

Describing 3D Geometric Primitives Using the Gaussian Sphere and the Gaussian Accumulator

Zahra Toony · Denis Laurendeau · Christian Gagné

Received: date / Accepted: date

Abstract Most complex object models are composed of basic parts or primitives. Being able to decompose a complex 3D model into such basic primitives is an important step in reverse engineering. Even when an algorithm can segment a complex model into its primitives, a description technique is still needed in order to identify the type of each primitive. Most feature extraction methods fail to describe these basic primitives or need a trained classifier on a database of prepared data to perform this identification. In this paper, we propose a method that can describe basic primitives such as planes, cones, cylinders, spheres, and tori as well as partial models of the latter four primitives. To achieve this task, we combine the concept of Gaussian sphere to a new concept introduced in this paper: the Gaussian accumulator. Comparison of the results of our method with other feature extractors reveals that our approach can distinguish all of these primitives from each other including partial models. Our method was also tested on real scanned data with noise and missing areas. The results show that our method is able to distinguish all of these models as well.

Keywords Shape description Principal primitives
Gaussian sphere Gaussian accumulator

Z. Toony · D. Laurendeau · C. Gagné
Computer Vision and System Laboratory, Department of Electrical and Computer Engineering, Université Laval, Québec, QC, Canada
E-mail: zahra.toony.1@ulaval.ca

D. Laurendeau
Tel.: +1 418-656-2979
Fax: +1 418-656-3159
E-mail: denis.laurendeau@gel.ulaval.ca

C. Gagné
E-mail: christian.gagne@gel.ulaval.ca

1 Introduction

The availability of fast and accurate 3D sensors has favored the development of different applications in assembly, inspection, Computer-Aided Design (CAD), reverse engineering, mechanical engineering, medicine, and entertainment, to list just a few. While 2D cameras capture 2D images of the surface of objects, either black-and-white or color, 3D cameras provide information on the geometry of an object surface. Today, newly introduced 3D cameras can acquire the appearance and geometry of objects concurrently.

One of the most difficult tasks in machine vision consists in object recognition/description. Over the past four decades, object recognition has been a very active research topic [30]. To identify an object and add a new object automatically to a database requires segmentation, tracking and classification algorithms [45]. Object recognition can be split into two categories. The first category refers to the methods dealing with 2D images and for which recognition is achieved using image intensity. The second category considers 3D geometric models built from point clouds or meshes for recognition [30]. A recognition method needs an embedded feature description process in order to recognize the type of each model.

It has been observed that 85% of complex objects found in industrial applications can be modeled by planes, cylinders, cones and spheres [36]. Adding the toroidal models in the set of primitives allows 95% of industrial objects to be described [36]. Discriminating or, in other words, describing the type of these simple/basic primitives is usually a difficult task. Most descriptors or methods that extract features from models work properly on complex models because they extract these features from different parts of an object which

have specific properties, e.g. high curvature parts. But since the basic models are similar in their different parts and there is no specific property in their parts, most of these methods fail to extract discriminative features. So, a specific feature descriptor is needed to describe each of these primitives. Some well-known descriptors such as SIFT [26] and Spin Images [20] are proposed for complex models, some other descriptors such as D2 [33] and Laplacian spectra descriptor [53] are proposed for simple models. More precisely, Laplacian spectra is introduced to extract features from CAD models.

In this paper, a method for describing the simple basic geometric primitives of 3D models is presented. The primitives of interest are planes, cylinders, cones, spheres, and tori, as well as partial instances of the latter four types. The method first decides whether a given unknown primitive belongs to the plane-cylinder-cone class or to the sphere-torus class. When the unknown primitive belongs to the plane-cylinder-cone class, the Gaussian sphere of the primitive is analyzed to find the correct primitive type. When the unknown primitive belongs to the sphere-torus class, a new concept, the Gaussian accumulator, is introduced to identify the correct type of primitive.

The goal of this paper is to propose a method to identify the type of geometric primitives to which parts of 3D models resulting from a segmentation process belong to. The method is based on the analysis of simple surface properties (e.g. normal to the surface, distribution of the normals) and does not require the use of a classifier for primitive identification. Experiments show that the method works well for CAD models or real scans of 3D objects and it can deal with complete or partial primitives. The proposed method finds applications in many fields such as reverse engineering of 3D parts, part-to-CAD analysis and quality control.

The motivation for this work is to support reverse engineering or part-to-CAD applications. In reverse engineering for instance, an object needs to be scanned by a 3D scanner. The scan then needs to be processed in order to find its basic geometric primitives that are then assembled as a CAD model. A common approach for processing the 3D scans is to use segmentation algorithms that identify the different segments of the object. It is then required to find to which type of geometric primitive (e.g. plane, sphere, cylinder, cone, torus) a segment belongs to. This paper proposes an approach for achieving this last task. For the applications targeted by this paper, an accurate 3D scanner (i.e. a metrologic sensor) is used since Kinect-like sensors do not provide the level of accuracy that is required in metrology.

The reminder of the paper is organized as follows. Related work is presented in Section 2. In Section 3, we expose the problem with more details and introduce the new descriptor that is proposed to discriminate basic primitives from each other. Section 4 shows the efficiency of the approach and presents experimental results obtained with different synthetic models. These results are compared with other methods. The method is also tested on real scanned models and on the results of a segmentation approach [48]. Finally, we conclude the paper in Section 5 and propose directions for future research.

2 Related Work

Consider a database of three-dimensional (3D) models. An application of 3D object recognition consists in matching objects in the scene with models in the database and to estimate their pose. Object pose is determined by six degrees of freedom in 3D (three translation parameters and three rotation parameters). So, the problem of 3D object recognition consists in recognizing a model in a 3D scene with respect to models in a database while considering its 3D pose.

The 3D data which contains geometric properties can be obtained from different sensors such as stereo cameras, time of flight laser scanners, active sensors such as the Microsoft Kinect or Panasonic DI-Imager [30]. All of these sensors capture a 2.5D scan of the object. In order to capture the entire 3D geometry model we need to capture scans from different viewpoints and then apply registration and integration algorithms in order to obtain the 3D model. There are also sensors such as the Creaform Go!Scan 3D scanner which captures the geometry of an object and, as an output, returns a complete 3D mesh model with no need for registration/integration.

Several issues should be considered when working with 3D models [30]: *occlusion*, when part of a 3D object is self-occluded or is occluded by another object; *clutter*, when a scene contains objects that are close to each other and may touch; *noise*, capturing the 3D model of an object always contains noise caused by the sensor and by the quality of scanning; and *sampling resolution*, the 3D model can be captured with different sensors and also with different accuracy levels which may affect the captured geometry.

The analysis and retrieval of a 3D mesh (a complete geometrical model) is commonly used in a number of real-world applications such as object recognition and image retrieval. The 3D meshes are generally very large in size and irregular in both shape and resolution. It is not easy to design descriptors capturing the geometric

structure of even the simple 3D meshes such as spheres, and cylinders.

Such feature sets are called shape descriptors which can describe shapes locally or globally [7]. A global descriptor represents a whole mesh as a feature vector which includes some information such as area, volume, statistical moments, Fourier coefficients, eigendecomposition of Laplace-Beltrami operator [8], wavelet parameters [34], diffusion embeddings [51], and skeleton based representations [22].

2.1 Global Descriptors

Several global descriptors are proposed in the literature to describe the whole object using a single compact representation of shape. These methods are efficient but sensitive to occlusion and clutter [30]. One of these well-known approaches, the D2 descriptor, was presented in 2002 by Osada et al. [33]. They first sample random pairs of points from the model and then calculate the Euclidean distance between each pair. They then construct a histogram based on the intervals of distances. This histogram is the global descriptor of the 3D model and works better for simple objects than for complicated 3D models. These descriptors are compared with each other using the L_1 norm.

A similar descriptor, the Surface-Pair-Relation Histograms, presented in [49], samples a pair of oriented points. Instead of computing only the distances between these points, it rather finds a four-dimensional feature vector that is invariant to rotation and translation. The features include distances and normal information. A classifier is then used to learn this 4D feature descriptor to recognize the objects. A robust recognition rate is obtained by using Kullback-Leibler and likelihood matching approaches. The method is still sensitive to noise.

The global descriptor presented in [17] calculates the surface normal and the curvature value. It then takes locally the depth value of each pixel of the object and groups them in a multidimensional histogram as a descriptor. Another global descriptor called Potential Well Space Embedding is proposed in [41]. It is based on comparing a 7D error function using the properties of the ICP (Iterative Closest Point) algorithm [4]. Object recognition is performed by comparing the feature vectors of runtime data with those of the preprocessed database.

The Spherical Harmonic Descriptor by Kazhdan et al. [21] is another global descriptor. This rotation invariant shape-descriptor is based on spherical harmonics. The main idea consists in decomposing a 3D model into

a collection of functions defined on concentric spheres and to use spherical harmonics to discard orientation information (phase) for each one. Another approach based on Laplacian spectra [53] has been proposed for CAD models. Each 3D model is simplified using the idea of progressive meshes introduced by Hoppe [18]. Then the perturbed Laplacian spectrum approach is applied to extract the features. Using these values, a spectral distribution is built for each model. These distributions are then compared using the Kullback-Leiber divergence measure.

Recently, in addition to global descriptors, local descriptors have also attracted attention [7, 5, 29]. In such approaches, a mesh is described by a set of local descriptors, each of them characterizing a specific neighborhood of vertices in the mesh. Each mesh can be described by a different number of descriptors. For local descriptors, a set of vertices is defined as the neighborhood on which the descriptor value is computed. These vertices are called interest points. The selection of interest points is important and allows the identification of variant or invariant features.

In the following, we cover the relevant work on interest point detectors, local descriptors for mesh objects, and the Bag of Words (BoW) approach in mesh retrieval as well.

2.2 Interest Point Detectors

In a given mesh, we wish to identify a set of vertices that can describe the shape of a surface well. There are many different ways to select these vertices, their neighborhood as well as the shape descriptors. A performance evaluation of several kinds of 3D detectors has been performed recently in [46]. As an example, in [29], the authors use Principal Component Analysis (PCA) to extract the features. For each vertex, they define a sphere centered on the vertex with a specific radius. Then, all vertices inside the sphere are defined as neighbors. A covariance matrix of the neighborhood is then constructed and the eigenvalues are calculated. And finally, the feature points are the vertices for which the ratio between the two leading eigenvalues is maximized.

