Octree Simplification of Polyhedral Solids

Dolors Ayala
Pere Brunet

Report LSI-95-1-R

Octree Simplificaction of Polyhedral Solids

Dolors Ayala Pere Brunet
Universitat Politecnica de Catalunya
Barcelona, Spain
e-mail: [ayala, brunet]@lsi.upc.es

January 12, 1995

Abstract

Multiresolution representations are becoming more and more important for the visualization, operation and interrogation of very complex scenes. In this paper, an automatic simplification algorithm for general two-manifold polyhedra is presented. The algorithm is based on the generation of an intermediate octree representation of the object and produces a set of valid two-manifold polyhedral approximations of the initial solid ensuring precise distance approximations between the surface of the initial object and the surface of the resulting solids. A restricted non-iterative version of the general algorithm is proposed and implemented for the particular case of isothetic polyhedra. In this case the proposed scheme is specially simple and generates a whole family of isothetic simplifications.

Keywords: geometric modelling, realtime interaction and visualization, multiresolution models, geometric simplification, octree representations.

1 Introduction

The need for multiresolution representations was already stated in 1976. In [5] J. Clark pointed out that objects which cover a small area in the screen could be rendered in a simplified version and he proposed a hierarchical model supporting several representations of an object. F. Crow [6] also proposed a data structure describing objects with several levels of detail. Not only objects that are very small or far away from the observer must be rendered in a simplified version but also objects that are moving quickly across the screen and which appear blurred or can be seen for a short amount of time [11].

Multiresolution representation approaches lead to the geometry simplification problem and make clear the need for automatic simplification methods. The goal is to have algorithms for the automatic generation of multiresolution models of an initial object or assembly. These algorithms must be able to simplify geometric models and generate new approximate models involving a lower number of geometric entities.

In this context, the problem which is addressed in this paper can be precisely defined as follows:

Given a two-manifold polyhedral solid S with a total of $face\#(S)$ planar faces and a set of decreasing tolerances $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_l$ with $\varepsilon_k < \varepsilon_{k-1}$, generate a set of two-manifold polyhedral solids $s_1, s_2, \ldots, s_{l-1}, s_l$ such that,

$$dist(surf(S), surf(s_k)) \leq \varepsilon_k$$

with a increasing number of faces, $face\#(s_k) \geq face\#(s_{k-1})$. The final set of solids s_k form a multiresolution approximating set to S, s_l being the closest approximation to S and s_1 being the coarser and simplest approximation.

The corresponding 2D problem - 2D polyline simplification - is a well known problem in 2D computational geometry. This problem can be formulated in a direct or inverse way. The direct or $min\#$ problem is the problem of minimizing the number of polygon vertices given an error ϵ. The inverse or $min\epsilon$ problem minimizes the error ϵ given the final number of polygon vertices [10]. Solutions exist for both the direct and inverse problem: strip trees [2], optimal approximation [13], Delaunay pyramid [7], flintstones [25], etc.

Some attemtps to extend these results to 3D have also been proposed. The prism tree [19] extends the strip tree and the flintstones have the corresponding 3D approach [25] [24].

The prism tree method represents polyhedra as ternary trees of truncated piramides representing enclosing boxes of parts of the polyhedron. The method is well suited for complex 3D real objects digitized as polyhedra with thousands of faces. Flintstones are also a hierarchical scheme based on the approximation of closed polyhedra without holes. Here, the bounding volumes are balls, the so called flintstones. This approximation approach is not simple, specifically at the first levels in which the polyhedron may self-intersect.

In [22], an algorithm is presented for reducing, in a specified rate, the number of triangles of a surface obtained with the marching cubes method. Turk [23] approximates a polygonal surface by a triangular one with a specified number of vertices. In [15] a similar method is presented; it provides a provable bound of the approximation error and it is applicable to arbitrary polyhedral models but it may produce self-intersecting polyhedra similarly to the

flintstones method.

Some methods directly simplify data obtained from 3D digitizing cameras minimizing the number of polygons when reconstructing the surface from these data points [8].

Other approaches [12] simplify an initial triangular mesh obtained from a scattered set of data points minimizing an energy function. These methods produce a new triangular mesh with a small number of vertices which approximates the data, while dealing with sharp edges and corners.

Finally, in [20] a method for approximating arbitrary polyhedra is presented. It is also based on the triangulation of the object surface. The method is very time efficient but may produce non-regular and non-solid objects with isolated faces, edges and points.

Except the two last papers, most of these approaches are well suited for the automatic simplification of models representing smooth curved surfaces, but are poorly suited for polyhedral models with well defined corners and sharp edges. Moreover, all of them deal only with triangular faces.

In the present paper, an algorithm for the generation of the multiresolution approximating set of an inicial two-manifold polyhedron S is proposed. The presented approach accepts arbitrary polyhedra and no triangulation process is needed: faces of the initial and simplified polyhedra may have any number of vertices. The algorithm can be used in a preprocess step in applications where visualization of very complex scenes is required, or in applications involving interference detection, interrogation or complex solid operations. In the visualization case, it is interesting to have a multiresolution representation of the objects in the scene, the objects far from the observer being represented at a coarser level (with simplification of the features that would be projected onto one or very few pixels). In the interference detection or interrogation cases, interference or computations within a certain tolerance can be computed using simplified representations s_k of the solid S.

Next section includes the basic definitions and outlines the proposed algorithm. Section 3 presents the general simplification algorithm, while an specific algorithm for the simplification of isothetic polyhedra is described in section 4. Finally, practical examples are presented and discussed in section 5.

2 Definitions and Proposed Approach

Before presenting the proposed approach, let us introduce the concepts of classical octrees (CO), maximal division classical octrees (MDCO) and terminal grey nodes (TG):

Classical octree (CO) : It is a representation based on an adaptive hierarchical space division [21]. The octree is an octal tree representing the subdivision process, and contains White (W) nodes, Black (B) nodes and Grey (G) nodes.

Maximal division classical octree (MDCO) [3]: It is an octree representation containing White nodes (W) which correspond to cubic regions completely outside the object, Black nodes (B) completely inside the object, Grey nodes (G) that contain part of the surface of the object and are therefore subdivided, and Terminal Grey nodes (TG) that

contain part of the surface of the object but are at the deepest level of the tree and consequently are not subdivided.

