Recovering Primitives in 3D CAD meshes

Roseline Bénière[a,c], Gérard Subsol[a], Gilles Gesquière[b], François Le Breton[c] and William Puech[a]

[a]LIRMM, Univ. Montpellier 2, CNRS, 161 rue Ada, 34392, France;
[b]LSIS, Univ. Aix-Marseille, CNRS, IUT, BP 90178, 13637 Arles Cedex, France;
[c]C4W, 219 rue Le Titien 34000 Montpellier, France

ABSTRACT

In an industrial context, recovering a continuous model is necessary to make modifications or to exchange data with a format including continuous representation of objects like STEP. But for many reasons, the initial continuous object can be lost after a display or an exchange with a discretized format. The mesh can also be deformed after a numerical computation. It is then important to have a method to create a new continuous model of the object from a mesh. In case of CAD object, the first step is to detect simple primitives like: plane, sphere, cone and cylinder from a 3D CAD mesh. This paper is focused on this step. This method of detection use curvature features to recover each primitive type. Segmentation is based on the curvature feature computed for each vertex. It permits to extract sub-meshes. Each one corresponds to a primitive. Parameters of these primitives are found with a fitting process according to the curvature features.

Keywords: Primitives, CAD, reverse engineering, surface fitting

1. INTRODUCTION

In the industry, it is important to have a continuous representation for a CAD model. This representation is appropriate to make modification, data exchange or comparison with manufactured object. These models are composed of primitives like planes, spheres, cylinders, cones or parametric patches. But to be displayed or to make numerical computation, the CAD model must be discretized to obtain a mesh. The initial continuous object can be lost. In others cases, it does not correspond anymore with the mesh, after a numerical computation, for example. Then a reverse engineering process is used to recover a continuous model. The method to extract planes, spheres, cylinders and cones, for a given mesh created from a CAD object, is proposed in this paper. In this work, we consider that the object has not been distorted by noise.

In previous work[1] , we have proposed a method to extract parametric surfaces. A method to detect the mesh part which corresponds to simple primitives is now proposed, allowing to extract shape feature of each primitive with curvature informations. Indeed the curvature behavior depends on the primitive type and of the primitive parameters. So we can be used it to make the segmentation according to the primitive type and to determine the parameters of the primitives.

The section 2 presents the previous work focused on this problem. In Section 3, we describe our method which consists in finding and computing primitive parameters in a same process, then we present the results in Section 4 and to finish we conclude Section 5.

2. OVERVIEW

These last years, many methods have been proposed to extracting primitives in a reverse engineering (one can find a review in Varady *et al.*[2]) They are generally composed of three steps:

1. Segmentation: the mesh is divided into sub-meshes, by computing at each vertex, some shape features and by aggregating the vertices with present similar features,

roseline.beniere; gerard.subsol; william.puech@lirmm.fr; gilles.gesquiere@lsis.org; flb@c4W.com

2. Classification: a primitive type is associated to each sub-mesh,
3. Fitting: the parameters of the primitive are computed for each sub-mesh.

In the method described by Benko *et al*[3,4] the shape features are based on the co-planarity between neighbor triangles in order to detect the sharp edges of the mesh. Then, a plane is fitted to each sub-mesh. If the plane is close enough to the sub-mesh, it will be kept. Otherwise, the sub-mesh is approximated with more complex primitives as a sphere or a cylinder or *etc ...* until it corresponds quite well to the sub-mesh. The result may be improved by adding some constraints as tangency between primitives.

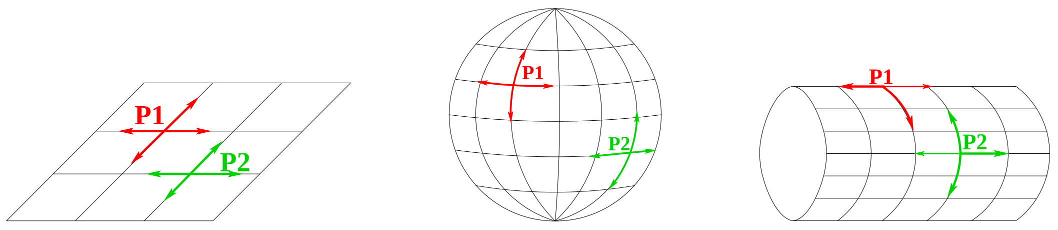
Several papers deal only with one step. For example, Bohm *et al*[5] and Lavva *et al*[6] expose techniques to segment and classify the sub-mesh, using curvature features. Sunil et Pande[7] base the segmentation and the classification steps on the dihedral angle and on the size of each triangle. Lukács *et al*[8] and Schnabel *et al*[9] propose solutions to fit primitives on a sub-mesh or a points cloud. The first method uses an approximation with all points whereas the second defines one primitive for each point group (for example, group of three points for a plane) and keeps the best one. Primitive recovering can be used also only to segment the mesh as in Attene *et al.*[10]

