



ORIGINAL ARTICLE

Surface Area Distribution Descriptor for object matching

Mohamed F. Gafar, Elsayed E. Hemayed*

Computer Engineering Department, Faculty of Engineering, Cairo University, Giza, Egypt

Received 29 October 2009; received in revised form 28 January 2010; accepted 12 February 2010

Available online 2 August 2010

KEYWORDS3D object recognition;
Volumetric descriptor;
Surface area distribution;
Shape matching

Abstract Matching 3D objects by their similarity is a fundamental problem in computer vision, computer graphics and many other fields. The main challenge in object matching is to find a suitable shape representation that can be used to accurately and quickly discriminate between similar and dissimilar shapes. In this paper we present a new volumetric descriptor to represent 3D objects. The proposed descriptor is used to match objects under rigid transformations including uniform scaling. The descriptor represents the object by dividing it into shells, acquiring the area distribution of the object through those shells. The computed areas are normalised to make the descriptor scale-invariant in addition to rotation and translation invariant. The effectiveness and stability of the proposed descriptor to noise and variant sampling density as well as the effectiveness of the similarity measures are analysed and demonstrated through experimental results.

© 2010 Cairo University. All rights reserved.

Introduction

With recent advances in technologies designed for the digital acquisition of 3D models there has been an increase in the availability and usage of 3D objects in a variety of applications. Examples of such applications include database models searching, industrial inspection, autonomous vehicles, surveillance and medical image analysis. As a result, there is a large collection of 3D objects available. This motivates the need to be able to retrieve 3D objects that are similar in shape to a given 3D object query. The accuracy of a 3D object retrieval system largely depends on finding a good descriptor that is able to represent the local and global characteristics of the 3D

object. Many descriptors have been provided in the research community. Surveys of such descriptors are available in the literature [1–4]. The descriptors can be categorised according to the way we think about the 3D object as:

Image descriptors: The object is projected onto one or several image planes producing renderings corresponding to depth maps [5], silhouettes [6], and others, from which descriptors can be derived. The effectiveness of the image based descriptors depends on the number of views taken of the object.

Surface descriptors: Surface descriptors regard the object as an ideal surface, infinitely thin, with precisely defined properties of differentiability. The descriptor is generated from the triangles on the surface. All computations are based on the relation between points or triangles on the surface. Examples of surface descriptors are curve fitting [7], geometric 3D moments [8], and pose-oblivious signature [9]. In the pose-oblivious signature, the object is described by a rectangular histogram generated by combining two histograms of the diameter function and the centricity function. The diameter function is defined for a vertex on the boundary of the object as a statistical averaging of the diameter in a cone around the direction opposite to the normal of the point. The

* Corresponding author. Tel.: +20 11 2306248; fax: +20 2 35723486.
E-mail address: hemayed@ieee.org (E.E. Hemayed).



centricity function is defined for a vertex as the average geodesic distance to all other vertices. The descriptor is invariant to transformation and surface sampling in addition to being pose invariant. In general, surface based descriptors are naturally scale-invariant and compact in terms of amount of data, but they are complex to manipulate.

Volumetric descriptors: The object is considered as a thickened surface that occupies some portion of volume in 3D space, or for watertight models as a boundary of a solid volumetric object. The descriptors are generated by cutting the volume into segments, which could be shells or sectors inside a boundary sphere, or cubes inside the boundary box [10–16]. The origin point of the boundary volume could be the centre of mass (in case of full matching) or a point on the object’s surface (in case of partial matching). A good descriptor is characterised by being invariant to translation and surface sampling. In general, volumetric descriptors are simple to construct. They approximate the object and make it simple to manipulate.

In this paper we present a descriptor that falls in the category of volumetric descriptors for full matching. We use spherical shells as partitions and the area inside each shell as the function to compute across the shells as we go from the centroid of the object to the furthest point on its surface. The proposed descriptor has the following advantages:

- It does not require any preprocessing to be generated.
- It is based on the area inside the shell, so it is surface sampling independent.
- It is normalised, so it is scale independent.
- Based on spherical shells makes it rotation independent.

The rest of this paper is organised as follows. First we discuss existing volumetric descriptors. Then we describe our descriptor’s generation algorithm. Finally, we explain and analyse the results from our experiments and provide a summary and suggestions for future work.

Related work

In volumetric descriptors, the volume occupied by the model is partitioned into cubes, shells or sectors. A selected function is computed across each partition. The function to be computed could be the number of vertices, length between the centroid and the vertices in the partition or area inside the partition.

Kazhdan et al. [12] proposed the Spherical Harmonic Representation tool as a means to transform rotation dependent shape descriptors into rotation independent ones. The key idea behind this tool was to describe a spherical function in terms of the amount of energy it contains at different frequencies. Since these values do not change when the function is rotated, the resulting descriptor is rotation invariant.