In [37], the eigendecomposition of the Laplace-Beltrami operator is first calculated over the mesh. Based on the smallest eigenvalues of the Laplace-Beltrami operator, the eigenfunctions are computed and the feature points are the local extremum points of these functions. Two other methods based on Laplace-Beltrami operator are proposed to find surface features. In [12] and [44], a Heat Kernel Signature (HKS) function is calculated over the surface using the Laplace-

Beltrami operator and extremum points of the HKS are considered as feature points.

Other methods are extracting feature points locally. Sipiran and Bustos [42] proposed an extension of the seminal Harris corner detection scheme [16] which considers a quadratic patch as the vicinity of a vertex to extract feature points.

Litman et al. [25] represent the surface as a tree using a graph-based diffusion formulation and then compute graph theoretic measures. Some other methods such as [50] exploit the difference of Gaussian (DoG) operators as a detector. They first apply a set of Gaussian filters on a mesh's curvature function and compute the mesh's octaves. To calculate the DoG functions they subtract each octave from its subsequent octave. The local maxima of the DoG function are selected as the interest points. It is in fact a Mesh-DoG operator. A Mesh-SIFT operator was proposed in [27] for which local features are scale-space extrema. Another method based on DoG was proposed in [6]. They first apply Gaussian filters to the mesh and find differences between the octaves to obtain DoG function. This procedure is then repeated on a different downsampled version of the input surface. The interest points are the local maxima of the DoG function, repeated at various downsampled ratios. Recently, a new method was proposed in [7], which uses an improved DoG based detector. For each interest point, an intrinsic scale detector is proposed to derive two scale invariant local features.

2.3 Local Descriptors

Feature detection methods are often used as a preprocessing step to describe a mesh or a surface with a reduced amount of data. A recent comprehensive evaluation of 3D local descriptors as well as a survey on local surface features have been proposed in [14] and [13], respectively. In [15], a new local reference frame has also been introduced in order to extract local features for 3D recognition. In the following we present two well-known local descriptors. The Spin Image mesh descriptor, which was proposed in [20], contains two dimensional accumulators. One is a (positive) perpendicular distance to the surface normal direction, β , and the other one is the signed (positive or negative) perpendicular distance to the tangent plane (P), α . The plane is determined by the vertex normal and position (n, p) . Thus, a rotation-invariant descriptor sums all accumulators of the vertex related to (α, β) in a radius around the interest point (see figure 1).

The Mesh-DOG and the Mesh-SIFT approaches that are presented in [50] and [27] respectively introduce a detector and also a descriptor where the Mesh-

DOG is invariant to rotation, translation and scale and the Mesh-SIFT is invariant to rotation and translation.

As 3D data sets are common in medical image processing applications, the 3D-SIFT scheme was proposed in [38]. This method was then applied to object recognition in CT volumes by [10]. In [7], two scale and rotation invariant local descriptors in 3D meshes are presented: 1) the Scale Invariant Spin Images, a scale invariant extension of original Spin Images [20]; 2) the Local Depth SIFT, the extension of local SIFT descriptor [26] in 3D meshes.

2.4 Bag of Words Approaches

The Bag of Words scheme, which is based on local descriptors, is used to retrieve images [43] or meshes [5, 9] from a large database of models. In a dataset of images or meshes, the BoW method first defines a feature dictionary from local descriptors and then quantizes the features to dictionary words. Afterwards, each data model is described by the histogram of its local features while each feature is represented by the dictionary word closest to it. When a query model is processed, its feature histogram is compared with all model histograms in the database to find similar models as the result [7].

In [32], SIFT descriptors are extracted from a set of depth images captured from different views of 3D objects. These descriptors are then quantized to prepare the dictionary words and the BoW approach is used for shape retrieval. Furuya and Ohbuchi [11] extended this approach to extract more local visual features using a dense depth image sampling. When spatial information is taken into account in the BoW approach, the result can be more accurate, so Bronstein et al. [5] propose to use the statistics of neighboring features

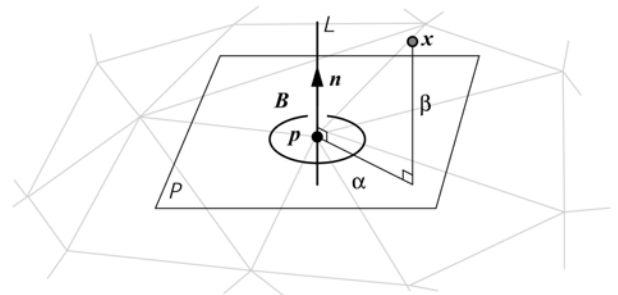


Fig. 1: Spin images: 3D point of a mesh with coordinate of p and normal n is shown in the figure. Two accumulators can be calculated given a 3D point: β , the radial distance to the surface normal line L and α , the axial distance above the tangent plane P [19].

as spatial information. A method which uses a visual dictionary and spatial knowledge is presented in [24]. After voxelizing the 3D surface, SURF features [2] are extracted from each voxel, used to build a visual vocabulary, and then a Hough Transform-like voting method is used to build the histogram for each set of words. An extension of this work is proposed in [23] to make the voting process invariant to rotation and scale. In [7], the BoW approach was simply used to illustrate the relevance of the proposed local features for high-level tasks such as mesh retrieval. This method yields robust 3D features compared to other state-of-the-art approaches which combine the SIFT and Spin image features at the same time. The authors present a structure to detect the interest points in 3D meshes and then compute their corresponding descriptors. As a descriptor, they first calculate a scale invariant formulation of spin images and then they adapt the classical SIFT feature extraction method to 3D data by considering the neighborhood of each interest point as a depth map and estimating its dominant angle using PCA to obtain rotation invariant features.

After extracting the features from the objects, either local features or global ones, the object recognition process can be carried out. Object recognition is mostly based on 2D images and video analysis. The most successful methods are based on statistical machine learning theory such as kernel methods. In order to use learning approaches, some proposed methods rely on a feature extraction process [1]. In particular, [31, 39] try to extract features from a set of 2D images taken from different views. Then, based on these features, the object is recognized without considering its 3D geometry [1].

Since most feature extraction methods fail to extract distinctive features from simple/basic models, we are proposing a method to extract features for such basic models. In addition, even when some descriptors can extract discriminative features from basic models, there is still a need for a learning method to achieve recognition. In other words, a classifier needs to be trained to learn the distinctive features. Our goal, here, is to present a description (recognition) strategy without using a classifier in order to avoid having trained such a classifier.

3 Proposed Method

As mentioned earlier, the basic geometric primitives being considered in this paper are planes, cones, cylinders, spheres, and tori, as well as partial instances of the latter four primitive types. The strategy for identifying

the type of a primitive consists of using surface normal information and the Gaussian sphere to first decide whether the primitive belongs to the plane-cone-cylinder class or to the sphere-torus class. The analysis of the Gaussian sphere and the introduction of the concept of the Gaussian accumulator then allows the correct type of the unknown primitive to be identified.

The work now for finding the type of an unknown geometric primitive from a 3D point cloud or a 3D mesh is a three-step process:

- STEP 1 Compute the normal vector at points in a point cloud or at vertices in a mesh of a primitive of an unknown type;
- STEP 2 Using surface normal information, build the unit Gaussian sphere and determine whether the primitive belongs to the plane-cone-cylinder class or the sphere-torus class. If the primitive belongs to the plane-cone-cylinder class, identify the type of primitive.
- STEP 3 When the primitive appears to belong to the sphere-torus class, the Gaussian accumulator, a concept introduced in this paper, is built and analyzed to find the primitive type.

Step two is useful because as shown in figure 2, normal vectors to the points of a plane map to a single point on the Gaussian sphere. Normal vectors to the points of a cylinder, map on a great circle of the Gaussian sphere while normal vectors to the points of a cone map on a small circle. The orientation of the circle depends on the orientation of the axis of the cylinder/cone.

It is thus clear that a proper analysis of the Gaussian sphere allows these types of primitives to be identified. However, as also shown in figure 2, normal vectors to the points on a sphere or on a torus distribute over the entire Gaussian sphere and it cannot be used to discriminate spheres from tori. So, step three is important to deal with these primitive types.

3.1 Computation of Surface Normals

The first step of the proposed approach consists in identifying the unit normal vector of the points of the surface of an unknown type primitive. Several methods have been proposed for computing normal vectors. For point cloud data, we use the method proposed in [52] while, for 3D meshes, we consider the normal of a vertex as the average of normals of its neighboring faces [35].

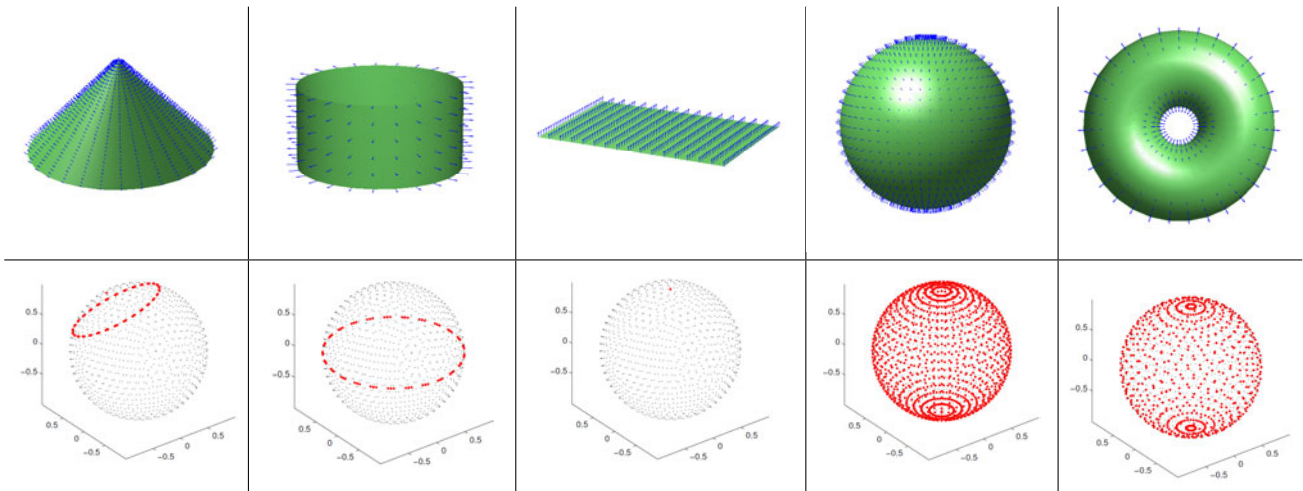


Fig. 2: 3D principal primitives with their normals on the model and on the Gaussian sphere.