Our method is based on shape features which are defined by discrete curvature. Many methods to compute the curvatures have been proposed. A review and a comparison of them can be find in Magid *et al.*[11] These methods can be classified on three main classes: quadrics fitting,[12] Gauss-Bonnet scheme and Euler formula.[13] In the following, we chose to use a method based on the Euler formula.[14]

3. PRIMITIVES DETECTION METHOD

The curvature at a point P on the surface S is defined by the minimum and the maximum curvature and by the two directions corresponding to the tangent vectors for which the extremal curvature are reached. They are called : *principal curvatures* (k_{min} and k_{max}) and *principal directions* ($\overrightarrow{Dir_{min}}$ and $\overrightarrow{Dir_{max}}$).

Simple primitives, like planes, spheres, cones and cylinders, have specific curvature characteristics. In the case of a plane (see Figure 1(a)), the value of curvature is equal to 0 at each point ($k_{min} = k_{max} = 0$). For a sphere (see Figure 1(b)) all the points have the same curvature and its value is equal to the inverse of the sphere radius ($k_{min} = k_{max} = \frac{1}{Radius}$). The cone or the cylinder points can be defined by one principal curvature equal to 0 with the corresponding principal direction following the cone or cylinder generating line. Furthermore, the other principal curvature is then equal to the inverse of the cone or cylinder radius, as shown Figure 1(c).



(a) (b) (c)

Figure 1: Primitive curvature features.

The curvature characteristics summarized on Table 1 will be used in our method, to segment the sub-meshes, classify and compute primitive parameters.

	k_{min}	k_{max}	$\overrightarrow{Dir_{min}}$	$\overrightarrow{Dir_{max}}$	Illustration
Plane	$= 0$	$= 0$	not defined	not defined	Figure 1(a)
Sphere	$= \frac{1}{Radius}$	$= \frac{1}{Radius}$	not defined	not defined	Figure 1(b)
Cone/Cylinder	$= 0$	$= \frac{1}{Radius}$	= generating line	not used	Figure 1(c)
	$= \frac{1}{Radius}$	$= 0$	not used	= generating line	

Table 1: Curvature features for each primitive type

The proposed approach is composed of three steps:

1. Principal curvatures and principal directions are computed for each mesh vertex;
2. Neighbor vertices having the same curvature characteristics are grouped in sub-meshes. A primitive type is associated to each sub-mesh, using characteristic features which allow to create it;
3. A primitive is fitted on each sub-mesh. The curvature values are used to find the primitive parameters, according to the primitive type.

3.1 Planes and Spheres

For each point which belongs to a plane or a sphere the curvature value is the same, so $k_{min} = k_{max}$. These values are equal to 0 for a plane and to $\frac{1}{Radius}$ for a sphere.

To find these primitives, neighbor vertices having $|k_{max} - k_{min}| < \epsilon_{sameCurv}$ are grouped in a same sub-mesh. A first vertex with such features is found and vertex neighbors are studied to propagate the sub-mesh.

