The Visual Computer manuscript No.
(will be inserted by the editor)

AB3D: action-based 3D descriptor for shape analysis

Zhige Xie · Yueshan Xiong · Kai Xu

Received: date / Accepted: date

Abstract High-level geometry processing has been a hot topic in graphics community. The functionality analysis of 3D models is an essential issue in this area. Existing 3D models often exhibit both large intra-class and inter-class variations in shape geometry and topology, making the consistent analysis of functionality challenging. Traditional 3D shape analysis methods which rely on geometric shape descriptors can not obtain satisfying results on these 3D models. We develop a new 3D shape descriptor based on the interactions between 3D models and virtual human actions, which is called Action-Based 3D Descriptor (AB3D). Due to the implied semantic meanings of virtual human actions, we obtain encouraging results on consistent segmentation based on AB3D. Finally, we present a method for recognition and reconstruction of scanned 3D indoor scenes using our AB3D. Experiments show that AB3D is a promising shape descriptor towards functionality analysis of 3D shapes.

This work was partially supported by the Research Fund for the Doctoral Program of Higher Education of China (No.20104307110003) , the National Natural Science Foundation of China (No. 61379103, 61202333, 61303185) and China Postdoctoral Science Foundation (No. 2012M520392).

Zhige Xie
State Key Laboratory of High Performance Computing, School of Computer Science, National University of Defense Technology,China
E-mail: zhigexie@gmail.com, Corresponding author.

Yueshan Xiong
State Key Laboratory of High Performance Computing, School of Computer Science, National University of Defense Technology,China
E-mail: ysxiong@nudt.edu.cn

Kai Xu
State Key Laboratory of High Performance Computing, School of Computer Science, National University of Defense Technology,China
E-mail: kevin.kai.xu@gmail.com, Corresponding author.

Keywords shape analysis · affordance · consistent segmentation · point cloud construction

1 Introduction

The rapid development of digital geometry processing technique has been greatly boosting the growth of digital geometry. Geometry processing is currently moving towards high-level shape analysis and understanding, aiming at discovering the underlying semantic information of a 3D shape. This gives the birth of the recent trend of high-level geometry processing [1].

Shape semantics reflects human knowledge about shape's geometry, structure, functionality, and their relationships. The semantic analysis of 3D models involves two aspects. One is classification, i.e., to which category a model belongs to. The other is about recognition/detection of the exact functionalities of various parts. Effective integration of the input shape knowledge is important to shape analysis. Many works have been devoted on structural analysis of 3D models, including model segmentation[2], consistent segmentation[3] , symmetry analysis[4] , etc. However, few of them focus on analyzing functional information of parts of 3D models, e.g. "back support" and "seat" of chair models. Macro Attene et al. [5] developed a system called "ShapeAnnotator", which needed users to create 3D shapes' ontology manually. Based on their ontology, they analyze a 3D model and obtain its geometric and semantic information. But their method requires extensive manual effort for ontology creation, and could not deal with 3D shape whose category is not contained in their ontology database.

In this paper, we try to utilize the implied semantic meanings of human actions to infer semantic information from 3D models automatically. To the best of our knowledge, the initiative work on using interactions between virtual human

and 3D shape to analyze functionality was presented by Tolga Abaci et al. [6]. However, their method limits to human hand actions. In contrast, we integrate the interactions of whole body, arm and hand actions into the same framework to analyze wider spectrum of functionalities. To achieve such integration, we formulate the semantic inference process as an optimization problem and then combine these interacting results together as a new action based shape descriptor. Ezer Bar-Aviv et al. [7] and Helmut Grabner et al. [8] both developed methods using virtual human whole body actions to interact with 3D shapes to get the semantic information, but their method can only deal with one action at a time. Different from their works, we use multiple sets of human actions to analyze input 3D models simultaneously, and utilize pattern classification method to classify these models.

Overview and contributions We propose a general framework of utilizing human actions to analyze functionalities of 3D models. We first develop a new 3D shape descriptor based on interactions between 3D models and virtual human actions, which is called Action-Based 3D Descriptor (AB3D). Based on AB3D, we develop a classification algorithm which can deal with 3D models exhibiting both large intra-class and inter-class variations as well. Furthermore, we obtain consistent segmentation results by introducing AB3D to the geometric descriptor based segmentation workflow. Finally, we present an algorithm for recognition and reconstruction of scanned 3D indoor scenes based on AB3D.

Our paper makes the following contributions:

(1) We develop a 3D shape descriptor based on human actions for 3D scene analysis. Based on interacting results computed from AB3D, we can perform labeled segmentation.

(2) We summarize three principles of interactions between virtual human actions and 3D models to guide best interacting transformation searching between actions and models. These principles are simple and can be easily implemented as an optimization problem.

(3) We adopt multiple levels of actions to handle ambiguities properly when dealing with 3D models which have multiple interacting ways.

(4) We use virtual human actions to reconstruct 3D models in scanned 3D indoor scenes.

2 Related Work

High-level geometry analysis. Traditional 3D shape analysis methods usually use geometric and topological [9] 3D shape descriptors, these methods analyze 3D models based on information from every triangle's normal direction, dihedral angle, average geodesic distance [10] and etc. Though these methods are useful in many traditional tasks, but they lack high-level semantic information. High-level geometry analysis [1] [11] [12] use human knowledge to assist 3D shape segmentation, matching, retrieval and etc. Macro Attene et al. [5] developed a system called "ShapeAnnotator", which needed users to create 3D shape ontology manually. Based on their ontology, they analyzed a 3D shape, and got its geometric and semantic information. Kalogerakis et al. [13] present a supervised learning based technique that employs information from shapes in a training set to segment a given shape, demonstrating significant improvement over single-shape segmentation algorithms. Wang et al. [14] proposed the concept of symmetry hierarchy on man-made 3D models. Xu et al. [15] introduced set evolution as a means to inspire creative 3D modeling. They all used different methods to combine human knowledge into their shape analysis procedures. But using human action itself as a knowledge resource has been rarely attempted in this area.

