:

- The averaged normalized coverage: the percentage of triangles with the same labels (foreground or background) found when using different user inputs per model, averaged across all models for each algorithm.

according to the stability test in [15]. Consequently, the stability is implicitly evaluated: if a good segmentation is repeated by multiple users, the average normalized coverage measure will be large. The stability is tested for the initial segmentation and the final segmentation, respectively. Fig. 9 shows the comparison of the stability test for each algorithm.

According to the results shown in Fig. 9 for the initial segmentation, we see that RWS and HAE are more stable, that is to say, they are less sensitive to different user inputs. Comparatively, EMC is more sensitive to different user inputs due to the greedy region growing method it employs. HFM is less stable because the isoline is more sensitive to different user inputs, GCS is also unstable because of the average accuracy of the segmentation it obtained. In comparison, the levels of stability of all five algorithms for the final segmentation shown in Fig. 9 are very close to each other. This result is not surprising. For example, although EMC and HFM are a bit more sensitive to the different strokes drawn by the user, they are both rapid at updating and involve more user interactions, which allows these two algorithms to obtain higher stability.

In particular, the stability test for the initial segmentation shows results closely correlated with Fig. 5. Both figures suggest almost the same relative performance of all five interactive algorithms.

6.4. Comparisons within shape classes

To understand how the segmentation algorithms work for specific types of objects, we also compare the suitability of the algorithms on different shape classes, i.e., man-made objects, organic objects, smooth objects, and objects with sharp features. Table 2 shows the performance of five algorithms for each object class. We rank the algorithms according to the Rand index metric averaged over models of each shape class in the table: 1 is the best and 5 is the worst. We find that no algorithm is the best for

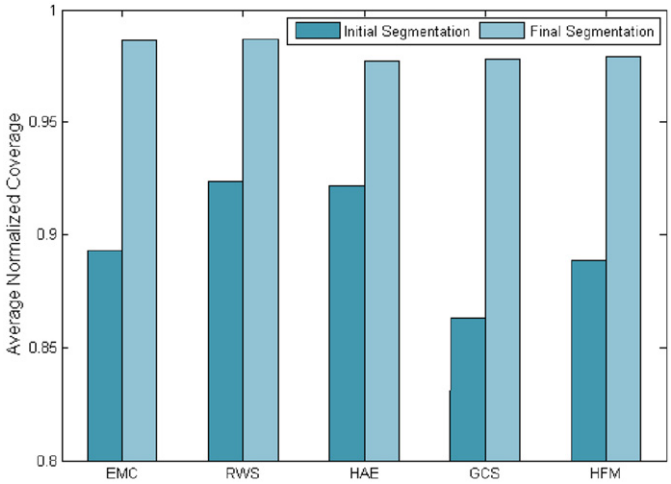

Fig. 9. Stability test for each algorithm.

Table 2
Comparison of the algorithms for different shape classes. Entries represent the rank of the algorithm according to the Rand index evaluation metric (1 is the best and 5 is the worst).

Shape class	EMC	RWS	HAE	GCS	HFM
Man-made	2	1	4	5	3
Organic	1	2	3	5	4
Smooth	1	2	3	5	4
Sharp	3	1	5	2	4
All	2	1	3	5	4