- After, for each sub-mesh, fitting is performed by a least square method. If $|\frac{k_{max}+k_{min}}{2}| < \epsilon_{isPlane}$, parameters which correspond to the coefficients of the implicit plane equation (1) are computed with a linear regression (see equation (2)).
$$ax + by + cz + d = 0 \tag{1}$$
$$\Sigma_i^n (ax_i + by_i + cz_i + d)^2 = 0 \tag{2}$$
- if $|\frac{k_{max}+k_{min}}{2}| > \epsilon_{isPlane}$ the sub-mesh corresponds to a sphere. The center and the radius of the sphere are approximated using equation (3) which represents the distance between a point and the surface of the sphere. A regression with a least square method is, also made in this case (equation (4)).
$$\sqrt{(x - x_{center})^2 + (y - y_{center})^2 + (z - z_{center})^2} - Radius \tag{3}$$
$$\Sigma_i^n ((x_i - x_{center})^2 + (y_i - y_{center})^2 + (z_i - z_{center})^2 - Radius^2)^2 = 0 \tag{4}$$
In this case, the curvature value is used to check approximated parameters. The radius approximated is compared with curvature average : $Q = |Radius - \frac{k_{max}+k_{min}}{2}|$. An important value of Q implies a problem on the process of extraction. For example, it can be produced by a mistake in segmentation when two spheres with a same radius are adjacent. When this problem is detected, the sub-mesh is segmented again by taking into account the normal to assess if the center is the same for each point.

3.2 Cones and Cylinders

A cylinder is a particular cone, so these two primitives have the same curvature behavior. Cone or Cylinder points have $k_{min} = 0$, $\overrightarrow{Dir_{min}}$ corresponding to the generating line and k_{max} is equal to the inverse of the cone or cylinder radius. Moreover if a point is translated with a vector equal to the normal multiplied by $\frac{1}{k_{max}}$, it will be on the rotation axis, as shown Figure 2. These primitives are characterized by two axis: the rotation axis and the generating line. The rotation axis is defined by a vector and by a point, which is the cone vertex or any axis point for cylinder. The generating line which is determined by an angle (between the rotation axis and the generating line) for cone and by a radius (which is the distance between the rotation axis and the generating line) for cylinder.

To know if two points are in the same cylinder or more generally in the same cone, the following criterion is defined. For these two points, a potential point on the axis is deduced by using the normal and the curvature. Then a potential axis is defined, and the angles between this axis and the directions equal to 0 are compared. If

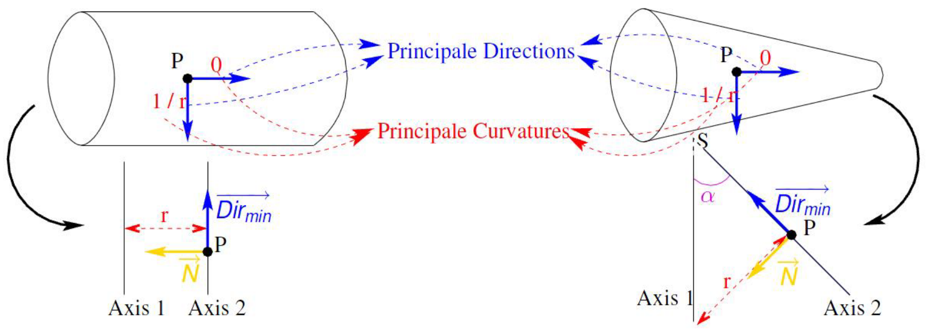


Figure 2: Cylinder and Cone features

the angles are equal, the two points belong to the same cone. On Figure 3, $P1$ and $P2$ are on the same cone, so $\alpha 1 = \alpha 2$.

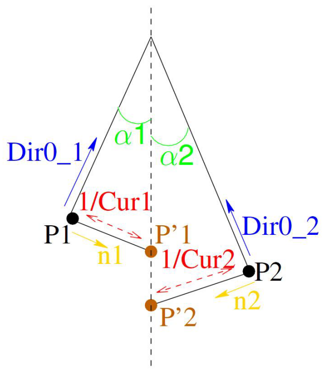


Figure 3: Criterion to test if two points belong to the same cone

This criterion is used to extract the sub-mesh corresponding to cone or cylinder. If two neighbor points follow a $k_{min} < \epsilon_{isConic}$, these points belong to a cone. Then to be sure that they belong to the same cone, the criterion is used. After their neighborhood are studied, to increase the sub-mesh.

Finally, for each points of the sub-mesh, the rotation axis is approximated with vertices computed using the normal and the inverse of the not null curvature. The generating axis is determined, computing the average of the angles, α, between the rotation axis and the direction, for which the curvature is null. If the $\alpha_{average} < \epsilon_{isCylinder}$, the sub-mesh correspond to a cylinder, and its radius is obtained by an average of the curvature inverse. Otherwise, it is a cone, and the generating line, is defined by the angle: $\alpha_{average}$.