Terminal grey nodes (TG) : They are grey nodes of the minimum allowed size. They obviously appear as terminal nodes at deepest level of the tree. Being grey nodes, they contain part of the surface of the object. On the other hand, given any point P of the object surface, it always exists a TG node n such that P is inside n. The eight vertices $v_i, i = 1..8$ of the cube associated with a TG node can be either white or black. White vertices are those outside s_k, whereas black vertices are known to be inside the final solid s_k. Vertex colors can be computed from the initial solid S or through node neighborhoods in the octree: vertices in contact with white nodes are white, whereas those located on black nodes of the MDCO are black.

Due to the precise definition of TG nodes (they contain three faces of the corresponding cube but are open at the rest of faces), it can be concluded that the set of TG nodes in a MDCO is a 6-connected set forming a (6, 26) 3D picture [16].

The colors of the eigth vertices of TG nodes allow 256 different configurations that can be grouped through complementary and rotational simetries onto the 14 equivalence classes shown in figure 1 [17] [26].

In this context, the proposed approach can be summarized as follows,

Step 1 : Build the MDCO representation c_l of the initial solid S. We will assume in the following that this octree has a depth of l levels, the edge size of the universe being 2^{l-1} times the edge size of the minimum size cubic nodes.

Step 2 : Obtain a multiresolution set c_1, c_2, ..., c_l of MDCO representations of the solid S. For every k, c_k is obtained by deleting level $k+1$ nodes in the octree c_{k+1} and considering grey nodes at level k as new TG nodes of the octree c_k. Finally, c_1 is a simple 1-level octree containing only the grey root node.

Step 3 : For every c_k, obtain a two-manifold polyhedral solid s_k such that the k-level MDCO representation of s_k is c_k. Obviously, c_k is a valid MDCO representation also for s_k, s_{k+1}, ..., s_l.

After step 3, a set of approximating solids s_1, ..., s_l has been obtained. Noting as ϵ the length of the main diagonal of c_l TG nodes, both the surface of S and the surface of s_k are completely contained in the set of TG nodes of c_k. Therefore,

$$dist(surf(S), surf(s_k)) \leq \epsilon \cdot 2^{l-k}$$

On the other hand, the second requirement in the problem definition,

$$face\#(s_k) \geq face\#(s_{k-1}), \forall k$$

will appear as a direct consequence of the proposed algorithm, as we will see in section 3.

Step 1 and step 2 are straightforward and need no further comments. MDCO generation is based on a simultaneous space subdivision and clipping of the planar faces of the initial solid S.

Next section is devoted to step 3 analysis. In this step, BRep elements (faces, edges and vertices) of solid s_k must be inferred from the information in the MDCO c_k and topological incidence relations must be generated.

3 Simplification Algorithm

3.1 Further definitions

Before presenting the proposed algorithm, this subsection introduces the concepts of closed MDCO, regular TG nodes, two-manifold TG nodes, two-manifold MDCO representations, TG maps and polyhedral TG_ maps. TG_ maps and Polyhedral TG_ maps will be the basis of the proposed algorithmn presented in section 3.2: the generation of the approximating solid involves the estimation of an starting TG_ map which is subsequently improved through modify operations until a final polyhedral TG_ map is obtained.

Closed MDCO : A MDCO is closed if every halfline starting from any black node or any black vertex of any TG node intersects at least one TG node of the MDCO.

Regular TG : A regular TG is a TG node having both white and black vertices. There is only one class corresponding to a non-regular node (class 0 in figure 1, with all vertices having the same color).

Two-manifold TG : Two-manifold TG nodes are regular TG nodes with a configuration of black and white vertices that is compatible with the surface of the solid within the node having a single connected component.

In order to classify classes from 1 to 13 into two-manifold and non-manifold TG nodes, we will follow Lorensen approach [17]. In his approach, nodes are traversed by a triangular surface; this results in classes 1, 2, 5, 8 and 11 being traversed by one sheet and the remaining by more than one sheet. In our case, nodes must be traversed by a boundary element of a polyhedron and, thus, only classes 4 and 13 are traversed by more than one sheet. For the remaining classes, a face, edge or vertex topology interior to the node can be found [1]. Nevertheless, resulting topologies for classes 7, 10 and 12 are concave vertices with eight or nine faces while all classes other than those have simpler topologies involving a maximum of four faces. In the present approach these three classes are not considered two-manifold in order to avoid complex topologies. In fact it can be shown that, under this assumption, the number of final faces traversing the TG node is reduced to a maximum of three faces [1].

Column A of table 1 shows the resulting classification of the 14 equivalence classes.

Two-manifold MDCO : A two-manifold MDCO is a closed MDCO such that every TG node is a two-manifold TG node.

class	A	B
0	non-regular	
1	2-manifold	in, on, join
2	2-manifold	in, on, join
3	2-manifold	on, join
4	non-manifold	
5	2-manifold	on, join
6	2-manifold	join
7	non-manifold	
8	2-manifold	in, on, join
9	2-manifold	in, on, join
10	non-manifold	
11	2-manifold	join
12	non-manifold	
13	non-manifold	

Table 1: A) Classification of nodes into two-manifold, non-manifold and regular. B) Classification of two-manifold nodes in types **in**, **on** and **join**.

It can be shown that non-manifold nodes can be subdivided into eight two-manifold sons [1]. Consequently, it is always possible to process the inicial MDCO obtaining a second two-manifold MDCO with TG nodes of two consecutive sizes.

TG_map : A TG_map is a classification of every TG node in a MDCO into **in**, **on** or **join** types. This classification is related to a certain underlying set of regions $r_1 \ldots r_n$ defined in the set of TG nodes of the MDCO. A region r_i is a connected set of 6-connected TG nodes. A TG node will be classified as **in** if the node is inside the region r_i, that is, the node is $in(r_i)$. It will be classified as **on** in the case it belongs to the boundary of a certain region i being $on(r_i)$, and there exists a value j such that the node is also $on(r_j)$. Finally, it will be classified as **join** if there exist at least three values i, j, k such that the node is $on(r_i)$, $on(r_j)$ and $on(r_k)$.

Column B of table 1 shows the classification of two-manifold nodes in types **in**, **on** or **join**.

In fact, a TG_ map uniquely determines the type of every TG node in the MDCO, solving all indeterminations in table 1-B. TG_maps are however not unique: for a given MDCO, every possible classification of its TG nodes into connected regions produces a different TG_map.

Polyhedral TG map : A TG_map is a polyhedral TG_map if there exists a polyhedron such that,

- The polyhedron has a vertex inside every **join** TG node of the map.
- For every pair of **join** TG nodes which are 6-connected through a chain of **on** TG nodes - a join_chain -, there exists an edge of the polyhedron which is interior to

(and only to) the **on** and **join** nodes of the chain.