Mian et al. [13] proposed a cubes grid model with a cube size half the mean resolution of the models in the library. In this method the surface area is then computed inside each cube in the grid (including only polygons with normals making an angle < 90 with the z -axis). This descriptor is transformation and surface sampling invariant. It also supports partial matching, but does require registration of the objects to complete the matching process.

Frome et al. [14] chose a point and considered it as a centre of a sphere that is divided into bins that are generated by equally spaced

Table 1 Generation of Area Distribution Descriptor.

Input: 3D triangles mesh
Output: Feature Vector
Step 1: Get the boundary sphere whose centre is the object’s centre of gravity
Step 2: Compute surface area distribution. Divide the sphere into shells and compute the area participation in each shell.
Step 3: Normalise the shells area by dividing the area in each shell by the total area of the object surface

boundaries in the azimuth Φ , equally spaced boundaries in the elevation dimension θ , and logarithmically spaced boundaries along radial dimension R . For each bin the number of vertices is computed and divided by the volume of the bin to compensate for large variation in bin sizes with radius and elevation. The cube root is taken to leave the descriptor robust to noise which causes points to cross over bin boundaries. The descriptor is invariant to transformation, but it is not surface sampling invariant.

Papadakis et al. [15] decomposed the 3D model into a set of spherical functions which represent not only the intersections of the corresponding surface with rays emanating from the origin, but also points in the direction of each ray which are closer to the origin than the furthest intersection point. The descriptor is invariant to transformation, but is not surface sampling invariant.

Ankerst et al. [16] represented the 3D shapes by using shape histograms for which several partitions of the space are possible. Partitions represent the bins that could be shells, sectors, or spider web. For each bin the number of vertices inside the bin is computed. The descriptor is invariant to transformation, but it is not surface sampling invariant. In general, most of the available volumetric descriptors are transformation invariant, but they miss the surface sampling invariance and require some preprocessing.

Methodology

In this paper we present a surface Area Distribution Descriptor. We generate a boundary sphere whose centre is the object’s centre of mass, and divide this sphere into shells. We then compute the participated triangles areas in each shell. The descriptor is generated by the algorithm shown in Table 1.

Computing the object’s centre of gravity

The centre of gravity of a 3D object having N triangular faces with vertices (v_1, v_2, v_3) is given by Eq. (1) [17], where R_i is the average of the vertices of the face (see Fig. 1), and A_i is twice the area of the face. The cost of computing the object’s centre of gravity is $O(N)$, as it depends on the number of faces N in the object.

$$C = \frac{\sum_{i=1}^N A_i R_i}{\sum_{i=1}^N A_i} \quad (1)$$

$$R_i = \frac{v_1 + v_2 + v_3}{3} \quad (2)$$

$$A_i = \|(v_2 - v_1) \times (v_3 - v_1)\| \quad (3)$$

Computing surface area distribution

To generate the descriptor we start by finding the maximum distance from the object's centre of gravity to the farthest point on the object's surface. This distance represents the radius of the boundary sphere for the object. By dividing the radius by the number of required shells, we generate a number of spheres whose centre is the object's centre of gravity. The volume between every two successive spheres represents a shell. For each shell we compute the area of intersection between the faces and the shell.

The maximum distance between the object's centre of gravity and the farther point on the object is computed using Eq. (4), where $dist$ is the Euclidian distance and is given in Eq. (5).

$$D_{Max} = \{Max(dist(C, v_i)); \quad i = 1 \dots N\} \quad (4)$$

$$dist(v_1, v_2) = \sqrt{(v2_x - v1_x)^2 + (v2_y - v1_y)^2 + (v2_z - v1_z)^2} \quad (5)$$

Considering the object as surrounded by a sphere whose centre is the object's centre of gravity and a radius equal to D_{max} , we decompose

Table 2 Intersection area shown rounded by orange border (top view).

Cases	Diagram	Intersected edges count	Area inside the sphere
3 In 0 Out		0	Triangle area Eq. (7)
2 In 1 Out		2	Intersection polygon area Eq. (8)
1 In 2 Out		2	Intersection polygon area (triangle) Eq. (7)
1 In 2 Out		3	Intersection polygon area Eq. (8)
0 In 3 Out		0	Zero
0 In 3 Out		0	Circle area Eq. (12)
0 In 3 Out		1	Circular segment area Eq. (13)
0 In 3 Out		2	Intersection polygon area Eq. (8)
0 In 3 Out		3	Intersection polygon area Eq. (8)

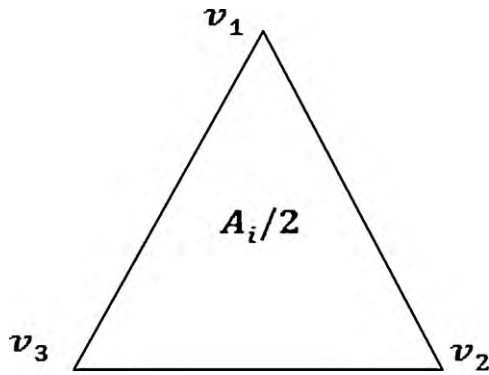


Fig. 1 Face's vertices.

this sphere into shells. Let the shell count be NS . Dividing the radius by NS , we generate NS shells cutting the faces of the object. Each shell has an outer and inner sphere. The cost of generating the shells is equal to the cost of finding the maximum distance plus the cost of dividing the maximum distance by the number of required shells. Thus the cost of generating the shells is $O(N)$.

