Multi-Resolution Rendering of Architectural Models

S. Belblidia, *J.P. Perrin, †J.C. Paul ‡

July 3, 1995

Abstract

This paper presents a method for representing complex models with various levels of detail. It is based on a geometric simplification algorithm and is applied to a scene described by a rooted-tree structure. In order to control the resulting image quality and the computation time, we propose two algorithms which allow to choose one representation of the scene.

Keywords : Computer graphics, image synthesis, realistic rendering, real-time rendering, computer-aided architectural design.

1 Introduction

Since hardware graphics accelerations have been developed in recent years, real-time rendering up-to-date is still a challenge when large geometrical models have to be displayed. In order to render the complexity of the geometry for a given viewpoint (view pyramid), partitioning the geometrical model into cells and pre-processing the potential cell-eye visibility [TS91] can reduce the task of the z-buffer and other rendering algorithms which directly depend on the geometric complexity of the model to be rendered.

Another strategy is to simplify the geometric modelization of the objects. One way is to define objects with various levels of detail (LOD) as in section 3. This paper presents a hierarchical elision strategy based on a clustering and merging algorithm [RB93] which enables the simplification of objects and provides various levels of detail for representing complex models (sections 4 and 5).

*CRAI/School of Architecture of Nancy, Postal address : Chateau du Montet, 54506 Vandoeuvre les Nancy. email :salim@norman.crai.ciril.fr

†CRAI/School of Architecture of Nancy, Postal address : Chateau du Montet, 54506 Vandoeuvre les Nancy. email :perrin@crai.ciril.fr

‡Joint research project in Computer Graphic and Computer Vision of the CNRS/INRIA/INPL/University of Nancy, Postal address : INRIA - BP 101 F-54602 Villers-les-Nancy Cedex. email : Paul@loria.fr

Using current workstations, and when the geometric model to be rendered is very large, the image quality is maintained but the frame-rate per time unit decreases. In order to maintain a real-time frame-rate, flight simulators generate simplified views of the environment and therefore, the quality of image decreases.

We would like to have a rendering system which allows the user to control either *the image quality or the frame-rate rendering*. Moreover, we want to fully optimize the capabilities of the hardware graphics accelerations even when the geometric complexity changes dramatically.
We propose with this aim in view two algorithms (detailed in sections 5.1 and 5.2) :

- image quality-oriented algorithm : if the capability of the graphics hardware is not full or if the expected image quality is lower than a certain threshold then it follows that we increase the LOD.
- real-time-oriented algorithm : if the previsible cost of the current frame is greater than a certain threshold we would then decrease the LOD.

Finally, we present some results (section 6) and future works (section 7).

2 Previous work

One of the first works in this field was James Clark's paper which has presented the LOD representation as a possible application of hierarchical structures with hidden surface removal and clipping optimization[Cla76]. In a more recent paper, Frank Crow used three descriptions of a chair at different LODs and suggested to automate the simplification process [Cro82].

In fact, some papers proposed various elision strategies to simplify the representation of objects. Level of detail representations can be produced by surface fitting techniques from sampled data points [SBD86]. Some of these scanned points are eliminated [JZ91] or automatically redistibuted [Tur92] to generate accurate polygonal representations. The simplification methods for an existing set of polygons are often more adapted to regular meshes [SZL92][HDD+94] while certain others run relatively well on polyhedral models [RB93].

Actually, the use of these hierarchical and geometrical methods aim to represent a complex model with a multitude of resolutions [Bla87] and visualize complex environments with interactive frame rates [FS93][MS95]. We also have these aims in view, but we think that some applications can require to control both computation time and image quality and we wish to develop these two approaches.

3 Database structuration

Before modelling with any CAD software, we first propose to organize the model and subdivide it into components according to semantic criteria. Each

of these components can be decomposed into smaller ones, and even if it is modelled once, it can be used several times in various positions. During the geometry structuration process, we have respected the following rules :

- an object is either composed of components or described by a set of polygons.
- it must be a logical entity not a group of non-connected components.
- a simple object must be as convex as possible (this restriction facilitates the simplification process).
- it must be modelled in a local coordinate system that matches its geometrical axes.