Affordance The concept of affordance has become a focus of attention within the machine vision and robotics community lately [16]. Affordance means one can detect the functionality of object through interaction between user and object itself [8]. Recently, methods have been proposed to detect objects based on human interactions. The human activity is annotated by extracting human motion from video data and used to indirectly identify objects [17]. Helmut Grabner et al. [8] presented an affordance detector which used whole human action to analyze the functionality of the scene. They utilized a probabilistic method to detect where is most suitable place for sitting with higher probability in the whole scene. But they failed to provide a general framework of using human actions to deeply analyze 3D models, e.g. segment 3D models into parts and label them according to their interactions with virtual human actions.

Consistent segmentation There have been various works on consistently segmenting a set of 3D shapes from the same class into semantic parts. Golovinskiy and Funkhouser [3] considered the consistent segmentation as a graph clustering problem. To deal with non-homogeneous part scales, Xu et al. [18] classified the shapes based on their styles and then establish part correspondences in each style group. Hu et al. [19] used subspace clustering to automatically cosegment a set of shapes from a common family into consistent parts. Oana Sidi et al. [20] introduced an unsupervised method to produce consistent segmentation for a set of 3D meshes through descriptor-space spectral analysis. Their work exploited the potential of spectral method in consistent segmentation. Martin Reuteret al. [21] proposed an approach which use topological features of Laplace-Beltrami eigenfunctions to accomplish consistent shape segmentation

and registration. Under the assumption that a set of 3D models with the same functionality have the same interacting ways with human, we can use virtual human actions to consistently segment a set of 3D shapes.

Scanned Scene Understanding 3D indoor point cloud reconstruction is particularly challenging due to object interferences, occlusions and overlapping which yield incomplete yet very complex scene arrangements. Silberman and Fergus [22] presented an algorithm for indoor scene segmentation which uses graph-cut to propagate classification labels of features to the full scene. Koppula et al. [23] addressed semantic labeling of 3D indoor scenes by fusing color, depth and contextual information together. Nan et al. [24] presented a search-classify approach which interleaves segmentation and classification in an iterative manner. Given the speciality of interactions with human in indoor scene, it is plausible to use a set of virtual human actions to recognize and reconstruct 3D models.

3 The computation of AB3D

In this paper, all 3D models/objects are represented as triangle meshes, possibly with multiple components. "Virtual human action" means the 3D mesh of a specific human action, such as a hand with some gesture or a full body with a standard pose.

The procedure to compute AB3D of a 3D object is divided into three stages. First, we collect three levels of standard virtual human actions from database. Second, we compute best interacting transformations between virtual human actions and the object. Finally, we combine these best interacting transformations to form a vector, which is the AB3D of the object. The whole procedure can be formalized as follows.

3.1 Three levels of virtual human actions

Part of our standard human actions were collected from Pose Pro [25] and others were downloaded from Google 3Dwarehouse [26]. These actions of virtual human are classified into three sets: whole body actions set Γ which contains chair_sitting, sofa_sitting, bed_sleeping, table_eating and bike_driving; arm actions set Σ which contains carrying, lifting, pushing, pulling and slicing; hand actions set Δ which contains hand_goblet, hand_screw, hand_cup, hand_plat and hand_fist. The three levels of actions are shown as Figure 1. Each set corresponds to one level of virtual human actions. We combine these three levels of virtual human actions as the total set of virtual human actions, i.e., $\Omega = \Gamma \bigcup \Sigma \bigcup \Delta$. Let A_m be one action of Ω.

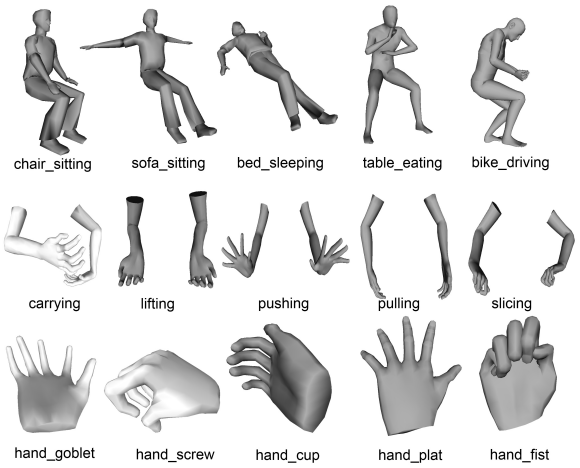

Fig. 1 The three levels of virtual human actions.

3.2 Interaction between single action and 3D object

Let Π be the set of 3D objects to be analyzed, and O_n be one object of Π. We denote the combined Axis-Aligned Bounding Box (AABB) as $B_{m,n}$ which contains both O_n's AABB and A_m's AABB . We hierarchically subdivide $B_{m,n}$ into small cubes. The total number of these cubes is l. Let C_j be one of these cubes, and its center vertex be V_j.

Let $T_{m,n}$ be the search space of transformations between A_m and O_n, and $t_{m,n}$ be one transformation of $T_{m,n}$. We also voxelize O_n and compute its 3D distance field $D_{m,n}$ [27]. The closest distance of C_j to O_n can be efficiently obtained by

$$d_{C_j,O_n} = D_{m,n}(t_{m,n}V_j) \tag{1}$$

After applying $t_{m,n}$ to A_m, the distance between A_m and O_n can be calculated as:

$$d_{m,n} = \sum_{j=1}^{l} d_{C_j,O_n} \tag{2}$$

where $d_{m,n}$ is the summation of distances between O_n and cubes C_j inside A_m.