3.2 Construction and analysis of the unit Gaussian sphere

Once the unit normal vector of each point at the surface of a primitive of unknown type has been found, it is easy to map its end point on the surface of the Gaussian sphere (examples of such mapping are shown in figure 2).

Principal Component Analysis (PCA) is used to find the plane passing through the points on the unit Gaussian sphere. The eigenvector corresponding to the smallest eigenvalue is the normal vector n to the PCA plane. Usually, in order to define a plane, we use a normal vector to the plane and a point on the plane. Having the normal of the PCA plane, we need a point to define the plane completely. The point can be considered as the mean of the points on the unit Gaussian sphere. However, using the mean causes a problem for cones with a cap.

As a matter of fact, as shown on the second column of the first line in figure 3, for a cone with a planar cap at its base, normal vectors near the base map on a different circle on the unit Gaussian sphere. This causes the mean value of the points to fall between this circle and the small circle corresponding to the normals of the rest of the cone. The final result is that the points on the Gaussian sphere may end up far from the PCA plane. This problem is easily solved if the median of the points on the Gaussian sphere is used instead of the mean, as illustrated on the third column of the first line in figure 3. For a cone without a cap, choosing the median does not have any adverse impact (second line in figure 3). In the rest of the paper, the term PCA plane refers to the plane passing through the median of the points on the Gaussian sphere.

Since the PCA plane is available, it is possible to analyze its behavior on the Gaussian sphere with respect to the different types of primitives of interest. Figure 4 summarizes observations made on 650 computer-generated primitive models set up by 3DsMax. The generated primitives cover the scale of man-made objects (from 1mm to 100mm). Table 1, lists the types of models as well as the number of each type (planes are not

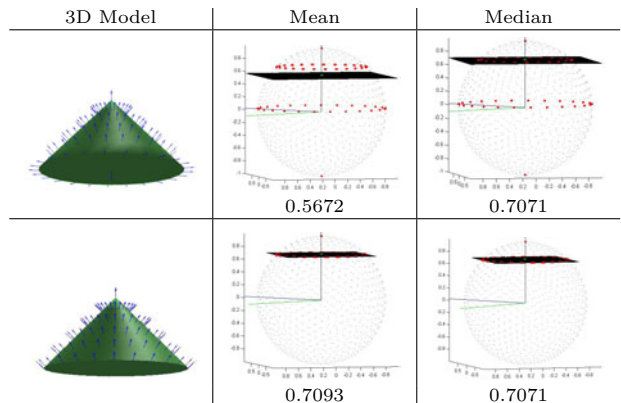


Fig. 3: Comparing the result of fitting a plane on the normals mapped on the Gaussian sphere using mean and median for a cone model with and without cap. The first row shows the result for a cone with a cap and the second row presents the result of a cone without a cap. The first column is the 3D model with the normals, the second and third columns show the normals on the Gaussian sphere with PCA fitted plane using the mean and the median, respectively. The values in the second and third columns represent the distance of the plane from the origin.

considered since they all map on a single point on the sphere).

For cylinders (full or partial, figure 4b), the points distribute as a great circle on the unit Gaussian sphere and so the PCA plane passes through the origin of the Gaussian sphere. For cones without a cap at the bottom (full or partial, figure 4c), the points distribute as a small circle on the unit Gaussian sphere and so the PCA plane is at a given distance from the origin of the Gaussian sphere depending on the aperture of the cone (the larger the aperture, the greater the distance). For cones with a cap at the bottom (full or partial, figure 4d), the points belonging to the conical part map on a small circle on the unit Gaussian sphere while points near the cap map on another circle near the origin.

For full spheres and full tori (figures 4e and 4f left) points spread almost everywhere on the Gaussian sphere and the PCA plane is any plane passing through the origin. For partial spheres and partial tori (figures 4e and 4f right), points distribute on the sphere depending on the shape but, again, the PCA plane is any plane passing through the points.

Based on the above observation and on experimental validation, it is possible to identify the type of primitives by analyzing the distribution of the normals on the Gaussian sphere in the vicinity of both sides of the PCA plane. First, each normal n_i on the Gaussian sphere is projected on the PCA plane normal vector n :

$$p_i = n_i \cdot n. \quad (1)$$

Considering all N normals mapped on the Gaussian sphere, the set $MAP(N)$ is obtained:

$$MAP(N) = \{p_i \mid i = 1, \dots, N\}. \quad (2)$$

Second, the following quantities are computed:

$$p_{MAX} = \max MAP(N), \quad (3)$$

Type of Primitive	# of models
Cylinder (full)	100
Cylinder (partial)	30
Cone without cap (full)	100
Cone without cap (partial)	30
Cone with cap (full)	100
Cone with cap (partial)	30
Sphere (full)	100
Sphere (partial)	30
Torus (full)	100
Torus (partial)	30

Table 1: 3D Models in GPrimDB [47]

$$p_{MIN} = \min MAP(N), \quad (4)$$

$$p = p_{MAX} - p_{MIN}, \quad (5)$$

$$\frac{p^2}{N} = Var(MAP(N)), \quad (6)$$

p_{MAX} and p_{MIN} are the maximum and minimum values of $MAP(N)$. p is the difference between the values while $\frac{p^2}{N}$ is the variance of $MAP(N)$. Let ϵ be a distance margin on both sides of the PCA plane, δ be a small positive value (i.e. $\delta < 1$), N_1 and N_2 be the number of points on the Gaussian sphere outside the distance margin on each side of the PCA plane respectively, and C a parameter. Figure 5 illustrates these parameters for a generic case of the Gaussian sphere.

If the points on the Gaussian sphere for which the normal distance to the PCA plane is included in the $[-\epsilon, +\epsilon]$ interval are removed, the situations presented in figure 6 will be observed for different types of primitives. How the value of ϵ is selected, will be discussed later (Algorithm 1).

Returning to figure 6: for a plane primitive (figure 6a), p and $\frac{p^2}{N}$ are small (even in the presence of noise for real scans) and the number of points on both sides of the PCA plane and outside the distance margin 2ϵ is small with respect to the total number of points N (i.e. $N_1 + N_2 < N$ with $\delta < 1$). This is explained by the fact that points on the Gaussian sphere map as a dense group near the true normal to the plane primitive (not the PCA plane).

For a full or partial cylinder (figure 6b) or a cone without a cap (full or partial, figure 6c), p and $\frac{p^2}{N}$ are also small since the points on the great circle (for cylinder) or on the small circle (for a cone without a cap) concentrate very close to the PCA plane and, consequently, $N_1 = N_2 = 0$. The distance d is the distance of the PCA plane from the origin of the Gaussian sphere. When these conditions are met, if distance d is zero, the primitive is a cylinder, and when $0 < d < 1$, then the primitive is a cone without a cap.

For a partial sphere, (figure 6d), or a partial torus (figure 6e), p is smaller than the threshold considered in algorithm 1, $\frac{p^2}{N}$ is large, N_1 and N_2 are not zero and, even for a large value of ϵ (compared to its value for full spheres or full tori which $\frac{N_1}{N_2}$ is large if $N_1 < N_2$ or $\frac{N_2}{N_1}$ is large if $N_2 < N_1$).

Finally, for a cone with a cap, (figure 6h), p is large, $\frac{p^2}{N}$ is large and either N_1 or N_2 is equal to zero. When $N_1 = 0$, N_2 is not small due to the points of the cap


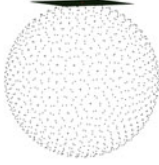

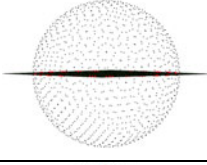
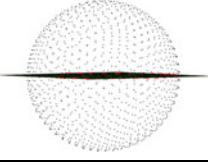

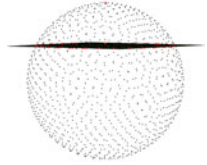
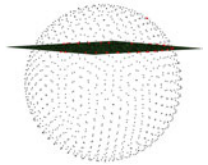

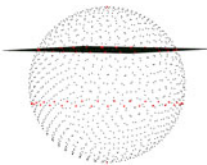
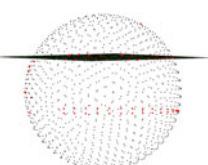

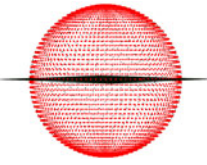
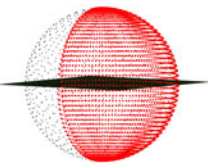
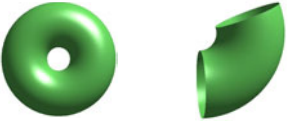
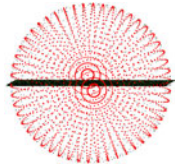

	Primitive Type	Primitive icon (full left, partial right)	Mapping on the Gaussian sphere (Full) (Partial)	
(a)	Plane			
(b)	Cylinder			
(c)	Cone without cap			
(d)	Cone with cap			
(e)	Sphere			
(f)	Torus			

Fig. 4: Typical unit Gaussian sphere for all types of primitives (full and partial) with the PCA plane passing through the median (black plane). For a plane, sphere, and torus the PCA plane is arbitrary since there is no privileged direction.

on the Gaussian sphere which map far from the PCA plane.

The value of C is used to determine whether an unknown primitive belongs to the plane-cylinder-cone class or to the sphere-torus class. When $C \leq t_1$, a given threshold, the primitive belongs to the plane-cylinder-cone (with or without cap) class and to the sphere-torus class otherwise.

If the primitive belongs to the plane-cylinder-cone class, it is a cylinder if the PCA plane is at distance $d = 0$ from the origin of the Gaussian sphere or it is a cone if $0 < d < 1$. In both cones, either N_1 or N_2 or

both are zero. When $d = 1$ and $N_1 + N_2$ is a small value or zero, it is a plane.

If the primitive belongs to the sphere-torus class, the Gaussian sphere does not contain enough information to allow the identification of the type of primitive, a situation that is clearly visible in figure 6d to 6g. The concept of Gaussian accumulator presented in the next section is introduced to achieve a necessary differentiation.