- For every region in the TG_map, there exists a planar face of the polyhedron which separates all white vertices from all black vertices for every **in** TG node in the region.

3.2 General algorithm

Let us assume that a closed MDCO c_k is given. On the other hand, we will also assume that the generation of the approximating solids $s_1, \ldots, s_k, \ldots, s_l$ is performed in a sequential way, so that the solid s_{k-1} is known at the moment the MDCO c_k is processed. A general description of the proposed algorithm is as follows:

```
compute_vertex_colors
obtain_two_manifold_MDCO
generate_starting (TG_map)
while not polyhedral (TG_map) do
 modify (TG_map)
endwhile
```

After the *compute_vertex_colors* procedure, for all TG nodes in the MDCO the color of every vertex $v_i, i = 1 \ldots 8$ of the corresponding cube has been computed. Vertex colors can be obtained using node neigborhood information in the MDCO when possible, or alternatively using the point in polyhedron test in the initial solid S.

The *obtain_two_manifold_MDCO* procedure guarantees that every TG node in the output tree is a two-manifold TG node.

The rest of this section describes the *polyhedral* test and the *generate_starting* and *modify* routines in the most general case of a polyhedral solid S.

3.3 Polyhedral test for a TG_map

The test for a polyhedral TG_map first obtains the explicit topology of the TG_map, and then detects if there exists a polyhedron with the obtained topology and with faces completely inside TG nodes of the TG_map. More specifically, the generation of the explicit topology of the TG_map is performed in the following way,

- A vertex of the polyhedron BRep is generated for every **join** node in the TG_map.
- An edge of the polyhedron BRep is generated for every join_chain connecting pairs of **join** nodes in a 6-connected way, and for every pair of neighbour **join** nodes (void join_chains).
- A face of the polyhedron BRep is generated for every edge loop so that its corresponding **on** and **join** nodes in the TG_map enclose either a region containing only **in** nodes or a null region.

At this moment, the TG_map is a polyhedral one if there exists a polyhedron with a vertex within each **join** TG node of the TG_map and such that,

- For every polyhedron edge and for every node - either **on** or **join** - of its corresponding join_chain in the TG_map, this node is 6-connected to another node of same join-chain through the node face which is intersected by the polyhedron edge.
- Every polyhedron face is planar and interpolates the polyhedron vertices belonging to the face.
- White vertices of **in** TG nodes in the TG_map region corresponding to a polyhedron face, are separated from Black vertices of the same nodes by the polyhedron face.

In this context, the algorithm for testing polyhedral TG_maps must solve an optimization problem: the best location of the polyhedron vertices within **join** nodes must be found, in order to minimize a weighted sum of the magnitudes dv_{ij}, nv_{ij} and nc_i,

$$sp = \alpha_1 \sum_{faces} \sum_{vertices} dv_{ij} + \alpha_2 \sum_{faces} \sum_{innodes} nv_{ij} + \alpha_3 \sum_{join-chains} nc_i$$

where,

- dv_{ij} is the distance from the weighted plane approximating the vertices of the polyhedron face i, to the polyhedron vertex j. Vertex j must be one of the vertices of the face i.
- nv_{ij} is the number of white/black vertices of TG **in** nodes that are not well separated by the weighted plane approximating the vertices of the polyhedron face i.
- nc_i is the number of faces of the nodes of the join_chain i which are intersected by the polyhedron edge i, and which are not 6-connected to nodes of the same chain.

In the case the optimization reaches a null minimum, it can be guaranteed that the tested TG_map is a polyhedral one:

```
function polyhedral (TG_map) return boolean
 initialize_polyhedron_vertex_locations
 compute_weighted_sum (sp)
 repeat
  change_vertex_locations_within_join_TG_nodes
  compute_weighted_sum (sp1)
  spa:= sp
  sp := sp1
 until sp ≥ spa
 return (sp < epsilon )
endfunction
```

3.4 Generate_starting procedure

This routine obtains a first TG_map for c_k, that will be afterwards refined in successive iterations with the modify routine. The algorithm is trivial in the case c_1; in other cases c_k, it uses the polyhedral TG_map obtained in the processing of the previous MDCO c_{k-1} as an initial guess, by computing a initial TG_map with the previous topology. In this way, the algorithm explodes a coherence based on the intrinsic hierarchical nature of the MDCO. More specifically,

A **join** TG node in the TG_map of c_k is generated for every **join** TG node in the polyhedral TG_map of c_{k-1}. This **join** node is searched among the TG son nodes of the **join** TG node in c_{k-1}. In the case where more than one TG son node exists, any of them can be choosen as a candidate.

In a second step, **join** TG nodes in c_k are grouped onto regions, following the same topology of the polyhedral TG_map of c_{k-1}.

This method, besides using the coherence inferred by the hierarchical structure of the MDCO, guarantees the second postcondition of our simplification problem: as it will be shown in the next section, the algorithm of the modify routine can only increase the number of regions of the TG_map, and therefore,

$$face\#(s_k) \geq face\#(s_{k-1}), \forall k$$

3.5 Modify procedure

This routine works in two steps. In the first one, the magnitude $\alpha_3 \sum_{join\_chains} nc_i$ is decreased by changing the location of **join** nodes in the TG_map while keeping its topology. This location change is performed by marking a previous **join** node as non **join** and marking a previous **in** or **on** node as the new **join** node. The candidate **join** node for the location changing is the node with the greatest sum of the nc_i values of the join_chains converging to it. Any location change of a **join** node induces a recomputation of the **on** nodes of the join_chains emerging from it.

In the second step, the magnitudes $\sum_{vertices} dv_{ij}$ and $\sum_{innodes} nv_{ij}$ for every region in the TG_map are analysed. The TG_map topology in the region with the highest weighted value of these magnitudes is modified by increasing the number of regions. This is performed through a split region or a create ring operation. New **join** and **on** TG nodes appear and are marked accordingly.

3.6 Particular cases

In the extreme case of a MDCO with a single TG node - the root node -, the two first preprocess steps of the algorithm convert it to a set of eight half-size TG nodes, each of them having seven white vertices and a black one. In this case, it is easy to see that the proposed algorithm generates the BRep of a cube with 6 orthogonal faces.

For the restricted class of isothetic polyhedra, a simple algorithm based on the direct classification of TG nodes can be derived. It will be presented and discussed in the next section.

4 Simplification Algorithm for Isothetic Polyhedra

An isothetic polyhedron is a polyhedron with all its edges and faces oriented in three orthogonal directions [14]. In this section, a simple version of the general simplification algorithm is presented for the particular case of isothetic polyhedra.