The resulting structure is a tree that contains nodes representing complex objects andleaves which are simple objects as in [Cla76]. In the modeller we have used, each object is modelled and stored in a separate file. We first implemented a file converter to transform the root "dxf" file into our format and obtain information about the inclusion and relative placement of all objects. A second converter creates the geometry of each simple object from its polygonal description generated by the modeller.

The second step is to create for each simple object a simplified version which is generated by a simplification algorithm. Each complex object is assigned a simplified form which is the result of the same simplification algorithm applied to the set of its simplified component versions using a coarser threshold. We can then describe each complex object either by this approximation or develop it into its components and recursively choose one possibility for each object (Figure 1). Two particular cases can occur :

- the simplified description of an object can be the same as its full version. It is always represented with the same resolution.
- the simplified description of an object can be null. The object does not appear when represented in its simple form.

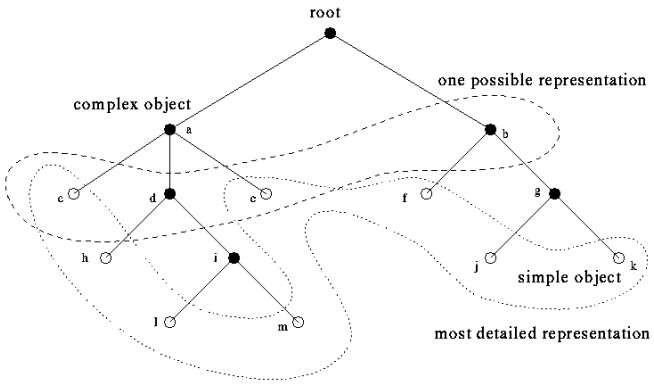


Figure 1: Hierarchical organization of a model

Above are two representations of the model. In the most detailed one, all the leaves of the model are used and the description include the objects c, h, l, m, e, f, j and k. One of the possible coarse representations include the objects c, S(d), e and S(b) where S(b) is the result of the simplification algorithm applied to S(b)+S(g) and so on. We have choosen not to include in the figure the simplified versions of the leaves in order to maintain design clarity.

4 Elision process

Instead of modelling objects at various LODs, we have implemented an automatic simplification algorithm based on vertex clustering and merging as in [RB93]. A value is assigned to each vertex according to its perceptual importance. The vertices are grouped into clusters then merged into a unique vertex representative of the cluster that is deterined as the vertex with the maximum weight or the center of mass of all cluster vertices. The simplification rate depends on the cluster size. The polygons are triangulated before moving the vertices to avoid non-planar deformations.

However, some problems have appeared while using the method so we have adapted it to solve for some specific cases :

- Object bounding box : On some facetted objects, the weights of the vertices computed with geometrical criteria can be equal. Since the vertices in each cluster are merged into the center of mass, this causes an important reduction of the object volume and some continuity problems can appear between contiguous objects.

  We have implemented a new merging procedure for the border clusters in order to conserve the bounding box limits. The representative vertex of the cluster is chosen to be the closest to the bounding box frontier. The merging procedure is modified inside the clusters which touch one or more bounding box frontier. In order to consider each frontier separately, the edge and angle clusters are subdivided respectively into two and three sub-clusters. This technique runs better on convex objects (Figure 2-b).

- Cluster interfaces : Some of the object vertices can stand on a cluster interface (symmetry axe, for example). When they are added to one of the two candidate clusters, this can introduce dissymetries after the merging procedure.

  We can consider the interfaces as separate clusters and recursively, their edges and angles also (Figure 2-c).