Algorithm 1 describes the steps for computing ϵ and C . Parameter ϵ is used to discard points on the Gaussian sphere near the PCA plane and the value of C is used to identify to which group the unknown primitive

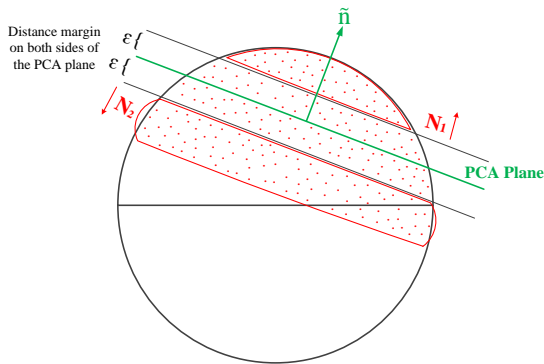


Fig. 5: Different parameters used for analyzing the Gaussian sphere.

belongs to. The formulas for computing ϵ and C are derived based on the observation of the 650 primitives generated with 3DsMax CAD software and on experiments run on scans of objects. Based on observation and experiments, the value of t_1 was empirically set to 0.1. This value is used for all experiments presented in this paper (see section 4).

3.3 Applying the Gaussian accumulator to the sphere-torus class

As demonstrated in the previous section, although the Gaussian sphere allows the unambiguous identification of planes, cylinders, cones, and partial instances of the latter two, it fails to identify spheres and tori since the surface normals of such primitives spread over its entire surface. However, careful observation of the distribution of surface normals on the Gaussian sphere (see figure 7) shows that this distribution is nevertheless different for spheres and tori. The top row of figure 7 shows a sphere and a torus with 4187 and 4176 vertices, respectively. The center and third row of figure 7 show side and top views of the distribution of the surface normals on the Gaussian sphere for both primitives. This distribution clearly differs for spheres and tori. Being able to measure this difference allows spheres to be differentiated from tori as well as partial instances of both.

In this work, it is proposed to use a spherical histogram called the Gaussian accumulator to analyze the distribution of surface normals for spheres and tori. The Gaussian accumulator is a unit sphere with $V = 1000$ vertices and tessellated into 1996 singular cells distributed evenly on the surface of the sphere (see figure 8 and 9 for examples of Gaussian accumulators with these properties). The choice of V will be discussed

Algorithm 1: The pseudo code to determine if the model is in the first category or in the second category.

Input: 3D model \mathbf{G} .

1. $\mathbf{N} \leftarrow$ Compute normals of the model \mathbf{G} .
2. Apply PCA on the normal points (\mathbf{N}).
3. $\tilde{n} \leftarrow$ Select the eigenvector related to the smallest eigenvalue as the normal of the plane.
4. $\tilde{P} \leftarrow$ Find the plane with the median of the normal points, \mathbf{N} , and the normal of the plane, \tilde{n} .
5. $\forall n_i \in \mathbf{N}$, map n_i on normal \tilde{n} :
 $MAP(i) = n_i \cdot \tilde{n}$.
6. **if** $\max\{MAP\} - \min\{MAP\} \leq 0.3$ **then**
if $Var(MAP) \leq 0.001$ **then**
 $\epsilon = 0.1$;
else
 $\epsilon = 0$;
end
else
 $\epsilon = \frac{\max MAP - \min MAP}{5}$;
end
7. Compute,
 $\forall n_i \in \mathbf{N}, ED(i) = \|n_i - Plane \tilde{P}\|_2$.
8. Remove all n_i from \mathbf{N} which $ED(i) \leq \epsilon$.
9. Select,
 $N_1 =$ the points of \mathbf{N} that are in the same direction of the \tilde{n} ,
 $N_2 =$ the points of \mathbf{N} that are in the opposite direction of the \tilde{n} ,
10. **if** $|N_1| = 0$ **or** $|N_2| = 0$ **then**
 $C = 0$;
else if $|N_1| + |N_2| \leq |N| \times 0.01$ **then**
 $C = 0$;
else if $|N_1| \leq |N_2|$ **then**
 $C = \frac{|N_1|}{|N_2|}$;
else
 $C = \frac{|N_2|}{|N_1|}$;
end

Output: C .

in more detail in section 4 concerning the experimental results. In the following the steps for calculating the Gaussian accumulator are presented:

- STEP 1 Prepare a unit sphere with $V = 1000$ vertices and tessellated into 1996 cells,
- STEP 2 For a given model, compute the surface normal at each vertex map the normals on the surface of the Gaussian accumulator,
- STEP 3 Using the k-d tree algorithm [3], find the corresponding cell and increment the count for this cell of the accumulator,
- STEP 4 Sort the voted Gaussian accumulator in order of decreasing count value (see figure 12 for examples of such accumulators for spheres and tori)

Primitive	Normals on the Gaussian sphere	Parameters
(a) Plane		δ_p small σ_p^2 small ϵ small, set to 0.1 $N_1 + N_2 < \alpha N$, $\alpha \ll 1$ $d = 1$ C set to 0
(b) Cylinder (full or partial)		δ_p small σ_p^2 small ϵ small, set to 0.1 $N_1 = 0$ $N_2 = 0$ $d = 0$ C set to 0
(c) Cone without cap (full or partial)		δ_p small σ_p^2 small ϵ small, set to 0.1 $N_1 = 0$ $N_2 = 0$ $0 < d < 1$ C set to 0
(d) Partial sphere		δ_p small σ_p^2 large ϵ small, set to 0 $N_1 \neq 0$ $N_2 \neq 0$ $d = 0$ $C = \frac{N_1}{N_2}$ if $N_1 < N_2$ or $\frac{N_2}{N_1}$ if $N_2 < N_1$
(e) Partial torus		δ_p small σ_p^2 large ϵ small, set to 0 $N_1 \neq 0$ $N_2 \neq 0$ $d = 0$ $C = \frac{N_1}{N_2}$ if $N_1 < N_2$ or $\frac{N_2}{N_1}$ if $N_2 < N_1$
(f) Full sphere		δ_p large σ_p^2 large ϵ large, set to $\frac{\delta_p}{5}$ $N_1 \neq 0$ $N_2 \neq 0$ $d = 0$ $C = \frac{N_1}{N_2}$ if $N_1 < N_2$ or $\frac{N_2}{N_1}$ if $N_2 < N_1$
(g) Full torus		δ_p large σ_p^2 large ϵ large, set to $\frac{\delta_p}{5}$ $N_1 \neq 0$ $N_2 \neq 0$ $d = 0$ $C = \frac{N_1}{N_2}$ if $N_1 < N_2$ or $\frac{N_2}{N_1}$ if $N_2 < N_1$
(h) Cone with cap (full or partial)		δ_p large σ_p^2 large ϵ large, set to $\frac{\delta_p}{5}$ $N_1 \neq 0$ if $N_2 = 0$ $N_2 \neq 0$ if $N_1 = 0$ $0 < d < 1$ C set to 0

Fig. 6: Illustration of normals distribution on the Gaussian sphere with respect to the PCA plane and parameters associated with each situation.

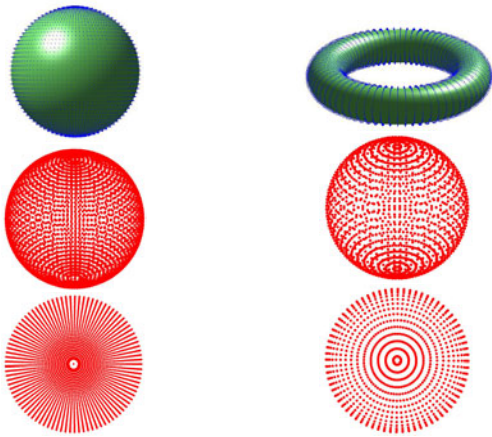


Fig. 7: The dispersion of normals on the Gaussian sphere for spheres and tori. The first row shows the 3D models, the second row and the third row represent the side view and top view of the 3D models normals, respectively.

STEP 5 Calculate the statistical moments (mean, variance, skewness, and Kurtosis) of the sorted Gaussian accumulator.

Before explaining how statistical moments can be used to discriminate spheres from tori, it is important to address an essential issue that arises in building the Gaussian accumulator. Depending on whether one wishes to differentiate between spheres and tori resulting from CAD models or real scans, the distribution of vertices on the model must be taken into account. As illustrated in figure 8, the distribution of vertices on a CAD model or a model obtained by decimation of a finer CAD model or a real scan is very different. For pure CAD models (built with 3DsMax for instance) of spheres and tori, one clearly observes poles on the triangulations of their vertices. At these poles, the concentration of vertices is much higher than elsewhere on the primitive (the equator for instance). This uneven distribution of vertices causes an uneven distribution of surface normals and, ultimately, leads to a biased Gaussian accumulator.

One way to circumvent this problem for CAD models consists in generating a sphere or a torus, for instance, with four times the desired number of vertices and then to decimate the model to the desired number of vertices using decimation algorithms such as the one available in Meshlab. The second and fourth columns in figure 8 show the result of the decimation of CAD models of a sphere and a torus with 6080 and 5704 triangles, respectively. The Gaussian accumulator for these models clearly shows that this processing eliminates the poles and yields suitable Gaussian accumulators.

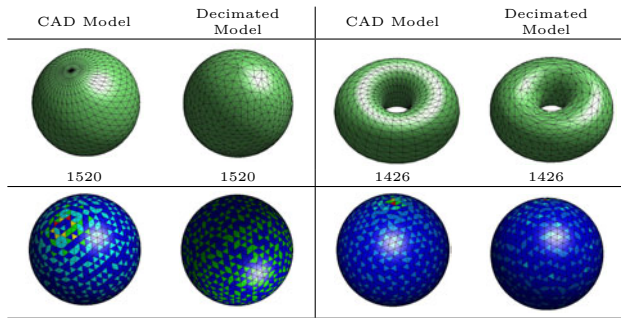


Fig. 8: Triangulations of the CAD models and their decimated models with their results of voting on the Gaussian accumulator. The first row shows the triangulation of the CAD models and their decimated ones using MeshLab. The result of voting on the Gaussian accumulator is shown in the third row. The values under the 3D models, in the second row, present the number of triangles for each model.

