www.cadanda.com
Computer-Aided Design AND APPLICATIONS™

CAD models from medical images using LAR

Alberto Paoluzzi[1], Antonio DiCarlo[2], Francesco Furiani[3], Miroslav Jiřík[4]

[1]Roma Tre University, apaoluzzi@me.com
[2]Roma Tre University, adicarlo@mac.com
[3]Roma Tre University, f.furio@gmail.com
[4]West Bohemia University, miroslav.jirik@gmail.com

ABSTRACT

In this paper we show the extraction of the portal vein subsystem of the liver using a novel topological representation of geometric data, called *Linear Algebraic Representation* (LAR). This representation scheme [11] is characterised by a very large domain, encompassing 2D and 3D engineering meshes, manifold and non-manifold geometric and solid models, and high-resolution 3D images. The novel representation scheme is being developed within the framework of the IEEE-SA P3333.2 - Draft Recommended Practice for Three-Dimensional (3D) Medical Modeling [20].

Keywords: Solid modeling, representation scheme, Linear Algebraic Representation, LAR, 3D medical images.

1 INTRODUCTION

Technological advances have made possible to acquire large sets of biomedical data at a fast rate and affordable costs. As a consequence, the ease of producing and collecting data in digital form is producing a gradual shift of paradigm from physical prototypes and experiments to virtual prototypes and mathematical modeling for simulation and prediction [2].

The ability of extracting geometrical models from medical imaging will produce a quantitative evidence-based medicine, where laboratory and clinical observation are being cumulated and made accessible for integrative research [17]. In our close future, knowledge we have on sub-systems, dimensional scales, and biophysical aspects, will be formalized, collected, catalogued, organized, shared and combined in any possible way, giving origin to personalized healthcare, providing integration across sub-systems, temporal and dimensional scales, and across medical and engineering disciplines, using physical-mathematical knowledge.

According with the availability of data, the interest is growing for physical simulations, typical of engineering CAD, to better understanding the physiology and behavior of human subsystems, on a single-patient basis, and using personalized models extracted from 3D images of patient body. A current research area provides 3D geometric guidelines to surgeons, about the design of liver's partial resections for care of tumors [19], using also low-Reynolds fluids-dynamics simulations [31].

Of course, great challenges are posed by old and new application fields, namely material science—think of soft tissues, engineered surfaces, nano-materials and meta-materials—and biomedicine, where

modeling and simulation issues range from the molecular/protein level to multi-scale modeling of subcellular organelles, cellular structures, tissues and organs. These problems raise demanding requirements, ranging from cooperative collaboration to multi-physics support, where different field equations imply different geometric structures at the level of basic descriptive data, to robustness toward scale mismatch in coupled problems, huge complexity of the simulation environment, or terascale number of elementary entities or agents.

In particular, the new applications of 3D medical imaging to personalized medicine, surgery simulation, virtual physiological human modeling, computer-based training of doctors, etc. require at the present time the convergence of techniques from computer imaging, computer graphics, geometric/solid modeling, and physical simulations.

A novel representation scheme is presented here, that unifies the treatment of images, meshes and polygonal data, and requires the minimum amount of storage for a "complete" geometric representation of human body parts and systems. The work introduced in this paper and in [11] as been developed within the framework of the IEEE-SA Project P3333.2 - Standard for "Three-Dimensional Model Creation Using Unprocessed 3D Medical Data".

1.1 Previous work

The foundational concept of *representation scheme* in solid modeling, as a mapping between mathematical models and computer representations, was defined at Rochester by the Production Automation Project leaded by Herbert Voelcker and Aristides Requicha at the end of seventies, and produced an enormous research work in the two following decades. This work on representation schemes includes, but is not limited to [30, 29, 33, 6, 5, 8, 12, 15, 40, 42, 41, 18, 32, 38, 36, 34, 21 26, 7, 14, 27, 35, 16, 28, 10, 3, 25, 24, 23, 1].

Concurrently, from research developments in numerical analysis, B-splines and NURBS emerged as the ubiquitous and most useful mathematical tool to support boundary representations of solids, and the first geometry kernels were created in some universities, in US and Europe. Later transformed into commercial software, and supported by huge investments, these geometric kernels became the foundational framework for all commercial solid modelers and the emerging business of PLM systems for aerospace, automotive, naval, and manufacturing industries.

