Recovering Primitives in 3D CAD meshes

Roseline Bénière
G. Subsol, G. Gesquière, F. Le Breton and W. Puech

LIRMM, Montpellier, France
C4W, Montpellier, France
LSIS, Arles, France

14 octobre 2010
Réunion ICAR

LIRMM
C4W

Custom CAD Software
In CAD an object is usually modeled by a structured combination of primitives
Objective

In CAD an object is usually modeled by a structured combination of primitives. But to use, we often need to discretized it into a 3D mesh.
Objective

In CAD an object is usually modeled by a structured combination of primitives. But to use, we often need to discretized it into a 3D mesh. And the initial model can be lost or not correspond anymore.
Objective

In CAD an object is usually modeled by a structured combination of primitives.
But to use, we often need to discretize it into a 3D mesh.
And the initial model can be lost or not correspond anymore.
So a primitive extraction algorithm is needed to reconstruct the initial representation.
Previews

Benkõ et al.
*Algorithms for reverse engineering boundary representation models*

Bohm et al.
*Curvature based range image classification for object*
PROC SPIE INT SOC OPT ENG 4197 : 211-220 2000
Previews

- Benkő et al.  
  *Algorithms for reverse engineering boundary representation models*  

- Bohm et al.  
  *Curvature based range image classification for object*  
  PROC SPIE INT SOC OPT ENG 4197 : 211-220 2000

Same Process :
Previews

Benkő et al.
*Algorithms for reverse engineering boundary representation models*

Bohm et al.
*Curvature based range image classification for object*
PROC SPIE INT SOC OPT ENG 4197 : 211-220 2000

Same Process:

1. Segmentation
Previews

- Benkõ et al.  
  *Algorithms for reverse engineering boundary representation models*  

- Bohm et al.  
  *Curvature based range image classification for object*  
  PROC SPIE INT SOC OPT ENG 4197 : 211-220 2000

Same Process:

1. Segmentation
2. Classification
Previews

Benkő et al.
Algorithms for reverse engineering boundary representation models

Bohm et al.
Curvature based range image classification for object
PROC SPIE INT SOC OPT ENG 4197: 211-220 2000

Same Process:
1. Segmentation
2. Classification
3. Fitting
Curvature 2D and 3D

Curvature 2D:

\[
\mathbf{N} \mathbf{P} \mathbf{C} \mathbf{R}
\]

\[ \Rightarrow K_p = \frac{1}{R} \]

Curvature 3D:

\[
\mathbf{N} \mathbf{S} \mathbf{P} \mathbf{C} \mathbf{PL}
\]

2 Principal Curvatures (\(k_{\text{max}}\) et \(k_{\text{min}}\))

2 Principal Directions (\(\text{Dir}_{\text{max}}\) et \(\text{Dir}_{\text{min}}\))

Normal Recovering Primitives in 3D CAD / R.Bénier
Curvature 2D and 3D

Curvature 2D:

\[ K_p = \frac{1}{R} \]
Curvature 2D and 3D

Curvature 2D:

Curvature 3D:

\[ K_p = \frac{1}{R} \]
Curvature 2D and 3D

Curvature 2D:

\[ K_p = \frac{1}{R} \]

Curvature 3D:

- 2 Principal Curvatures \((k_{max} \text{ et } k_{min})\)
- 2 Principal Directions \((Dir_{max} \text{ et } Dir_{min})\)
- Normal
Primitive curvature features

The points contained in Plane, Sphere, Cone or Cylinder have specific features on curvature:
Primitive curvature features

The points contained in Plane, Sphere, Cone or Cylinder have specific features on curvature:

- **Plane** ⇒ $k_{\text{max}} = k_{\text{min}} = 0$

Diagram:

![Diagram showing plane and curvature](image)
Primitive curvature features

The points contained in Plane, Sphere, Cone or Cylinder have specific features on curvature:

- **Plane** $\Rightarrow k_{max} = k_{min} = 0$

- **Sphere** $\Rightarrow k_{max} = k_{min} = \frac{1}{r_{Sp}} \neq 0$
Primitive curvature features

The points contained in Plane, Sphere, Cone or Cylinder have specific features on curvature:

- **Plane** ⇒ $k_{max} = k_{min} = 0$

- **Sphere** ⇒ $k_{max} = k_{min} = \frac{1}{r_{Sp}} \neq 0$

- **Cylinder** ⇒ $k_{min} = 0$ et $k_{max} = \frac{1}{r_{Cy}}$

  $Dir_{min} = \text{Generating line}$
Primitive curvature features

The points contained in Plane, Sphere, Cone or Cylinder have specific features on curvature:

- **Plane** ⇒ \( k_{\text{max}} = k_{\text{min}} = 0 \)

- **Sphere** ⇒ \( k_{\text{max}} = k_{\text{min}} = \frac{1}{r_{\text{Sp}}} \neq 0 \)

- **Cylinder** ⇒ \( k_{\text{min}} = 0 \) et \( k_{\text{max}} = \frac{1}{r_{\text{Cy}}} \)
  
  \( \text{Dir}_{\text{min}} = \text{Generating line} \)

- **Cone** ⇒ idem Cylinder but with a variable radius
Discrete Curvature

In a mesh, we compute a discrete curvature for each point.

Discrete Curvature

In a mesh, we compute a discrete curvature for each point.
Discrete Curvature

In a mesh, we compute a discrete curvature for each point. We use the Euler formula

\[ k_n = k_{max} \cos^2(\theta) + k_{min} \sin^2(\theta) \]

With \( \theta \) the angle between \( n \) and \( \text{Dir}_{max} \).

The neighbors are studied to approximate \( k_{max}, k_{min} \) and \( \theta \).
In a mesh, we compute a discrete curvature for each point. We use the Euler formula

\[ k_n = k_{\text{max}} \cos^2(\theta) + k_{\text{min}} \sin^2(\theta) \]

With \( \theta \) the angle between \( n \) and \( \text{Dir}_{\text{max}} \). The neighbors are studied to approximate \( k_{\text{max}}, k_{\text{min}} \) and \( \theta \). To determine the point which will be used, we fix a \textit{k-neighborhood}. 

**Discrete Curvature**
Discrete Curvature

In a mesh, we compute a discrete curvature for each point. We use the Euler formula

\[ k_n = k_{\text{max}} \cos^2(\theta) + k_{\text{min}} \sin^2(\theta) \]

With \( \theta \) the angle between \( n \) and \( \text{Dir}_{\text{max}} \). The neighbors are studied to approximate \( k_{\text{max}}, k_{\text{min}} \) and \( \theta \).

To determine the point which will be used, we fix a \textit{k-neighborhood}.

- Concave Point
- Convex Point
- Saddle Point
- Plane Point
- Spheric Point
Planes Extraction

- From Curvatures
Planes Extraction

- From Curvatures
- Group all adjacent points with $k_{max} = k_{min} = 0$
Planes Extraction

- From Curvatures
- Group all adjacent points with $k_{max} = k_{min} = 0$
Planes Extraction

- From Curvatures
- Group all adjacent points with $k_{\text{max}} = k_{\text{min}} = 0$
- Equation Coefficients: $ax + by + cz + d = 0$ are approximated by a least square regression
Planes Extraction

- From Curvatures
- Group all adjacent points with $k_{max} = k_{min} = 0$
- Equation Coefficients: $ax + by + cz + d = 0$ are approximated by a least square regression
Spheres Extraction

- From Curvatures
Spheres Extraction

- From Curvatures
- Group all adjacent points with $k_{max} = k_{min} \approx K$
Spheres Extraction

- From Curvatures
- Group all adjacent points with $k_{max} = k_{min} \approx K$
Spheres Extraction

- From Curvatures
- Group all adjacent points with $k_{max} = k_{min} \approx K$
- The radius and the center are approximated by a least square method
  ⇒ The average of the curvature inverse is used to validate it
Spheres Extraction

- From Curvatures
- Group all adjacent points with $k_{\text{max}} = k_{\text{min}} \approx K$
- The radius and the center are approximated by a least square method
⇒ The average of the curvature inverse is used to validate it
Cones/Cylinders Extraction

Features of Cones and Cylinders:

Principale Directions
Principale Curvatures

Axis 1
Axis 2

r
1/r

1/Cur1
1/Cur2

α

α

r

P

P'

P

P'

S

P

Axis 1
Axis 2
Cones/Cylinders Extraction

Features of Cones and Cylinders:

- **Principale Directions**
- **Principale Curvatures**

Criterion of belonging to a same cone:
If \( P_1 \) and \( P_2 \) are in the same cone
\[ \Rightarrow \alpha_1 = \alpha_2 \]
Cones/Cylinders Extraction

- From Curvatures

\[ \text{pointAxisTheo} = \text{point} + \text{normal} \times \text{radius} \]

\[ \Rightarrow \text{Rotation Axis} \]

\[ \Rightarrow \alpha \text{ angle between Axis and Dir} \]

\[ \alpha = \pi \Rightarrow \text{Cylinder: Average curvature} \]

\[ \alpha \neq \pi \Rightarrow \text{Cone: Intersection between each plane created by the two principal directions of one point} \]

\[ \Rightarrow \text{Vertex} \]
Cones/Cylinders Extraction

- From Curvatures
- Group adjacent points by the criterion of belonging
Cones/Cylinders Extraction

- From Curvatures
- Group adjacent points by the criterion of belonging
Cones/Cylinders Extraction

- From Curvatures
- Group adjacent points by the criterion of belonging
- For each point: \( \text{pointAxisTheo} = \text{point} + \text{normal} \times \text{radius} \)
  \( \Rightarrow \) Rotation Axis \( \Rightarrow \alpha \) angle between Axis and \( \text{Dir}_{=0} \)
  - \( \alpha = \pi \Rightarrow \) Cylinder: Average curvature \( \Rightarrow \) Radius
  - \( \alpha \neq \pi \Rightarrow \) Cone: Intersection between each plane created by the two principal directions of one point \( \Rightarrow \) Vertex
Cones/Cylinders Extraction

- From Curvatures
- Group adjacent points by the criterion of belonging
- For each point: \( \text{pointAxisTheo} = \text{point} + \text{normal} \times \text{radius} \)
  \[ \Rightarrow \text{Rotation Axis} \Rightarrow \alpha \text{ angle between Axis and } \text{Dir}_{\alpha=0} \]
  - \( \alpha = \pi \Rightarrow \text{Cylinder: Average curvature } \Rightarrow \text{Radius} \)
  - \( \alpha \neq \pi \Rightarrow \text{Cone: Intersection between each plane created by the two principal directions of one point } \Rightarrow \text{Vertex} \)
Results

Recovering Primitives in 3D CAD / R.Bénière
Results

http://shapes.aimatshape.net
Sparse Mesh

The mesh can not have many points
Sparse Mesh

The mesh cannot have many points
⇒ The curvature computation is not correct
Sparse Mesh

The mesh can not have many point
⇒ The curvature computation is not correct
⇒ The primitive extraction is disturbed
Sparse Mesh ⇒ Segmentation

A solution can be to segment the mesh (by the dihedral angle for example)
Sparse Mesh ⇒ Segmentation

A solution can be to segment the mesh (by the dihedral angle for example) ⇒ The curvature computation is better
Sparse Mesh $\Rightarrow$ Segmentation

A solution can be to segment the mesh (by the dihedral angle for example)

$\Rightarrow$ The curvature computation is better

$\Rightarrow$ The primitive extraction is improved
Noisy Mesh

The mesh cannot have many points.
Noisy Mesh

The mesh cannot have many points
⇒ The curvature computation is not correct
Noisy Mesh

The mesh can not have many point
⇒ The curvature computation is not correct
⇒ The primitive extraction is disturbed
Conclusion and Future Work

Our method take a Mesh
Conclusion and Future Work

Our method takes a Mesh $\Rightarrow$ extract primitives.
Conclusion and Future Work

Our method takes a Mesh $\Rightarrow$ extract primitives.

Future Work
- Cut and Fuse Primitives to reconstruct the continue representation
Conclusion and Future Work

Our method takes a Mesh \[\rightarrow\] extract primitives.

Future Work
- Cut and Fuse Primitives to reconstruct the continuous representation
- Add a Segmentation step to the method
Conclusion and Future Work

Our method takes a Mesh $\rightarrow$ extract primitives.

Future Work
- Cut and Fuse Primitives to reconstruct the continue representation
- Add a Segmentation step to the method
- Deal with noisy mesh
Thanks for your attention

QUESTIONS?

Site: www.lirmm.fr/~beniere
Mail: roseline.beniere@lirmm.fr
C4W site: www.c4w.com