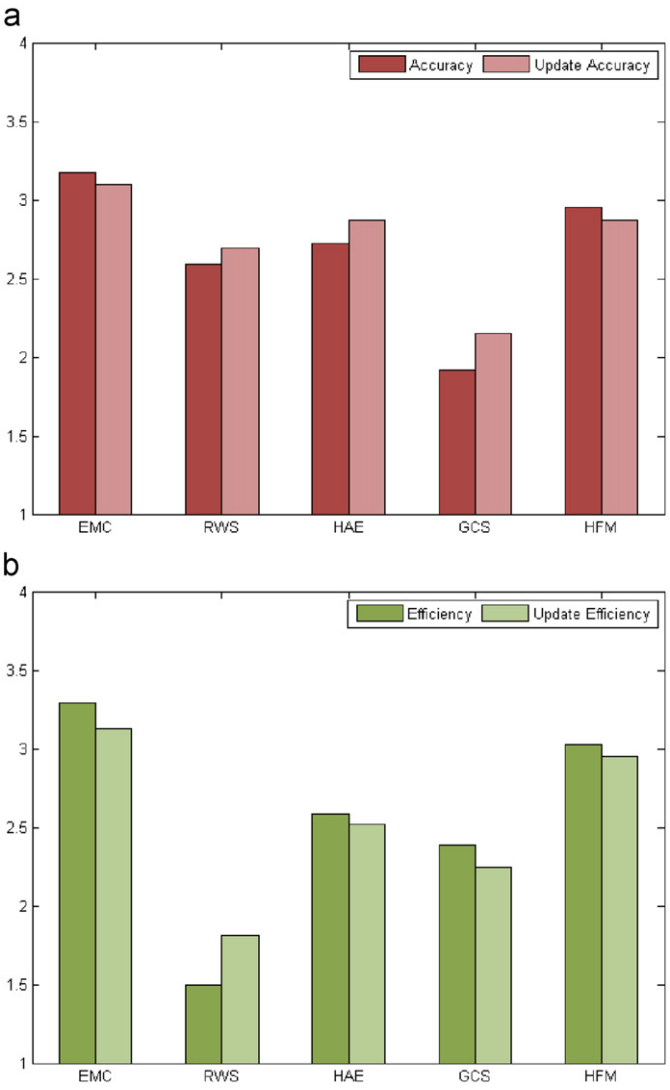

Fig. 10. The survey of perceived accuracy and efficiency. (a) Perceived accuracy; (b) perceived efficiency.

every class, and the algorithms were not particularly designed for specific shape classes.

6.5. User feedback

According to the answers indicated by the participants on a scale of 1–5 in the questionnaire, we have measured the perceived accuracy and efficiency, averaged across the answers of all participants. Both the initial segmentation and the update were taken into account in the questionnaire. The survey results are shown in Fig. 10. From the results, we can see that the perceived accuracy and efficiency are roughly consistent with the average measured accuracy and efficiency, except that EMC and HFM received higher ranks compared with their performance in terms of accuracy profiles (Fig. 4).

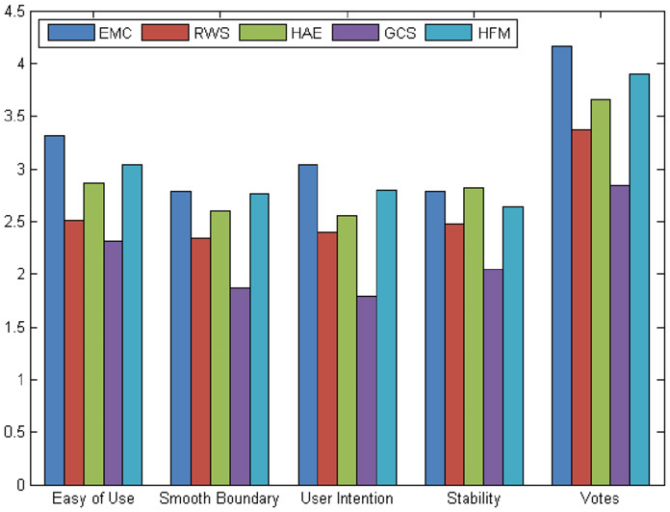

Fig. 11. Results of user feedback for each algorithm.

In addition to the accuracy and efficiency, the participants were also asked to rate each of the evaluated algorithms on a scale of 1–5 for a series of questions listed in the questionnaire, again with higher ranks indicating better performance. Fig. 11 shows the survey results of the questionnaire.