These modifications reduce the simplification rate but allow for a better visual result. To evaluate these two criteria, we propose to compare after each simplification process two values : 1) the average and/or maximum displacement of the vertices in regard to the object size 2) the percentage of eliminated vertices. We have reducedthe number of polygons of some objects up to one half without loosing their general aspect. Some examples of these

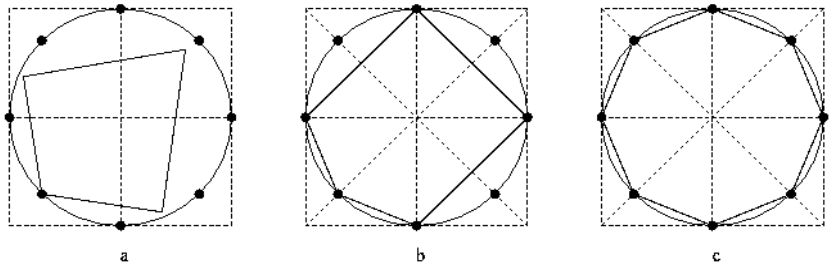


Figure 2: A 2D example of the simplification result : a) simple form algorithm, b) conserving the bounding box, c) separating the cluster interfaces.

objects and the corresponding values are shown in section 6.

We have adapted this algorithm to our hierarchical structure. First, each simple object is simplified using a specific cluster size. For each complex object, the simplification process is applied to the collection of the simplified versions of its components with coarser criteria. The resulting set of polygons can be used to represent the complex object instead of displaying its components.

5 Managing the structure

The goal of this final stage is to produce for a given viewpoint the right representation. In the case of a hierarchical structure, we must determine the tree configuration by developing some nodes and not others. Let us consider a complex object O composed of two simple objects O1 and O2 (Figure 3) . Their respective simplified versions are S(O), S(O1) and S(O2). There are five possible configurations of this model :

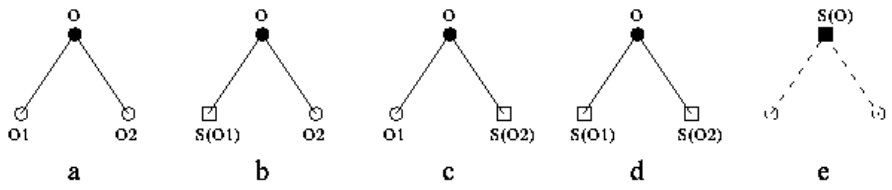


Figure 3: Multiple representations of a model

Funkhouser and Séquin use heuristics to evaluate for a given object the benefit and the cost of its representation in a given level of detail [FS93]. We have used these two notions applied to two kinds of applications (image quality or real-time) :

5.1 Image quality context

In this context, the main rule for level of detail selection is :

if predicted_image_quality of object O ¡ quality_threshold then
use a better representation of O

The distance from the object to the camera is the main criterion that we consider (other criteria will be developed in future works) to anticipate the image quality. Each object is assigned a distance threshold d^{th} that defines its spherical proximity area. We suppose that the approximation of an object is negligible if the projection of the removed details is smaller than the pixel size *ps*. Therefore the distance d^{th} has a straightforward relationship with the size of the cell previously used in the elision process (Figure 4). The following evaluation of this distance for a pixel in the center of the screen is available for other pixels :

$$d^{th} = (1/ps) * f * vd; \quad (1)$$

d^{th} : distance camera-object
f : focal distance
ps : pixel size
vd : maximum vertex displacement = cell diagonal

For instance, an object simplified with a cluster size of 10 cm is displayed in its simplified form if the camera (f=50mm, ps=0.18mm) is farther than 72 m (d¿d^{th}), and in its full description if the same camera is closer (d¡d^{th}).

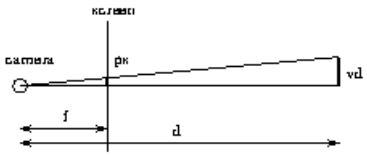


Figure 4: Object-camera distance and removed details size

For a given object, we will determine whether the camera is inside or outside its proximity area to choose between one of its two representations. For instance, there are five possible configurations of the model described in figure 3. They correspond to the five regions which result from the the proximity areas of O, O1 and O2 (Figure 5). The following procedure is a recursive traversal algorithm beginning at the tree root which determines the right configuration :

representation (object o, camera c)
rep_o = null;

```
if inside_proximity_area ( c, o )
   if is_a_simple_object ( o )
      rep_o = full_description ( o );
   else
      for each sub-object so of o
         add_to ( rep_o, representation ( so, c ) );
else
   rep_o = simplified_form ( o );
return rep_o;
```

