Multi-resolution Geometric Modeling

Leila De Floriani

- Motivations and Requirements
- A general framework for multi-resolution models
- Selective refinement queries
  - Selective refinement algorithms

12/10/14 1

Multi-resolutiom Level-Of-Detail (LOD)

Mesh at full resolution

- Generic term related to both the *size* of a mesh and its *fidelity* in representing a shape:
  - *Resolution:* size of triangles forming a mesh
  - *Accuracy:* difference between the mesh and the shape it represents (*approximation error*)
- It is usually true that:

  higher accuracy ⇔ higher resolution ⇔ larger mesh size

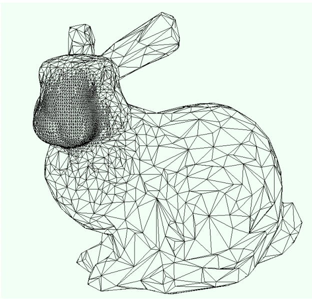
Uniform LOD

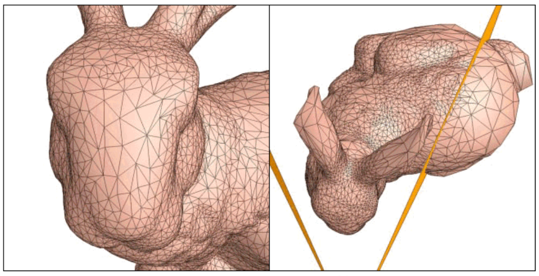
- *Uniform LOD:*
  - resolution/accuracy is constant over the whole mesh
- *Variable LOD:*
  - resolution/accuracy may be variable through space and time

Variable LOD

12/10/14 2

An example of variable LOD





(b) Front view and (c) Top view ($\tau = 0.1\%$; 10,528 faces)

[Image courtesy of H. Hoppe]

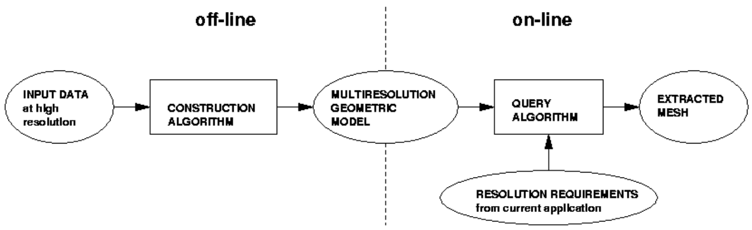
Why a Multi-resolution Model?

- Accuracy may vary in different parts of a shape
- *Accuracy* related to the *mesh resolution* and to the *mesh size*
- Need for *locally adapting* the resolution of a mesh in different parts of the object
- Two ways of tackling this problem:
  - *on-the-fly construction of a mesh at certain LOD* through simplification techniques:
    - Simplification algorithms with an accurate error evaluation are time consuming
    - It is difficult to generate variable-resolution representations
  - *use of a multi-resolution model*

Multi-resolution Model

Comprehensive structure built *off-line* which

- preprocesses and organizes a collection of alternative mesh representations of a spatial object
- can be efficiently queried according to parameters specified by an application task to extract adaptive meshes *on-line*



Requirements for a Multi-resolution Model

- Provide a *virtually continuous* range of meshes representing a shape (surface, scalar field) at different resolutions
- The *number of different meshes*, which can be extracted from the model, must not be fixed a priori, but be a *function of the data size*
- *Resolution* of any extracted mesh can be *variable* in different parts of the shape
- Support to *efficient query processing* (e.g., extraction of surface or scalar field representations in real time)
- *Size of the multi-resolution model* should not be much higher than the size of the mesh at full resolution

Variable-resolution Multi-resolution Models

- Variable-resolution models are continuous multi-resolution models from which it is possible to extract selectively refined meshes (where the portion of the mesh to be refined is defined at query time).
- Collection of mesh modifications describing small portions of a shape at different LODs
- Dependency relation that allows selecting subsets of modifications (according to application-dependent criteria)
- We introduce a framework for describing variable-resolution models: the *Multi-Tessellation (MT)*