Obviously, most of the users preferred EMC and HFM over the other three algorithms. From Figs. 6 and 7, it can be seen that both EMC and HFM give good initial segmentation and are rapid at updating which can involve more additional interactions to provide iterative improvements in the for segmentation, despite their performance in terms of stability profiles (Fig. 9).

6.6. Comparison with automatic algorithms

In this part, we compare the interactive segmentation algorithms with the automatic ones to better understand their performance. We pick one of the automatic algorithms, the randomized cuts algorithm [10], for comparison as it combines many other automatic segmentation algorithms and performs relatively better than the other automatic algorithms, as shown in [6]. The segmentation results obtained by the randomized cuts algorithm were from the Princeton segmentation database [6]. We discarded those segmentation results that are not meaningful for fair comparison. Fig. 12 shows the comparison results on both the average boundary accuracy (NCD) and the average region accuracy (RI and NGCE). From the results we can see that the randomized cuts algorithm performs worse than EMC and RWS. It is surprising that it performs better than some of the other interactive algorithms.

6.7. Summary

The above experimental results are useful not only to study the properties of the current interactive segmentation algorithms but also to inspire new interactive segmentation techniques. Several observations on the characteristics of the evaluated algorithms, which are believed to be helpful and beneficial, include the following:

- In terms of the average measured accuracy, EMC and RWS surpass the performance of the other three algorithms.

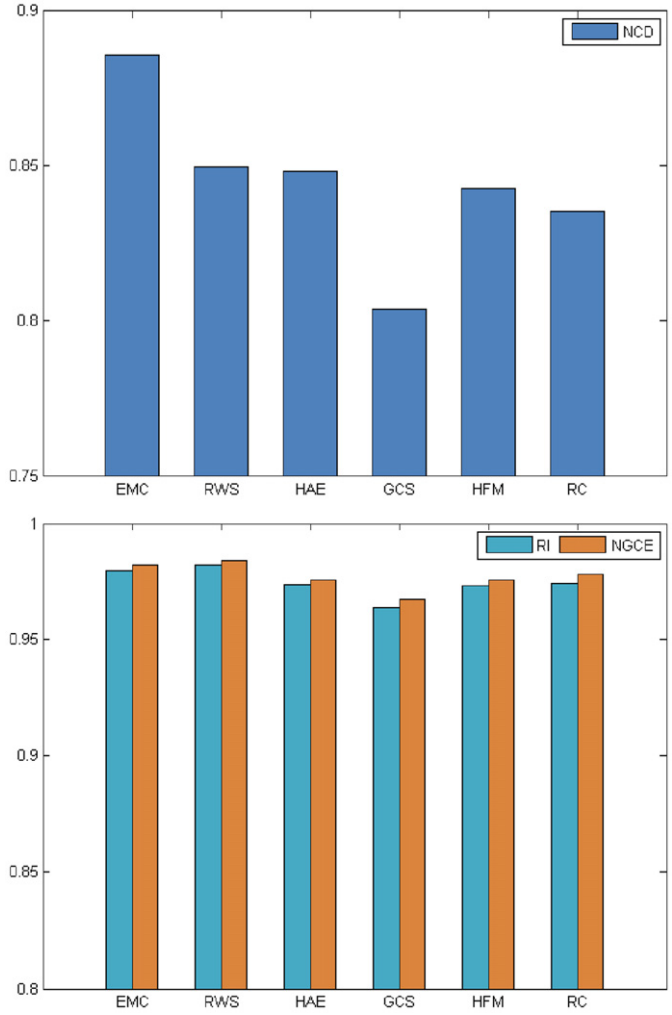

Fig. 12. Comparison of interactive segmentation algorithms with one of the automatic algorithms, the randomized cuts algorithm (RC) [10]. Upper, average boundary accuracy (NCD); Lower, average region accuracy (RI and NGCE).

