Decomposition of a 3D triangular mesh into quadrangulated patches

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

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

May 21st
GRAPP 2010

Objective

Decompose a triangular mesh into a set of quadrangulated patches.

Objective

Decompose a triangular mesh into a set of quadrangulated patches.

A patch:

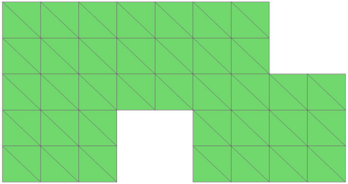
- is constituted of quads
- has a rectangular grid structure

Objective

Decompose a triangular mesh into a set of quadrangulated patches.

A patch:

- is constituted of quads
- has a rectangular grid structure

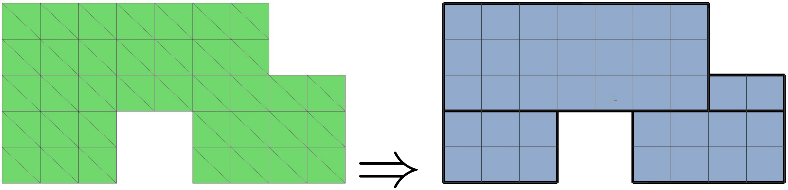


Objective

Decompose a triangular mesh into a set of quadrangulated patches.

A patch:

- is constituted of quads
- has a rectangular grid structure



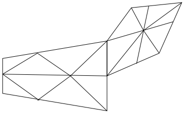
Motivation

Patches can be used for:

- interpolating or approximating a surface by a continuous representation,
- making reverse engineering to recognize the grid of the control points,
- compressing 3D mesh geometry without describing the topology,
- applying subdivision schemes,
- doing numerical simulation based on finite elements.

Constraints

Assumption: the mesh coordinates are exact ⇒ do not change the shape:

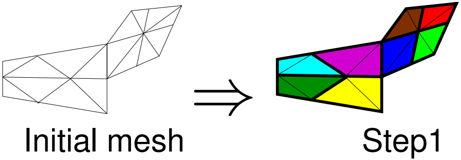


Initial mesh

Constraints

Assumption: the mesh coordinates are exact ⇒ do not change the shape:

- the vertices must be preserved,
- the edges are derived from the original triangular mesh.

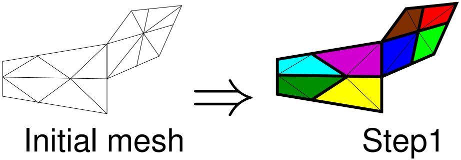


Constraints

Assumption: the mesh coordinates are exact ⇒ do not change the shape:

- the vertices must be preserved,
- the edges are derived from the original triangular mesh.

To create patches ⇒ last constraint:



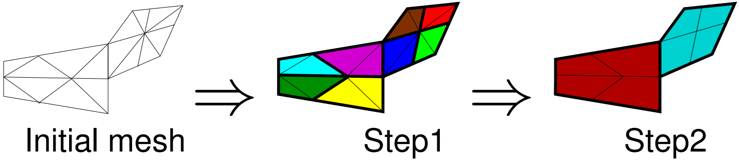
Constraints

Assumption: the mesh coordinates are exact ⇒ do not change the shape:

- the vertices must be preserved,
- the edges are derived from the original triangular mesh.

To create patches ⇒ last constraint:

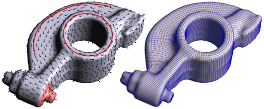
- the quadrangulated meshes are decomposed into quad rectangular grids.



State of the art: Triangular to quadrangular mesh

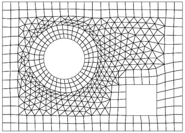
Remeshing algorithms

- Huang *et al.*
  Spectral quadrangulation with orientation and alignment control
  ACM trans. Graph. 27(5):1-9 2008



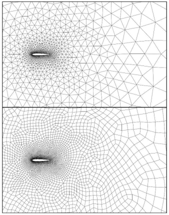
Advancing front algorithms

- Owen *et al.*
  Advancing front quadrilateral meshing using triangle transformations.
  7th International Meshing Roundtable:409-428 1998.



Merging algorithms

- Borouchaki and Frey.
  Adaptive Triangular-Quadrilateral Mesh Generation.
  International Journal for Numerical Methods in Engineering 1998.