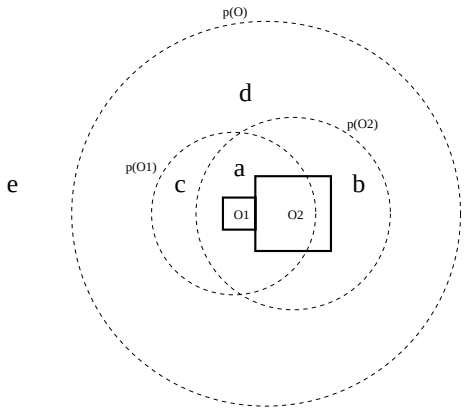


Figure 5: Regions of the space where the camera generates different configurations of the model

If the camera (with a 50mm focal distance) stands in the region c, the model will be represented as O1+S(O2). Moreover, if the model is viewed from the region e, its representation will be S(O).

5.2 Real-time context

The main rule in this case is :

```
if predicted_rendering_time of object O ¿ target_time
   use a lower representation of O
```

We suppose that the task complexity is proportional to the number of polygons and pixels for a given z-buffer capability (in terms of storage memory space and computation time). The algorithm then determines for each visibe object whether an improved representation than that of its simplified form is "computable" within the target time. Otherwise, the simple version of the object is displayed and its components are ignored.

```
representation ( object o, camera c, target_time t)
   rep_o = null;
   if required_time ( better_description(c) ) ¡ t then
      if is_a_simple_object ( o ) then
         rep_o = full_description ( o );
      else
```

```
        for each sub-object so of o
            sub_t = sub_object_target_time ( o, so, t );
            add_to ( rep_o, representation ( so, c, sub_t ) );
    else if required_time ( simplified_form(c) ) ¡ t then
        rep_o = simplified_form ( o );
    return rep_o;
```

This algorithm applied to the object O runs this way :

```
if ( S(O1)+S(O2) ) is computable then
    if O1 is computable then display O1;
    else display S(O1);
    if O2 is computable then display O2;
    else display S(O2);
else display S(O);
```

6 Discussion and results

We have choosen an architectural illustration and we present here a LOD model of the "Porte Héré" arch of "Place Stanislas" in Nancy (Figure 6).

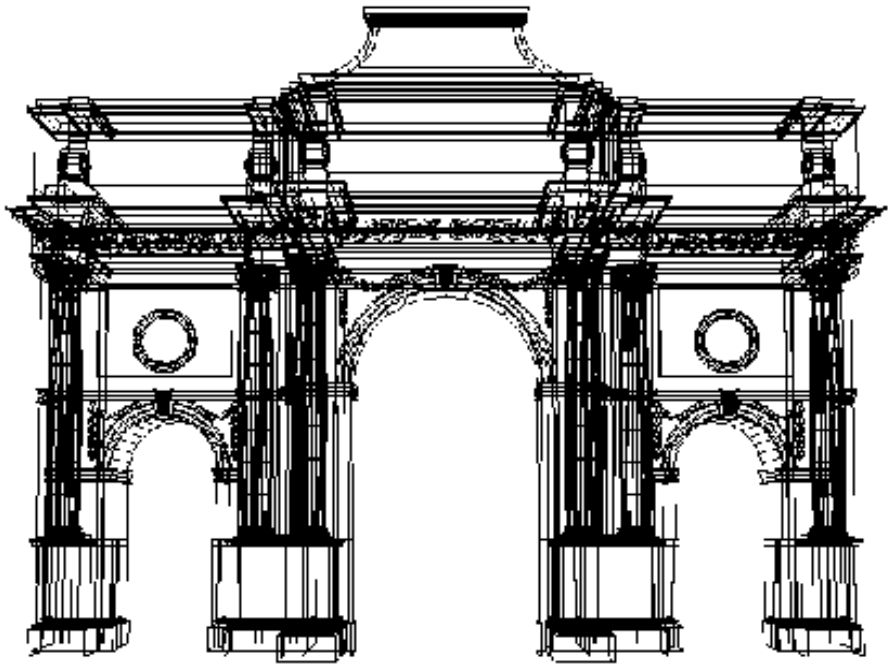


Figure 6: "Porte Héré" arch in Nancy

We have subdivided it into three main parts : a central arch and two lateral archs. Each of them is then organize as a superposition of horizontal levels. The most detailed object is the column capital. The database contains nearly 25000 polygons and is composed of 38 simple objects structured in a tree with 4 levels of depth (Figure 7). Theoretically, there are more than one million different representations of this arch.