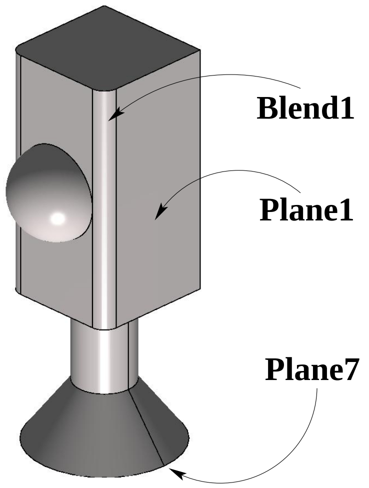
4. EXPERIMENTAL RESULTS

The method was implemented as a plug-in in the *C4W*[*] software, *3D Shop*[†]. To assess our method, several tests were done: we took B-Rep models which we discretized in order to apply our method. Then, we compare the original and the computed parameters, with a precision 0.001.

4.1 Assessment of the method

A simple CAD object *TestObject*, shown Figure 4, was created with 7 planes, 1 sphere, 1 cylinder and 1 cone. It contains also 2 blending surface between planes, which can be considered as cylinders, due to the constant radius. All parameters of the primitives are given in the Table of Figure 4. The object is then discretized by using the standard function of C4W.

[*]C4W is a company which develops innovative CAD solutions and, supports this work (http://www.c4w.com).
[†]www.c4w.com/3d-cad-software.html



		Parameter Values
Sphere	Center (x;y;z)	0;20;0
	Radius	15
Cylinder	Axis (x;y;z)	0;0;1
	Radius	10
Blend1	Axis (x;y;z)	0;0;1
	Radius	5
Cone	Axis (x;y;z)	0;0;1
	Angle	0.643
Plane 1	Coefficients (a;b;c;d)	-0.05;0;0;1
Plane 7	Coefficients (a;b;c;d)	0;0;-0.011;1

Figure 4: CAD object: *TestObject* with its parameters

Figures 5, 6 and 7 show three meshes, *MeshObject1*, *MeshObject2* and *MeshObject3*, obtained by discretizing the same CAD object at different resolutions (respectively 55,318; 6,434 and 1,712 triangles). This will allow one to check the behavior and the robustness of our method. All the recovered primitives are shown in Figures 10, 11 and 12.