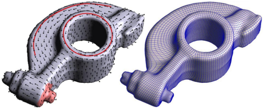
State of the art: Triangular to quadrangular mesh

Remeshing algorithms

- Huang *et al.*

  Spectral quadrangulation with orientation and alignment control

  ACM trans. Graph. 27(5):1-9 2008

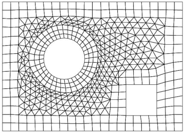


Advancing front algorithms

- Owen *et al.*

  Advancing front quadrilateral meshing using triangle transformations.

  7th International Meshing Roundtable:409-428 1998.

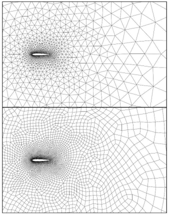


Merging algorithms

- Borouchaki and Frey.

  Adaptive Triangular-Quadrilateral Mesh Generation.

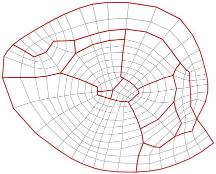
  International Journal for Numerical Methods in Engineering 1998.



State of the art: Quadrangular meshes to patches

Decomposition into patches

- Eppstein *et al*.
  Motorcycle graphs: Canonical mesh partitioning.
  Comput. Graph. forum, 27(5):1477-1486 2008.



Outline

1. Presentation of the Decomposition Method
   - Computation of a Quality Coefficient
   - Construction of Quadrangulated Areas
   - Decomposition into Quadrangulated Patches
2. Experimental Results
   - First Results
   - Threshold Variations
   - CAD Objects
3. Conclusion and Future Work
   - Conclusion
   - Future Work

Our method

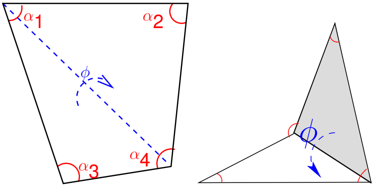
3 steps:

1. Computation of a quality coefficient for each pair of adjacent triangles
2. Construction of quadrangulated areas, using the quality coefficients
3. Decomposition into quadrangulated patches from quadrangulated areas

1) Computation of a Quality Coefficient

Computation of the quality coefficient Q based on:

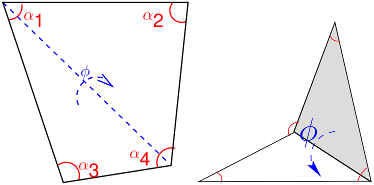
- dihedral angle (ϕ)
- angles between connected edges (α_i)



1) Computation of a Quality Coefficient

Computation of the quality coefficient Q based on:

- dihedral angle (ϕ)
- angles between connected edges (α_i)

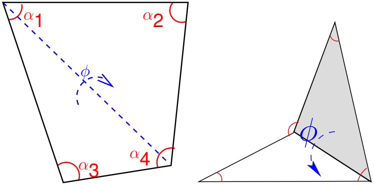


$$Q = \begin{cases} 2\pi & \text{if } \phi < \phi_{min} \\ \frac{1}{4}\sum_{i=1}^{4} |\frac{\pi}{2} - \alpha_i| & \text{elsewhere.} \end{cases}$$

1) Computation of a Quality Coefficient

Computation of the quality coefficient Q based on:

- dihedral angle (ϕ)
- angles between connected edges (α_i)



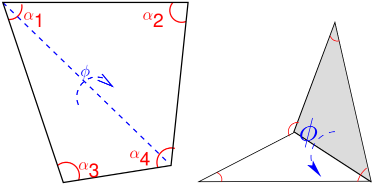
$$Q = \begin{cases} 2\pi & \text{if } \phi < \phi_{min} \\ \frac{1}{4}\sum_{i=1}^{4} |\frac{\pi}{2} - \alpha_i| & \text{elsewhere.} \end{cases}$$

if $Q \approx 0$

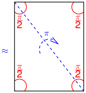
1) Computation of a Quality Coefficient

Computation of the quality coefficient Q based on:

- dihedral angle (ϕ)
- angles between connected edges (α_i)



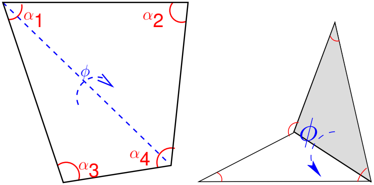
$$Q = \begin{cases} 2\pi & \text{if } \phi < \phi_{min} \\ \frac{1}{4}\sum_{i=1}^{4} \left|\frac{\pi}{2} - \alpha_i\right| & \text{elsewhere.} \end{cases}$$

if $Q \approx 0 \Rightarrow$ quad $\approx$ planar rectangle 

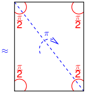
1) Computation of a Quality Coefficient

Computation of the quality coefficient Q based on:

- dihedral angle (ϕ)
- angles between connected edges (α_i)



$$Q = \begin{cases} 2\pi & \text{if } \phi < \phi_{min} \\ \frac{1}{4}\sum_{i=1}^{4} |\frac{\pi}{2} - \alpha_i| & \text{elsewhere.} \end{cases}$$

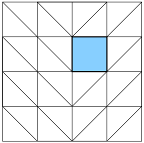


if $Q \approx 0 \Rightarrow$ quad $\approx$ planar rectangle
$\Rightarrow$ quad with good quality

2) Construction of Quadrangulated Areas

Iterative construction of quadrangulated areas:

Start with the best Q.

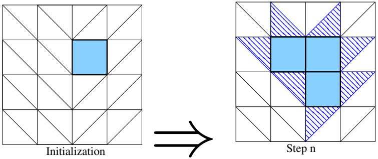


Initialization

2) Construction of Quadrangulated Areas

Iterative construction of quadrangulated areas:

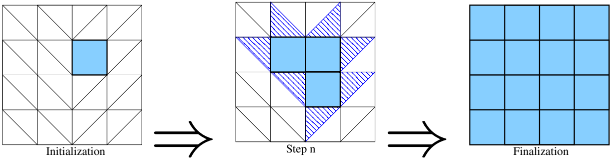
Find the quad with the best Q in the neighborhood



2) Construction of Quadrangulated Areas

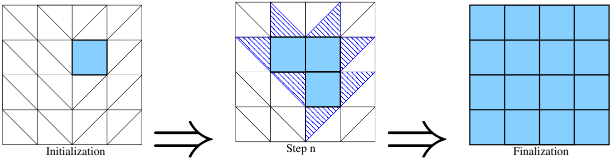
Iterative construction of quadrangulated areas:

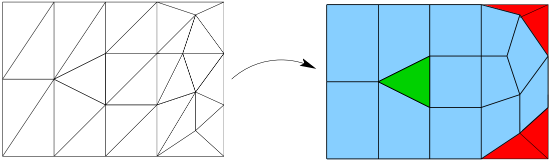
No new quad can be created.



2) Construction of Quadrangulated Areas

Iterative construction of quadrangulated areas:



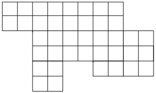


Left triangles:

- isolated triangles 🟩
- triangles of quads with $Q > Q_{max}$ 🟥

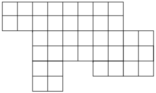
3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”

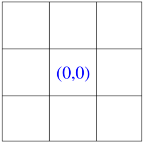


3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



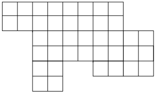
Each quad is labeled with a position using neighbors



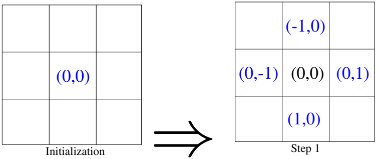
Initialization

3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors

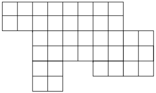


Initialization

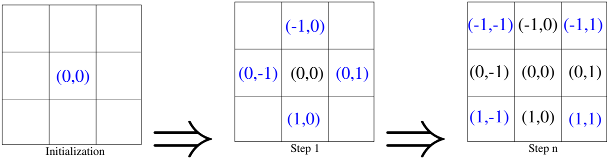
Step 1

3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”

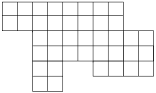


Each quad is labeled with a position using neighbors

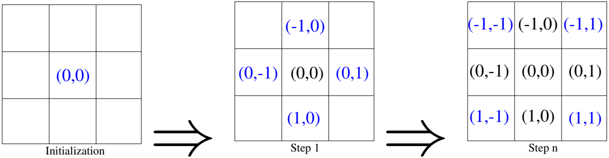


3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors



	(0,0)	

Initialization

⇒

	(-1,0)	
(0,-1)	(0,0)	(0,1)
	(1,0)	

