Boundary Generation from Voxel-based Volume Representations

R. Joan-Arinyo
J. Solé

Report LSI-96-20-R

Boundary Generation from Voxel-based Volume Representations

R. Joan-Arinyo* and J. Solé
Universitat Politècnica de Catalunya
Departament de Llenguatges i Sistemes Informàtics
Av. Diagonal 647, 8a, E-08028 Barcelona

Abstract

A method to generate polygonal boundary representations from voxel-based volume representations is presented. As an auxiliary data structure, the algorithm makes use of a face octree extracted from the voxel-based volume representation. The precision of the computed boundary is given by the length of the main diagonal of the voxels in the initial representation. Experimental results that illustrate the method are presented.

1 Introduction

Voxel-based volume models are widely used in scientific visualization, [6, 8]. This is because voxels are the simplest way of representing spatial data stored in 3D arrays, and because this kind of data is what remote sensing and scanning technology generates from nondestructive examination of the internal structure of objects. Unfortunately, voxel-based volume schemes have some disadvantages; the huge amount of storage requirements to encode even simple solids, high time-consuming rendering algorithms, and the difficulty to apply shape recognition, geometric measurements or a series of geometric operations are some of them.

Hyerarchical spatial encodings like classical octrees [9, 14, 13] and extended octrees [1, 4, 5] have been proposed to reduce storage requeriments. They allow the use of simple algorithms to perform boolean operations between solids. Face octrees are an intermediate scheme between classical octrees and extended octrees; they are more concise than classical octrees and are particularly well suited to approximate representations of objects

*Partially Supported by CICYT, the Spanish agency for research funding, under grant TIC95-0630-C05-04

with complex surface boundaries [2, 3]. The most important drawback in this representations is that they are not closed models.

There are several reasons that justify the design of an efficient algorithm to convert huge voxel data representations into simple, closed, polygonal boundary representations. First, the conversion will yield a reduction of the amount of storage needed for the representation. Second, rendering of solids represented as polygonal boundary representations will be faster than rendering original voxel-based models, and the quality of the generated images will be improved. Third, simple, closed, polygonal boundary representations can be used for measurements and to performing some geometric operations. Several methods have been reported in the literature. For a detailed discussion see [10].

In this paper we present a new algorithm to extract a polygonal boundary representation from a voxel-based volume representation. The algorithm reads as input a voxel-based representation consisting of a binary three-dimensional array that encodes in each cell either the presence or the absence of solid matter. The algorithm consists of three major steps. First, a set of non planar quadrilaterals that covers the set of voxels in the boundary of the volume data set is generated. Then, a face octree is extracted from the volume data representation. Finally, a valid boundary representation is derived from the set of non planar quadrilaterals with the help of the face octree computed in the previous step. The planes associated with face octree nodes are used as embedding planes where quadrilaterals inside each octree node are projected. This has the effect of "ironing" high frequencies derived from the voxel data set. Non planar quadrilaterals spanning two or more octree nodes are split into two triangles to complete the boundary construction. The extracted representation is a correct boundary representation because it is self-intersection-free and closed. The precision of the boundary representation is given by the length of the main diagonal of the voxels in the initial data set except for some subsets of voxels forming walls or crests that are one voxel thick.

2 Face Octrees

Just for the sake of completness let us recall the basic concepts involved in octree representations. Octrees are one of the classical approximate decomposition schemes. They are trees that represent solids by encoding the recursive subdivision of a cubic finite universe where the solids are placed, [9, 12].