4.1 TG node types

The fourteen equivalence classes of the TG nodes configurations are completely defined in the isothetic case.

Figure 2 shows the fourteen equivalence classes and corresponding types.

There are six classes corresponding to two-manifold nodes: one Face-type node, one Edge-type node and four Vertex-type nodes corresponding to the four possible isothetic vertices [14]. One class corresponds to the non-regular node and the remainding seven classes correspond to non-manifold nodes.

4.2 Node subdivision

In order to obtain a two-manifold MDCO in the preprocess part of the simplification algorithm, our approach consists on subdividing every non-manifold node into its eight sons in such a way that all resulting nodes are two-manifold TG nodes. The resulting MDCO will contain TG nodes of two consecutive sizes.

We can decide without loss of generality, that the isothetic boundary traversing TG nodes intersects its edges - those having end vertices of different color - following a black proximity criterium: the intersection point is located at the edge at a 0.25 distance from its black vertex and at a 0.75 distance from the white one. Following this, it can be easily shown [1] that all non-manifold nodes can be subdivided into eight two-manifold sons. Furthermore, applying this black proximity criterium implies subdividing also class 11 node, thus considering it as a non-manifold node. Table 2 shows the resulting subdivisions.

Non-regular nodes are also subdivided into eight two-manifold sons in a straightforward way. In this case, subdivision is performed by taking into account the node neighborhoods. Figure 3 shows the three possible neighborhoods of non-regular nodes together with the resulting surface [1]. It can be observed that in fact, non-regular TG nodes represent non-regular parts of the surface of the final solid at the present approximation level. The reconstruction scheme presented in figure 3 is necessary in order to ensure the distance approximation between the surface of the initial object and the surface of its simplification, in the regions with non-regular TG nodes.

4.3 Algorithm.

In this section, the application of the general algorithm introduced in section 3.2 to this particular isothetic case is presented.

The *compute_vertex_colors* procedure is the general case routine.

class	0	1	2	3	4	5	6	7
3	W	W	W	W	1	W	W	1
$\bar{3}$	8	5	5	8	W	2	2	W
4	W	W	W	1	1	W	W	W
$\bar{4}$	5	5	5	W	W	5	5	5
6	W	W	W	1	2	2	W	W
$\bar{6}$	5	2	5	W	W	W	5	2
7	W	W	W	1	W	1	1	W
11	5	W	2	W	5	2	W	W
12	5	2	W	W	2	W	W	1
13	W	1	1	W	1	W	W	1

Table 2: Node subdivision for the isothetic case (the numbering of sons is the same of the numbering of vertices shown in figure1).

Obtain_two_manifold_MDCO processes all non-manifold and non-regular nodes (section 4.2), obtaining a MDCO with two-manifold TG nodes of two consecutive sizes. Subdivision of non-manifold nodes is done by traversing the MDCO and processing such nodes in an independent way. Processing non-regular nodes, instead, needs neighbour information and must use techniques for neighbour-finding[21]

The *generate_starting* procedure generates an initial TG_map. In this particular isothetic case, two-manifold nodes can be unambiguosly classified in types **in** (class 8), **on** (class 2) and **join** (classes 1, 5 and 9) which correspon to a face, an edge and three vertices (see section 4.1) Therefore, the TG_map is unique and defines completely the geometry and topology of the embedded polyhedron. Thus, it can be stated that this initial TG_map is already a polyhedral TG_map and hence, in this particular case, the general algorithm becomes sequential, the while loop being superfluous.

Consequently, the *generate_starting* procedure includes, as its final step, the reconstruction of the polyhedron. This reconstruction algorithm is similar to the Extended Octree to BRep conversion algorithm and is based on the fact that all the required information is in Vertex-type nodes [4], [18]. The algorithm traverses the MDCO and processes all Vertex-type TG nodes (classes 1, 5 and 9), inferring the embedded geometry by using the black proximity criterion i.e. choosig the boundary to pass at a 0.25 distance from black vertices of the node, for the non-subdivided nodes, and at a 0.5, for the subdivided ones.

It is important to remark that the isothetic polyhedron reconstruction algorithm presented here does not produce holes as some isosurface generation methods do. Figure 4 shows the continuous reconstruction of the presented method in the case in which the marching cubes algorithm produces a hole in the surface [9].

5 Examples

Figure 5 shows two examples of an isothetic scene. This scene corresponds to the "Pla Cerda" architectonic project of Barcelona. The initial MDCO of figure 5 b), c_l, has been obtained, and several successive MDCO simplifications have been generated by simply deleting successive levels in the initial MDCO. Finally, figure 6 shows several successive simplifications of the example obtained from the presented isothetic algorithm.

6 Conclusions and future work

An automatic simplification algorithm for general two-manifold polyhedra has been presented. The algorithm is based on the generation of an intermediate MDCO representation of the solid and produces a set of two-manifold polyhedral approximations of the initial solid ensuring precise distance approximations between the surface of the initial object and the surface of the resulting solids. In the general case the algorithm is computationally expensive, involving the generation and modification of TG_maps together with testing their closeness to polyhedral_TG_maps. Nevertheless, the time complexity of the algorithm can be usually be afforded, provided that the multiresolution simplification is performed in a - batch - preprocess prior to their use in interactive applications. A remarcable advantage of the proposed algorithm is that all geometric algorithms are concentrated in the polyhedral test for TG_maps, once the MDCO of the initial object has been generated. This is specially important in terms of robustness considerations.

A restricted non-iterative version of the general algorithm has been proposed and implemented for the particular case of isothetic polyhedra. In this case the proposed scheme is specially simple and generates a whole family of isothetic simplifications.

Future work includes the study and implementation of specific algorithms for other specific classes of initial solids S, the robustness analysis of the geometric algorithms and the implementation of the general algorithm, including more efficient algorithms for the *Generate_starting* and *Modify* routines.

Besides polyhedron simplification, the presented algorithm can be used in solid operations in solid modelers: step 3 of the proposed algorithm can be used, for instance, for robust boolean operations among a large set of solids. By converting the initial solids onto their MDCO representations, performing boolean operations in the discrete octree space, and back-converting the resulting MDCO onto a final BRep, an error bounded by eps on the surface of the final solid can be guaranteed. However, standard BRep algorithms involve robustness problems and cannot ensure this bounded precission, mainly when a sequence of boolean operations has to be performed - boundary evaluation of CSG trees, for instance -.