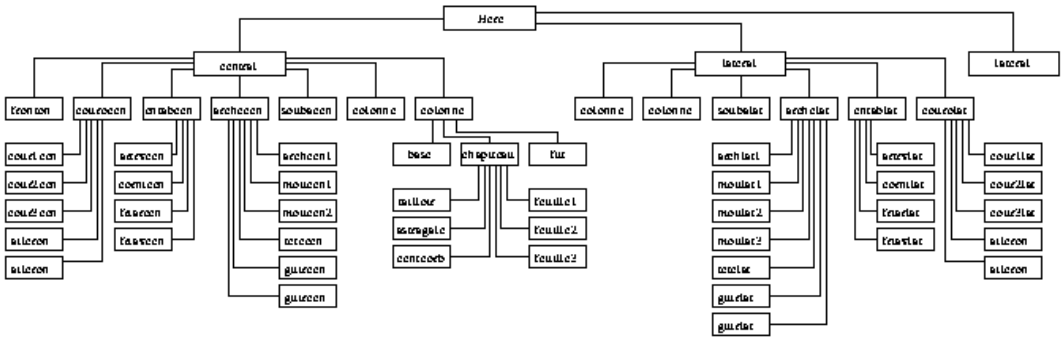


Figure 7: Organization of "Porte Héré" database

The simplification process applied to the simple object "fut" (column shaft) gives the following results using the same subdivision properties with different options (Figure 8) :

We have operated the same tests on the complex object "colonne" (column). Below are the resulting sub-tree organization and certain possible representations of the column (Figure 9) . We notice that the objects "feuille2" and "feuille3" have no simplified version and do not appear when used in their simple form.

7 Future works

We are working on the implementation of the two algorithms described in sections 5.1 and 5.2 in order to evaluate : 1) the computation time needed to compute an image with given quality criteria 2) the image quality resulting from a fixed target frame-time.

Since determining for each point of view the right configuration of the scene is cumbersome, we are working on a new method to subdivide the 3D-space in regions where the model has the same aspect.

We will also incorporate other criteria for the level of detail selection such as the proximity of an object to the center of the screen or its importance in a given application.

Finally, after testing our algorithms on architectural monuments with many ornemental details, we will operate on indoor architectural spaces like

offices or industrial buildings.

Authors

Salim Belblidia is a PhD student at the School of Architecture of Nancy University. He received an architecture diploma from Algiers School of Architecture in 1990 and a master degree in Computer Science from Saint-Etienne University in 1992.
S.Belblidia can be contacted at :
CRAI/Ecole d'Architecture de Nancy, Chteau du Montet, 54506 Vandoeuvre-ls-Nancy France.
E-mail : salim@norman.crai.ciril.fr

Jean-Pierre Perrin is the Director of the Research Center in Architecture and Engineering . He is also a professor at the School of Architecture of Nancy.
J.P Perrin can be contacted at :
CRAI/Ecole d'Architecture de Nancy, Chteau du Montet, 54506 Vandoeuvre-ls-Nancy France.
E-mail : perrin@crai.ciril.fr

Jean-Claude Paul is the Scientific Director of the joint research project in Computer Graphic and Computer Vision of the CNRS/INRIA/INPL/Nancy University.
J.C. Paul can be contacted at :
INRIA - Campus Scientifique BP 101, F-54602 Villers-lès-Nancy Cedex (France).
E-mail : paul@loria.fr

References

[Bla87] E. H. Blake. A metric for computing adaptive detail in animated scenes using object oriented programming. In G. Marechal, editor, *Proceeding of the International Conference Eurographics'87*, pages 295–307, August 1987.