- In terms of the average measured efficiency, EMC outperforms the other four algorithms, followed by HFM, while HAE performs worst.
- In terms of the average measured stability, RWS and HAE provide better performance than the other three algorithms for the initial segmentation, and EMC and HFM give comparable performance for the final segmentation despite their poorer performance for the initial segmentation.
- EMC requires the least amount of time and produces fairly accurate segmentation results, and RWS provides comparable segmentations.
- Both EMC and HFM allow highly efficient refinement during the updating process.
- HAE gives the best initial segmentation result and tends to produce smaller changes as additional interactions are added compared with the other four algorithms.

From the above results and observations, it is clear to see that no interactive algorithm is better than all the others. However, it is quite surprising that the first published interactive mesh segmentation algorithm, EMC, performs relatively better than the others across most of the performance factors such as accuracy and efficiency. Probably the reasons are (1) the region growing scheme in EMC is very efficient; (2) the feature-aware isophotic metric considered in the region growing fully respects the minima rule and saliency theory so that the obtained regions are accurate in terms of capturing the geometry features; (3) the simple scheme allows very quick feedback during the update process.

In the questionnaire, most of the users preferred EMC and HFM over the other three methods. This is probably because EMC and HFM allow highly efficient refinement (and thus quick feedback) during the update processing. We are much inspired by this observation, which suggests a critical insight on designing an interactive system. If a user wants to perform some interactions in using a program, he/she wants to have instant update feedback once he expresses his intention interactively. He might not care about how many interactions are required, but he does care about how fast feedback he will get in the operations. A quick update process should be required in designing an interactive system. On the other hand, too many interactions will make the job tedious. Thus, a good interactive program should require as few interactions as possible and give instant feedback during the interaction.

In summary, the evaluation results and insights on the performance of all algorithms will be useful for researchers and practitioners who need to pick an effective mesh segmentation algorithm for their applications.

7. Conclusion

In this paper, we have presented the first thorough evaluation of five foreground/background sketch-based interactive mesh segmentation algorithms, considering five criteria for the comparison. To evaluate the interactive algorithms effectively, we developed an intuitive sketch-based interface with consistent capabilities for participants to segment the part of interest from models in a uniform way. The evaluation was carried out via extensive user experiments, and several interesting insights were discovered. We believe that our work will help better understand the mechanisms of the various sketch-based interactive mesh segmentation algorithms and will inspire further studies on this topic.

Currently, the interactive segmentation algorithms are merely evaluated based on the corpus via a series of user experiments. In addition to the stroke variation of user inputs, there also exist more advanced issues to be concerned for examining partition compatibility of interactive algorithms, such as the noise and topology variation of models. In the future we plan to enlarge the corpus in terms of type of models and ground-truth, hoping that the expanded corpus could offer a wealthier comparison of interactive algorithms. It is also worthwhile to evaluate the different sketch-based user interfaces for mesh cutting, such as the foreground/background sketches, the cross-boundary brushes [25], and the new paint mesh cutting approach [7].

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61070071), the 973 National Key Basic Research Foundation of China (2009CB320801), and the Fundamental Research Funds for the Central Universities (2010QNA3039).

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.cag.2011.03.038.

References

[1] Antini G, Berretti S, Bimbo AD, Pala P. 3D mesh partitioning for retrieval by parts applications. In: Proceedings of IEEE international conference on multimedia and expo; 2006. p. 1210–3.

[2] Attene M, Katz S, Mortara M, Patane G, Spagnuolo M, Tal A. Mesh segmentation—a comparative study. In: Proceedings of IEEE international conference on shape modeling and applications; 2006. p. 1–7.

[3] Benhabiles H, Vandeborre J-P, Lavoué G, Daoudi M. A framework for the objective evaluation of segmentation algorithms using a ground-truth of human segmented 3D-models. In: Proceedings of IEEE international conference on shape modeling and applications, Beijing, China, short paper; June 26–28, 2009.

[4] Benhabiles H, Vandeborre J-P, Lavoué G, Daoudi M. A comparative study of existing metrics for 3D-mesh segmentation evaluation. The Visual Computer 2010;26(12):1451–66.