This process works well for CAD models but it is however impractical for real scans since nothing prevents the user from accumulating points more densely in specific areas of the scanned objects. A better and more general approach that is proposed in this paper rather consists in implementing a Gaussian accumulator cell incrementation procedure that takes into account the density of triangles in the neighborhood of each surface normal (i.e. at each vertex of the triangular mesh). This incrementation method, which is called

Density-Weighted-Voting, works as follows: The first-ring set of neighboring faces is found for each vertex i of the mesh. This set is composed of all triangles having i as a vertex and is called N_{1i} . The density S_i at vertex i is defined as:

$$S_i = \frac{\sum_{j \in N_{1i}} area_j}{N_{1i}}, \quad (7)$$

where, N_{1i} is the size of N_{1i} and $area_j$ is the area of the j^{th} triangle in the set. In order to find the area of a triangle, we simply identify two adjacent side vectors of the triangle and then the area is half the cross product norm of these two adjacent vectors [28]:

$$area_j = \frac{1}{2} \|u_j \times v_j\|, \quad (8)$$

where, u_j and v_j are the two adjacent side vectors of triangle j . The surface normal at i is mapped on the Gaussian accumulator and the cell corresponding to n_i is incremented by S_i (instead of 1). This incrementation strategy prevents a cell from being incremented several times because of a large concentration of triangles on the mesh. Figure 9 shows the results achieved

by Density-Weighted-Voting compared to regular voting. The last column indicates that the Gaussian accumulator built with Density-Weighted-Voting is more uniform for spheres and tori (see the color code at the rightmost side of the figure).

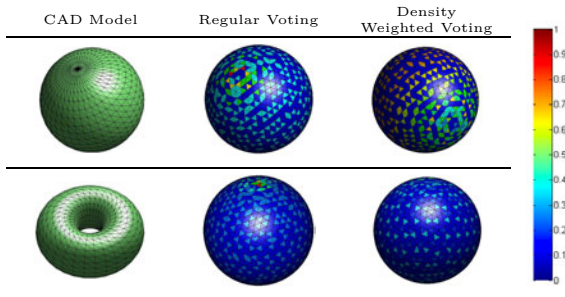


Fig. 9: The difference between regular voting of the Gaussian accumulator in comparison with our proposed Density Weighted Voting approach for the CAD models with poles.

Once the Gaussian accumulator is available, its cells are first sorted in order of decreasing count value and statistical moments are computed on the sorted histogram. In this work, variance, skewness, and Kurtosis are computed and analyzed in order to check whether or not they can be used to discriminate between spheres and tori. The Gaussian accumulator for the CAD models of spheres and tori (see table 1 for the number of models in each category) were built and the three above statistical moments were computed. It is important to mention that the CAD models for spheres and tori in the database span a wide spectrum of sizes and scales of man-made objects. In addition, both sparse and dense models with respect to the number of vertices are included in the database. For the experiments made in this work, the Gaussian accumulator was composed of 1000 vertices and 1996 cells.

Figure 10 shows plots of different combinations of statistical moments computed on the Gaussian accumulator of the spheres and tori in the database. Figure 10a plots the skewness of the Gaussian accumulator over its variance, figure 10b plots the Kurtosis over the variance and figure 10c plots the Kurtosis over the skewness. Finally, figure 10d plots the skewness over the variance scaled by the Kurtosis value of the Gaussian accumulator for each model in the database. Figure 10a and 10b suggest that the skewness and Kurtosis seem potentially relevant to separate spheres from tori. Focusing on the skewness, figure 11, which plots the value of this moment for all sphere/torus models in the database, reveals that this moment alone can be used to discrimi-

nate these two types of primitives by mere thresholding:

$$\begin{aligned} \text{skewness} < \tau &\rightarrow \text{sphere}, \\ \text{skewness} > \tau &\rightarrow \text{torus}. \end{aligned} \quad (9)$$

(with $\tau = 2$)

Careful observation of the area near the threshold value in figure 11 also shows that a very small number of tori (1) have a skewness value smaller than the defined τ and could thus be misclassified as spheres. However these models are tori with very few vertices and do not show many geometric details. Such instances of tori are not likely to be found in real-world applications. As pointed out above, simple thresholding of the skewness value suffices to differentiate spheres from tori. However, should one want to train a classifier, this would be possible since as spheres in figure 12, the sorted histogram of the Gaussian accumulator of spheres and tori have enough richness to be used as feature vectors.

4 Experimental Results

This section presents the results obtained by the method presented in this paper for recognizing different primitive types. The method was tested on CAD models of geometric primitives, 3D models of primitives scanned with a 3D sensor and, finally, on primitives extracted from complex models of scanned objects using the segmentation approach described in [48]. The results show that the proposed method is successful in identifying the types of primitives of interest (plane, cylinders, cones, spheres, and tori).

For all experiments reported in this paper, the Gaussian accumulator is composed of 1000 vertices generating 1996 triangular cells which creates a uniform tessellation at the surface of the sphere. The Particle Sample Sphere method described in [40] was used for tessellating the sphere.

4.1 Experimental results on the CAD models

As mentioned in figure 12, one may want to use the proposed accumulator as a descriptor. For this purpose the results of the proposed method applied on the GPrimDB database of CAD models [47] are compared to those based on improved SIFT and Spin-Images introduced in [7], the Laplacian spectra method [53] proposed for CAD models, the Spherical Harmonic Descriptor (SHD) introduced in [21] and D2 descriptor presented in [33]. The GPrimDB database contains 520 CAD models: 100 cones, 100 cylinders, 100 spheres, 100

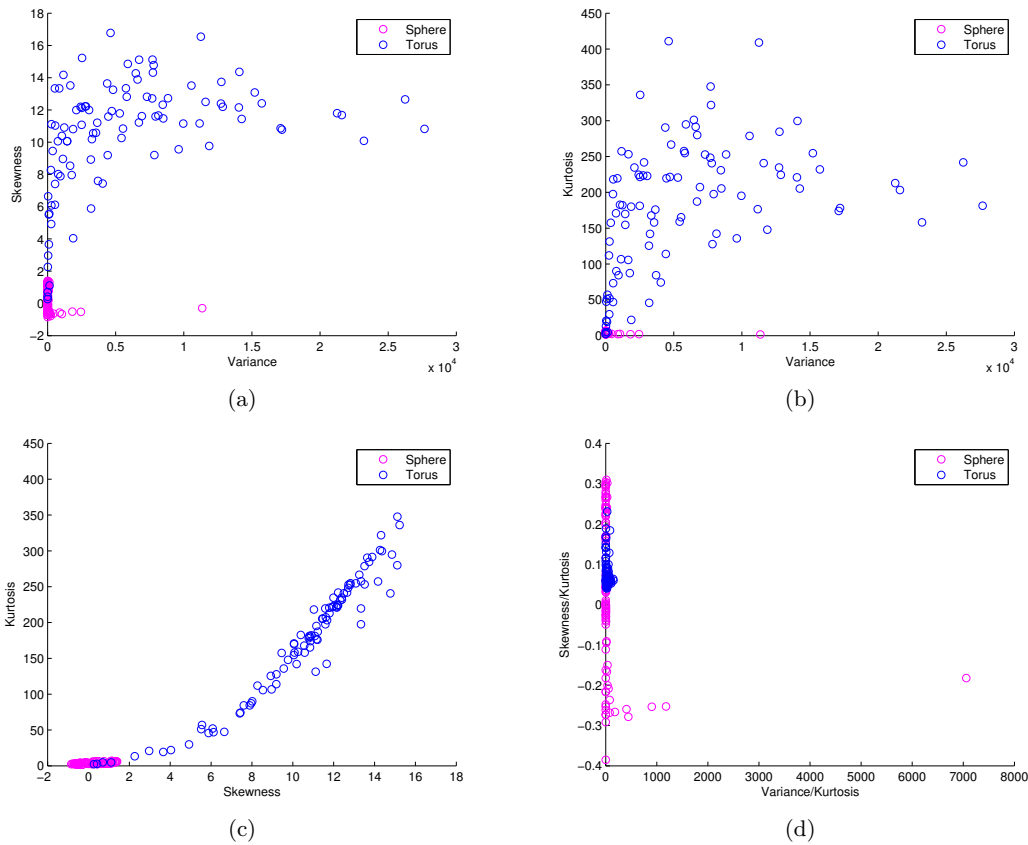


Fig. 10: The different plots of the Gaussian accumulator's statistical moment. Part (a), shows the Skewness of the Gaussian accumulator over its variance. The Kurtosis over the variance and the Kurtosis over the Skewness are shown in parts (b) and (c), respectively. Part (d) presents the plot of Skewness over the variance proportional to the Kurtosis value.

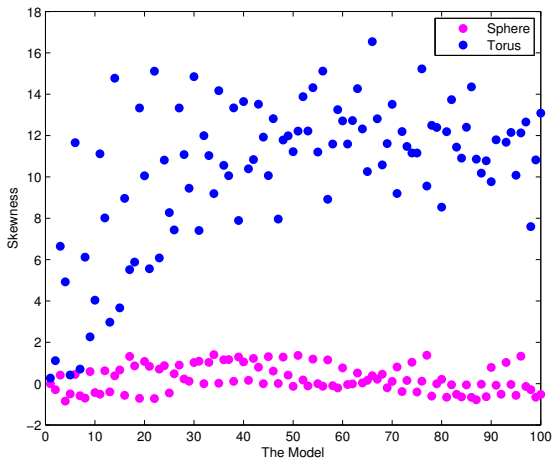


Fig. 11: The Skewness value of the Gaussian accumulator for 100 spheres and 100 tori.

tori, 30 partial cones, 30 partial cylinders, 30 partial spheres and, 30 partial tori. The SIFT/Spin-based are used with 15 words in the Bag of Words while the D2 descriptor uses a 1024-bin histogram. Figure 13 shows the results achieved by the 5 approaches listed above. These results show that SIFT, Spin-Images, Laplacian spectra, and SHD approaches do not have enough richness to differentiate between the different types of primitives. Only the D2 descriptor seems to have enough discriminative power to identify primitives while a classifier needs to be trained to separate the primitives. However, the D2 descriptor lacks the discriminative power in the presence of partial primitives as shown in figure 14. For instance, depending on the model, partial cylinders have D2 descriptors that are similar to tori, some other are similar to cones and the rest are similar to complete cylinders. Partial spheres of some models have D2 descriptors that are very similar to cones while some partial tori are similar to cylinders.