At the present time, the information and communication world and its technologies are changing at a furious pace but, conversely, despite the tremendous amount of research done and the continuing technical advances in computer-aided design, geometric computing and scientific visualization, the most widely used software tools in the PLM industry still follow the basic approach established about twenty years ago, and centered around non-manifold topology, boundary representation, NURBS curves and surfaces. Such systems therefore require appropriate pre-processors (usually Delaunay triangulations) towards PDE solvers and post-processors towards graphics renderers and user interfaces. A lot of work was also done about using computational clusters and porting pre-existing software tools to novel system architectures, more and more oriented toward vector processing and many-core chips. However, it is our firmly held belief that time is ripe to radically rethink the fundamentals of solid modeling.

1.2 Paper organization

This paper is organized as follows. Section 1 introduces the aim and the goals of the ongoing implementation of the LAR prototype. Section 2 gives a simple introduction to the main definitions of the Linear Algebraic Representation, providing also some simple examples of cellular spaces. Section 3 discusses the principal aspects of the LAR of model topology, and discusses a few notable algorithms. Section 4 is dedicated to the LAR of images, showing how to map the structure of a d-dimensional image to a cellular complex of d-cuboids, and discussing how to reduce the standard morphology operators over images to the composition of topological operators over cellular complexes. This section also contains the main result of the paper: the extraction of a well-defined solid model of the liver's vein system from 3D images. The Conclusion section 5 summarizes the current situation of the prototype implementation and provides some clues about the future development.

2 LINEAR ALGEBRAIC REPRESENTATION

A representation scheme is a mapping between the mathematical spaces to be represented by a computer system and their symbolic representation in computer memory [30]. The *Linear Algebraic Representation* (LAR) scheme [11], uses *Combinatorial Cellular Complexes* (CCC) as mathematical domain [4], and various compressed representations of *sparse matrices* [9] as codomain.

Since LAR provides a *complete* representation of the topology of the represented space, the matrix $[\partial_d]$ of the boundary operator shall be used to compute the coordinate representation [c] of the boundary chain of *any subset c* of cells, though a single operation of SpMV multiplication [9] between the CSR representation of $[\partial]$ and the CSC representation of the [*c*] chain.

Even more importantly, the matrices of coboundary operators $[\delta^0]$, $[\delta^1]$, and $[\delta^2]$, computable in the LAR scheme by means of multiplications between sparse matrices, respectively provide the discrete *gradient*, *curl*, and *divergence* upon the given space decomposition *S*. The *Laplacian* operator Δ, a primary ingredient in most physical-mathematical equations, is computed as a combination of *boundary* and *coboundary* operators. Last but not least, the standard operators of *mathematical morphology* on images (dilation, erosion, opening and closing) [13] are obtained by *SpMSpM* product of matrices of topological incidences with boundary and coboundary matrices.

For the sake of concreteness, in the following section we use some *python* notations and expressions, that would be often directly computable in our open-source prototype software system https://github.com/cvdlab/lar-cc, in the course of continuous development.

2.1 Some basic definitions and examples

The first important concept introduced by LAR is the definition of the model of a cell complex, as made by a list of vertices, given as lists of coordinates, and by one or two topological relations. The "list" structures we refer to, here and in the following of the paper, are modern data structures, typycal of languages like *python* or *javascript*, where the access to a list element, given its ordinal value, requires constant time.

Definition 1 (LAR model). *A* LAR model *is either a pair* `V,FV`*, or a triple* `V,FV,EV`*, where:*

1. `V` *is the list of vertices, given as lists of coordinates;*
2. `FV` *is a* cell-vertex *relation, given as a list of cells, where each cell is given as a list of vertex indices.*
3. `EV` *is a* facet-vertex *relation, given as a list of cells, where each cell is given as a list of vertex indices and "facet" stands for* $(d-1)$*-face of a d-cell*