The feature vector FV is computed by aggregating the intersection area of every face F_i with every shell S_j (see Fig. 2), where NS is the number of shells and N is the number of faces. This generates a feature vector with length equal to the number of shells. The cost of generating the feature vector depends on the number of faces and the number of shells which is $O(N*NS)$.

$$FV = \left\{ FV_j = \left(\sum_{i=1}^N Area(S_j \cap F_i) \right); \quad j = 1 \dots NS \right\} \quad (6)$$

For each vertex in the triangle face we acquire the container shell. We then find the intersection areas between the face and the shell's inner and outer spheres. By subtracting these two areas, we compute the participated area in each shell.

To find the intersection area of a sphere and a face, we compute the intersection points between the sphere and every face's edges. Table 2 shows the possible cases for the sphere and the triangle

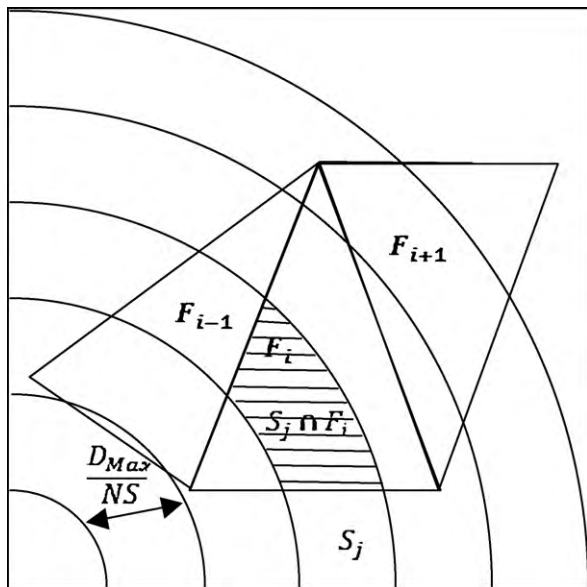


Fig. 2 The intersection between surface triangle and the shells.

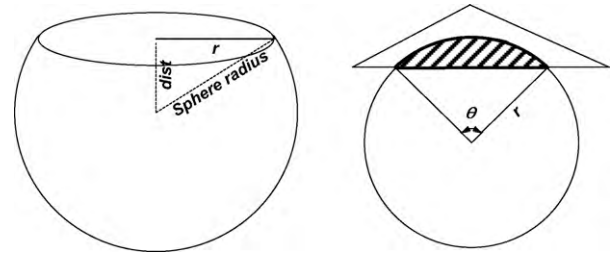


Fig. 3 Left: The radius of the circle resulting from the intersection of the plane and the sphere. Right: The circular segment is hashed.

intersection and the area computation for each case. We assign each vertex of the face a label to describe its location in relation to the sphere. The labels are In and Out. Based on the labels count for the triangle and the number of edges that intersect with the sphere we compute the area.

The area of the intersection region can be computed according to Eqs. (7)–(13). The area of a triangle with vertices v_0, v_1, v_2 is computed using Eq. (7) and the area of the planar polygon with vertices $v_0 \dots v_{n-1}$ and a normal N can be computed using Eq. (8) [18].

$$A = \frac{1}{2} |(v_1 - v_0) \times (v_2 - v_0)| \quad (7)$$

$$A = \frac{N}{2} \cdot \sum_{i=0}^{n-1} v_i \times v_{i+1} \quad (8)$$

To compute the area of the circle and the circular segment we need to compute the radius of the circle. First we compute the distance between the sphere's centre and the triangle's plane. To find the triangle's plane we use the three vertices to generate the equation of the plane using Eq. (9) [19].

$$ax + by + cz + d = 0 \quad (9)$$

The perpendicular distance (see Fig. 3) between the sphere's centre C and the plane can be computed using Eq. (10). The radius of the circle r can then be easily computed using Pythagorean Theorem (Eq. (11)).

$$dist = \frac{|aC_x + bC_y + cC_z + d|}{\sqrt{a^2 + b^2 + c^2}} \quad (10)$$

$$r = \sqrt{\text{sphere radius}^2 - dist^2} \quad (11)$$

The area of the circle is computed using Eq. (12) and the area of the circular segment [20] using Eq. (13).

$$A = \pi r^2 \quad (12)$$

$$A = \frac{r^2}{2} (\theta - \sin(\theta)) \quad (13)$$

The participated triangle's area in each shell is the difference between the area inside the outer shell and the area inside the inner shell. The feature vector is the summation of the participated triangle's areas in each shell. The algorithm used to compute the feature vector is shown in Table 3.