7 Acknowledgements

The authors would like to thank Isabel Navazo, Frits Post and Alvar Vinacua for the fruitful ideas and discussions held during the development of the present work and to Robert Juan and Anna Puig for their helpfull comments. They also would like to thank Carlos Andujar

for his support in the implementation of the algorithms and Txatxo Sabater of the UPC Department of Architectonic Composition for supplying the "Pla Cerda" scene.

References

[1] D. Ayala and P. Brunet. MDCO to BRep conversion algorithm. Technical report, UPC-Llenguatges i Sistemes Informàtics, 1995.

[2] D.H. Ballard. Strip trees: a hierarchical representation for curves. *Communications ACM*, 24(5):1 - 28, 1981.

[3] P. Brunet, R. Juan, I. Navazo, A. Puig, J. Sole, and D. Tost. *Scientific Visualization. Advances and challenges*, chapter Modeling and visualization through data compression, pages 157 - 170. Academic Press, 1988.

[4] P. Brunet and I. Navazo. Geometric modeling using exact octree representation of polyhedral objects. In C.E. Vandoni, editor, *Eurographics'85*, pages 159 - 169. Elsevier Science Publishers B.V., 1985.

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

[6] F.C. Crow. A more flexible image generation environement. *Computer Graphics*, 16(3):9 - 18, 1982.

[7] L. DeFloriani. A pyramidal data structure for triangle based surface description. *IEEE Computer Graphics and Applications*, 9(2):67 - 80, 1989.

[8] M.J. DeHaemer and M.J. Zyda. Simplification of objects rendered by polygonal approximations. *Computers and Graphics*, 15(2):175 - 184, 1991.

[9] M.J. Düurst. Additional reference to marching cubes. *Computer Graphics*, 22(2), 1988.

[10] D. Eu and G. Toussaint. On approximating polygonal curves in two and three dimensions. Technical Report SOCS 92.15, McGill University, 1992.

[11] T.A. Funkhouser, C.H. Sequin, and S.J. Teller. Management of large amounts of data in interactive building walkthroughs. In *ACM SIGGRAPH, Computer Graphics 1992 Symposium on interactive 3D graphics*, pages 11 - 20, 1992.

[12] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *SIGGRAPH'93, Computer Graphics Proceedings, Anual Conference Series*, pages 19 - 26, 1993.

[13] H. Imai and M. Iri. Polygonal approximation of a curve, formulations and algorithms. In G.T. Toussaint, editor, *Computational Morphology*, pages 71 - 86. Elsevier Science Publishers B.V. (North Holland), 1988.

[14] R. Juan-Arinyo. Isothetic polyhedra and monotone boolean formulae. Technical Report LSI-94-3-R, UPC-Llenguatges i Sistemes Informatics, 1994.

[15] A.D. Kalvin and R.H. Taylor. Superfaces: Polyhedral approximation with bounded error. Technical Report RC 19135 (82286), IBM Research Division, 1993.

[16] T. Y. Kong and A. Rosenfeld. Digital topology: Introduction and survey. *Computer Vision, Graphics and Image Processing*, 48:357 – 393, 1989.

[17] W.E. Lorensen and H.E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):44 – 50, 1987.

[18] I. Navazo, D. Ayala, and P. Brunet. A geometric modeller based on the exact octree representation of polyhedra. *Computer Graphics Forum*, 5(2):91 – 104, 1986.

[19] J. Ponce and O. Faugeras. An object centered hierarchical representation for 3D objects: the prism tree. *Computer Vision, Graphics and Image Processing*, 38:1 – 28, 1987.

[20] J. Rossignac and P. Borrel. Multiresolution 3D aproximations for rendering complex scenes. In B. Falcidieno and T.L. Kunii, editors, *Modeling in Computed Graphics*, pages 455 – 465. Springer-Verlag, 1993.

[21] H. Samet. *Applications of Spatial Data Structures.* Addison-Wesley, 1989.

[22] W.J. Schroeder, J.A. Zarge, and W.E. Lorensen. Decimation of triangle meshes. *Computer graphics*, 26(2):65 – 70, 1992.

[23] G. Turk. Re-tiling polygonal surfaces. *Computer Graphics*, 26(2):55 – 64, 1992.

[24] R. C. Veltkamp. 3D computational morphology. In R.J. Hubbold and R. Juan, editors, *Eurographics'93*, pages 115 – 127. Blackwell Publishers, 1993.

[25] R.C. Veltkamp. *Closed Object Boundaries from Scattered Points.* PhD thesis, Erasmus University Rotterdam, 1992.

[26] J. Wilhelms and A. Van Gelder. Topological considerations in isosurface generation. *Computer Graphics*, 24(5):79 – 86, 1990.

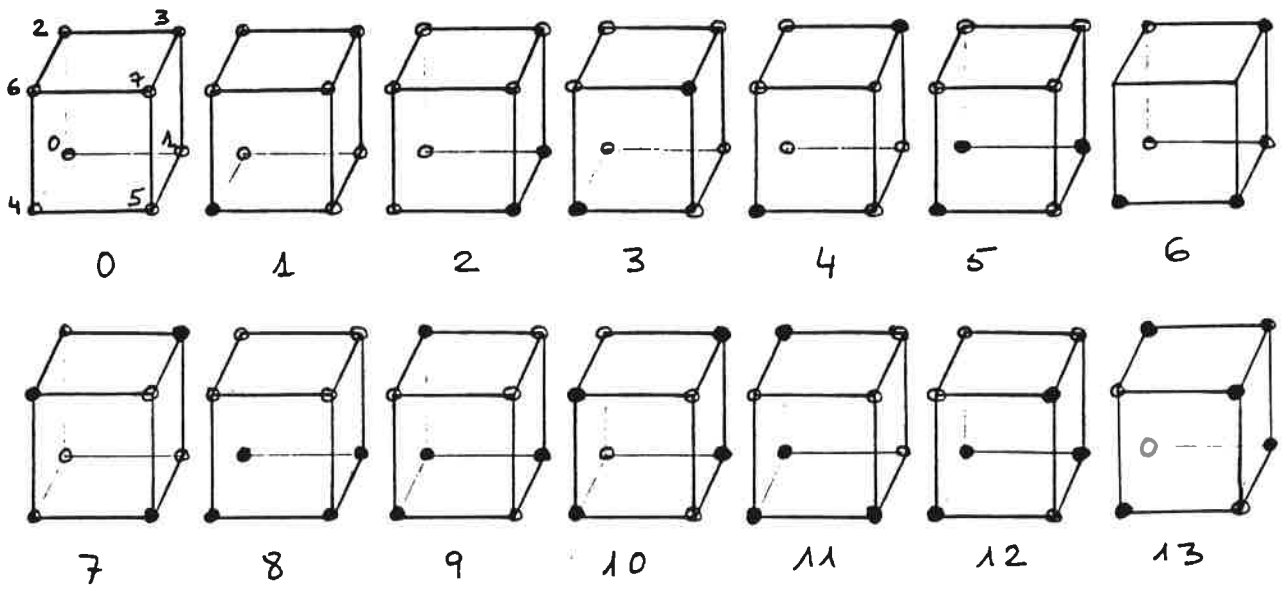

Figure 1: The 14 equivalence classes. (Class 0 includes the selected numbering of vertices)