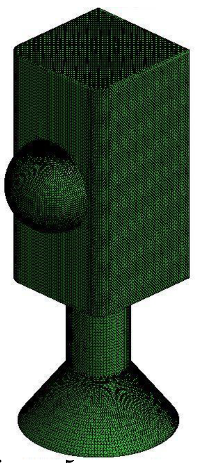


Figure 5: *MeshObject1*: 55,318 triangles

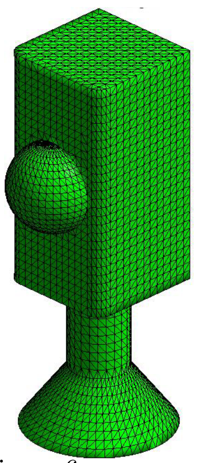


Figure 6: *MeshObject2*: 6,434 triangles

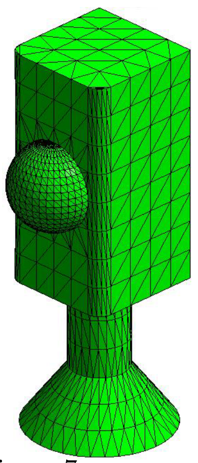


Figure 7: *MeshObject3*: 1,712 triangles

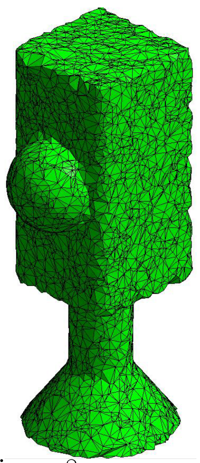


Figure 8: *MeshObjectIrreg*: 9,938 triangles

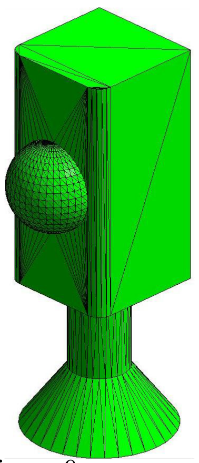


Figure 9: *MeshObjectSparse*: 1,034 triangles

On the *MeshObject1* and *MeshObject2*, all primitives are recovered and then the object can be completely reconstructed. To evaluate our method, an error is computed by the relative difference between initial and reconstructed parameters for each primitive. For the sphere, the error is lower than 0.01% for the center and the radius on the mesh *MeshObject1*, *MeshObject2* and *MeshObject3*. In the case of the cone and the cylinders, the errors corresponding to the deviation angle for the rotation axis, are lower than 0.01%. For the cone, the error for the angle, between the two axis, is 0.01% for *MeshObject1* and 0.2% for *MeshObject2* and *MeshObject3*. For the cylinder, the radius error are 0.02% for *MeshObject1* and 0.2% for *MeshObject2*. The radius error of the cylinders for the blends, are included between 1% and 4%. To compare the planes, the angle given by the normal of the reconstructed object and of the reference one is studied. For all planes in the three meshes, if the plane is detected, the angle error is lower than 0.01%. First results on a simple use case are correct. The detected primitives are very closed to the reference one and appear sufficient in the frame of our industrial application.

The object *TestObject*, was discretized in a irregular way to obtain the mesh *MeshTestIrregular*, shown Figure

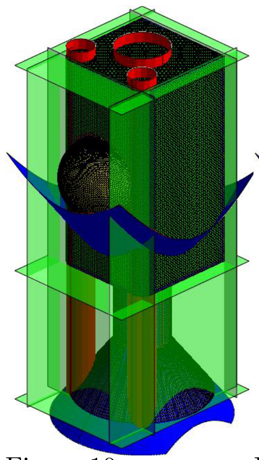


Figure 10: Primitives from *MeshObject1*

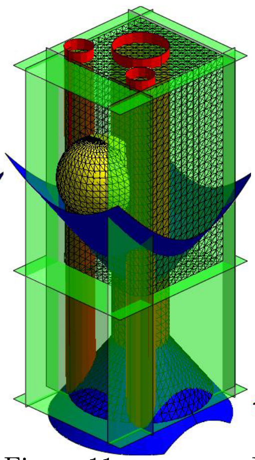


Figure 11: Primitives from *MeshObject2*

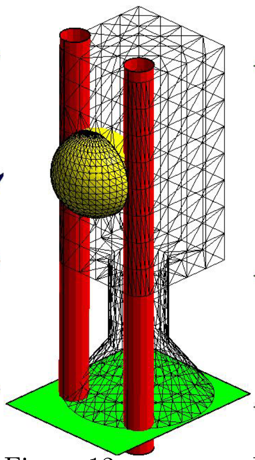


Figure 12: Primitives from *MeshObject3*

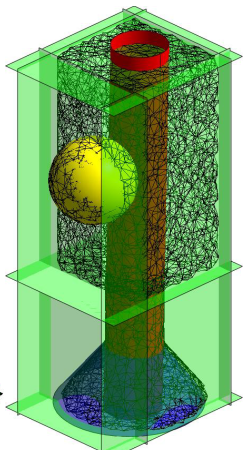


Figure 13: Primitives from *MeshObjectIrreg*

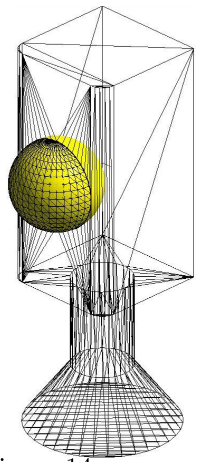


Figure 14: Primitives from *MeshObjectSparse*

8. To create this mesh, we use points on the B-Rep model, but they are not extract using the isoparametrics. Figure 13 presents the result: in this case all primitives are extracted except the blends. The error parameter values for planes are lower than 0.01% and for the sphere, the cone and the cylinder, they are included between 0.03% et 1.99%.