The nodes allowed in classical octrees are either terminal or non-terminal nodes. Terminal nodes are labeled as black or white depending on whether they are fully inside or fully outside the solid boundary. The solid boundary is approximated by a layer of cubes of a minimum specified size; they are also

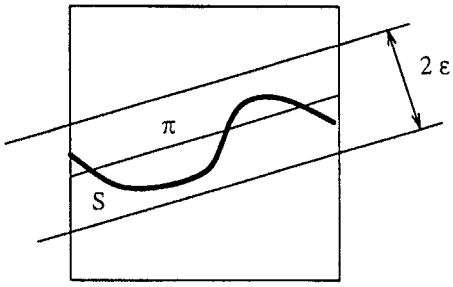

Figure 1: Face node and associated band.

labeled as black or white depending on some criterion [9, 14]. Non-terminal nodes are labeled as grey nodes.

In order to avoid the verbosity and the approximate character of the classical octree scheme, several proposals of new octree encodings have appeared in the literature. Extended octrees were introduced by Ayala *et al.*, [1] and polytrees were introduced by Carlbom *et al.*, [4]. Besides the white and black nodes, these octrees also represent face, edge and vertex nodes. These new types of nodes are terminal nodes and allow the exact representation of the solid boundary of polyhedra while reducing verbosity and keeping a reasonable complexity for the boolean operations.

Face octrees were introduced by Brunet in [2]. A face octree is an octree with white, black, face and grey nodes together with a tolerance ε that controls the degree of approximation of the representation. White, black and grey nodes are defined as in classical octrees. Face nodes contain a connected part of the object boundary and each of them has an associated equation of some plane, π, that approximates the boundary S within the node with a given tolerance ε. See Figure 1. Grey nodes are those that can not be labeled as white, black or face nodes. They represent regions of the object surface that are not flat enough. When the recursive subdivision reaches the minimum predefined node size, grey nodes are terminal grey nodes. Face octrees are halfway between classical and extended octrees; they are more concise than classical octrees and are well suited to approximate representations of objects with complex surface boundaries [3, 11].

3 The Algorithm

The algorithm has three major steps. First, a set of non planar quadrilaterals that covers the set of boundary nodes in the volume data is generated. Then a face octree is extracted from the volume data representation. Finally, a valid boundary representation is derived from the set of non planar quadrilaterals with the help of the face octree computed in the previous step.

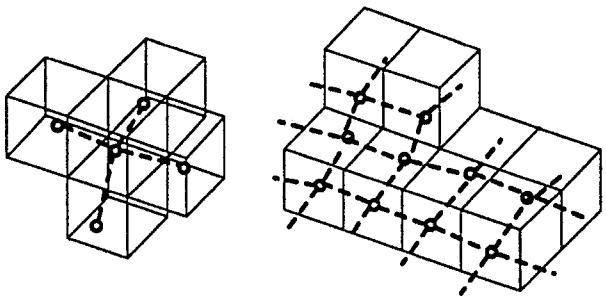

Figure 2: Interconnecting representative points.

3.1 Definitions

It is assumed that the voxels in the representation are labeled as either black or white; they represent space regions that are, respectively, inside or outside the original solid. Two voxels in the representation are *neighbours* if they share a face, an edge or a vertex. We will refer to either a *vertex neighbour*, an *edge neighbour* or a *face neighbour* whenever we want to make explicit the neighbourhood relationship between two given voxels. A voxel that has at least one white voxel in its neighbourhood is said to be a *boundary voxel.*

3.2 Non Planar Quadrilaterals

A set of non planar quadrilaterals that covers the set of boundary voxels is computed in the form of a geometrically deformed model (GDM), [7, 10]. The GDM is an elastic network shaped by a set of points as follows. We start by associating a point with each boundary voxel; we shall call these points the *representative points.* Next each representative point is placed in the boundary voxel center and a link is defined between each representative point and every point that represents a face neighbour of the considered voxel. See Figure 2. A cost function is associated with every representative point in the network. The network is relaxed by minimization of the cost function while each representative point is constrained to stay inside the voxel limits it represents. If the links between representative points are interpreted as edges, the relaxed network of representative points defines a set of non planar quadrilaterals which covers the set of boundary voxels.