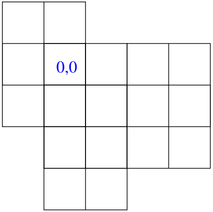
Step 1

⇒

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

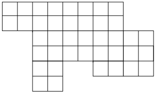
Step n

Example:

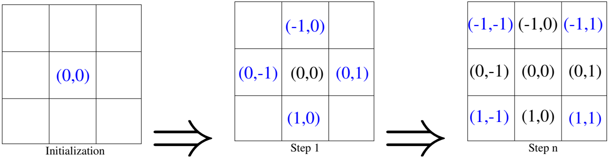


3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors



	(0,0)	

Initialization

⇒

	(-1,0)	
(0,-1)	(0,0)	(0,1)
	(1,0)	

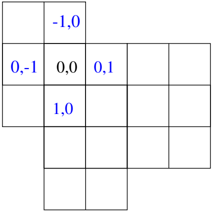
Step 1

⇒

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

Step n

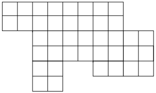
Example:



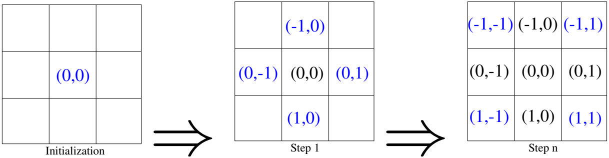
	-1,0			
0,-1	0,0	0,1		
	1,0			

3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors



	(0,0)	

Initialization

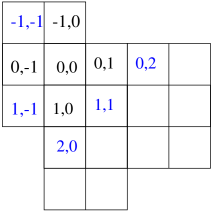
	(-1,0)	
(0,-1)	(0,0)	(0,1)
	(1,0)	

Step 1

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

Step n

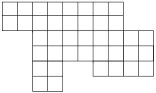
Example:



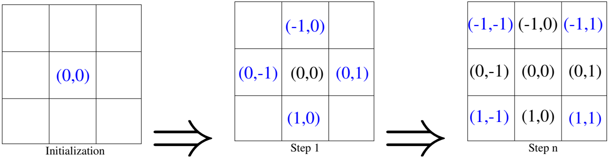
-1,-1	-1,0			
0,-1	0,0	0,1	0,2	
1,-1	1,0	1,1		
	2,0			

3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors



	(0,0)	

Initialization

⇒

	(-1,0)	
(0,-1)	(0,0)	(0,1)
	(1,0)	

Step 1

⇒

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

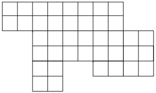
Step n

Example:

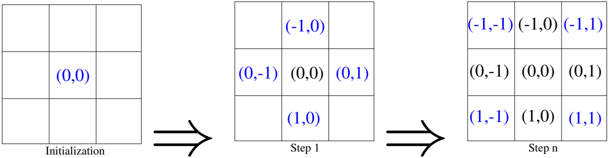
-1,-1	-1,0			
0,-1	0,0	0,1	0,2	0,3
1,-1	1,0	1,1	1,2	
	2,0	2,1		
	3,0			

3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors



	(0,0)	

Initialization

⇒

	(-1,0)	
(0,-1)	(0,0)	(0,1)
	(1,0)	

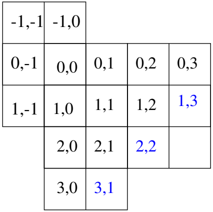
Step 1

⇒

(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(1,1)

Step n

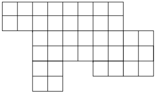
Example:



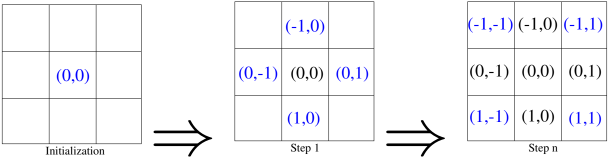
-1,-1	-1,0			
0,-1	0,0	0,1	0,2	0,3
1,-1	1,0	1,1	1,2	1,3
	2,0	2,1	2,2	
	3,0	3,1		

3) Decomposition into Quadrangulated Patches

3.1) The quads are arranged into “rectilinear polygons”



Each quad is labeled with a position using neighbors

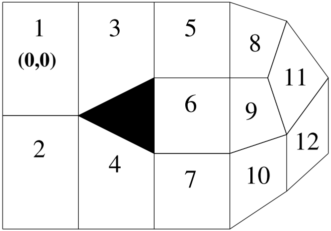


Example:

-1,-1	-1,0			
0,-1	0,0	0,1	0,2	0,3
1,-1	1,0	1,1	1,2	1,3
	2,0	2,1	2,2	2,3
	3,0	3,1		

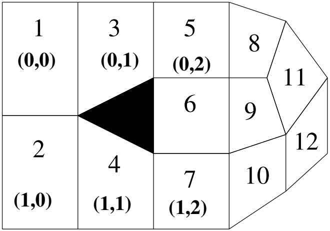
3) Decomposition into Quadrangulated Patches

Problems:



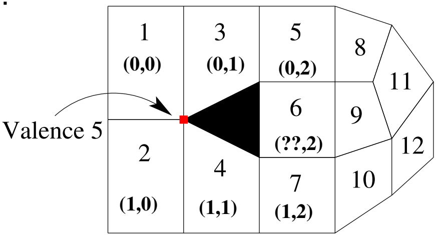
3) Decomposition into Quadrangulated Patches

Problems:



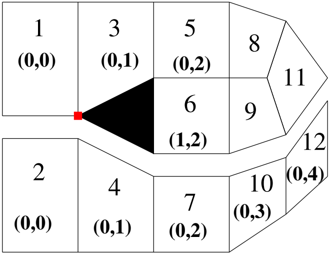
3) Decomposition into Quadrangulated Patches

Problems:



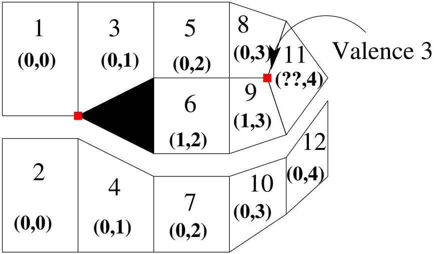
3) Decomposition into Quadrangulated Patches

Problems:



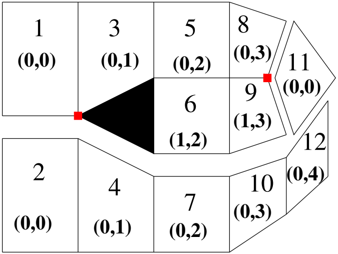
3) Decomposition into Quadrangulated Patches

Problems:



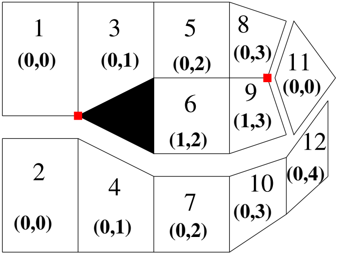
3) Decomposition into Quadrangulated Patches

Problems:

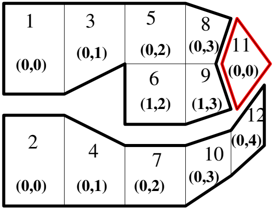


3) Decomposition into Quadrangulated Patches

Problems:

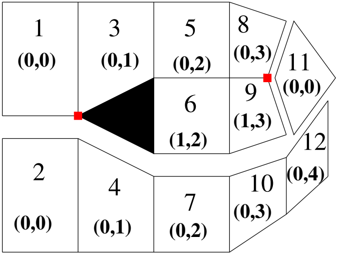


Decomposition into rectilinear polygons ⟹

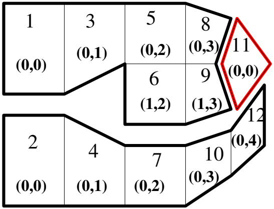


3) Decomposition into Quadrangulated Patches

Problems:



Decomposition into rectilinear polygons ⇒



Rectilinear polygons constituted by only one quad are not kept.

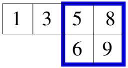
3) Decomposition into Quadrangulated Patches

3.2) The rectilinear polygons are decomposed into patches:
⇒ same number of rows for each column.

3) Decomposition into Quadrangulated Patches

3.2) The rectilinear polygons are decomposed into patches:
⇒ same number of rows for each column.

Iterative computation of the patches:

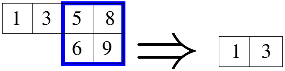


1	3	5	8
		6	9

3) Decomposition into Quadrangulated Patches

3.2) The rectilinear polygons are decomposed into patches:
⇒ same number of rows for each column.