Due to the weak number of triangles in *MeshObject3*, few primitives are found. In this particular case, the curvature is not precisely computed. The neighborhood of each vertex is not sufficient and the computation is disturbed by neighbors belonging to others primitives. A particular case is exhibited in the Figure 9(*MeshObjectSparse*). Only a sphere is extracted (Figure 14), whereas the mesh is an exact discretization of the B-Rep model. This sparse mesh problem can be avoided by introducing a pre-segmentation step to restrict the vertex neighborhood. For instance, by analyzing the dihedral angles, it is possible to compute a simple segmentation as shown Figure 15 where each sub-mesh is described with a different color. With this pre-processing, all primitives are found (Figure 16). The parameters of planes and sphere are approximated with an error lower than 0.01%. For the cone and cylinders, errors on axis are lower than 0.01%. The cone angle error is equal to 0.07%. the radius cylinder is equal to 0.3% but the blend radius is quite large, 21.8%. All of these results are synthesized in Table 2.

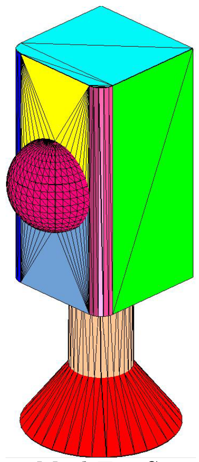


Figure 15: *MeshTestSparse* segmented

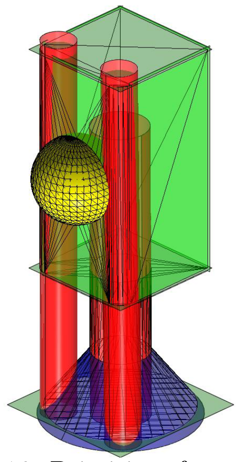


Figure 16: Primitives from *MeshObject4* segmented

		MeshObject1	*MeshObject2*	*MeshObject3*	*MeshObjectIrregular*	*MeshObjectSparse*
Sphere	Center (x;y;z)	0;19.999;0	0;19.999;0	0;19.999;0	0.07;19.912;-0.003	0;19.999;0
	Radius	14.999	14.999	14.999	15.104	14.999
Cylinder	Axis (x;y;z)	0;0;1	0;0;1	Not Detected	-0.019;0.0043;1	0.003;0;1
	Radius	9.998	9.997	Not Detected	9.924	9.975
Blend1	Axis (x;y;z)	0;0;1	0;0;1	0.001;0.002;1	Not Detected	0.006;-0.007;1
	Radius	5.054	5.222	5.064	Not Detected	6.092
Cone	Axis (x;y;z)	0;0;1	0;0;1	Not Detected	-0.0351;0.0147;1	0.002;0;1
	Angle	0.643	0.644	Not Detected	0.631	0.644
Plane 1	Coefficients (a;b;c;d)	-0.05;0;0;1	-0.05;0;0;1	Not Detected	-0.05;0;0;1	-0.05;0;0;1
Plane 7	Coefficients (a;b;c;d)	0;0;-0.011;1	0;0;-0.011;1	0;0;-0.011;1	0;0;-0.011;1	0;0;-0.011;1

Table 2: Table of primitive parameters computed

4.2 Industrial CAD data

4.2.1 Motor

The same process was performed on a complex CAD object, a motor (see Figure 17) which was provided by *Aim@SHAPE Project*[‡]. The discretization of the B-rep model gives 22,482 triangles (see Figure 18). Our method detect many primitives, 26 planes and 25 cylinders (see Figure 19). A comparison between the original and the approximated parameters is presented in the Table 3 for three primitives. The results are very satisfying.