This set of quadrilaterals contains quadrilaterals from which a valid boundary representation can be derived plus some extra quadrilaterals that must be pruned. The undesired quadrilaterals are generated when boundary voxels are arranged forming crests with a thickness of one or two voxels, as illustrated in Figure 3. When the crest is one voxel thick, extra quadrilaterals are dangling quadrilaterals. They are characterized by the fact that the quadrilaterals on the top of the crest have at least one vertex shared by less

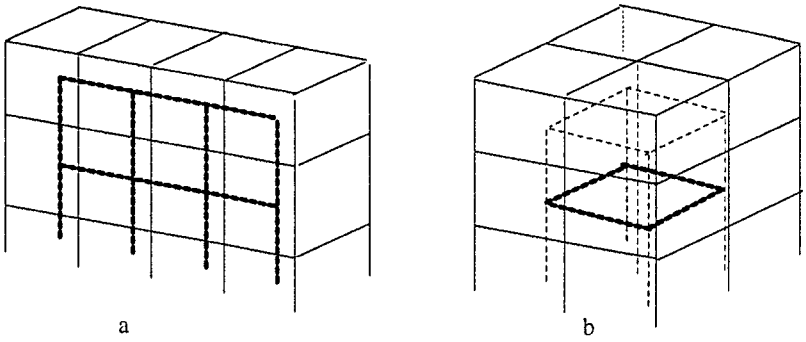

Figure 3: Extra quadrilaterals in bold dashed lines. a) Dangling quadrilaterals. b) Internal quadrilaterals.

than four quadrilaterals. These quadrilaterals are recursively pruned starting from those on the top of each crest. Note that as a result of this pruning some information present in the initial data set is lost. When the crest has a thickness of two voxels, quadrilaterals would be located inside the final boundary. In this case, the number of quadrilaterals sharing each vertex of each extra quadrilateral is greater than the number of representative points to which the vertex is connected. These extra quadrilaterals are pruned by a linear walk through the set of quadrilaterals.

It is worth to note that since all the boundary voxels are used to compute the set of quadrilaterals our method does not suffer from the branching problem, [10], that is, the algorithm naturaly handles the situation where several branches of voxels depart from a compact set of voxels.

3.3 Face Octree Generation

A face octree is computed by applying the procedure reported in [7] to the set of relaxed representative points obtained in Section 3.2.

First, a plane is associated with each boundary voxel in the volume representation as follows. For each boundary voxel, the representative points in its face neigbourhood are sorted circularly taking as center the representative point of the considered voxel. Then, for each triangle defined by the center and two consecutive sorted representative points, a normal is computed. Finally, a normal defined as the average of the normals to the triangles weighted with the respective triangle area is associated with the voxel considered. The plane defined by the average normal and the representative point is the GDM plane.

The face octree is extracted from the relaxed representative points and the GDM planes associated to the corresponding boundary voxels, by means of a compaction process. Initially a face octree induced by a space subdivision coincident with the planes that define the volume data voxels is defined.

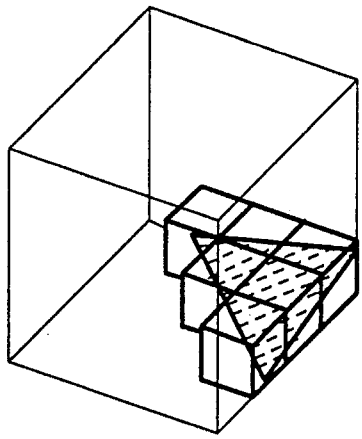

Figure 4: Face nodes compaction.

The face nodes in this face octree are the boundary voxels with the associated GDM planes. The construction of this first face octree guarantees that the space subdivision defined by face octrees generated in the compaction process will never intersect the interior of a data voxel. The extraction process compacts face nodes in the face octree as much as possible while preserving the volume data precision given by the length of the main diagonal in the data voxels. Compaction (see Figure 4) is performed by trying to fit a plane that is both interior to all boundary voxels inside the region bounded by the current face octree node and does not intersect any other voxel in the node, [7].