The approach presented in this paper can deal successfully with complete and partial models of each type

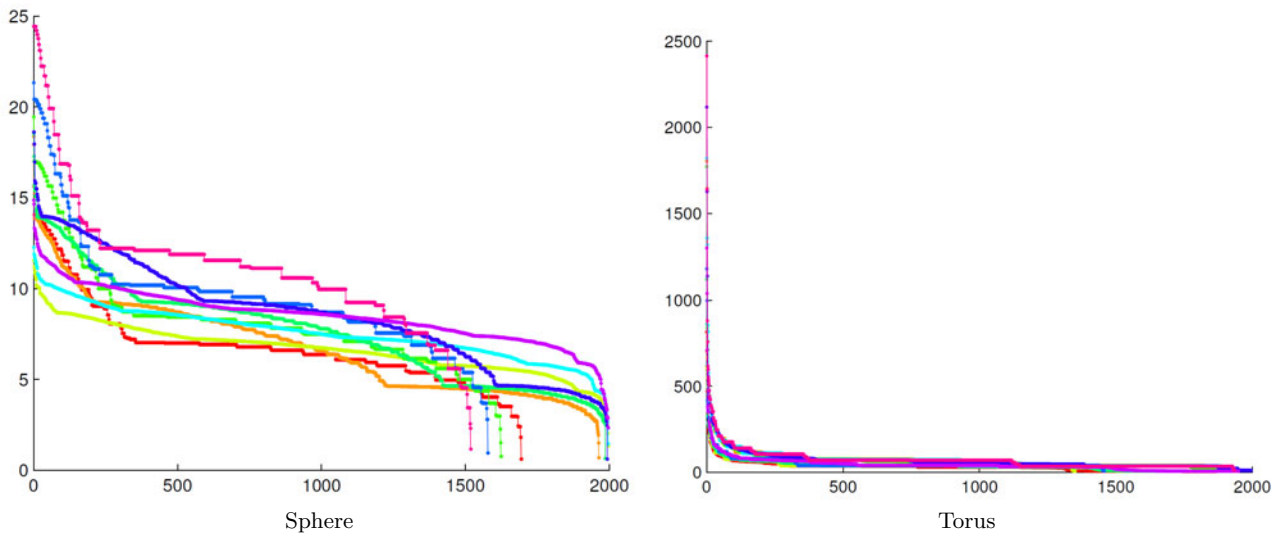


Fig. 12: The sorted histogram of Gaussian accumulator values for 10 spheres and 10 tori. The histograms emphasize the similarity between spheres and between tori.

of primitive. As an example, the normals of a partial cone are making a small circle on the Gaussian sphere while, for a partial cylinder, the normals make a great circle which is similar to complete cones and cylinders. So, the method is able to discriminate between partial cones and partial cylinders. Figure 15 shows the result of the PCA plane fitting on the normals of partial cylinders and cones. The results indicate that using the distance of the plane from the origin, we can discriminate partial cylinders from partial cones as well as their complete models. In the figure, there are both large and small slices of a cylinder and a cone. For the cone with cap, because of large numbers of points on the faces between the model and the cap, the calculation of the normal vector using PCA changes a little but we are still able to determine if it is a cone or a cylinder using the distance of the plane from the origin while for the cone without a cap, the calculation of the plane's normal is accurate.

As discussed in the previous section 3.1, a value must be computed for parameter C for some types of primitives (see Algorithm 1). Table 2 lists the max, min, and mean value of parameter C for the different types of models in the database. From the table, the value of the threshold t_1 for separating primitives in the sphere-torus category from the other types of primitives was set to 0.1.

The resolution of the Gaussian accumulator for separating spheres from tori was investigated. Figure 16 shows the plots of the skewness for each sphere and torus primitive in the database, full or partial, with different resolutions (i.e. numbers of vertices and numbers of triangular cells) for the Gaussian accumulator.

Based on experiments conducted with different parameter values, a sphere with 1000 vertices and 1996 triangular cells was chosen empirically. The figure also shows that for these parameter values, a value of 2 for the threshold t_2 in Algorithm 1 can discriminate between spheres and tori. It can be observed from the figure that the number of vertices or, more generally, the number of triangular cells on the Gaussian accumulator does not have a significant impact on the recognition result. All plots present a similar threshold for discriminating spheres from tori, full or partial.

3D Model	Min	Max	Mean
Sphere	0.5714	1	0.9461
Partial Sphere	0.2821	1	0.8014
Torus	0.8696	1	0.9945
Partial Torus	0.6774	1	0.9756
Cylinder	0	0	0
Partial Cylinder	0	0	0
Cone without Cap	0	0	0
Partial Cone without Cap	0	0	0
Cone with Cap	0	0.0625	0.0081
Partial Cone with Cap	0	0.0217	0.0018

Table 2: Min, max, and mean value of C value introduced in Algorithm 1.

4.2 Real Scanned Models

This section presents the results of our method for real scanned models and complex models composed of several different primitive types. The first row of figure 17 shows a complex CAD model which is segmented into its primitives using the 3D-NCuts segmentation approach [48] with the primitives identified by the proposed approach. The second row shows the results of segmentation for a mug model using the 3D-NCuts method and the result of primitive identification using our approach. For the mug model, since we wanted to show the results of primitive identification for the cylin-

der without a cap and torus (handle of the mug), we did not consider the bottom part of the mug. The C value is given for each primitive and, for the primitives in the first category (plane-cone-cylinder), the distance from the origin is presented as the d value. The primitives in the second category (sphere-torus) can be distinguished using the skewness which is presented as the S value.

In order to test our approach on real scanned models, we used the Creaform Go!SCAN handheld 3D scanner to scan a sphere, a torus, two cones, a plane, a cylinder, a partial sphere, and two partial tori. The results are shown in figure 18. The first column shows the 3D scanned models. The second column shows the num-

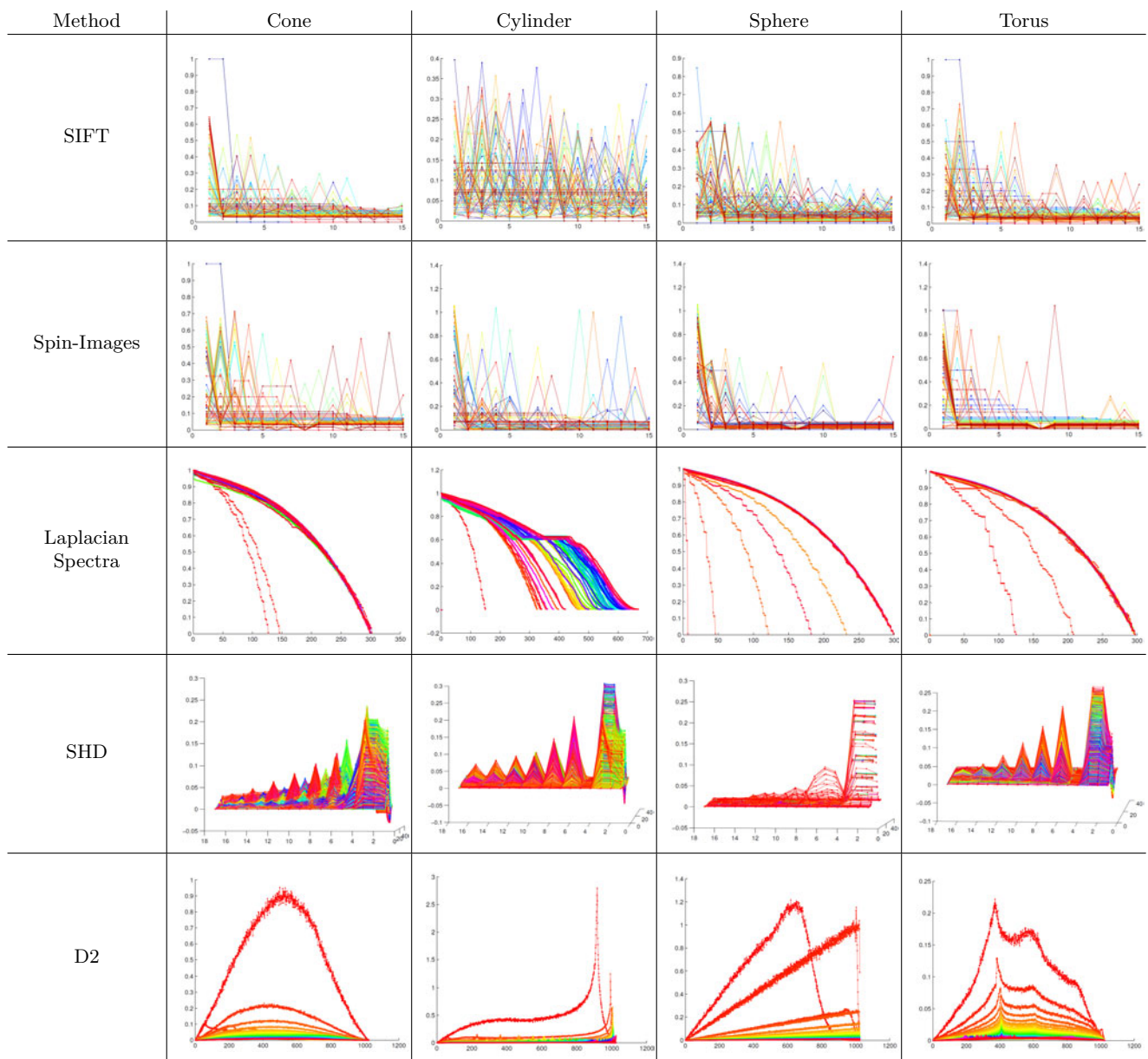


Fig. 13: Comparison between feature extraction methods for our principal primitives.