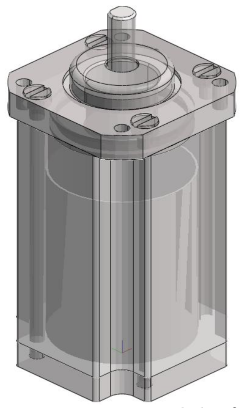


Figure 17: B-Rep model of a motor

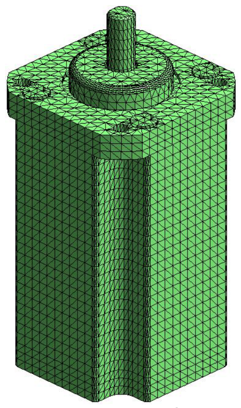


Figure 18: *MotorMesh*: 22,482 triangles

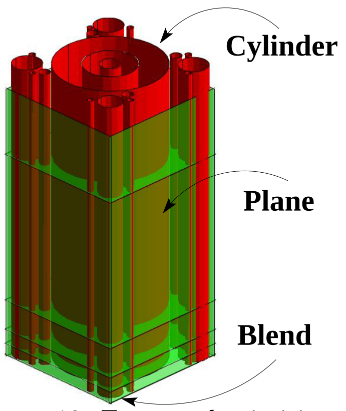


Figure 19: Extracted primitives

		Original Values	*MotorMesh*
Cylinder	Axis (x;y;z)	0;0;1	0;0;0.999
	Radius	33.5	33.499
Blend	Axis (x;y;z)	0;0;1	0;0;0.999
	Radius	7	7.079
Plane	Coefficients (a;b;c;d)	0;0.024;0;1	0;0.024;0;1

Table 3: Table of primitive parameters computed from *MotorMesh*

4.2.2 Pump

Our method was also tested on a real industrial object, corresponding to an oil pump commercialize by the *Viking Pump*[§] company. The B-Rep model (see Figure 20), was discretized to obtain the mesh presented in

[‡]http://shapes.aimatshape.net/viewgroup.php?id=1242
[§]http://www.vikingpump.com/en/engineering/3dmodels.html

Figure 21, which contains 158,746 triangles. Figures 22 and 23 present primitives which are extracted from the mesh. 1 sphere, 9 planes, 13 cylinders and 1 cone are found. Moreover 31 cylindrical blends are detected, (see Figure 24). The Table 4 shows a comparison between the exact (in millimeters) and the computed parameters for three primitives.

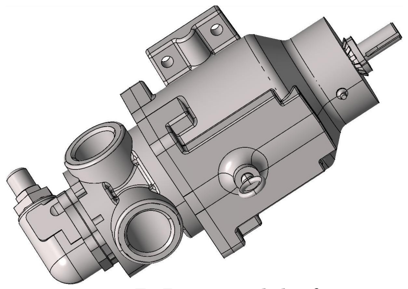


Figure 20: B-Rep model of a pump

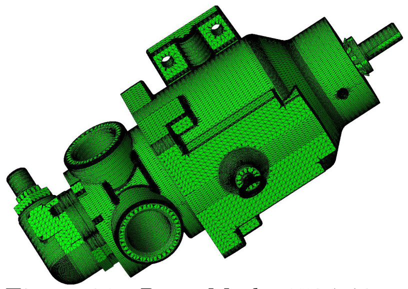


Figure 21: *PumpMesh*: 158,746 triangles

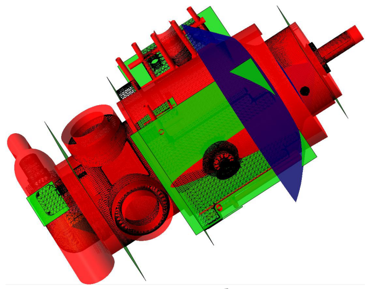


Figure 22: Extracted primitives

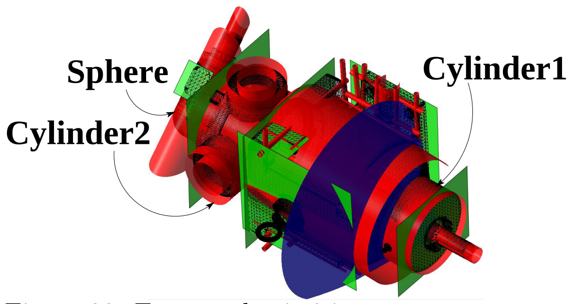


Figure 23: Extracted primitives

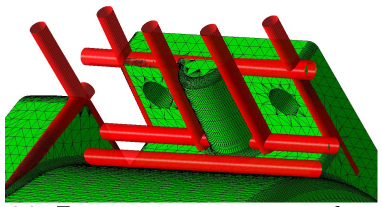


Figure 24: Zoom on some extracted cylindrical blends

		Original Values	*PumpMesh*
Cylinder1	Axis (x;y;z)	1;0;0	0.999;-0.011;-0.004
	Radius	49.669	49.669
Cylinder2	Axis (x;y;z)	0;0;1	-0.006;1;0.002
	Radius	34.162	34.175
Sphere	Center (x;y;z)	409.175;367.654;515.722	409.167;367.682;515.780
	Radius	23.114	23.088