Normalisation

To support scaling independency we normalise the feature vector. We divide the feature vector by the total area of the object's faces. The normalised feature vector (NFV) is computed using Eq.

Table 3 Participation area computation.

PA is participated area
 For each shell s_j in shells
 For each face F_i in mesh faces
 A_o = Intersection area with the outer sphere
 A_i = Intersection area with the inner sphere
 $PA = A_o - A_i$
 $FV_{j+} = PA$
 Next face
 Next shell

(14), where FV is the feature vector and A is the object's total area computed as the summation of all faces of the object.

$$NFV = \frac{FV}{A} \quad (14)$$

$$A = \sum_{i=1}^N A_i \quad (15)$$

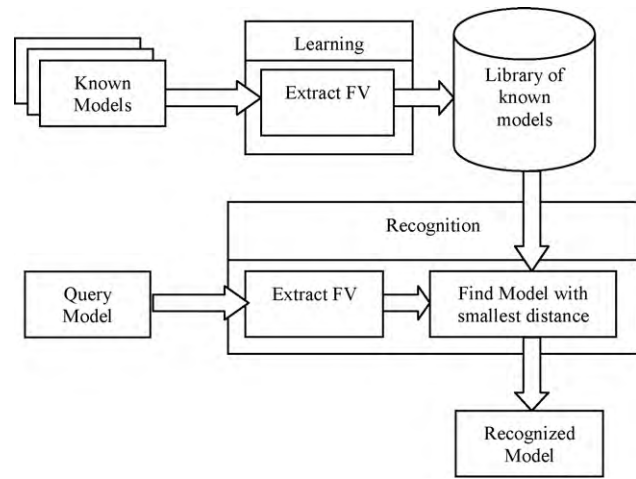
The cost for computing the normalised feature vector depends on the number of shells, which means it is $O(NS)$. The cost of computing the total area is ignored as it has been computed while computing the centre of gravity of the object. From the computation steps discussed previously, we can state that these steps can be used in manifold and non-manifold triangle meshes.

The merits of the Area Distribution Descriptor can be summarised as follows:

1. *Translation independent*: Because it takes the centre point of the object as the origin of the shells.
2. *Rotation independent*: Because it divides the object into spherical shells.
3. *Scaling independent*: Because it normalises the vector based on the total surface area.
4. *Surface sampling independent*: Because it takes the area distribution, not the points' distribution.

Object recognition











The proposed surface Area Distribution Descriptor can be used for describing and recognising 3D models in a library. Fig. 4 shows the block diagram of the object recognition process. This can be broken into two phases: the learning phase (offline) and the recognition

**Fig. 4** The block diagram for the recognition process.

phase (online). In the learning phase, we compute the feature vector for the object and store it in the library of known objects. In the recognition phase, we compute the feature vector for the query object and then compare it to the feature vectors of the known objects. The comparison is based on the distance function. The known object with the smallest distance is considered to be the best match for the query object. A survey of the distance functions that can be used to compare feature vectors is presented by Cha [21]. In our experiments we compare different methods to compute the distance (Euclidian, Chi-Square, Intersection, TaniMoto).

The recognition phase is considered a k -NN search problem (with $k=1$) where the unknown object is compared to all known objects. The recognition computation time in this case is $O(N)$, where N is the number of known objects. Thus, this approach is applicable in the case of a small number of known objects. It will be more efficient if we consider higher values for k when the number of objects increases. In this case, nearby objects (based on k) are clustered in one class. In this case, the recognition process is performed in two steps; getting the nearest class to the unknown object then getting the nearest object from within this class. The recognition computation time in this case is $O(k) + O(N/k)$. In this paper, we focus more on presenting the developed descriptor. Thus the selection of k and analysing the recognition process time are beyond the scope of this paper.

Table 4 Models used in the experiments.

Model					
Name	Deinonych	Dino	Plateosaurus	Raptor	Dino1043
Vertices count	13695	23984	8306	12794	554
faces count	26894	47903	16062	25601	1023
Model					
Name	Nessyb	Pthi	Dragon2	Vp4009-dragon	Dinopet
Vertices count	4920	3098	54319	4922	8047
Faces count	9552	6170	108588	9831	15945