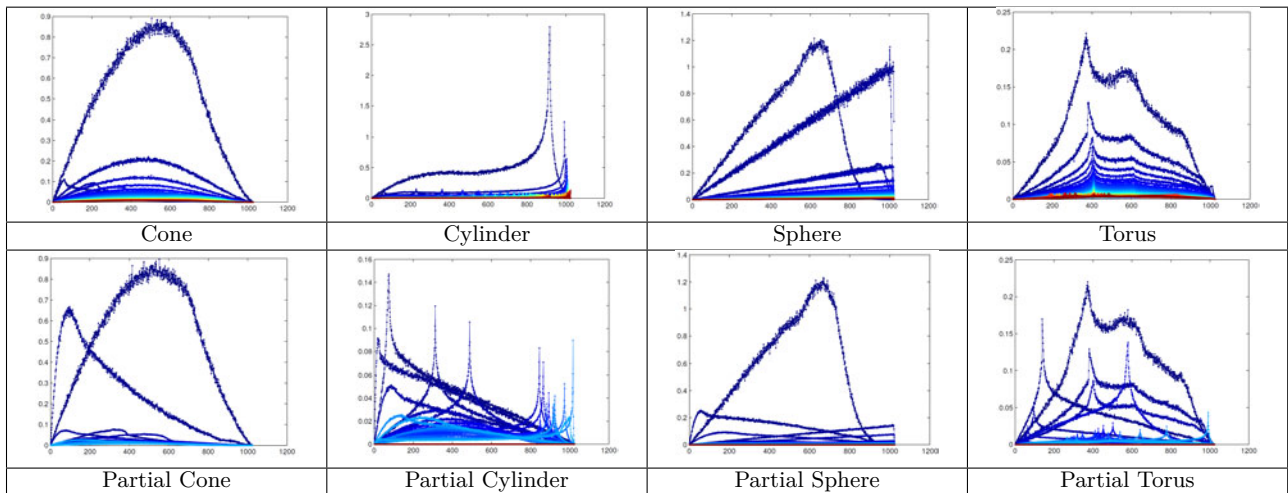


Fig. 14: D2 descriptor for the principal primitives and their partial models.

ber of vertices, the third column presents the C value and, finally the last column shows either the d value for the distance of the PCA plane from the origin or the S value for the skewness value. The C values greater than $t_1 = 0.1$, correspond to the models in the first category (i.e. the first four models). In the first category, if d is close to one then the model is a plane (first row), if it is close to zero then the model is cylinder (second row), otherwise it is a cone (third and fourth rows). In the second category, if S is greater than $t_2 = 2$ then the model is a torus (last three rows), otherwise it is a sphere (fifth and sixth rows). The last row shows a partial torus with missing data but using the C and S values, mentioned above, the partial torus is identified correctly to be a torus.

5 Conclusion

In this paper, we proposed a method for recognizing simple geometric primitives. At first, we identify the normals of the model, map them all on a Gaussian sphere and then we fit a plane using PCA. Based on the proposed algorithm, when a parameter C is smaller than threshold t_1 , the model is in the first category (plane-cone-cylinders); otherwise, it is in the second category (sphere-torus). For the first category we determine the distance of the PCA plane from the origin and for the second category we calculate the skewness value of the Gaussian accumulator. Comparing our method with approaches based on descriptors, our method does not require any classifier and is also able to recognize partial models, real noisy models as well as objects with missing data.

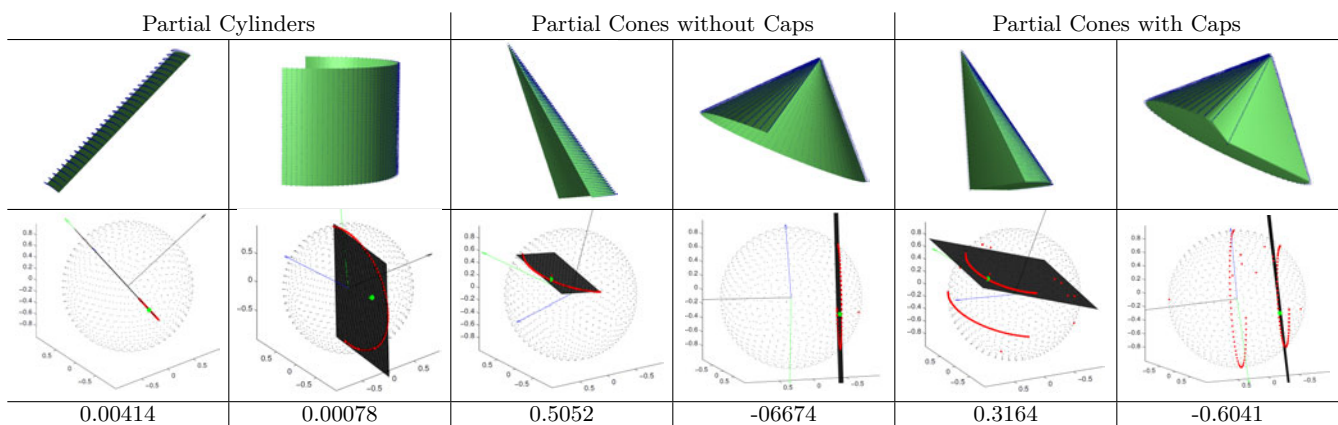


Fig. 15: Partial cylinders and cones on the Gaussian sphere with the distance of the PCA plane from the origin which is shown in the last row.

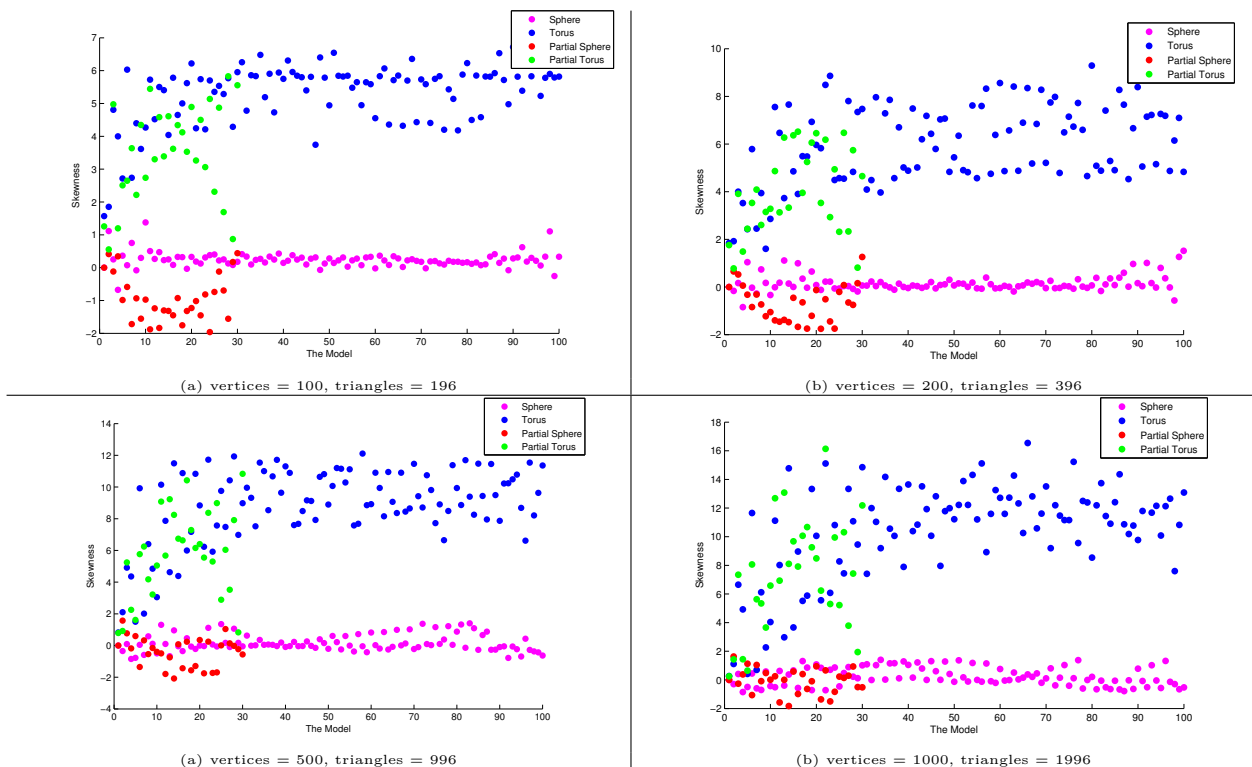


Fig. 16: The result of the Gaussian accumulator for partial spheres and partial tori as well as complete spheres and tori with different number of cells. The values in each part show the number of vertices and triangles (cells) for the accumulator.

Acknowledgements This work was supported by the NSERC-Creaform Industrial Research Chair on 3D Sensing. Z. Toony was supported by a FRQNT post-graduate scholarship.

References

1. Baareh, A.K., Sheta, A.F., Al-Batah, M.S.: Feature based 3D object recognition using artificial neural networks. *Int. Journal of Computer Applications* **44** (2012)
2. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: 9th European Conf. on Computer Vision (ECCV), pp. 404–417. Springer (2006)
3. Bentley, J.L.: Multidimensional binary search trees used for associative searching. *Communications of the ACM* **18**(9), 509–517 (1975)
4. Besl, P.J., McKay, N.D.: Method for registration of 3-D shapes. *Robotics-DL tentative* pp. 586–606 (1992)
5. Bronstein, A.M., Bronstein, M.M., Guibas, L.J., Ovsjanikov, M.: Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. on Graphics (TOG)* **30**(1), 1 (2011)
6. Castellani, U., Cristani, M., Fantoni, S., Murino, V.: Sparse points matching by combining 3d mesh saliency with statistical descriptors. *Computer Graphics Forum* **27**(2), 643–652 (2008)
7. Darom, T., Keller, Y.: Scale-invariant features for 3-d mesh models. *IEEE Trans. on Image Processing* **21**(5), 2758–2769 (2012)
8. Dubrovina, A., Kimmel, R.: Matching shapes by eigendecomposition of the Laplace-Beltrami operator. *Proc. of 3D Data Processing, Visualization and Transmission (3DPVT)* **2**(3) (2010)
9. Fehr, J., Streicher, A., Burkhardt, H.: A bag of features approach for 3D shape retrieval. In: *Advances in Visual Computing*, pp. 34–43. Springer (2009)
10. Flitton, G.T., Breckon, T.P., Bouallagu, N.M.: Object recognition using 3D SIFT in complex CT volumes. *Proc. of the British Machine Vision Conf. (BMVC)* pp. 1–12 (2010)
11. Furuya, T., Ohbuchi, R.: Dense sampling and fast encoding for 3D model retrieval using bag-of-visual features. *Proc. of the ACM Int. Conf. on Image and Video Retrieval* p. 26 (2009)
12. Gebal, K., Bærentzen, J.A., Aanæs, H., Larsen, R.: Shape analysis using the auto diffusion function. *Computer Graphics Forum* **28**(5), 1405–1413 (2009)
13. Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J.: 3d object recognition in cluttered scenes with local surface features: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36**(11), 2270–2287 (2014)
14. Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., Kwok, N.M.: A comprehensive performance evaluation of 3d local feature descriptors. *International Journal of Computer Vision* pp. 1–24 (2015)
15. Guo, Y., Sohel, F., Bennamoun, M., Lu, M., Wan, J.: Rotational projection statistics for 3d local surface description and object recognition. *International journal of computer vision* **105**(1), 63–86 (2013)
16. Harris, C., Stephens, M.: A combined corner and edge detector. *Alvey Vision Conf.* **15**, 50 (1988)
17. Hetzel, G., Leibe, B., Levi, P., Schiele, B.: 3d object recognition from range images using local feature histograms.