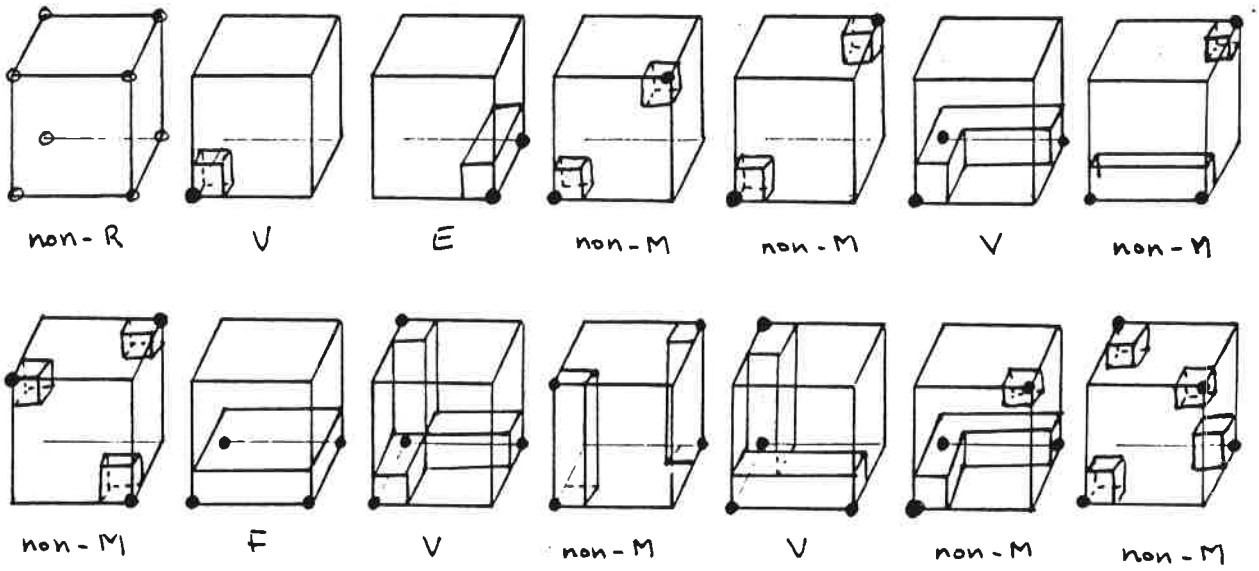

Figure 2: The 14 equivalence classes for the isothetic case

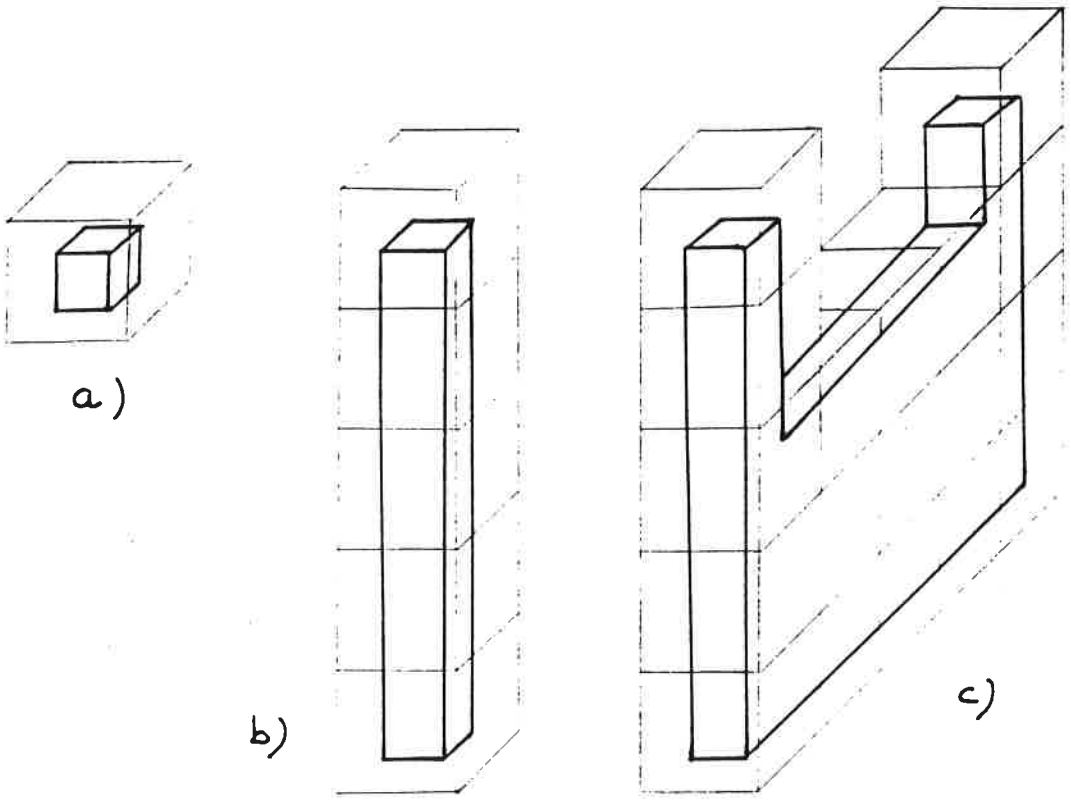

Figure 3: Neighborhoods for non-regular nodes. a) an isolated node; b) a column of nodes; c) a slice of nodes.