2.1.1 Examples

Some very simple examples of 0D, 1D, and 2D models follows. They are displayed in Figure 1. The user should remember that the LAR representation scheme is *dimension-independent*. The same data structures and algorithms can be used in every (low-dimensional) Euclidean space.

```
""" 0D, 1D, 2D model examples """
V = [[0.,0.],[1.,0.],[0.,1.],[1.,1.],[0.5,0.5]]
VV = [[0],[1],[2],[3],[4]]
EV = [[0,1],[0,2],[0,4],[1,3],[1,4],[2,3],[2,4],[3,4]]
FV = [[0,1,4],[1,3,4],[2,3,4],[0,2,4]]
model0d, model1d, model2d, complex2d = (V,VV), (V,EV), (V,FV), (V,VV+EV+FV)
```

Definition 2 (Complete LAR model). *A* complete *d*-model *is a triple* `V,FV,EV`*, where the relations* `FV` *and* `EV` *define the d-cells and* $(d-1)$*-cells, respectively.*

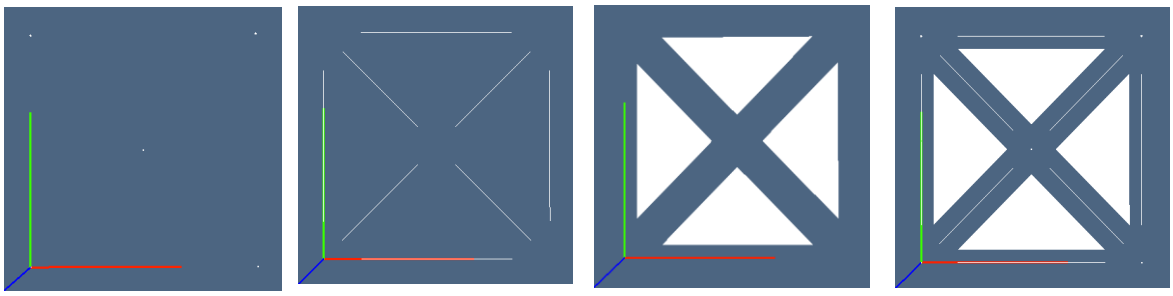


Fig. 1: Images of *model0d, model1d, model2d, complex2d*, drawn exploded.

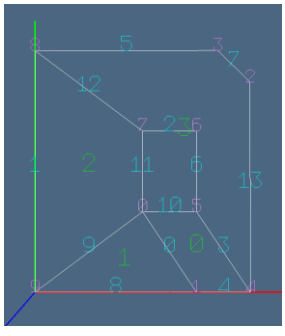
Remark 1 (Complete LAR model). *In order to compute the boundary and coboundary operators* ∂_d *and* $\delta^{d-1} = \partial_d{}^T$, *both FV and EV are needed: in general it is needed the description of* d- *and* $(d-1)$-*cells. Actually, for several important meshes and grids, where the number of vertices of each cell is* constant *(e.g. complexes of triangles, quadrilaterals, hexahedra, and regular simplicial complexes), the pair V, FV is sufficient, since EV can be efficiently computed from FV.*

3 BASIC REPRESENTATIONS

A few basic representation of topology are used in the LARCC library. They include some common sparse matrix representations: CSR (Compressed Sparse Row), CSC (Compressed Sparse Column), COO (Coordinate Representation), and BRC (Binary Row Compressed).

3.1 BRC (Binary Row Compressed)

We denote as BRC (*Binary Row Compressed*) the standard input representation of the LAR-CC computational framework. A BRC representation is an *array of arrays of integers*, with no requirement of equal length for the component arrays. The BRC format is used to represent a (normally sparse) binary matrix. Each component array corresponds to a matrix row, and contains the indices of columns that store a 1 value. No storage is used for 0 values.



```
V = [[0.395,0.296],[0.593,0.0],[0.79,
 0.773],[0.671,0.889],[0.79,0.0],
 [0.593,0.296],[0.593,0.593],
 [0.395,0.593],[0.0,0.889],[0.0,0.0]]

FV = [[0,1,4,5],[0,1,9],[0,7,8,9],
   [2,3,4,5,6,7,8]]

EV = [[0,1],[8,9],[6,7],[4,5],[1,4],
   [3,8],[5,6],[2,3],[1,9],[0,9],[0,5],
   [0,7],[7,8],[2,4]]
```