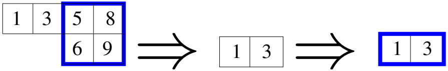
Iterative computation of the patches:



3) Decomposition into Quadrangulated Patches

3.2) The rectilinear polygons are decomposed into patches:
⇒ same number of rows for each column.

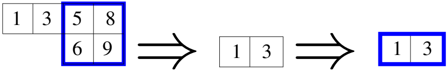
Iterative computation of the patches:



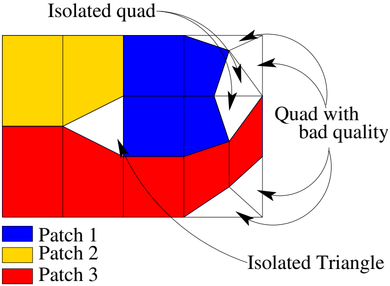
3) Decomposition into Quadrangulated Patches

3.2) The rectilinear polygons are decomposed into patches:
⇒ same number of rows for each column.

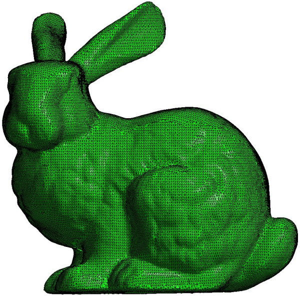
Iterative computation of the patches:



Final result:

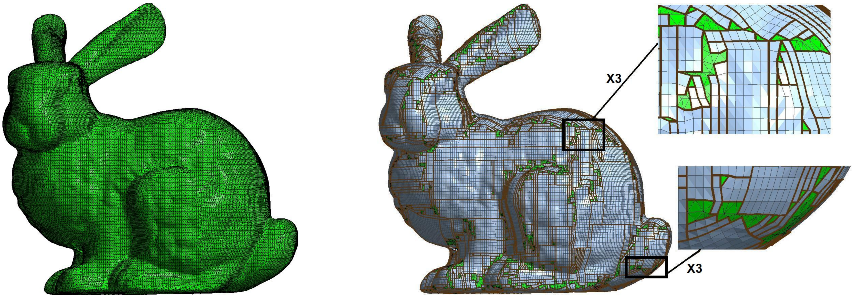


First Results



Stanford Bunny mesh: 69,451 triangles.

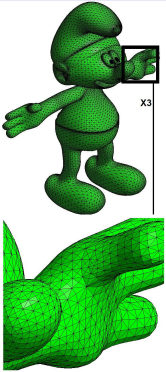
First Results



Stanford Bunny mesh: 69,451 triangles.

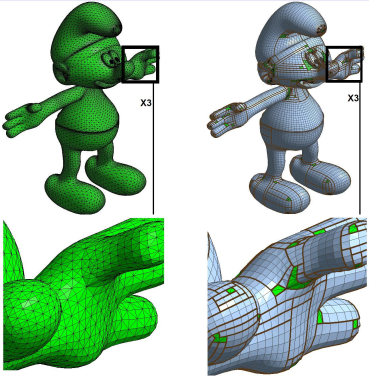
Q_{max}	ϕ_{min}	# patches	Covering	Time
$\frac{\pi}{2}$	$\frac{5\pi}{6}$	1,932	89.98%	4 min

Threshold Variations



Smurf mesh: 64,320 triangles.

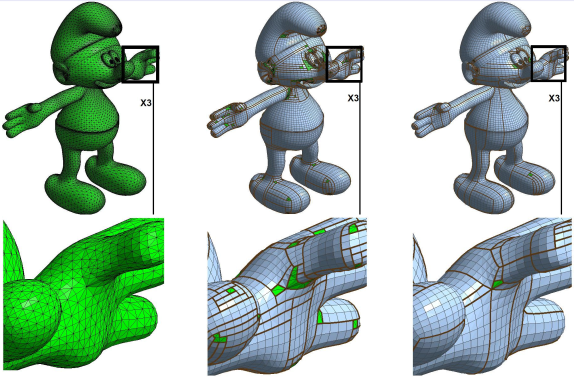
Threshold Variations



Smurf mesh: 64,320 triangles.