[Cla76] J. H. Clark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 19(10):547–554, October 1976.

[Cro82] F. C. Crow. A more flexible image generation. In *Computer Graphics (Siggraph'82 proc.)*, volume 16, pages 9–18, July 1982.

[FS93] T. A. Funkhouser and C. H. Séquin. Adaptative display algorithm for interactive frame rates during visualization of complex virtual environments. In J. T. Kajiya, editor, *Computer Graphics (Siggraph'93 proc.), Annual Conference Series*, pages 247–254, August 1993.

[HDD+94] H. Hoppe, T. D. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In A. S. Glassner, editor, *Computer Graphics (Siggraph'94 proc.), Annual Conference Series*, pages 295–302, July 1994.

[JZ91] M. J. De Haemer Jr. and M. J. Zyda. Simplification of objects rendered by polygonal approximations. *Computers and Graphics*, 15(2):175–184, 1991.

[MS95] P. W. C. Maciel and P. Shirley. Visual navigation of large environments using textured clusters. In *Proceedings 1995 Symposium on Interactive 3D Graphics*, pages 95–102, June 1995.

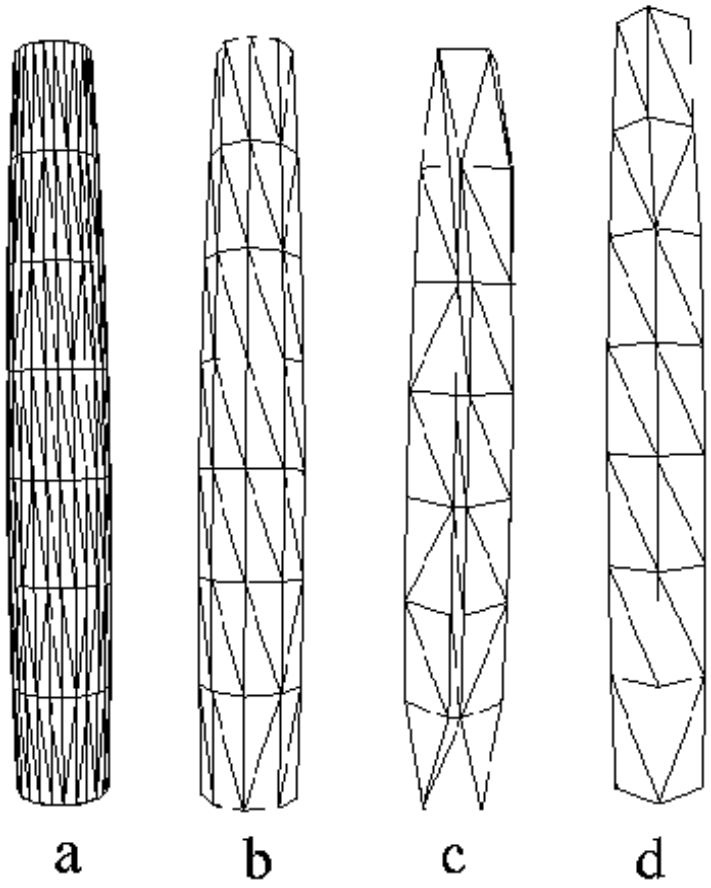
[RB93] J. Rossignac and P. Borrel. Multi-resolution 3d approximations for rendering complex scenes. In *Conference on Geometric Modeling in Computer Graphics*, pages 453–465, June 1993.

[SBD86] F. J. M. Schmitt, B. A. Barsky, and W. H. Du. An adaptive subdivision method for surface-fitting from sampled data. In D. C. Evans and R. J. Athay, editors, *Computer Graphics (Siggraph'86 proc.)*, volume 20, pages 179–188, August 1986.

[SZL92] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. In E. E. Catmull, editor, *Computer Graphics (Siggraph'92 proc.)*, volume 26, pages 65–70, July 1992.