Table 4: Table of primitive parameters computed from *PumpMesh*

5. CONCLUSION AND PERSPECTIVES

In this paper, a new method to detect and to compute their parameters from a 3D mesh is presented. These two steps are based on the curvature features. This method gives preliminary correct results with CAD meshes, even if they are a composed of a lot of objects. The results appear good enough to be used in our industrial process.

To improve our method, we have first to modify the computation of the cone and cylinder parameters. Like for planes and spheres an approximation on points cloud, should give better results than averages. Nevertheless we plan to keep the latter method in order to make verification. Others primitives like torus or revolution surfaces which present particular curvature features, could also be considered.

The tests which we have done by pre-segmenting the mesh show that it improves the result, so we plan to integrate this step in our method. For the first tests, the segmentation is based on dihedral angle and this local criterion gives good result on simple objects. But a global criterion like curvature variation for example, should give a real and general improvement.

To generalize our method, we now have to adapt our method to take into account noisy meshes, resulting from 3D-scanning for example. The process will be the same, but the thresholds used during the detection and the fitting, have to be adapted.

To finish, this method should be developed by adding a step to trim the primitives with correct boundaries. The boundaries could be extracted by computing the intersections between the continuous representation of the adjacent primitives.

REFERENCES

[1] Bénière, R., Subsol, G., Gesquière, G., Le Breton, F., and Puech, W., "Decomposition of a 3D Triangular Mesh into Quadrangulated Patches," *International Conference on Computer Graphics Theory and Application (GRAPP 2010)* , 96–103 (2010).

[2] Varady, T., Martin, R., and Cox, J., "Reverse engineering of geometric models–an introduction," *Computer-Aided Design* **29**(4), 255–268 (1997).

[3] Benkõ, P., Martin, R., and Várady, T., "Algorithms for reverse engineering boundary representation models," *Computer-Aided Design* **33**(11), 839–851 (2001).

[4] Benkö, P., Kós, G., Várady, T., Andor, L., and Martin, R. R., "Constrained fitting in reverse engineering," *Computer Aided Geometric Design* **19**(3), 173–205 (2002).

[5] Bohm, J. and Brenner, C., "Curvature based range image classification for object recognition," in [*PROC SPIE INT SOC OPT ENG*], **4197**, 211–220 (2000).

[6] Lavva, I., Hameiri, E., and Shimshoni, I., "Robust methods for geometric primitive recovery and estimation from range images," *IEEE Trans. Systems, Man and Cybernetics* **37**, 826–845 (jun 2007).

[7] Sunil, V. B. and Pande, S. S., "Automatic recognition of features from freeform surface CAD models," *Computer-Aided Design* **40**(4), 502–517 (2008).

[8] Lukács, G., Martin, R., and Marshall, D., "Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation," *Computer Vision-ECCV'98* **1406**, 671–686 (1998).

[9] Schnabel, R., Wahl, R., and Klein, R., "Efficient RANSAC for point-cloud shape detection," *Comput. Graph. Forum* **26**(2), 214–226 (2007).

[10] Attene, M., Falcidieno, B., and Spagnuolo, M., "Hierarchical mesh segmentation based on fitting primitives," *The Visual Computer* **22**(3), 181–193 (2006).

[11] Magid, E., Soldea, O., and Rivlin, E., "A comparison of Gaussian and mean curvature estimation methods on triangular meshes of range image data," *Computer Vision and Image Understanding* **107**, 139–159 (sep 2007).

[12] Douros, I. and Buxton, B., "Three-dimensional surface curvature estimation using quadric surface patches," *Scanning 2002 Proceedings* (2002).

[13] Chen, X. and Schmitt, F., "Intrinsic surface properties from surface triangulation," in [*ECCV*], **588**, 739–743 (mai 1992).

[14] Dong, C.-s. and Wang, G.-z., "Curvatures estimation on triangular mesh," *Journal of Zhejiang University - Science A* **6**(1), 128–136 (2005).