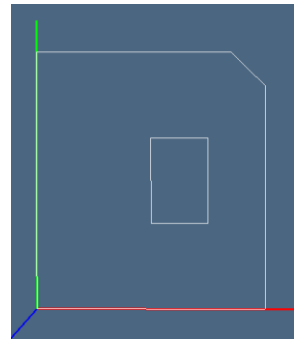


Figure 2: (a) LAR model with *0*-, *1*-, and *2*-cells; (b) The LAR model: triple V, FV, EV of *vertices*, *faces* and *edges* (indexed on vertices); (c) the extracted boundary. Let us notice that 2-cells may have different numbers of vertices, and that they can be either *convex* or *non-convex*.

```
FV = [[0,1,4,5],
      [0,1,9],
      [0,7,8,9],
      [2,3,4,5,6,7,8]]
```

$$\mapsto M_2 = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} \mapsto$$

```
<4x10 sparse
matrix of type
'numpy.uint8'
with 18 stored
elements in CSR
format>
```

3.2 Compressed Sparse Row (CSR) matrix storage

General purpose representations aimed to mix a description of the boundary with a description of the interior of the model are mostly used for physical simulation, whereas computer graphics applications usually prefer some simple representation of the boundary, like the set of triangles or quads on the boundary surfaces, efficiently supported by LAR.

For example, the *triangle-mesh* geometry representation used in the STL format for stereolitography, the open standard AMF (Additive Manufacturing File) for describing objects for additive manufacturing processes (3D printing), and the ISO/ASTM standard to describe the shape and composition of any 3D object to be fabricated on 3D printers, can be considered, for what concerns the geometry, as special cases of the LAR representation.

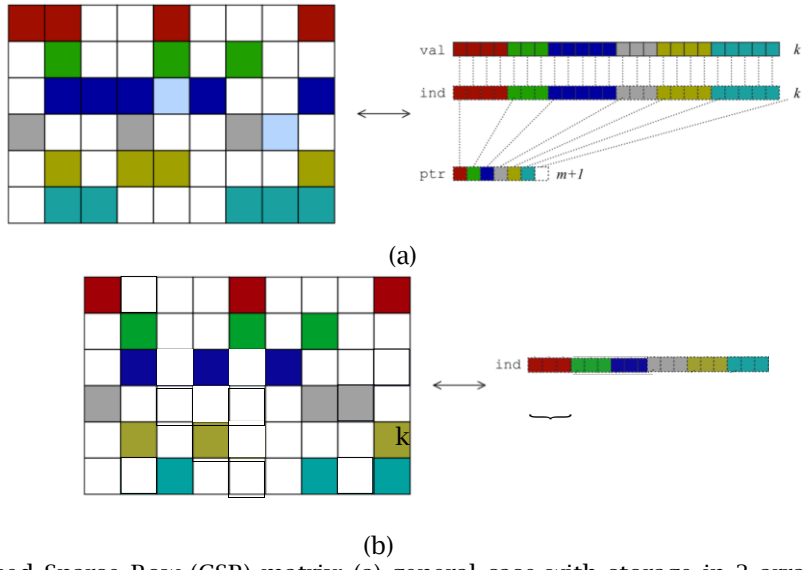


Figure 3: Compressed Sparse Row (CSR) matrix: (a) general case with storage in 3 arrays (image from [39]); (b) special case: LAR of *d*-meshes, with binary values and non-zeros in constant number *k* for each row, with $k = d + 1$ (regular *simplicial d*-complexes) or $k = 2^d$ (regular *cuboidal d*-complexes).

3D triangulations The unstructured representations of the object's interior often consist of 3D tetrahedra, storing for each tetrahedron the 4 references to vertices, and the 4 references to the adjacent tetrahedra, sharing a common triangle facet [22]. This adjacency information shall be efficiently reconstructed by the LAR representation, whose long-term storage and transmission consists of only 4 vertex indices per tetrahedral cell.

Solid boundary representations Most frequently, the representation scheme of topology used by *solid modellers* is a decompositive representation of the boundary, to be coupled with *meshing* of the interior just in case of need. The boundary is usually decomposed into *faces*, with boundaries of faces represented in turn by a decomposition into *edges*, given as pairs of *vertices*. In case of *manifold* representations, to store only a subset of the binary incidence relationships between such boundary elements is sufficient. The usual *non-manifold* representations often include some set of pointers

between incident pairs of boundary elements, usually circularly ordered to discriminate locally between the interior and the exterior, so at least doubling the storage size of the representation. Conversely, LAR includes only lists of cells as unordered lists of vertex indices, and manages equally well both manifold and non manifold models.