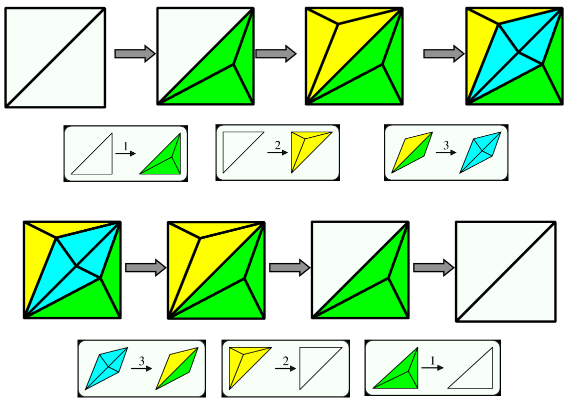
- This definition is:
  - independent of the properties of the modifications
  - dimension-independent

Ingredients for a multi-resolution model: modifications

In simplification - refinement or coarsening - an initial mesh undergoes a sequence of updates.

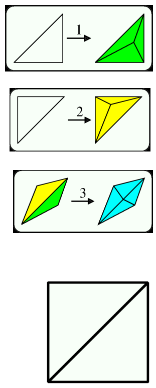
Refinement: from coarse to fine

Coarsening: from fine to coarse



Ingredients for a multi-resolution model: modifications

- We denote a modification M as a pair M=(M^-, M^+) where
  - M^- is the set of cells removed by M
  - M^+ is the set of cells inserted by M
- M is a *refinement modification* if Σ_2 has more cells than Σ_1; it is a *coarsening modification*, otherwise.
- Recall that modifications on a a simplicial complex that produce a simplicial complex as result are called *conforming*.
- We denote as M_0 a *d*-dimensional mesh, which is the mesh at the coarsest resolution (called the *base mesh*).
- {M_1, M_2,…,M_h}: a set of *d-dimensional conforming refinement modifications* M_i=(M_i^-, M_i^+) such that, for any *d*-simplex σ in M_i-, σ belongs either to M_0 or to exactly one M_j+ (with j≠i).

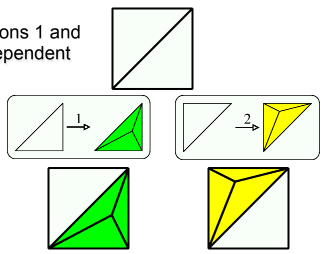


Base mesh

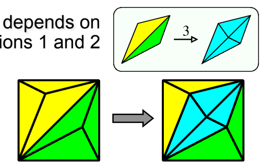
Ingredients for a multi-resolution model: dependency relation

A modification M_j *directly depends* on a modification M_i, with i≠j,
if and only if
M_j removes some *d*-simplex introduced by M_i

Modifications 1 and 2 are independent



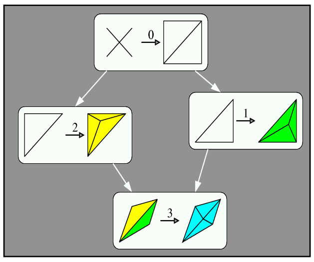
Modification 3 depends on both modifications 1 and 2



Multi-resolution model as a partially ordered set

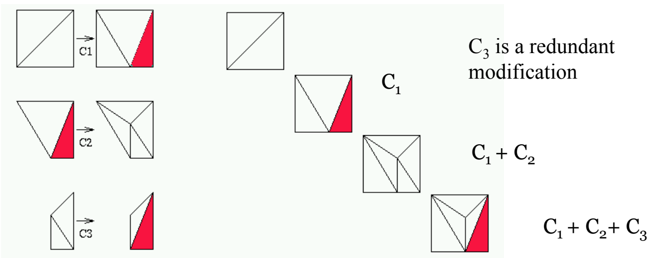
- If the transitive closure ∠ of the direct dependency relation is a *partial order*, then $\boldsymbol{M}=(\Sigma_0, \{M_1, M_2,\ldots,M_h\}, \angle)$ is an *LOD Model*