3.3 Contact area and number of intersecting points

The contact area between A_m and O_n is also very important to our method. To compute the contact area, we first determine whether each triangle of O_n and AABB of A_m is intersected or not. $S_{m,n}$ is the summation of all the areas of triangles which are intersected. We also use RAPID [28] to compute the number of intersecting points $N_{m,n}$ between A_m and O_n.

3.4 Best interacting transformation between single action and 3D object

Based on the observations from interaction between human and objects in everyday life, we propose three heuristic guiding principles of best interacting transformation. (i) The distance between virtual human and object should be as small as possible. (ii) The contact area should be as large as possible. (iii) Intersections between virtual human action and object should be avoided. Based on these principles, we formulate the process of finding best interacting transformation between A_m and O_n as an optimization problem,

$$t^*_{m,n} = \underset{T_{m,n}}{argmin}(d_{m,n} - \beta * S_{m,n} + \alpha * N_{m,n}) \quad (3)$$

where α and β are weights to balance between intersecting points and contact area. The three terms in the right side of equation (3) correspond to the three observations respectively. We solve equation (3) using traversal method in transformation search space $T_{m,n}$. The details will be discussed in section 4.2.

3.5 AB3D of 3D object

The final step is to determine which level of actions should be chosen for O_n. As our observation, the level of virtual human actions highly depends on the volume of its AABB. Let its volume be V_n. We first compute average volume of human actions of each level, then compare V_n to them. If V_n is significantly bigger than some levels' average volumes, we will omit these levels'actions. By computing the best transformations between O_n and a number of m virtual human actions, we can obtain AB3D of O_n which is defined as a vector:

$$AB3D(Object_n) = (t^*_{1,n}, t^*_{n,n} \ldots t^*_{m,n}) \quad (4)$$

4 3D shape classification based on AB3D

In this section, we will develop our 3D shape classification algorithm based on AB3D. Section 4.1 describes details of preprocessing of 3D objects. Section 4.2 proposes algorithm 1, which computes the best interacting transformation between single virtual human action and 3D object. Section 4.3 presents algorithm 2, which computes AB3D of 3D object. Section 4.4 shows how to use Support Vector Machine (SVM) to classify input 3D models. The overview of our AB3D classification algorithm is depicted as Figure 2.

4.1 Preprocessing of 3D models

AB3D requires 3D models have approximately correct sizes as in our daily life. This is critical to our algorithm because all our features are deducted from interactions between virtual human action and 3D models. If one model of our database has the wrong size, we will get incorrect AB3D. Fortunately, most of the models used in this paper are collected from Google 3Dwarehouse [26] , most of which have the correct relative sizes. The meshes with poor quality or improper sizes will be discarded. Finally, we use principal component analysis (PCA) to adjust the main three eigenvectors aligned to the axis of the coordinate. The method of Fu et al. [29] is used to make sure that models are correctly oriented in coordinate system.

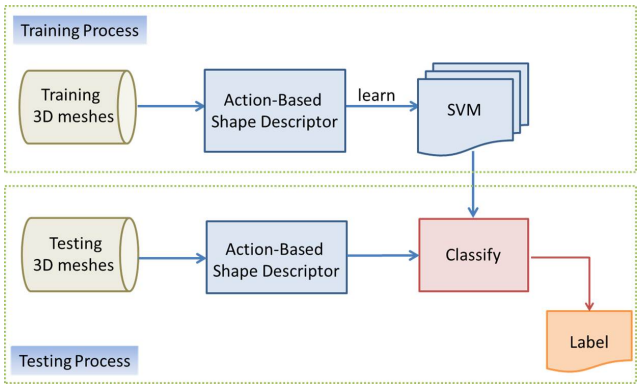

Fig. 2 Block-diagram overview of our method.

4.2 Computation of best interacting transformation between single action and 3D object

For a single action A_m and one 3D object O_n , we use Algorithm 1 to compute the best transformation $t^*_{m,n}$ between them. The resolution of voxelization is determined by which actions set A_m belongs to. If A_m is one of the whole body set, the resolution is 3.33 cm^3; otherwise, the resolution is 1 cm^3. The rigid transformation space is also regularly sampled on the same resolution. The rotation transformation is sampled at severy 30 degrees step only in x-y plane, because we have adjusted the 3D models using PCA and Fu et al. 's method [29]. All our AB3D computation is performed in the transformation space which combine the rigid and rotation transformation together.

Here we describe the details to solve equation (3). At first, we utilize RAPID [28] to do collision detection between A_m and O_n. When these two models are in the boundaries of collision, we store these boundary transformation. Finally, we do traversal search for every boundary transformation. We substitute these boundary transformations into equation (3). $t^*_{m,n}$ is the transformation which minimize right side of equation (3). When A_m belongs to whole body action set, weights α and β are set to be 0.8 and 0.2; otherwise, α and β are set to be 0.5 and 0.5.

Algorithm 1 Compute best transformation between single virtual human action and 3D object

Input:
Single virtual human action, A_m; 3D object, O_n.
Output:
Best transformation between A_m and O_n : $t^*_{m,n}$
Compute the combined AABB $B_{m,n}$, spatially subdivide $B_{m,n}$
Voxelize O_n and compute 3D distance field $D_{m,n}$ in $B_{m,n}$
for each $t_{m,n}$ in $T_{m,n}$ **do**
  for each C_j in $B_{m,n}$ **do**
    Use RAPID to compute $N_{m,n}$ and determine the boundaries of collision detection
  end for
end for
for each $t_{m,n}$ in the boundary transformations **do**
  Use equation (2) to compute $d_{m,n}$
  Use equation (3) to compute $t^*_{m,n}$
end for
return $t^*_{m,n}$

4.3 Computation of AB3D

The computation of AB3D of one 3D object O_n is detailed in Algorithm 2.There are many complicated situations when interacting with 3D objects. As a result, using multiple actions is a natural choice for us. Take a middle size vase with a handle for example. If we just use hand actions to interact with it, we may get wrong classification because the way of interaction between this vase and some cups with handles are similar. If we use multi-level actions which contains both arm actions and hand actions to interact with it, we can combine these best interacting transformations together as input vector for classification method. As a result of richer information, we can get more accurate functional analysis results.