Comparison with Baumgart's scheme The common reference term for comparing the memory requirements of a solid boundary representation in 3D is the *Winged-Edge* scheme by Baumgart [5], that makes use of relation tables with a storage occupancy $8|E| + |V| + |F|$, where *F, E, V* stand for the sets of boundary faces, edges and vertices, respectively. An *equivalent* LAR representation of topology of the boundary of a 3D solid (B-rep) needs the *storage* of the only CSR(M_2) sparse matrix, corresponding to the *FV* incidence relation, and the *computation* of the CSR(M_1) sparse matrix, for the *EV* relation, for a total memory size of $2|E| + 2|E|$, according to [40].

3.3 Some LAR-based operations and algorithms

Several LAR-based algorithms are already implemented in our prototype implementation *lar-cc*, currently written in *python*. They include at the present time, mostly implementd in *MapReduce* style:

- generation of 0- and 1-dimensional cellular complexes;
- generation of simplicial and cuboidal *d*-complexes;
- computation of the matrices of *hk-incidence* operators ($0 \le h, k \le d$);
- conputation of $(d-1)$-*facets* of *d*-cells, and hence combined with efficient sorting and removal of duplicates, the computation of all *k-skeletons* ($0 \le k \le d$);
- computation of matrices of *boundary* and *coboundary* operators for both *oriented* and *non-oriented* complexes;
- *Cartesian product* of cellular complexes;
- *extrusion* of simplicial complexes
- computation of *integrals* of polynomials over polyhedral domains (2D, 3D);
- generation of *hierarchical* chains (structures) and cochains over a cellular complex;
- computation of the cellular complex generated as *assembly* (hierarchical structures) of LAR models and affine transforms;

4 LAR OF IMAGES

In this section we mainly discuss how to map a *d*-image, with normally $d \in \{2, 3\}$, to the coordinate representation of one or more *chains* (subsets of voxels) within the linear space Cd generated by the *cellular complex* corresponding in (generalized) row-major order to the image voxels, using LAR.

4.1 From *d*-images to (co)chains

In order to produce the coordinate representation of a chain in a multidimensional image (or *d*-image) we need: (a) to choose a basis of image elements, i.e. of *d*-cells, and in particular to fix a total ordering of image voxels; (b) to map the multi-index, selecting a single *d*-cell of the image, to a single integer mapping the cell to its ordinal position within the chosen basis ordering.

As an example, the generic hexahedral 3-cell (with 8 vertices), depending on three indices *i, j, k* is obtained as convex combination of the vertices indexed as:

$$\begin{aligned}\texttt{cell}[i,j,k] = [&(i,j,k),\ (i+1,j,k),\ (i,j+1,k),\ (i,j,k+1),\\ &(i+1,j+1,k),\ (i+1,j,k+1),\ (i,j+1,k+1),\ (i+1,j+1,k+1)]\end{aligned}$$

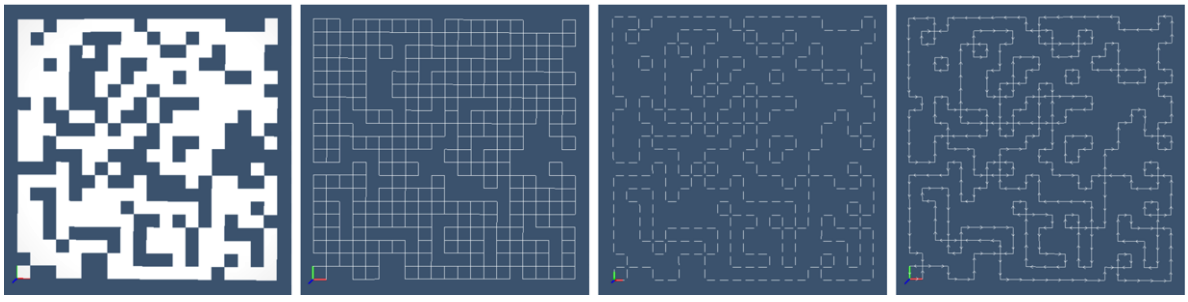


Figure 4: The orientation of the boundary of a random cuboidal 2-complex; (a) 2-cells; (b) 1-cells; (c) exploded boundary 1-chain; (d) *oriented* boundary 1-chain.