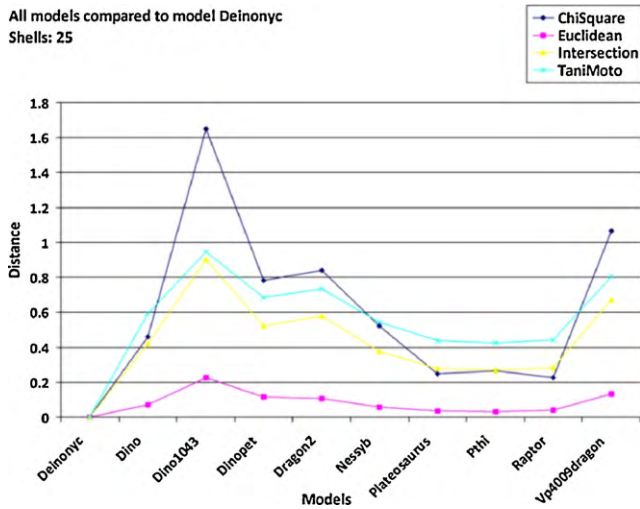


Fig. 5 Distance between models and model Deinonyc.

Results and discussion

In our experiments we used dinosaurs' models from the INRIA Gamma research dataset [22] and dinosaurs models from the Princeton Shape Benchmark [23]. Table 4 shows some objects from INRIA dataset. In these experiments we studied the stability of the descriptor against noise, variant sampling density, rotation and scale. We also studied the effectiveness of the distance computation methods for measuring the similarity between the models' descriptors.

Distance computation

In this experiment we compared each model with all other models. For each comparison we computed the distance using four different methods (Chi-Square, Euclidian, Intersection, TaniMoto). The following are the formulas for computing these distances [21]:

Euclidian distance

$$dist(FV1, FV2) = \sum_{j=1}^{NS} (FV1_j - FV2_j)^2 \quad (16)$$

Chi-Square distance

$$dist(FV1, FV2) = \sum_{j=1}^{NS} \frac{(FV1_j - FV2_j)^2}{(FV1_j + FV2_j)} \quad (17)$$

Intersection distance

$$dist(FV1, FV2) = 1 - \sum_{j=1}^{NS} \text{Min}(FV1_j, FV2_j) \quad (18)$$

TaniMoto distance

$$dist(FV1, FV2) = \frac{\sum_{j=1}^{NS} (\text{Max}(FV1_j, FV2_j) - \text{Min}(FV1_j, FV2_j))}{\sum_{j=1}^{NS} \text{Max}(FV1_j, FV2_j)} \quad (19)$$

In this experiment we considered the shell numbers to be 25. Fig. 5 shows the results for model "Deinonyc" compared to the other 10 models. By comparing the slope of the "Chi-Square" line with other lines, we can find that it changes more sharply between models. That means it is the most distinguishable distance computation method for the given experimental data. After "Chi-Square" comes "TaniMoto", as it has a higher distance between the model and others. The conclusion of this experiment is that "Chi-Square" is the best distance computation method for the "Area Distribution Descriptor".

Descriptor stability

Several experiments were conducted to analyse the stability of the descriptor to different factors; noise level, sampling density, object rotation and object scaling. The details and the analysis of these experiments are discussed below. Since the proposed descriptor is affected by the number of shells, we have shown the results for different shells count (5, 25, 55, 75 and 105). The selection of the number of shells is determined by the density and the size of the mesh triangles. A larger number of shells means a longer computation time and lower tolerance to noise. On the other hand, a smaller number of shells means less computation time but less discriminating power since most of the object details are lumped together in the large shells. In our experiments, we found that 25 shells are good enough for the size of objects that we considered in our experiment.

Noise effect: In this experiment we added Gaussian noise to the model with different deviations. For the applied Gaussian noise we set the mean value μ to be the length of the edge. For the deviation value σ we changed it in the range [0,1] with increasing steps equal to 0.1. Table 5 shows the algorithm used to apply noise on a model.

We used different shells counts, from 5 to 125 with an increasing step of 10. Fig. 6 shows the results for the "Chi-Square" method and shells count (5, 25, 55, 75, and 105). As can be seen in the figure, when the deviation is less than 0.5 the effect of the noise on the distance is small: the distance from the original object ranges between [0,0.02]. Starting from deviation 0.5, a small shells count gives better results than a high shells count. The conclusion of this experiment is that increasing the shells more than 25 makes the noise effect increase if the deviation is more than 0.4.

Sampling density effect: The sampling density in 3D models is represented by the number of faces. In our experiment, we acquired

Table 5 Applying noise on given mesh.