[TS91] S. J. Teller and C. H. Séquin. Visibility preprocessing for interactive walkthroughs. In T. W. Sederberg, editor, *Computer Graphics (Siggraph'91 proc.)*, volume 25, pages 61–69, July 1991.

[Tur92] G. Turk. Re-tiling polygonal surfaces. In E. E. Catmull, editor, *Computer Graphics (Siggraph'92 proc.)*, volume 26, pages 55–64, July 1992.



% of eliminated vertices	0 %	67 %	71 %	83 %
Number of triangles	380	124	108	60
Average vertex displacement	0 %	1.7 %	5.3 %	4.6 %

Figure 8: Results of the simplification process :a) original model, b) conserving the clusters interfaces, c) conserving the bounding box, d) maximum simplification

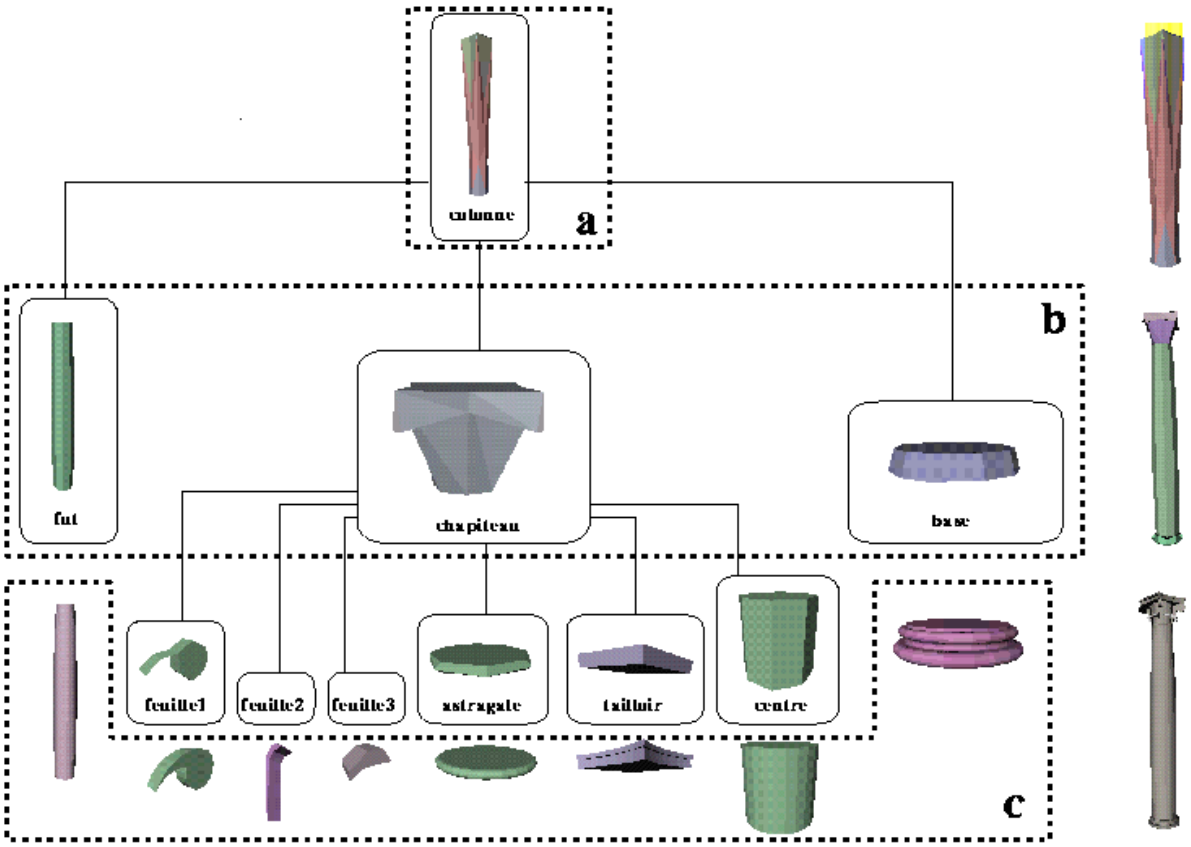


Figure 9: The sub-tree organization and some possible representations of the column