- An LOD model can be viewed as a Directed Acyclic Graph (DAG) in which:
  - *nodes* are modifications
  - *arcs* are direct dependency links



Multi-resolution model as a partially ordered set (cont'd)

- The direct dependency relation defines a partial order when all modifications in $\{M_1, M_2,\ldots,M_h\}$ are *non-redundant*
- A *non-redundant* modification with respect to a set of modification does not recreate *d*-simplexes eliminated by other modifications in the set



C_3 is a redundant modification

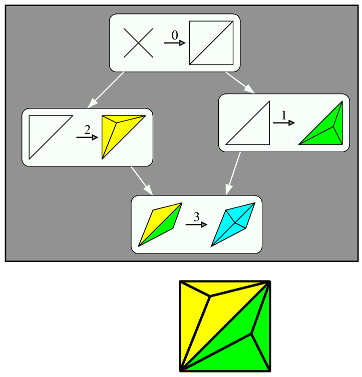
Closed subsets of modifications and extracted meshes

- A subset S of modifications in an LOD model is *closed* with respect to the partial orderif and only if:

  $M_j \in S \Rightarrow M_i \in S \ \forall \ M_i \angle M_j$

- A closed subset S and the base mesh Σ_0 define an *extracted mesh* Σ_S obtained from applying the modifications in S to Σ_0

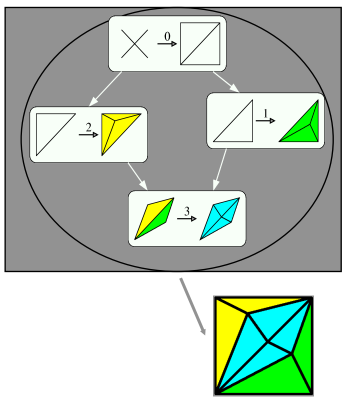
- Closed subsets correspond to *cuts* in the DAG



An extracted mesh: initial mesh + 1 and 2 (or 2 and 1)

Reference mesh

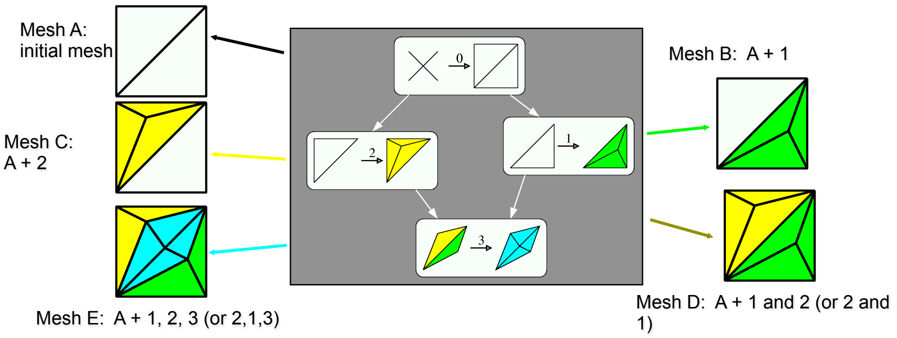
Any sequence of modifications in ***M*** corresponding to a total order extending ∠ produces the mesh at full resolution, called the *reference mesh*



Reference mesh

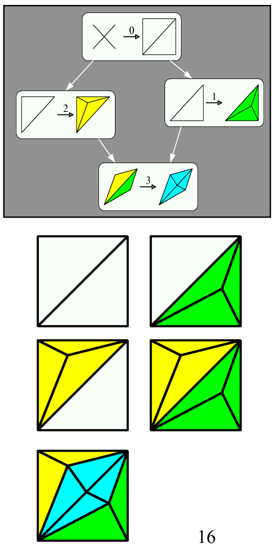
Closed subsets of modifications and extracted meshes

From a multi-resolution model, we can extract any mesh obtained from the coarsest mesh by applying any sequence of refinement updates compatible with the dependency relation