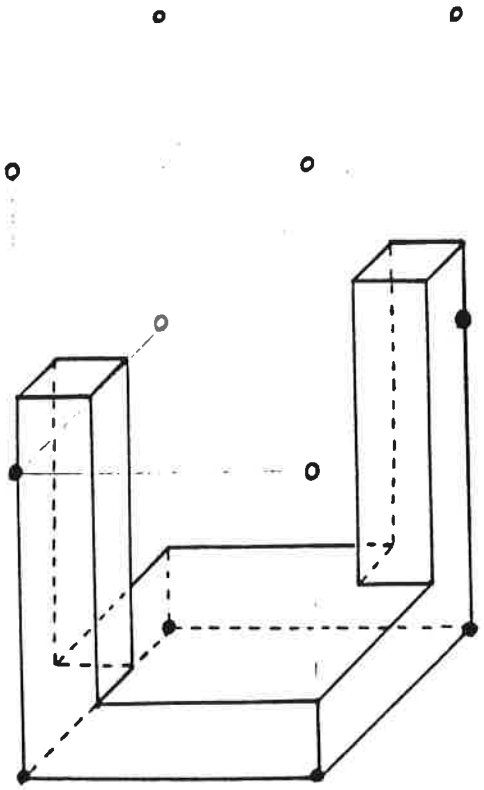

Figure 4: Polyhedron surface generated from two neighbour TG nodes of class 3 (see figure 2)

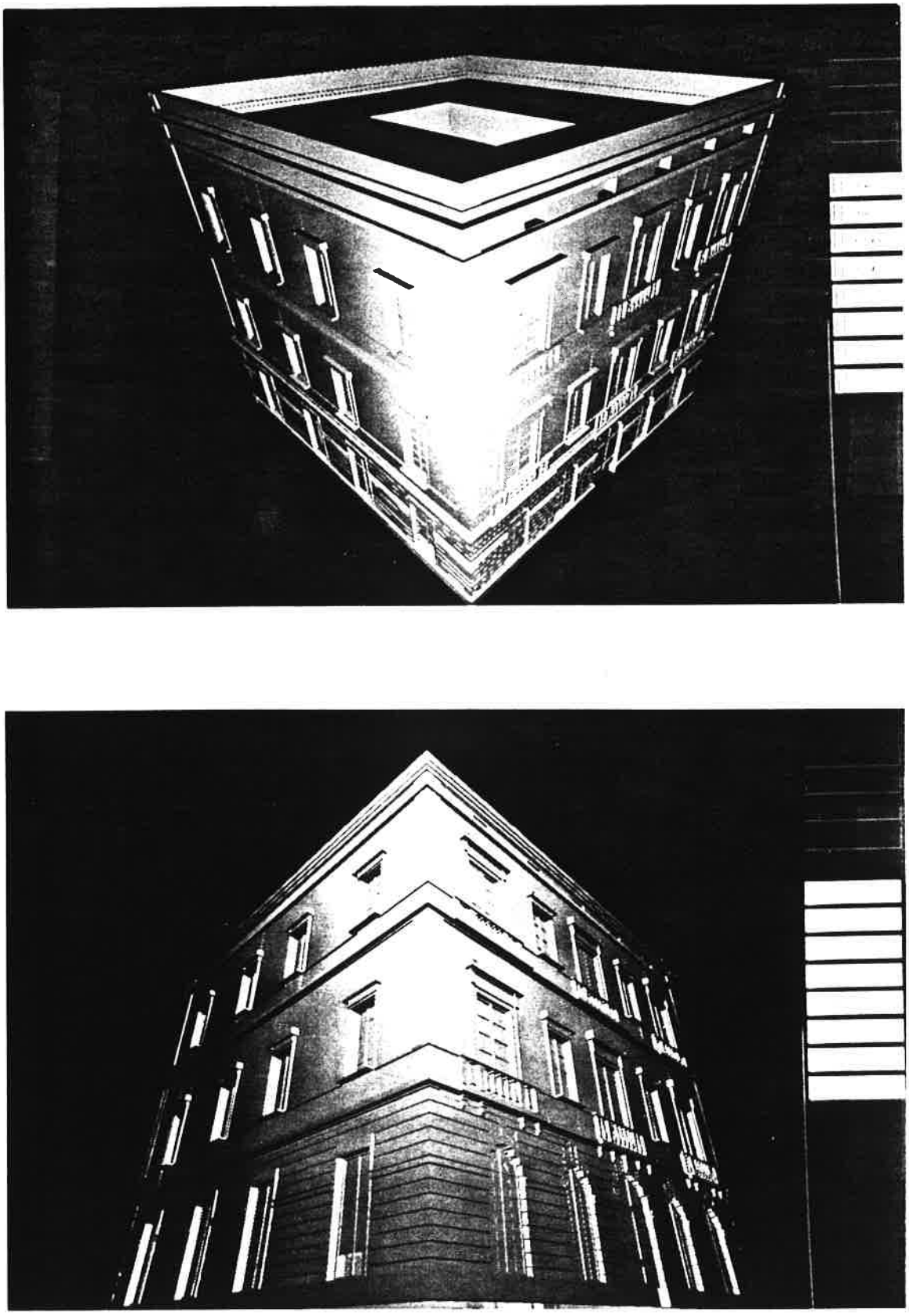

Figure 5. a) Isothetic solid. A building.