Algorithm 2 AB3D of 3D object

Input:
Three levels standard virtual human actions Ω; 3D object, O_n.
Output:
AB3D(O_n)
for each virtual human action A_m in Ω **do**
  if V_n is bigger than 3 times of mean volume of actions level A_m belongs to **then**
    Set $t^*_{m,n}$ = NULL
  else
    Use Algorithm 1 to compute $t^*_{m,n}$
  end if
end for
Use equation (4) to compute $AB3D(O_n)$
return $AB3D(O_n)$

4.4 Classification algorithm using SVM based on AB3D

In this subsection, we will focus on how to use AB3D as feature vectors to classify objects. We use Support Vector Machine (SVM) [30] which is one of the most popular algorithm for classification.

We use $AB3D(Object_n)$ as input feature for SVM. 50% of the dataset is used as training model set, and others are used as test model set. As shown in Figure 2, in training process, we use one-vs-all strategy [31] to train a single SVM classifier for every category, so we have n SVM classifiers for n categories of models. In the testing process, we compute every object's AB3D as input, and compute the probabilities of labeling tags from their corresponding SVM classifiers. Finally, we vote based on these probabilities to determine the final labeling results of test 3D objects.

5 Results

5.1 Dataset

We construct an 3D model dataset because existing model database failed to collect 3D models which have inter-class variations in appearance and can be easily interacted with as well. These 3D models are popular nowadays. We searched key words "modern chair", "modern sofa", "modern bed", "modern tabel" on Google 3DWarehouse [26], and then we searched "chair", "sofa", "bed", "table" . These "modern" models were found in the first several results pages.

We downloaded hundreds of "modern" 3D models from these four categories, discarded all the tags attached in these models, and labeled them as their corresponding categories, i.e., "bed", "chair", "sofa", and "table". The number of triangles of these models varies from hundreds to millions. Figure 3 shows part of our 3D models dataset.

Fig. 3 Part of our 3D model dataset. Note that this dataset has both great large intra-class and inter-class variations in appearance.

Part of our standard human actions were collected from Pose Pro [25] and others were download from Google 3D-warehouse [26]. We used Quadirc Edge Collapse Decimation filter in MeshLab [32] to simplify these models to around 2000 triangles. The three levels of virtual human actions set are shown as Figure 1.

5.2 Effectiveness of interacting transformation search

To validate the effectiveness of AB3D, we choose a bottle model and a goblet model from princeton shape benchmark (PSB) [33], and use them to interact with the human hand action set. Note that we adjust the sizes of these two PSB models according to their sizes in our daily life. Figure 4 visualizes the result of our algorithm. From these results, we can conclude that AB3D generated by our algorithm are close to the way of interactions between human and objects in everyday life. Specifically, the result between fingers and goblets shows that our algorithm can be used to get precise interacting transformations.

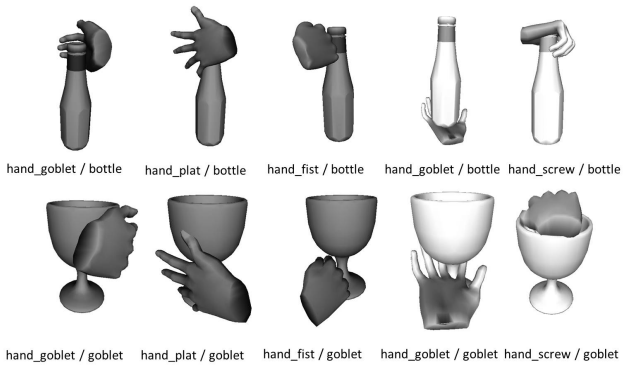

Fig. 4 The best transformations between four hand actions and two 3D models from PSB computed by Algorithm 1. Every entry below the transformation result contains hand action name (before the slash) and model name (after the slash).

5.3 Performance Evaluation

To validate the retrieval performance of our classification algorithm, we choose 335 models of four categories described in subsection 5.1. We chose four virtual human actions from whole body action set, which are chair_sitting, sofa_sitting, bed_sleeping and table_eating. We split the whole 3D models dataset into two parts: 50% of the dataset is used as training data and the rest is used as test data.

We compare our method against two classical shape descriptors: rotation invariant spherical harmonic (SPH) [34] and LightField Descriptor (LFD) [35] with AB3D on our dataset. The reason why we choose LFD is that LFD-like method is still one of the most powerful shape descriptors in 3D model classification and retrieval contest [36] [37]. In the experiment, we first use SPH and LFD to compute the 3D objects' descriptors, and then we use k-nearest neighbor(kNN) [38] to classify these models. Finally we also apply our AB3D based SVM classification algorithm to classify these 3D models. We use libSVM [39] toolbox written in MATLAB, and adopt the one-vs-all strategy [31] to train four SVM classifiers. These four SVM classifiers are used to classify these test models.

Figure 5 shows plots of precision-recall curves for of SPH, LFD and our method. Tables 1 shows the performance of one-vs-all SVM classifiers according to each class. On this dataset, the average classification precision of our algorithm is around 78%, while SPH is around 20% and LFD is around 30%. We can safely come to a conclusion that, for 3D models which have inter-class variations in appearance and can be easily interacted with, our algorithm can exceed these two geometric descriptor based algorithms.