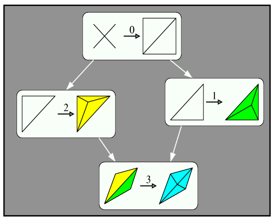
Properties of multi-resolution models

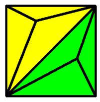
- This definition of LOD model is
  - independent of the specific modification
  - based on a "natural" notion of dependency
  - dimension-independent
- *Fundamental result*: The number of possible meshes that can be built from the *d*-simplexes in an LOD model ***M*** equals the number of closed sets in ***M.***
- Expressive power of an LOD model depends on the number of its closed sets.
- Intuitively, the number of closed sets will be high if the modifications are small, i.e., they involve a restricted number of simplexes. Examples of such modifications are vertex splits, or vertex insertions.
- *High expressive power* ⇒ capability of adapting to variable LOD ⇒ a large number of different meshes can be extracted from the model.



Properties of multi-resolution models (cont'd)

- We define the *growth factor* for a closed set S:
  - #*d*-simplexes in a closed set S / #*d*-simplexes in the corresponding extracted mesh Σ_S
- If the growth factor for a closed set is bounded by a constant, then the size of the extracted mesh Σ_S and of its corresponding closed set are of the same order of magnitude.
- If the above is true for every closed of a given LOD model ***M***, we say that ***M*** has a *linear growth.*
- This implies that extracting a mesh Σ_S has a complexity linear in the number of *d*-simplexes in S.
- It can be shown that if the number of *d*-simplexes deleted by any modification M in ***M*** is bounded by a constant, then ***M*** has a *linear growth.*
- The above is true for common modifications, like edge collapse/vertex split or vertex insertion/vertex removal



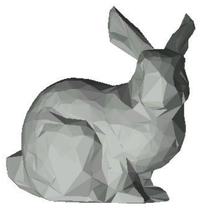


An extracted mesh: initial mesh + 1 and 2 (or 2 and 1): S= {0,1,2}

Selective refinement queries

Extract from an LOD model a mesh satisfying some application-dependent requirements based on LOD

uniform high resolution



uniform low resolution

high resolution just on the head

LOD Criterion for Selective Refinement

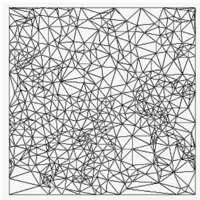
- Selective refinement is driven by local LOD parameters, usually by the approximation error (accuracy)
- Also, the size of the extracted mesh is often used as a criterion to stop retrieval (due to storage space constraints)
- *Approximation error* is usually associated with the *d*-simplexes of an LOD model.
- *LOD criterion* τ : Boolean function defined over the *d*-simplexes σ of an LOD model:
  $\tau(\sigma)$ = *true* if σ satisfies a specific local LOD parameter
- An extracted mesh Σ satisfies a given *LOD criterion* τ if and only if $\tau(\sigma)$ = true for all *d*-simplexes σ of Σ

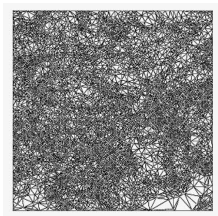
Examples of LOD criteria

- *Uniform LOD:*
  - $\tau(\sigma)$= *true* if the error associated with σ is less or equal to a constant threshold
- *Variable LOD:*
  - $\tau(\sigma)$= *true* if the error associated with σ is less or equal to the maximum over σ of a threshold function *f* defined at each point of the domain (e.g., a view-dependent function)
- *Special case of variable LOD for scalar fields:*
  - the threshold function *f* depends on the value of the field at each point of the domain (e.g, $f(p)\leq\varepsilon$ at a set of interesting field values, arbitrarily large otherwise)

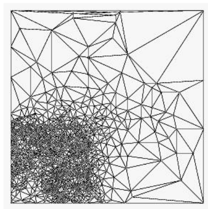
Examples of LOD criteria on a terrain

Uniform LODs





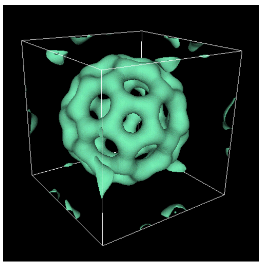
Variable LOD based on a region in the domain