P is percentage of the edge length
 $GetRandom(\mu, \sigma)$ is a method that returns a random number based on Gaussian distribution with mean μ and deviation σ
 L is edge length
 L_{new} is the new edge length after applying noise
 v_1, v_2 are start and end point of the edge

```

P = 0
Do
  P = P + 0.1
  For each face in mesh faces
    For each edge in face
       $L_{new} = GetRandom(L, P * L)$ 
       $v_2 = v_1 + L_{new}(v_2 - v_1)$  //move the second vertex
    Next edge
  Next face
While P ≤ 1

```

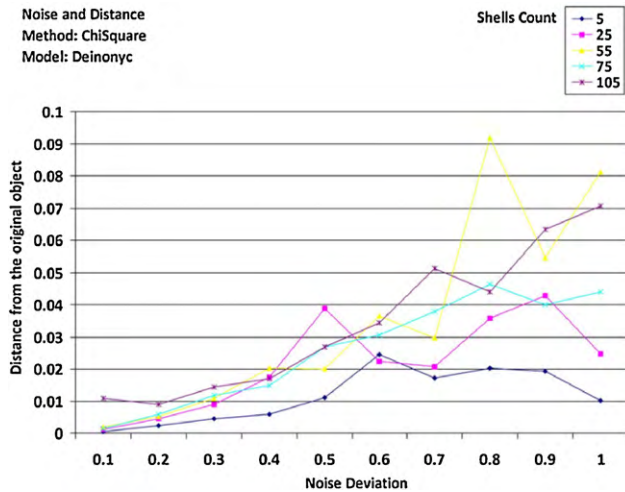


Fig. 6 Noise deviation vs. distance.

one model and generated models with different sampling densities from it. The generated models' faces count is a percentage of the original model's faces count, ranging from 0.1 to 2. We compared the original model with the generated models. Fig. 7 shows the results for "Chi-Square" method and shells count (5, 25, 55, 75, and 105). As can be seen in the figure, the effect of surface sampling change (in the period [0.1, 2.0]) is very small for all the shells count. It is less than 0.01 except for the sampling percentage 0.1 of the original object, and the distance is still less than 0.07 which is still a small value. The conclusion of this experiment is that the descriptor can be considered sampling density independent.

Rotation effect: In this experiment we rotated the object from 0 to 360° (with an increasing step of 10°), comparing it to the original object. Fig. 8 shows the results for the "Chi-Square" method and shells count (5, 25, 55, 75, and 105). As can be seen, the effect of rotation is negligible. The distance between the rotated object and the original object for any rotation degree is less than 1.40E−26, which can be considered 0. The conclusion of this experiment is that the descriptor is rotation independent.

Scaling effect: In this experiment we scaled the object 0.1 to 3 times the original object (with an increasing step of 0.1), comparing in each case with the original object. Fig. 9 shows the results for the "Chi-Square" method and shells count (5, 25, 55, 75, and 105).

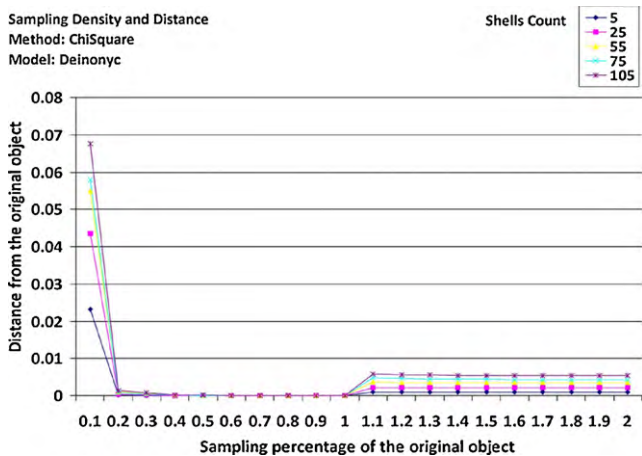


Fig. 7 Sampling density vs. distance.

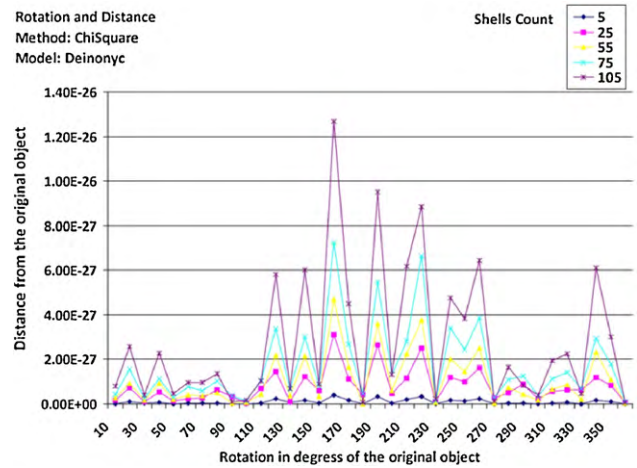


Fig. 8 Rotation angle vs. distance.

As can be seen in the figure, the effect of scaling is negligible. The distance between the scaled object and the original object for any scale ratio is less than 3.00E−26, which can be considered 0. The conclusion of this experiment is that the descriptor is scale independent.