It is worth to note that the face octree computed so far, approximates the local planarity of the boundary voxels in the volume data.

3.4 Boundary Generation

We have a set of boundary voxels each with an associated representative point that has been relaxed. We have a set of non planar quadrilaterals defined on this set of relaxed points that covers the boundary voxel set. Furthermore, there is a face octree that induces a space subdivision compatible with the voxels in the volume data; i.e., the interior of each boundary voxel is either inside or outside a given face node in the face octree. Generating a valid boundary representation from these informations is straightforward.

For each face node in the face octree, those relaxed representative points of data voxels that are inside the octree node are projected on the plane associated with the node by projecting orthogonally the representative points. See plane π in Figure 5. As a result, projections of non planar quadrilaterals whose vertices are inside a given face node are planar quadrilaterals embedded in the plane associated with the face node, and are inside the same face

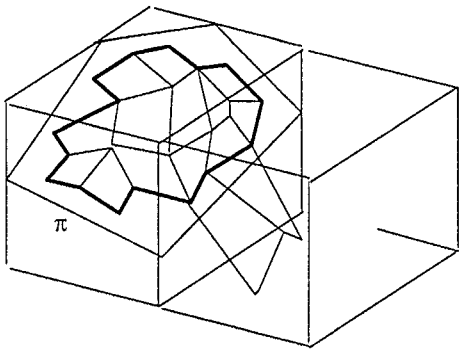

Figure 5: Quadrilaterals generated by projection of relaxed representative points (thin line) and planar polygon defined (bold line).

node, too. These projections are collapsed on one planar polygon defined by the subset of the projections of representative points such that for each projection at least one of the quadrilateral edges incident on it is partially inside and partially outside the face node. Polygon in boldline in Figure 5 shows an example.

Quadrilaterals generated by projection of non planar quadrilaterals whose vertices belong to data voxels located in different face nodes obviously have different vertices inside different face nodes; in general they are non planar and are split into two triangles. This completes a valid boundary representation. Figure 6 shows two face nodes in the octree that are neighbours. Quadrilaterals q_1 to q_6 are projections of non planar quadrilaterals; q_1 and q_2 are inside node a, q_5 and q_6 are inside node b; all of them are planar quadrilaterals embedded in the plane associated with the respective face octree node and will be included in the corresponding planar polygon. Quadrilaterals q_3 and q_4 have vertices in different face octree nodes: in general they are not planar and are split into two triangles. For an in depth discussion see [15].

The resulting boundary representation approximates the initial volume data with a precision given by the length of the main diagonal of the voxel in the initial voxel-based volume representation, [7, 15], except for those subsets of voxels forming crests one voxel thick that have been lost in the pruning process.

4 Results

The method has been implemented and applied to voxel data sets generated from solids defined with a geometric solid modeler. We present three case studies. For each case study we have sampled the geometry of the initial solid models with three different precision levels. The lowest level was generated

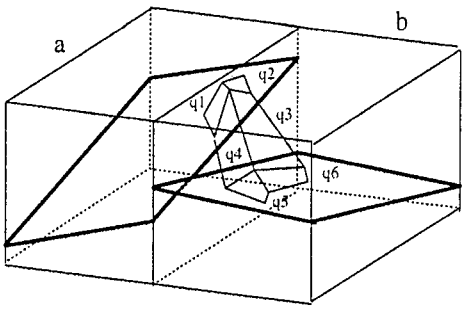

Figure 6: Projected quadrilaterals. Quadrilaterals q_1 and q_2 are inside node a. Quadrilaterals q_5 and q_6 are inside node b. Quadrilaterals q_3 and q_4 have vertices in both nodes.

by dividing into 32 voxels the edges of the cubic space region where the solids were embedded, for the intermediate level each edge was divided into 64 voxels, and for the highest precision the edges were divided into 128 voxels. Numerical values are shown in Table 1. The second column gives the precision, the third gives the number of voxels in the data set, the fourth is the number of nodes in the face octree, the fifth column gives the number of polygons in the boundary representation, and the last gives the ratio of the number of polygons in the boundary representation with respect to the number of voxels in the volume data set. From the numerical results it is clear that, except for the lowest precision level, the polygonal boundaries generated are much more compact than the initial volume data sets. As one could expect, the larger the precision the larger the resulting compaction.