SVM	Category	Test Number	Correct Number	Correct Ratio
1	Bed	170	161	94.7059%
2	Chair	170	154	90.5882%
3	Sofa	170	143	84.1176%
4	Table	170	160	94.1176%

Table 1 Performance of four one-vs-all SVM classifiers on our affordance dataset. 170 test 3D models are input to every SVM classifier, and the Correct Ratios are the classification accuracy rates of these 4 SVM classifiers respectively.

5.4 Analysis of our algorithm

Our experimental platform is a personal computer with a Pentium(R) Dual-Core 2.6GHz CPU, and 2 GB RAM. We implement algorithm of computing AB3D in C++, and one-vs-all SVM training and testing procedures in MATLAB. Table 2 is time-consumption of computing AB3D for some typical models. Because the total size of AB3D features is less than 10 KB, the one-vs-all SVM training and testing operations can be accomplished less than 10 seconds.

The complexity of Algorithm 1 is $O(n \log n)$. This algorithm first uses RAPID [28] to compute boundaries of collision detection between action and object. The number of boundary transformations is constant, which varies from 8 to 32. And the number of rotation transformations according to every rigid transformation is 12. As a result, the time-consumption of our algorithm is constant times of RAPID algorithm's. Given the fact that RAPID has the complexity of $O(n \log n)$, Algorithm 1 also has the same complexity.

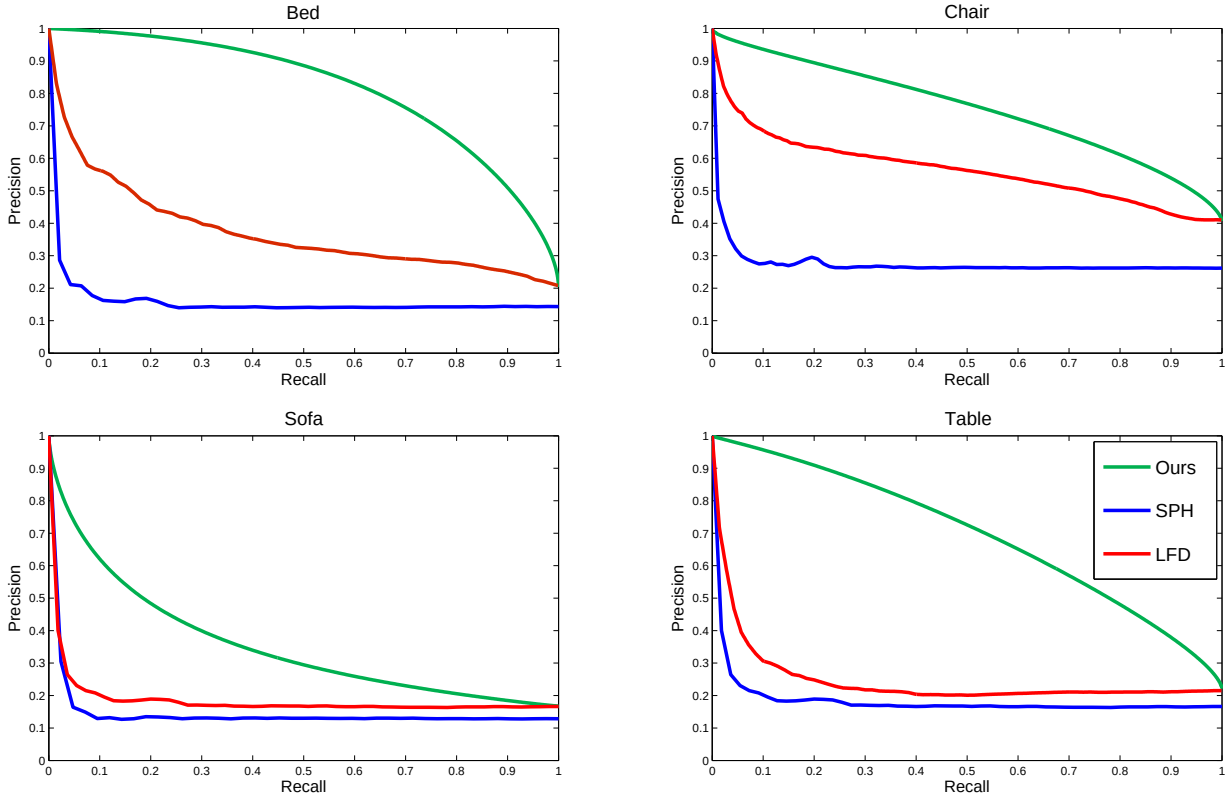

Fig. 5 Comparisons of our AB3D-based method, SPH[34] and LFD [35] on our affordance dataset. The curves are "Precision" vs. "Recall" curves of each method. Green curves represent performance of our algorithm, blue curves represent performance of SPH, and red curves represent performance of LFD. The results of SPH and LFD are generated by PSB utility [33]. For the convenience of comparison, result curves have been interpolated.

3D Model	Running time	Number of triangles
newchair288	486.594 s	195550
newsofa18	40.562 s	28740
newbed210	7.844 s	3423
newchair52	0.172 s	36

Table 2 Running time of computing AB3D for four typical 3D models.

6 Applications

In this section, we show two interesting applications based on AB3D. Since AB3D can reflect the implied meanings of interactions between 3D models and virtual human actions, it can be utilized by different applications in geometry processing area.

6.1 Consistent segmentation

Unlike methods like [3], we use AB3D to consistently segment a set of 3D models. We believe that interactions between human and objects convey consistent semantic information of these objects.