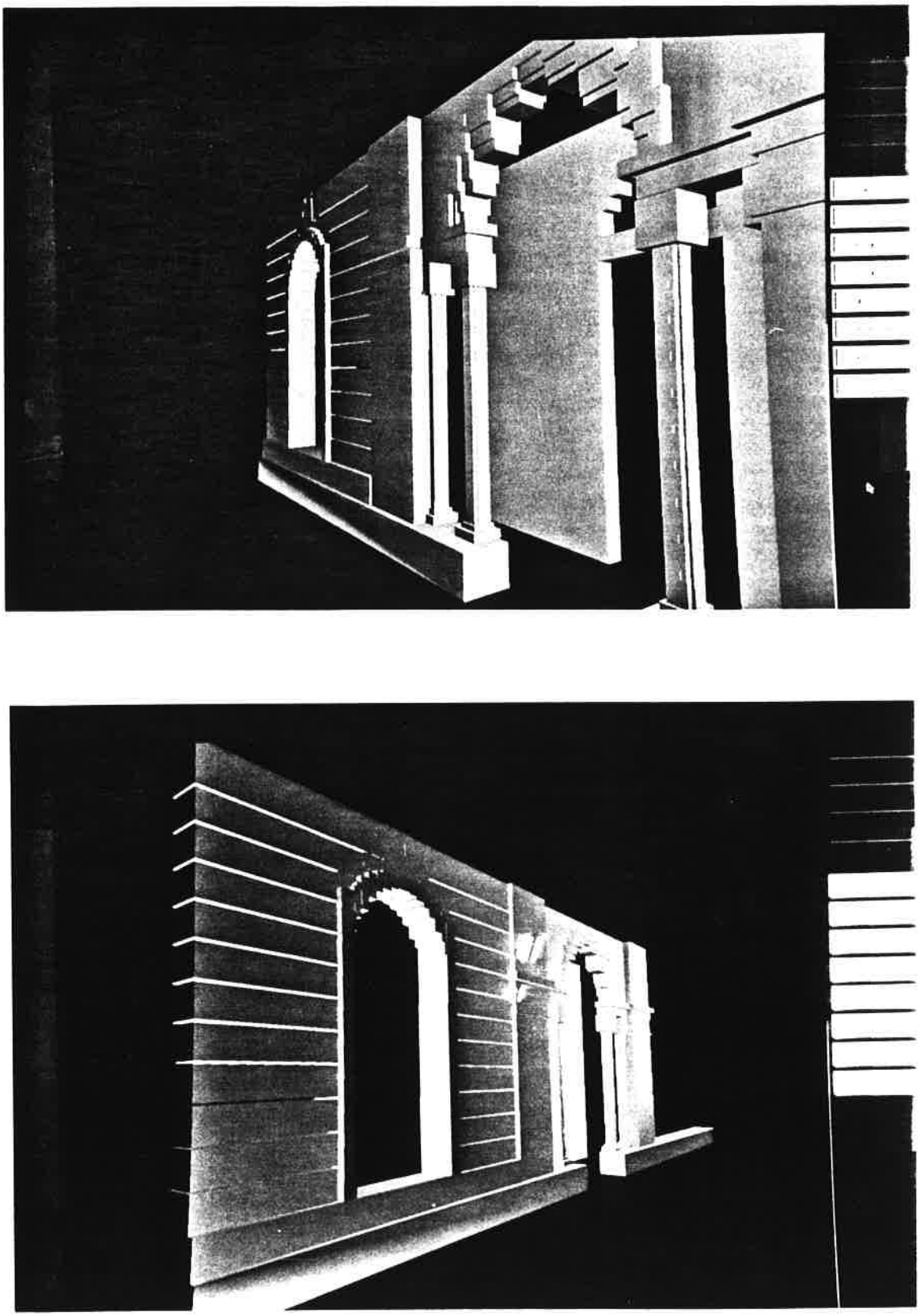

Figure 5. b) Isothetic solid. A front of a building.

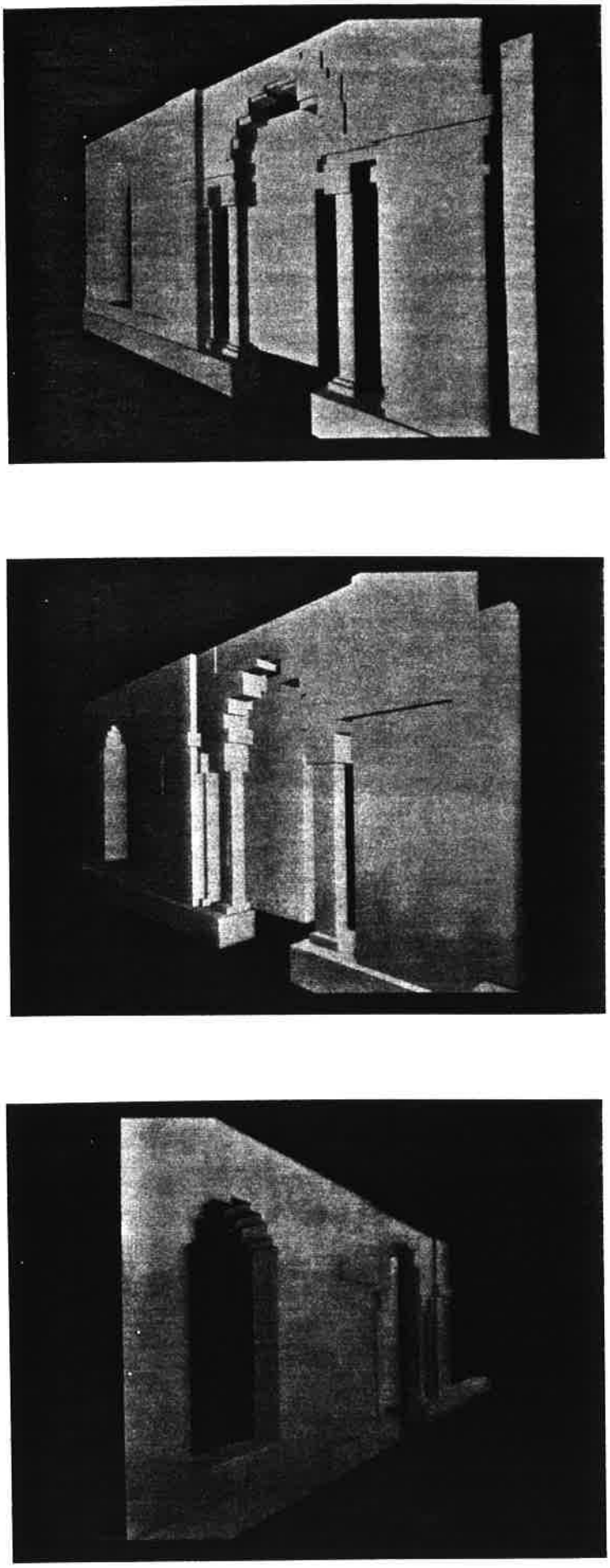

Figure 6. Successive simplifications of model in figure 5 b).

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

Recent Research Reports

LSI–94–45–R "On RNC approximate counting", Josep Díaz, María J. Serna, and Paul Spirakis.

LSI–94–46–R "Prototipatge semàntic d'un model conceptual deductiu" (written in Catalan), C. Farré and M.R. Sancho.

LSI–94–47–R "Generació i simplificació automàtica del Model d'Esdenivents Interns corresponent a un model conceptual deductiu" (written in Catalan), C. Farré and M.R. Sancho.

LSI–94–48–R "B-Skip trees, a data structure between skip lists and B-trees", Joaquim Gabarró and Xavier Messeguer.

LSI–94–49–R "A posteriori knowledge: from ambiguous knowledge or undefined information to knowledge", Matías Alvarado.

LSI–94–50–R "An approach to the control of completeness based on metaknowledge", Jordi Alvarez and Núria Castell.

LSI–95–1–R "Octree simplification of polyhedral solids", Dolors Ayala and Pere Brunet.

Copies of reports can be ordered from:

Nuria Sánchez
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Pau Gargallo, 5
08028 Barcelona, Spain
secrelsi@lsi.upc.es