Computation time

In this experiment we studied the time required to generate the descriptor for models with different numbers of faces, ranging from 60 faces to 108,588 faces. In our experiments we used a laptop with Intel Core 2 CPU 2.0 GHz and 2 GB memory, running Windows 2008 Server 32 bit operating system. The application was built on the .Net framework 3.5 with C# language. Fig. 10 shows the results for models with faces count less than 50,000 with shells counts (5, 25, 55, 75, and 105). As can be seen, increasing the shells to more than 25 makes the descriptor's generation time increase for faces count greater than 10,000. For shells count 25 the descriptor generation time is less than 10s for faces count up to 10,000. The conclusion of this experiment is that using shells count more than 25 is time consuming.

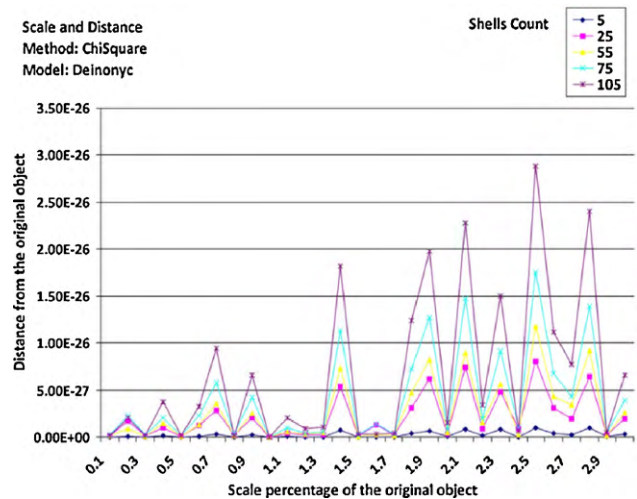


Fig. 9 Scaling percentage vs. distance.

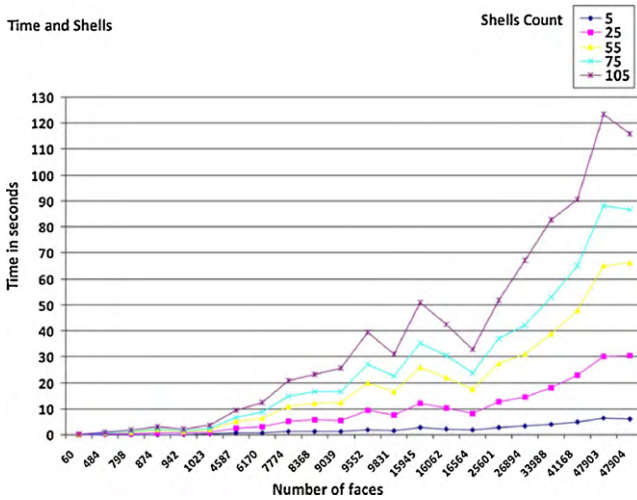


Fig. 10 Computation time vs. faces count.

To estimate the computation time of the descriptor generation algorithm, we list the algorithm along with the step number in Table 6. Here number of faces is N and number of shells is NS . The algorithm computation time can be computed as follows. The computation time of the descriptor is in $O(N*NS)$.

Time to compute the centre of gravity

$$\text{Step 1} = T(N) = O(N) \quad (20)$$

Time to generate shells

$$\text{Step 2} = T(N) = O(N) \quad (21)$$

Time to compute participated area

$$\text{Step 3, 4, 5, 6, 7, 8, 9, 10} = T(N * NS * 4) = O(N * NS) \quad (22)$$

Time to normalise the feature vector

$$\text{Step 11} = T(NS) = O(NS) \quad (23)$$

In summary, the “Area Distribution Descriptor” is independent of scaling, rotation and sampling density. The most effective

Table 6 Feature vector generation.

Input: 3D triangles mesh
Output: Feature vector (FV)
1 Get the boundary sphere whose centre is the object's centre of gravity
$C = \frac{\sum_{i=1}^N A_i R_i}{\sum_{i=1}^N A_i}$
2 Find the radius of the bounding sphere and divide this sphere into shells
3 For each shell s_j in shells
4 For each face F_i in mesh faces
5 A_o = Intersection area with the outer sphere
6 A_i = Intersection area with the inner sphere
7 $PA = A_o - A_i$
8 $FV_{j+} = PA$
9 Next face
10 Next shell
11 Normalise the shells area by dividing the area in each shell by the total area of the object surface

distance computation method to compare the descriptor is “Chi-Square”. The descriptor is less sensitive to Gaussian noise with deviation up to 0.4 of the edge length. Using more than 25 shells would be time consuming for computation and comparison of the descriptor.

Conclusions

In this paper we have presented a new volumetric descriptor based on the surface area distribution. The proposed descriptor is used to match objects under rigid transformations including uniform scaling. The descriptor represents the object by dividing it into shells and acquiring the area distribution of the object through those shells. The computed areas are normalised to make the descriptor scale-invariant in addition to rotation and translation invariant. The descriptor construction process can be used in manifold and non-manifold triangle meshes. The experimental results demonstrate the effectiveness and stability of the descriptor to noise and variant sampling density. Results also showed that the descriptor is less sensitive to Gaussian noise with deviation up to 0.4 of the edge length. The most effective distance computation method to compare the descriptor is “Chi-Square”. The results also showed that using more than 25 shells would be time consuming for computation and comparison of the descriptor. In this paper, we considered only full object matching and will address partial object matching in future work.