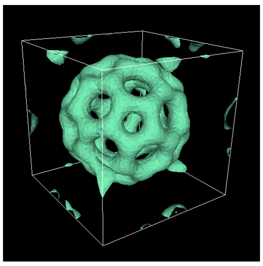
Uniform LOD

Input parameters: an accuracy threshold E

Error(t)≤E for every tetrahedron in the extracted mesh



Buckyball
12.5 million tetrahedra



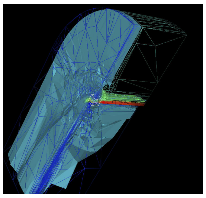
E = 5% of the field range
274,460 tetrahedra

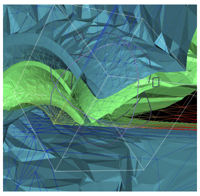
Variable LOD based on spatial location

Input parameters: a *Region of interest* (*ROI) R* and *two accuracy thresholds* E_1 and E_2 $(E_1<E_2)$:

- Error(t) ≤ E_1 for each tetrahedron t interesecting R
- Error(t) ≤ E_2 for any other tetrahedron

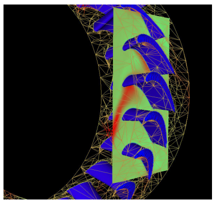
ROI = box
E=0.01% in the ROI
size = 1/3 of reference mesh





BluntFin: 222,528 tetrahedra

ROI =cross plane
E= 2% in the ROI
size = 7% of reference mesh

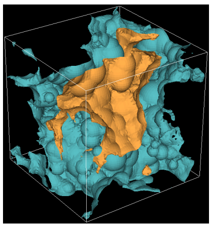


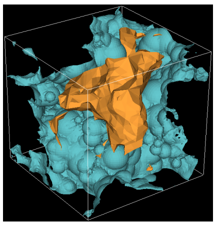
Turbine Blade 576,566 tetrahedra

Variable LOD based on field values

Input parameters: a *collection of field values FV* and *two accuracy thresholds* E_1 and E_2 $(E_1<E_2)$:

- Error(t) ≤ E_1 for every tetrahedron t interesecting the isosurfaces of values in FV
- Error(t) ≤ E_2 for any other tetrahedron





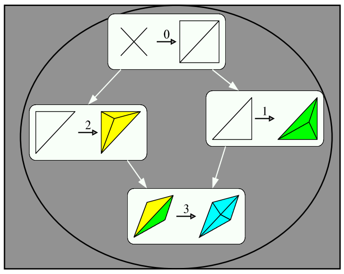
E = 0.1% along the blue isosurface

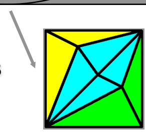
25% size of the mesh at uniform LOD with error = 0.1%

Plasma: 1,500,282 tetrahedra

Formulation of the general selective refinement query

- *General selective refinement query:* given an LOD model ***M*** and an LOD criterion τ
  extract from ***M*** the mesh of minimum size Σ_S satisfying τ





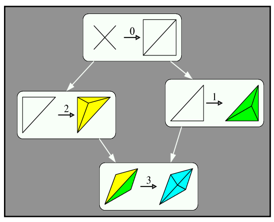
Modifications 1, 2 and 3 form a closed set

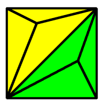
Extracted mesh

- It can be shown to be equivalent to:
  extract from ***M*** the mesh Σ_S associated with the closed subset S of minimum cardinality such that Σ_S satisfies τ.

Algorithms for selective refinement

- They are based on a traversal of the *partially ordered* set defining the LOD model
- They construct a closed set S of modifications
- The mesh associated with such closed set S is the solution to the query
- Two approaches:
  - *top-down*: it generates a mesh from scratch
  - *incremental*: it modifies a previously extracted mesh by locally refining it or by locally coarsening it





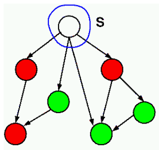
An extracted mesh: S= {0,1,2}

Top-down approach

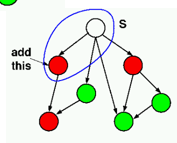
Input: LOD model and LOD criterion
Output: mesh satisfying LOD criterion