Our consistent segmentation approach is proposed as follows. (1) We use method described in paper [18] to over-segment a set of 3D models. We compute AABBs of these over-segmented parts, denoting the centers of these AABBs as $P_1, P_2, \ldots, P_n$. (2) We manually choose virtual human action for these models. For example, we choose chair_sitting action for a set of chairs. Then we compute their AB3D. (3) We denote the number we want to segment this set of 3D models as n. We segment the action into n interacting parts, denoting the center of their AABBs as $C_1, C_2, \ldots, C_n$. (4) We transform virtual human action into the scene based on AB3D. According to distances from every over-segmented part's center to $C_1, C_2, \ldots, C_n$, we tag each part as i when the distance between this part's center and C_i ($i \in (1,n)$) is the minimal.

Figure 6 shows the consistent segmentation results to a set of chairs. Figure 6(a) depicts the over-segmented results on these chairs. Figure 6(b) shows the AB3D computed by Algorithm 1. Then we segment the virtual human model into three interacting parts: "torso", "upper-leg " and "lower-leg ". The center of three parts is C_1, C_2, C_3. Figure 6(c) shows the final segmentation results. Note that these tags have semantic meanings due to their relations to virtual human parts. In the process of interaction between human and chairs, we can naturally relate "torso" to chair's back-supporting

function parts, “upper-leg” to supporting function parts, and “lower-leg” to legs parts (though legs parts are not directly interacted, but man’s lower legs are closer to chair legs when he is sitting). As a result, $tag1$ is “back support”, $tag2$ is “support”, and $tag3$ is “leg”.

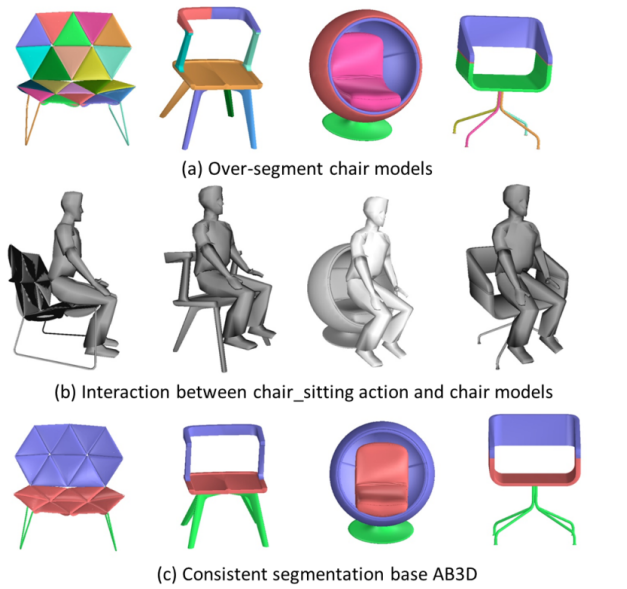

Fig. 6 The three stages of our method for consistent segmentation based AB3D. In stage (a), we first over-segment input 3D models. Then in stage (b), we transform chair_sitting action to these four chairs according to their AB3Ds respectively. In stage (c), we finish consistent segmentation through our proposed approach.

6.2 Recognition and reconstruction of scanned 3D indoor scenes

Nan et al. [24] presented a search-classify approach which interleaved segmentation and classification in an iterative manner. Using a robust classifier they traversed the scene and gradually propagated classification information. Then they reinforced classification by a template fitting step which yields a scene reconstruction.

AB3D can deal with the issue of recognition and reconstruction of scanned 3D indoor scenes. Firstly, AB3D can classify 3D models which could be well interacted with human, as described in previous sections. Indoor scenes are filled with furnitures which can be easily interacted with. Secondly, AB3D is robust when dealing with noisy point cloud data. When virtual human is interacting with objects, the most important information is the silhouettes of the object’s main parts.

Generally speaking, the point cloud data collected from 3D scanned device preserves main silhouette while having missing data in details. After down sampling, the silhouettes of the point cloud always correctly reflect the shape’s main parts, which can provide the information in the process of interacting with virtual human action. As a result, AB3D could obtain good results when dealing with point cloud data in most situations. All we have to change to algorithms described in section 4 is replacing the collision detection algorithm between meshes by collision detection algorithm between mesh and point cloud.

In this subsection, we present an algorithm using AB3D to recognize 3D shape in 3D point cloud scene. As shown in Figure 7, the workflow is: (a) We correct the height of the whole scene, and delete the walls and floor from the scene. (b) We use kNN (k-Nearest Neighbor) to cluster the point cloud. We choose the most complicated point cloud group (a table with four chairs) to run our algorithm, and the other groups will be done with the same procedure. (c) We over-segment the point cloud scene, generating a dozens of point cloud patches. (d) Then, we compute these patches’ center points, and connect them into a graph. The above 4 steps are similar to [24]. These steps only consider x and y coordinates, because the information in z coordinate has great ambiguities. As a result, kNN will make mistakes when classifying these pathes. Nan et al. [24] use their search-classify method to deal with these ambiguities.

In contrast, we use method based AB3D as follows to solve these ambiguities. (e) We use ZPclustering [40] to cluster the patches’ center points again. After ZPclustering, we can calculate clusters’ center points, which will be use as coordinate origins in AB3D computation. (f) In this step, we only choose two virtual human actions (i.e., chair_sitting and table_eating) when computing AB3D, because these two actions are enough to differentiate chairs and tables. The search space is AABB of the corresponding point cloud clusters. After computing AB3D, we can find the most suitable sitting positions. These positions are shown as four virtual human chair_sitting actions as in Figure 7(f). (g) We use 3D mesh of chair_sitting action to construct a bounding box. We use these four bounding boxes as decision boundaries to accomplish these patches’ final clustering. (h) After final clustering, we get more accurate cluster result. We tag clusters whose centers are near chair_sitting action as “chair” and others as “table”. As far as this step, we finish the recognition of the point cloud scene. (i) To accelerate reconstructing speed and improve stability of our method at the same time, we down sample the whole point cloud scene. (j) At this final step, we compare every point cloud cluster to the template models, and choose the most similar 3D models.