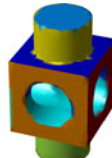
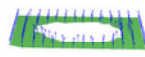


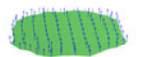





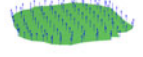



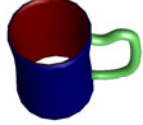
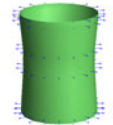
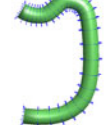
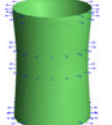
3D model	3D-NCuts segmentation result	Primitives with their values			
		 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane
		 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane	 $C = 0, d = 0.0013$ Cylinder	 $C = 0, d = 3.7 \times 10^{-4}$ Cylinder
		 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane	 $C = 0, d = 1$ Plane
		 $C = 0, d = 3 \times 10^{-14}$ Cylinder	 $C = 0.8454, S = 10.226$ Torus	 $C = 0, d = 2.8 \times 10^{-14}$ Cylinder	

Fig. 17: Primitive detection using our approach for the segments of two complex models.

- Proc. of IEEE Computer Society Conf. on Computer Vision and Pattern Recognition(CVPR) **2**, II 394 (2001)
18. Hoppe, H.: Progressive meshes. Proc. of the 23rd Annual Conf. on Computer Graphics and Interactive Techniques pp. 99 108 (1996)
 19. Johnson, A.E.: Spin-images: a representation for 3-D surface matching. Ph.D. thesis, Citeseer (1997)
 20. Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3d scenes. IEEE Trans. on Pattern Analysis and Machine Intelligence **21**(5), 433 449 (1999)
 21. Kazhdan, M., Funkhouser, T., Rusinkiewicz, S.: Rotation invariant spherical harmonic representation of 3D shape descriptors. Proc. of Eurographics/ACM SIGGRAPH Sym. on Geometry processing pp. 156 164 (2003)
 22. Kin-Chung Au, O., Tai, C.L., Cohen-Or, D., Zheng, Y., Fu, H.: Electors voting for fast automatic shape correspondence. Computer Graphics Forum **29**(2), 645 654 (2010)
 23. Knopp, J., Prasad, M., Van Gool, L.: Orientation invariant 3D object classification using hough transform based methods. Proc. of the ACM Workshop on 3D Object Retrieval pp. 15 20 (2010)
 24. Knopp, J., Prasad, M., Willems, G., Timofte, R., Van Gool, L.: Hough transform and 3D SURF for robust three dimensional classification. In: 11th European Conf. on Computer Vision (ECCV), pp. 589 602. Springer (2010)
 25. Litman, R., Bronstein, A.M., Bronstein, M.M.: Disunion-geometric maximally stable component detection in deformable shapes. Computers & Graphics **35**(3), 549 560 (2011)
 26. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. Journal of Computer Vision **60**(2), 91 110 (2004)
 27. Maes, C., Fabry, T., Keustermans, J., Smeets, D., Suetens, P., Vandermeulen, D.: Feature detection on 3D face surfaces for pose normalisation and recognition. 4th IEEE Int. Conf. on Biometrics: Theory Applications and Systems (BTAS) pp. 1 6 (2010)
 28. Mansfield, M., O'Sullivan, C.: Understanding physics. John Wiley & Sons (2012)
 29. Mian, A., Bennamoun, M., Owens, R.: On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. Int. Journal of Computer Vision **89**(2-3), 348 361 (2010)
 30. Mohamad, M.: 3D object recognition using local shape descriptors. Technical Report No. 2013-614 (2013)
 31. Murase, H., Nayar, S.K.: Visual learning and recognition of 3-D objects from appearance. Int. Journal of Computer Vision **14**(1), 5 24 (1995)
 32. Ohbuchi, R., Osada, K., Furuya, T., Banno, T.: Salient local visual features for shape-based 3D model retrieval. IEEE Int. Conf. on Shape Modeling and Applications pp. 93 102 (2008)
 33. Osada, R., Funkhouser, T., Chazelle, B., Dobkin, D.: Shape distributions. ACM Transactions on Graphics (TOG) **21**(4), 807 832 (2002)
 34. Paquet, E., Rioux, M., Murching, A., Naveen, T., Tabatabai, A.: Description of shape information for 2-D and 3-D objects. Signal Processing: Image Communication **16**(1), 103 122 (2000)

35. Peyré, G.: Toolbox Graph. <http://www.mathworks.com/matlabcentral/fileexchange/5355-toolbox-graph/> (2004). [Online; accessed June-2004]
36. Rabbani, T., Van Den Heuvel, F.: Efficient Hough transform for automatic detection of cylinders in point clouds. *Proc. of ISPRS Workshop on Laser Scanning* **3**, 60–65 (2005)
37. Ruggeri, M.R., Patanè, G., Spagnuolo, M., Saupe, D.: Spectral-driven isometry-invariant matching of 3d shapes. *Int. Journal of Computer Vision* **89**(2-3), 248–265 (2010)
38. Scovanner, P., Ali, S., Shah, M.: A 3-dimensional sift descriptor and its application to action recognition. *Proc. of the 15th Int. Conf. on Multimedia* pp. 357–360 (2007)
39. Selinger, A., Nelson, R.C.: A perceptual grouping hierarchy for appearance-based 3d object recognition. *Computer Vision and Image Understanding* **76**(1), 83–92 (1999)
40. Semechko, A.: Particle sample sphere code for uniform tessellation of a unit sphere. <http://www.mathworks.com/matlabcentral/fileexchange/37004-uniform-sampling-of-a-sphere> (2012). [Online; accessed June-2012]
41. Shang, L., Greenspan, M.: Real-time object recognition in sparse range images using error surface embedding. *Int. Journal of Computer Vision* **89**(2-3), 211–228 (2010)
42. Sipiran, I., Bustos, B.: A robust 3d interest points detector based on harris operator. *Eurographics Workshop on 3D Object Retrieval* **1**, 7–14 (2010)
43. Sivic, J., Zisserman, A.: Video google: Efficient visual search of videos. In: *Toward Category-Level Object Recognition*, pp. 127–144. Springer (2006)
44. Sun, J., Ovsjanikov, M., Guibas, L.: A concise and provably informative multi-scale signature based on heat diffusion. *Computer Graphics Forum* **28**(5), 1383–1392 (2009)
45. Teichman, A., Levinson, J., Thrun, S.: Towards 3D object recognition via classification of arbitrary object tracks. *IEEE Int. Conf. on Robotics and Automation (ICRA)* pp. 4034–4041 (2011)
46. Tombari, F., Salti, S., Di Stefano, L.: Performance evaluation of 3d keypoint detectors. *International Journal of Computer Vision* **102**(1-3), 198–220 (2013)
47. Toony, Z., Laurendeau, D., Gagné, C.: PGP2X: Principal geometric primitives parameters extraction. to appear in *Proc. of the 10th Int. Conf. on Computer Graphics Theory and Applications (GRAPP)* (2015)
48. Toony, Z., Laurendeau, D., Giguère, P., Gagné, C.: 3D-NCuts: Adapting normalized cuts to 3D triangulated surface segmentation. *Proc. of the 9th Int. Conf. on Computer Graphics Theory and Applications (GRAPP)* (2014)
49. Wahl, E., Hillenbrand, U., Hirzinger, G.: Surface-pair-relation histograms: a statistical 3D-shape representation for rapid classification. *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on* pp. 474–481 (2003)
50. Zaharescu, A., Boyer, E., Varanasi, K., Horaud, R.: Surface feature detection and description with applications to mesh matching. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* pp. 373–380 (2009)
51. Zhang, C., Chen, T.: Efficient feature extraction for 2D/3D objects in mesh representation. *Proc. of Int. Conf. on Image Processing* **3**, 935–938 (2001)
52. Zhang, J., Cao, J., Liu, X., Wang, J., Liu, J., Shi, X.: Point cloud normal estimation via low-rank subspace clustering. *Computers & Graphics* **37**(6), 697–706 (2013)
53. Zhu, K., Wong, Y.S., Loh, H.T., Lu, W.F.: 3D CAD model retrieval with perturbed laplacian spectra. *Computers in Industry* **63**(1), 1–11 (2012)

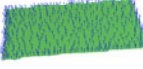
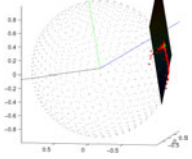

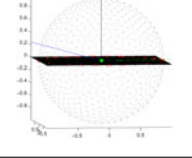
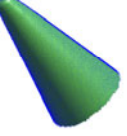
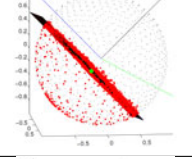

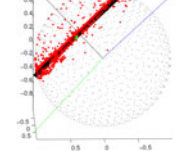
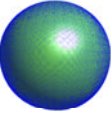
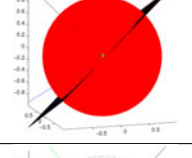
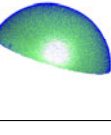
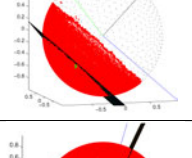
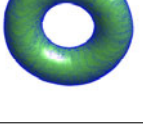
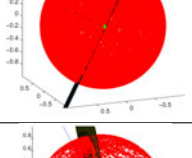

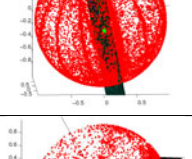

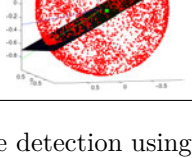
3D scanned model	Normals on the Gaussian Sphere	Number of Vertices	C Value	d or Skewness Value
		530	$C = 0$	$d = -0.9737$
		868	$C = 0$	$d = 0$
		11213	$C = 0$	$d = -0.2349$
		10238	$C = 0$	$d = 0.5011$
		96368	$C = 0.9794$	$S = -0.7569$
		13306	$C = 0.3633$	$S = -0.6538$
		96221	$C = 0.9746$	$S = 6.8488$
		21357	$C = 0.9033$	$S = 2.4912$
		6095	$C = 0.9207$	$S = 7.9608$

Fig. 18: Primitive detection using our approach for real scanned models.