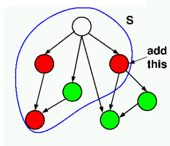
Initialization step: set S ⇐ empty
Generic step:

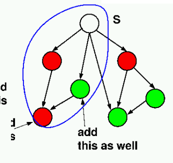
- Add a *red modification* u to S
- If set S *U* {u} is not closed, then recursively add all modifications preceding u on which u depends

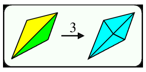


Red modifications are necessary to satisfy the LOD criterion









Modification 3 is a *red modification* if and only if
the yellow or the green triangle does not satisfy the LOD criterion

Modification 3 is a *green modification* if the yellow and the green triangles satisfy the LOD criterion

Top-down approach: implementation issues

- Efficiently implemented as a *depth-first traversal*
- Often implemented by using a *priority queue* (ordered according to decreasing error values): modifications having larger error values are performed first
  - the resulting algorithm is *interruptible* (i.e., intermediate steps produce approximations of the resulting mesh at intermediate resolutions)
  - it uses *extra storage* for encoding the priority queue (in our experiments on an LOD model based on tetrahedron bisection: *size of the queue* = 10-16% of the size of the extracted mesh)

Incremental approach

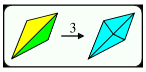
Input: LOD model, previously extracted mesh M, and new LOD criterion

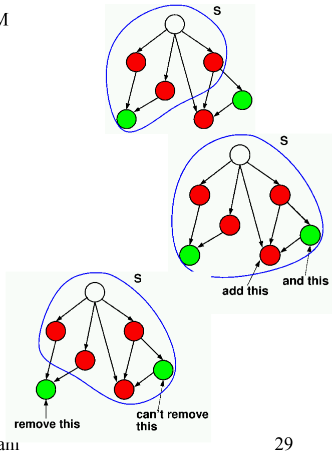
Red modifications are necessary to satisfy the new LOD criterion

Initialization step: set S ⇐ set of modifications corresponding to M

Generic step:

- Add *red modifications* to S, and *all* those modifications that are necessary to maintain S a closed set
- Remove *green modifications* if possible





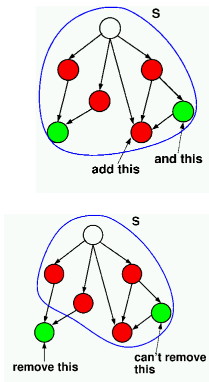
Modification 3 is a *red modification* if the yellow or the green triangle does not satisfy the LOD criterion

Modification 3 is a *green modification* if the yellow and the green triangles satisfy the LOD criterion

Incremental approach (cont'd)

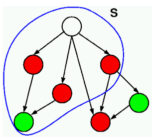
- Incremental algorithm perform both refinement and coarsening modifications, while top-down algorithm perform only refinements.
- A refinement modification *u* is *forced* by applying all modifications which are not in S and precede *u* in the partial order relation
- A coarsening modification is never "forced": it is applied only if;
  - it does not have direct descendants (like in this example)

  or
  - none of its descendants is necessary to satisfy the LOD criterion (they are all green modifications according to our convention)



Incremental approach: implementation issues

- *Direct algorithm:* all refinement modifications which do not satisfy the new LOD criterion are performed first, and the coarsening modifications whenever feasible.
- *Priority-based algorithm*: interleaving refinement and coarsening modifications by using
  - a *split queue* containing candidate refinement modifications
  - a *merge queue* containing candidate coarsening modifications
- A *priority-based* algorithm is *interruptible* and the size of the extracted mesh can be effectively used as a parameter.



Optimality Issues

- We can prove that the *top-down approach* produces the mesh of *minimum size* satisfying the LOD criterion.
- *The incremental* approach produces a mesh of *minimum size* provided that the LOD model ***M*** does not contain any modification $M=(\Sigma_1, \Sigma_2)$ such that all the d-simplexes of Σ_1 satisfy τ,but some of the d-simplexes in Σ_2 does not satisfy τ.
- For *error-based LOD criteria*: this means that the error must monotonically decreases at each modification. In the example below, the error associated with the blue triangles must be smaller than the one associated with the green and yellow triangles