Colour plates 7, 9 and 8 illustrate the quality of the boundary representations for solids in Table 1. The pictures were computed for the 128 by 128 by 128 voxel data sets. In each of the three sets of colour plates, the

Solid	precision	# voxels	# fo nodes	# polygons	# polygons / # voxels
A	32	1320	653	1718	1.30
	64	7599	1373	5185	0.68
	128	53645	2886	16768	0.31
B	32	4502	1351	4285	0.95
	64	19773	2596	11470	0.58
	128	229593	16662	51753	0.23
C	32	4926	1510	4965	1.00
	64	33373	3175	14694	0.44
	128	246365	8582	48272	0.19

Table 1: Case studies.

plate on the left shows the set of data voxels, the plate in the middle shows the associated face octree, and the plate on the right shows the boundary representation. In plates on the right, the planar polygons resulting from

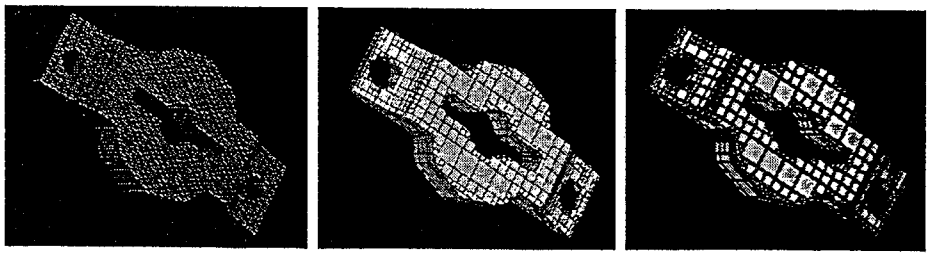

Figure 7: Case study solid A.

collapsing projections of non planar quadrilaterals on the octree face nodes planes can be easily distinguished as clean areas with uniform colour. The dark lines are the chains of triangles resulting from splitting the non planar quadrilaterals which sew the gaps between planar polygons.

5 Conclusions

Given a voxel-based model, the algorithm presented generates a non self-intersecting, closed, polygonal boundary representation, with a high degree of compaction. The degree of approximation of the initial data set is preserved excep for those parts where there are crests of voxels one voxel thick. We have demonstrated the use of this algorithm in several practical examples.

The algorithm is based on GDM techniques. It has been used in computing both a collection of non planar quadrilaterals that approximate the

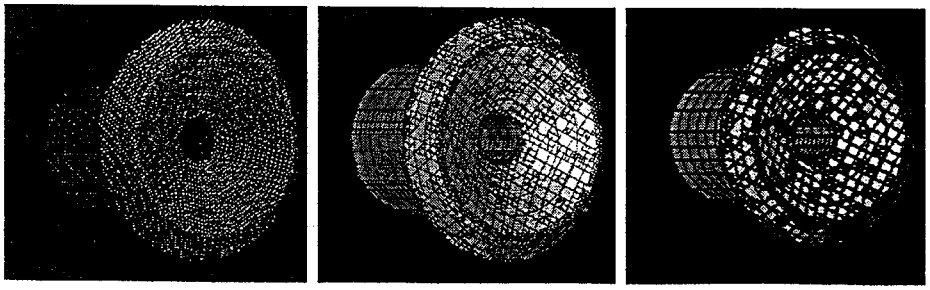

Figure 8: Case study solid B.