References

- [1] Atmosukarto I, Shapiro LG. A salient-point signature for 3D object retrieval. Proceedings of the 1st International ACM Conference on Multimedia Information Retrieval Mir2008 Co Located with the 2008 ACM International Conference on Multimedia mm 08 New York, NY, USA: Association for Computing Machinery (ACM); 2008.
- [2] Bustos B, Keim D, Saupe D, Schreck T. Content-based 3D object retrieval. IEEE Comput Graph Appl 2007;27(4):22–7.
- [3] Iyer N, Jayanti S, Lou K, Kalyanaraman Y, Ramani K. Three-dimensional shape searching: state-of-the-art review and future trends. Comput Aided Des 2005;37(5 SPEC.ISS.):509–30.
- [4] Tangelder JWH, Veltkamp RC. A survey of content based 3D shape retrieval methods. Multimed Tools Appl 2008;39(3):441–71.
- [5] Chen DY, Tian XP, Shen YT, Ouhyoung M. On visual similarity based 3D model retrieval. Comput Graph Forum 2003;22(3):223–32.
- [6] Heczko M, Keim DA, Saupe D, Vranic DV. Verfahren zur Ähnlichkeitssuche auf 3D-Objekten [Methods for similarity search on 3D databases]. Datenbank spektrum 2002;2(2):54–63.
- [7] Zaharia T, Preteux F. 3D shape-based retrieval within the MPEG-7 framework. In: Proceedings of SPIE—The International Society for Optical Engineering. San Jose, CA, USA: Institut National des Telecommunications (France); 2001. pp. 133–145.
- [8] Paquet E, Rioux M, Murching A, Naveen T, Tabatabai A. Description of shape information for 2-D and 3-D objects. Signal Process Image Commun 2000;16(1):103–22.
- [9] Gal R, Shamir A, Cohen D. Pose-oblivious shape signature. IEEE Trans Visual Comput Graph 2007;13(2):261–70.
- [10] Vranic DV, Saupe D. 3D shape descriptor based on 3D Fourier transform. In: Fazeekas K, editor. Proceedings of the EURASIP Conference on Digital Signal Processing for Multimedia Communications and Services. 2001.
- [11] Suzuki MT, Kato T, Otsu N. A similarity retrieval of 3D polygonal models using rotation invariant shape descriptors. IEEE Int Conf Syst Man Cybern 2000;4:2946–52.
- [12] Kazhdan M, Funkhouser T, Rusinkiewicz S. Rotation invariant spherical harmonic representation of 3D shape descriptors. In: Proceedings

- of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing: Eurographics Association. 2003.
- [13] Mian AS, Bennamoun M, Owens RA. 3D recognition and segmentation of objects in cluttered scene. Proceedings - Seventh IEEE Workshop on Applications of Computer Vision, WACV 2005, art. no. 4129453, pp. 8–13; IEEE Computer Society Washington, DC, USA; 2005.
- [14] Frome A, Huber D, Kolluri R, Bülow T, Malik J. Recognizing objects in range data using regional point descriptors. *Lect Notes Comput Sci* 2004;3023:224–37.
- [15] Papadakis P, Pratikakis I, Perantonis S, Theoharis T. Efficient 3D shape matching and retrieval using a concrete radialized spherical projection representation. *Pattern Recogn* 2007;40(9):2437–52.
- [16] Ankerst M, Kastenmüller G, Kriegel HP, Seidl T. 3D shape histograms for similarity search and classification in spatial databases. Berlin/Heidelberg: Springer; 1999.
- [17] Bourke P. Calculating the area and centroid of a polygon; 1988. Available: Local.wasp.uwa.edu.au/~pbourke/geometry/polyarea/.
- [18] Goldman RN. Area of planar polygons and volume of polyhedra. In: Arvo J, editor. *Graphics gems II*. New York: Academic Press Inc.; 1991. p. 170–1.
- [19] Vince JA. *Mathematics for computer graphics*. 2nd ed. Springer; 2005.
- [20] Weisstein EW. Circular Segment; 1999. Available: <http://mathworld.wolfram.com/CircularSegment.html>.
- [21] Cha SH. Comprehensive survey on distance/similarity measures between probability density functions. *Int J Math Models Methods Appl Sci* 2007;1(4):300–7.
- [22] INRIA GAMMA Group. 3D meshes research database; 2008. Available: <http://www-roc.inria.fr/gamma/gamma/download/>.
- [23] Princeton shape benchmark; 2005. Available: <http://shape.cs.princeton.edu/benchmark/>.