7 Conclusion, limitations, and future work

In this paper, we develop a new 3D descriptor, AB3D, which is computed from interactions between virtual human ac-

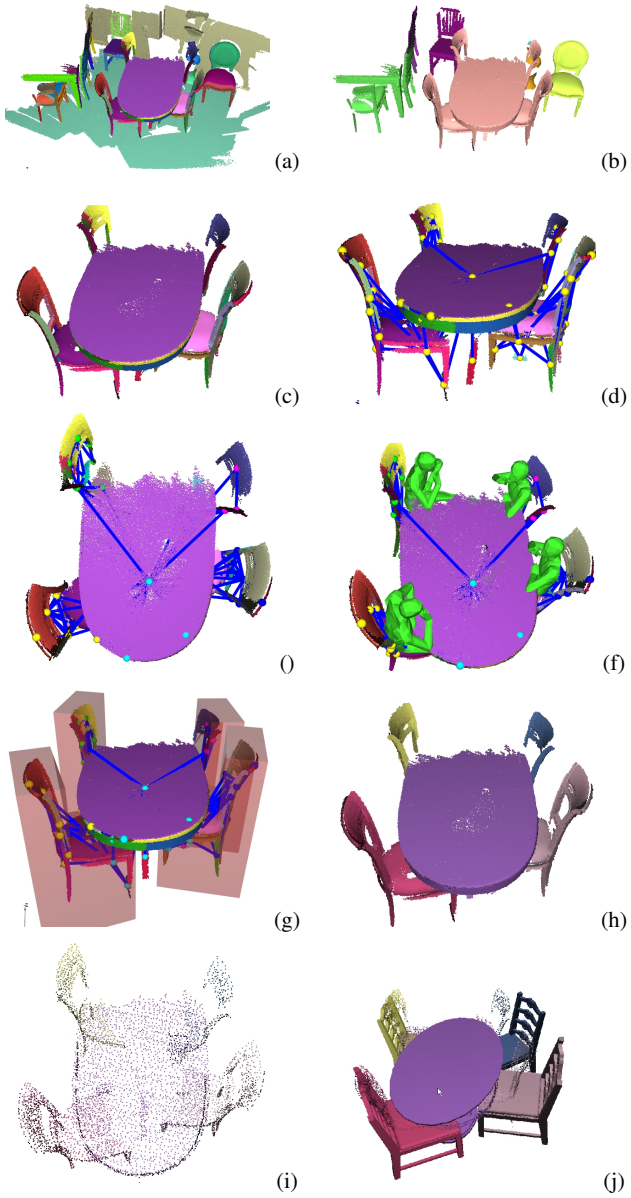

Fig. 7 The process of recognition and reconstruction of scanned 3D indoor scenes based on AB3D.

tions and 3D models. AB3D can utilize the abundant information implied in human actions, which is useful in 3D models functional analysis. We first formalize the workflow to compute AB3D, then we provide algorithms to compute AB3D for a 3D model. We conduct experiments to validate the effectiveness of our method. At last, we show two useful applications based on AB3D: consistent segmentation and reconstruction of scanned 3D indoor scenes.

Limitations AB3D can not deal with 3D models which is not easily interacted with virtual human actions. For these models, we should combine our framework to other traditional geometric descriptors to obtain correct analysis results. In subsection 6.1 and 6.2, in the action selecting step we must manually choose appropriate virtual human actions. When dealing with 3D models having a large number of triangles, our algorithm can not achieve real time so far. All these issues will limit AB3D's application.

Future work In the future, we will use GPU to accelerate speed of AB3D computation. We also want to extend our method to obtain a fully automatic algorithm for reconstruction of indoor point cloud scene. And we agree that rather than only use action-based 3D descriptor, combining with geometric descriptors when analyzing 3D shapes could be a potentially fruitful direction for further research. We regard our work as only an initial step in the direction of actions based 3D shape analysis. We believe that AB3D is a promising shape descriptor towards functionality analysis.

Acknowledgements The authors would like to thank the anonymous reviewers for their helpful comments. We also thank Liangliang Nan for providing his code of point cloud reconstruction. The original idea of this paper stemmed from research discussions between the first author and YanzhenWang, who first worked on the problem under the supervision of Dr. Hao (Richard) Zhang at Simon Fraser University.

References

1. Niloy J. Mitra, Michael Wand, Hao Zhang, Daniel Cohen-Or, and Martin Bokeloh. Structure-aware shape processing. In *EUROGRAPHICS State-of-the-art Report*, 2013.
2. Xiaobai Chen, Aleksey Golovinskiy, and Thomas Funkhouser. A benchmark for 3d mesh segmentation. *ACM Transactions on Graphics (TOG)*, 28(3):73, 2009.
3. Aleksey Golovinskiy and Thomas Funkhouser. Consistent segmentation of 3d models. *Computers & Graphics*, 33(3):262–269, 2009.
4. Kai Xu, Hao Zhang, Andrea Tagliasacchi, Ligang Liu, Guo Li, Min Meng, and Yueshan Xiong. Partial intrinsic reflectional symmetry of 3d shapes. In *ACM Transactions on Graphics (TOG)*, volume 28, page 138. ACM, 2009.
5. Marco Attene, Francesco Robbiano, Michela Spagnuolo, and Bianca Falcidieno. Characterization of 3d shape parts for semantic annotation. *Computer-Aided Design*, 41(10):756–763, 2009.
6. Tolga Abaci and Frédéric Vexo. Bridging geometry and semantics for object manipulation and grasping. In *In Proceedings of the 1st workshop towards Semantic Virtual Environments (SVE05*, pages 110–119. ACM press, 2005.
7. Ezer Bar-Aviv and Ehud Rivlin. Functional 3d object classification using simulation of embodied agent. In *BMVC*, pages 307–316, 2006.
8. Helmut Grabner, Juergen Gall, and Luc Van Gool. What makes a chair a chair? In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1529–1536. IEEE, 2011.
9. Marco Attene, Silvia Biasotti, Michela Mortara, Giuseppe Patanè, Michela Spagnuolo, and Bianca Falcidieno. Computational methods for understanding 3d shapes. *Computers & Graphics*, 30(3):323–333, 2006.
10. Silvia Biasotti, Bianca Falcidieno, and Michela Spagnuolo. Extended reeb graphs for surface understanding and description. In Gunilla Borgefors, Ingela Nyström, and GabriellaSanniti Baja,