Q_{max}	ϕ_{min}	# patches	Covering	Time
$\frac{\pi}{2}$	$\frac{5\pi}{6}$	931	91.39%	2 min

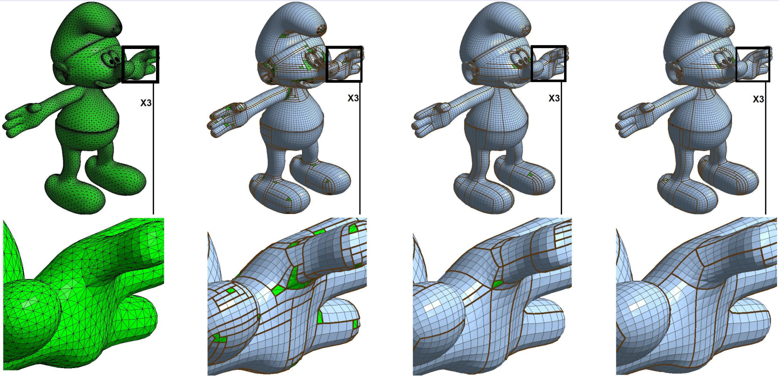
Threshold Variations



Smurf mesh: 64,320 triangles.

Q_{max}	ϕ_{min}	# patches	Covering	Time
$\frac{\pi}{2}$	$\frac{5\pi}{6}$	931	91.39%	2 min
π ↗	$\frac{5\pi}{6}$	519 ↘	98.52% ↗	4 min

Threshold Variations

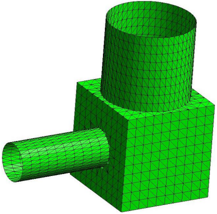
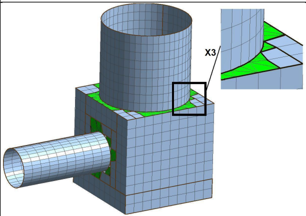


Smurf mesh: 64,320 triangles.

Q_{max}	ϕ_{min}	# patches	Covering	Time
$\frac{\pi}{2}$	$\frac{5\pi}{6}$	931	91.39%	2 min
π ↗	$\frac{5\pi}{6}$	519 ↘	98.52% ↗	4 min
2π ↗	2π ↗	502 ↘	98.56% ↗	5 min 30 sec

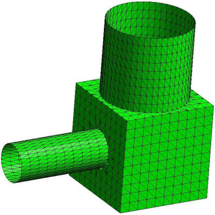
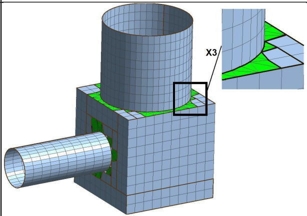
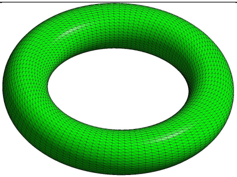
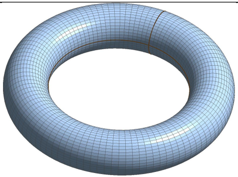
CAD Objects

$$Q_{max} = \frac{\pi}{2} \; / \; \phi_{min} = \frac{5\pi}{6}$$

	Initial mesh	Result	#triangles	#patches	Covering
CubeCylinders			2,608	21	95.47%

CAD Objects

$$Q_{max} = \frac{\pi}{2} \ / \ \phi_{min} = \frac{5\pi}{6}$$

	Initial mesh	Result	#triangles	#patches	Covering
CubeCylinders			2,608	21	95.47%
Torus			9,384	1	100%

Conclusion

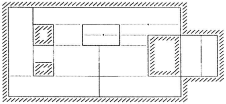
Our method:

- decomposes a triangular mesh into quadrangulated patches,
- has the particularity to use only the vertices and the edges of the triangular mesh,
- is implemented in the C4W framework.

Future Work

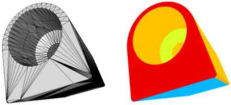
- Define other quality coefficients,
- Improve the quad propagation to minimize the number of isolated triangles,
- Optimize the rectilinear polygon search,

Soltan et al.
Minimum Dissection of a Rectilinear Polygon with Arbitrary Holes into Rectangles
Discrete and Computational Geometry 9(1):57-59 1993



- Use feature lines to guide the patch boundaries.

Lavoué et al.
A new CAD mesh segmentation method, based on curvature tensor analysis
Computer-Aided Design 37(10):975-987 2005



Thanks for your attention

QUESTIONS?

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

Roseline Bénière, G. Subsol, G. Gesquière, F. Le Breton and W. Puech,
Decomposition of a 3D triangular mesh into quadrangulated patches,
GRAPP, *Angers*, 2010