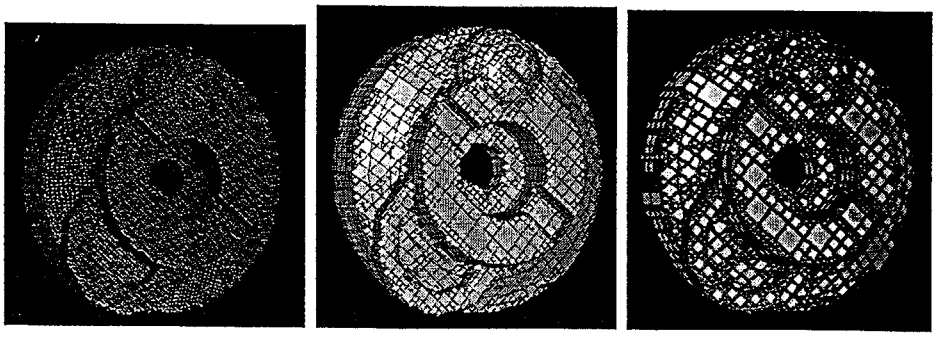

Figure 9: Case study solid C.

data set boundary, and in computing a face octree which works as an auxiliar data structure. As a result, the algorithm inherits several interesting properties exhibited by GDM models. One interesting property is that the cost of extracting the boundary representation depends on the complexity of the object, not on the size of the original voxel data set. Furthermore, since all the boundary voxels are involved in the computation of the GDM, the algorithm explicitly handles the branching problem.

Even with the most efficient algorithms, the extraction of surfaces from large sets of volume data is always a time-consuming problem. To reduce computation time, a parallel version of the presented algorithm is now being programmed on a CM-2 SIMD machine. This is part of a larger project comprising parallelization of a number of geometric operations on solids representations.

References

[1] D. Ayala, P. Brunet, R. Juan, and I. Navazo. Object representation by means of nonminimal division of quadtrees and octrees. *ACM Transactions on Computer Graphics*, 4:41–59, 1985.

[2] P. Brunet. Face octrees: Involved algorithms and applications. Technical Report LSI–90–14, Universitat Politècnica de Catalunya, Department of LiSI, 1990.

[3] P. Brunet, I. Navazo, and A. Vinacua. A modelling scheme for the approximate representation of closed surfaces. *Computing*, 8:75–90, 1993.

[4] I. Carlbom, I. Chakravarty, and D. Vanderschel. A hierarchical data structure for representing the spatial decomposition of 3-D objects. *IEEE Computer Graphics and Applications*, pages 24–31, April 1985.

[5] M.J. Dürst and T.L. Kunii. Integrated polytrees: A generalized model for integrating spatial decomposition and boundary representation. In *Theory and Practice of Solid Modeling*, Springer Verlag, 1989.

[6] G.T. Hermann and H.K. Liu. Three-dimensional display of human organs from computer tomograms. *Computer Graphics and Image Processing*, 9, (1):1–21, 1979.

[7] R. Juan-Arinyo and J. Solé. Constructing face octrees from voxel-based volume representations. *Computer-Aided Design*, 27(10):783–791, 1995.

[8] B.H. McCormick, T.A. De Fanti, and M.D. Brown (Eds). Visualization in scientific computing. *Computer Graphics*, 21:1–21, 1987.

[9] D. Meagher. Efficient synthetic image generation of arbitrary 3D objects. In *IEEE Computer Society Conf. on Pattern Rec. and Image processing*, pages 473–478, 1982.

[10] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, and M.J. Wozny. Geometrically deformed models: A method for extracting closed geometric models from volume data. *Computer Graphics*, 25, 4:217–226, 1991.

[11] N. Pla-Garcia. Recovering a smooth boundary representation from an edge quadtree and from a face octree. *Computer Graphics Forum*, 13(4):189–198, 1994.