[5] Brown S, Morse B, Barrett W. Interactive part selection for mesh and point models using hierarchical graph-cut partitioning. In: Proceedings of graphics interface; 2009. p. 23–30.

[6] Chen X, Golovinskiy A, Funkhouser T. A benchmark for 3D mesh segmentation. ACM Transactions on Graphics 2009;28(3). [Proceedings of SIGGRAPH].

[7] Fan L, Liu L, Liu K. Paint mesh cutting. Computer Graphic Forum 2011;30(2). [Proceedings of Eurographics].

[8] Funkhouser T, Kazhdan M, Shilane P, Min P, Kiefer W, Tal A, et al. Modeling by example. ACM Transactions on Graphics 2004;23(3):652–63. [Proceedings of SIGGRAPH].

[9] Ge F, Wang S, Liu T. New benchmark for image segmentation evaluation. Electronic Imaging 2007;16(3).

[10] Golovinskiy A, Funkhouser T. Randomized cuts for 3d mesh analysis. ACM Transactions on Graphics 2008;27(5). [Proceedings of SIGGRAPH Asia, Article No. 145.].

[11] Grady L. Random walks for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 2006;28(11):1768–83.

[12] Gregory AD, State A, Lin MC, Manocha D, Livingston MA. Interactive surface decomposition for polyhedral morphing. The Visual Computer 1999;15(9): 453–70.

[13] Ji Z, Liu L, Chen Z, Wang G. Easy mesh cutting. Computer Graphic Forum 2006;25(3):283–91. [Proceedings of Eurographics].

[14] Katz S, Tal A. Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Transactions on Graphics 2003;22(3):954–61. [Proceedings of SIGGRAPH].

[15] Lai YK, Hu SM, Martin RR, Rosin PL. Fast mesh segmentation using random walks. In: Proceedings of ACM symposium on solid and physical modeling; 2008. p. 183–91.

[16] Lévy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. ACM Transactions on Graphics 2002;21(3):362–71. [Proceedings of SIGGRAPH].

[17] Li Y, Sun J, Tang CK, Shum HY. Lazy snapping. In: ACM Transactions on Graphics; 2004. p. 303–8. [Proceedings of SIGGRAPH].

[18] McGuinness K, O'Connor NE. A comparative evaluation of interactive segmentation algorithms. Pattern Recognition 2010;43(2):434–44.

[19] Meng M, Ji Z, Liu L. Sketching mesh segmentation based on feature preserving harmonic field. Journal of Computer-Aided Design and Computer Graphics 2008;20(9):1146–52. [in Chinese, also see its technical report in English].

[20] Shamir A. A survey on mesh segmentation techniques. Computer Graphics Forum 2008;27(6):1539–56.

[21] Tan KH, Ahuja N. Selecting objects with freehand sketches. In: Proceedings of ICCV; 2001. p. 337–44.

[22] Wu HY, Pan C, Pan J, Yang Q, Ma S. A sketch-based interactive framework for real-time mesh segmentation. In: Proceedings of computer graphics international; 2007.

[23] Xiao C, Fu H, Tai C-L. Hierarchical aggregation for efficient shape extraction. The Visual Computer 2009;25(3):267–78.

[24] Zhang J, Wu C, Cai J, Zheng J, cheng Tai X. Mesh snapping: robust interactive mesh cutting using fast geodesic curvature flow. Computer Graphics Forum 2010;29(2):517–26. [Proceedings of Eurographics].

[25] Zheng Y, Tai CL. Mesh decomposition with cross-boundary brushes. Computer Graphics Forum 2010;29. [Proceedings of Eurographics].

[26] Zöckler M, Stalling D, Hege H-C. Fast and intuitive generation of geometric shape transitions. The Visual Computer 2000;16(5):241–53.

[27] Zuckerberger E, Tal A, Shlafman S. Polyhedral surface decomposition with applications. Computers & Graphics 2002;26(5):733–43.