editors, *Discrete Geometry for Computer Imagery*, volume 1953 of *Lecture Notes in Computer Science*, pages 185–197. Springer Berlin Heidelberg, 2000.
11. Niloy Mitra, Michael Wand, Hao Richard Zhang, Daniel Cohen-Or, Vladimir Kim, and Qi-Xing Huang. Structure-aware shape processing. In *SIGGRAPH Asia 2013 Courses*, page 1. ACM, 2013.
12. Vladimir G Kim, Wilmot Li, Niloy J Mitra, Siddhartha Chaudhuri, Stephen DiVerdi, and Thomas Funkhouser. Learning part-based templates from large collections of 3d shapes. *ACM Transactions on Graphics (TOG)*, 32(4):70, 2013.
13. Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3d mesh segmentation and labeling. *ACM Transactions on Graphics (TOG)*, 29(4):102, 2010.
14. Yanzhen Wang, Kai Xu, Jun Li, Hao Zhang, Ariel Shamir, Ligang Liu, Zhiquan Cheng, and Yueshan Xiong. Symmetry hierarchy of man-made objects. In *Computer Graphics Forum*, volume 30, pages 287–296. Wiley Online Library, 2011.
15. Kai Xu, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. Fit and diverse: Set evolution for inspiring 3d shape galleries. *ACM Transactions on Graphics (TOG)*, 31(4):57, 2012.
16. Yun Jiang, Hema Koppula, and Ashutosh Saxena. Hallucinated humans as the hidden context for labeling 3d scenes. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2993–3000. IEEE, 2013.
17. Abhinav Gupta and Larry S Davis. Objects in action: An approach for combining action understanding and object perception. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.
18. Kai Xu, Honghua Li, Hao Zhang, Daniel Cohen-Or, Yueshan Xiong, and Zhi-Quan Cheng. Style-content separation by anisotropic part scales. *ACM Transactions on Graphics (TOG)*, 29(6):184, 2010.
19. Ruizhen Hu, Lubin Fan, and Ligang Liu. Co-segmentation of 3d shapes via subspace clustering. In *Computer Graphics Forum*, volume 31, pages 1703–1713. Wiley Online Library, 2012.
20. Oana Sidi, Oliver van Kaick, Yanir Kleiman, Hao Zhang, and Daniel Cohen-Or. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. In *ACM Transactions on Graphics (TOG)*, volume 30, page 126. ACM, 2011.
21. Martin Reuter. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *International Journal of Computer Vision*, 89(2-3):287–308, 2010.
22. Nathan Silberman and Rob Fergus. Indoor scene segmentation using a structured light sensor. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 601–608. IEEE, 2011.
23. Hema S Koppula, Abhishek Anand, Thorsten Joachims, and Ashutosh Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *Advances in Neural Information Processing Systems*, pages 244–252, 2011.
24. Liangliang Nan, Ke Xie, and Andrei Sharf. A search-classify approach for cluttered indoor scene understanding. *ACM Transactions on Graphics (TOG)*, 31(6):137, 2012.
25. Poser pro 2010. `http://poser.smithmicro.com/poserpro.html`, 2010.
26. Google 3dwarehouse. `http://www.sketchup.google.com/3dwarehouse/`, 2013.
27. Mark W Jones, J Andreas Baerentzen, and Milos Sramek. 3d distance fields: A survey of techniques and applications. *Visualization and Computer Graphics, IEEE Transactions on*, 12(4):581–599, 2006.
28. Stefan Gottschalk, Ming C Lin, and Dinesh Manocha. Obbtree: a hierarchical structure for rapid interference detection. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 171–180. ACM, 1996.
29. Hongbo Fu, Daniel Cohen-Or, Gideon Dror, and Alla Sheffer. Upright orientation of man-made objects. In *ACM Transactions on Graphics (TOG)*, volume 27, page 42. ACM, 2008.
30. Vladimir Vapnik. *The nature of statistical learning theory*. springer, 2000.
31. Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.
32. Paolo Cignoni, Massimiliano Corsini, and Guido Ranzuglia. Meshlab: an open-source 3d mesh processing system. *Ercim news*, 73:45–46, 2008.
33. Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *Shape Modeling Applications, 2004. Proceedings*, pages 167–178. IEEE, 2004.
34. Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 156–164. Eurographics Association, 2003.
35. Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003.
36. R. C. Veltkamp and G. J. Giezeman. Shrec10 track: Large scale retrieval. In *3DOR*, pages 63–69. Wiley Online Library, 2010.
37. Afzal Godil andXianfang Sun Lian Zhouhui. Visual similarity based 3d shape retrieval using bag-of-features. In *Shape Modeling International Conference (SMI), 2010*, pages 63–69. IEEE, 2010.
38. Keinosuke Fukunaga and Patrenahalli M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *Computers, IEEE Transactions on*, 100(7):750–753, 1975.
39. Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.
40. Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2004.