[12] A. Requicha. Representations for rigid solids: Theory, methods, and systems. *Computing Surveys of the ACM*, 12:437–464, 1980.

[13] H. Samet. *Applications of Spatial Data Structures.* Addison Wesley Publ., Reading, MA, 1989.

[14] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison Wesley Publ., Reading, MA, 1989.

[15] J. Solé. *Parallel Operations on Octree Representations Schemes.* PhD thesis, Department LiSI, Universitat Politècnica de Catalunya, 1996.

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

Research Reports – 1996

LSI-96-1-R "(Pure) Logic out of Probability", Ton Sales.

LSI-96-2-R "Automatic Generation of Multiresolution Boundary Representations", C. Andújar, D. Ayala, P. Brunet, R. Joan-Arinyo, and J. Solé.

LSI-96-3-R "A Frame-Dependent Oracle for Linear Hierarchical Radiosity: A Step towards Frame-to-Frame Coherent Radiosity", Ignacio Martin, Dani Tost, and Xavier Pueyo.

LSI-96-4-R "Skip-Trees, an Alternative Data Structure to Skip-Lists in a Concurrent Approach", Xavier Messeguer.

LSI-96-5-R "Change of Belief in SKL Model Frames (Automatization Based on Analytic Tableaux)", Matías Alvarado and Gustavo Núñez.

LSI-96-6-R "Compressibility of Infinite Binary Sequences", José L. Balcázar, Ricard Gavaldà, and Montserrat Hermo.

LSI-96-7-R "A Proposal for Word Sense Disambiguation using Conceptual Distance", Eneko Agirre and German Rigau.

LSI-96-8-R "Word Sense Disambiguation Using Conceptual Density", Eneko Agirre and German Rigau.

LSI-96-9-R "Towards Learning a Constraint Grammar from Annotated Corpora Using Decision Trees", Lluis Màrquez and Horacio Rodríguez.

LSI-96-10-R "POS Tagging Using Relaxation Labelling", Lluís Padró..

LSI-96-11-R "Hybrid Techniques for Training HMM Part-of-Speech Taggers", Ted Briscoe, Greg Grefenstette, Lluís Padró, and Iskander Serail.

LSI-96-12-R "Using Bidirectional Chart Parsing for Corpus Analysis", A. Ageno and H. Rodríguez.

LSI-96-13-R "Limited Logical Belief Analysis", Antonio Moreno.

LSI-96-14-R "Logic as General Rationality: A Survey", Ton Sales.

LSI-96-15-R "A Syntactic Characterization of Bounded-Rank Decision Trees in Terms of Decision Lists", Nicola Galesi.

LSI-96-16-R "Algebraic Transformation of Unary Partial Algebras I: Double-Pushout Approach", P. Burmeister, F. Rosselló, J. Torrens, and G. Valiente.

LSI-96-17-R "Rewriting in Categories of Spans", Miquel Monserrat, Francesc Rosselló, Joan Torrens, and Gabriel Valiente.

LSI-96-18-R "Strong Law for the Depth of Circuits", Tatsuie Tsukiji and Fatos Xhafa.

LSI-96-19-R "Learning Causal Networks from Data", Ramon Sangüesa i Solé.

LSI-96-20-R "Boundary Generation from Voxel-based Volume Representations", R. Joan-Arinyo and J. Solé.

LSI-96-21-R "Exact Learning of Subclasses of CDNF Formulas with Membership Queries", Carlos Domingo.

LSI-96-22-R "Modeling the Thermal Behavior of Biosphere 2 in a Non-Controlled Environment Using Bond Graphs", Angela Nebot, François E. Cellier, and Francisco Mugica.

LSI-96-1-R "Obtaining Synchronization-Free Code with Maximum Parallelism", Ricard Gavaldá, Eduard Ayguadé, and Jordi Torres.

Hardcopies 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

See also the Department WWW pages, http://www-lsi.upc.es/www/