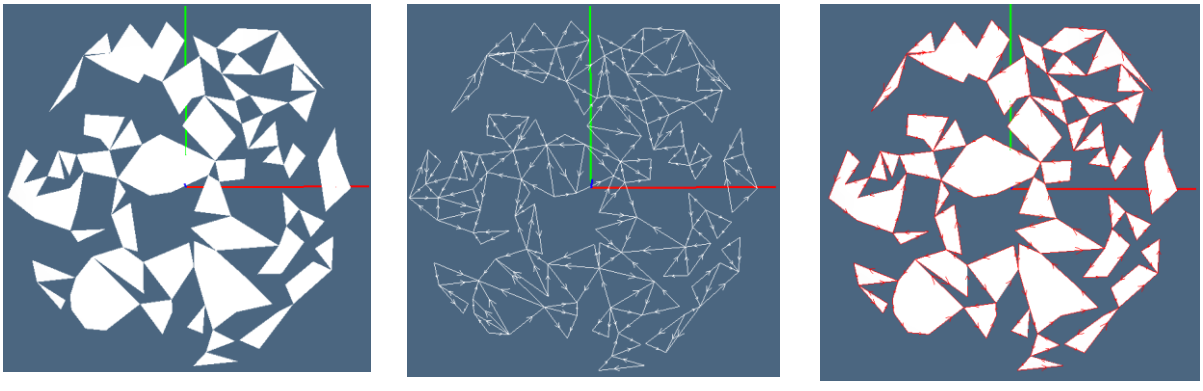


Figure 5: The orientation of the boundary of a random simplicial 2-complex; (a) 2-cells; (b) 1-cells; (c) *oriented* boundary 1-chain (red).

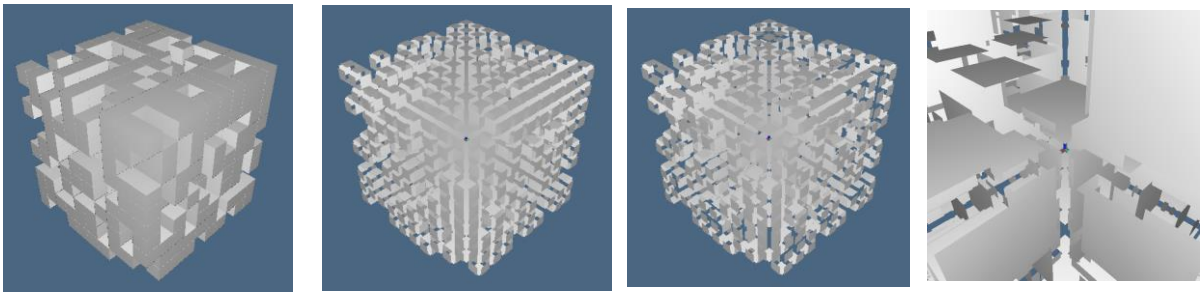


Figure 6: Extraction of oriented boundary of a cuboidal 3-complex: (a) Cellular 3-complex; (b) complex of 2-cells; (c) (exploded) oriented boundary; (d) view from the boundary interior.

Assuming that vertices are located over a 3D lattice of points with integer coordinates, it is easy to see that an explicit storage of coordinates is not required, since a bijective mapping exists between the ordinal index of cells and the tuples of coordinates of their vertices.

In Figure 7 it is shown our model of a d-image as a cuboidal grid with integer coordinates. Every d-cell is identified by a d-tuple of integer coordinates, which must be mapped to a single integer, in order to compute the basis vector (in C_d) corresponding to the cell, here in the linear space C_3 of *chains* of 3-cells (voxels). Such ideas are formalized in the following paragraphs.

Figure 7: 3D image portions as 3-cell complexes: (a) image portion seen explosed; (b) *divide et impera* paradigm; (c) reconstruction by removal of double cells, with sort-based *MapReduce* algorithm.

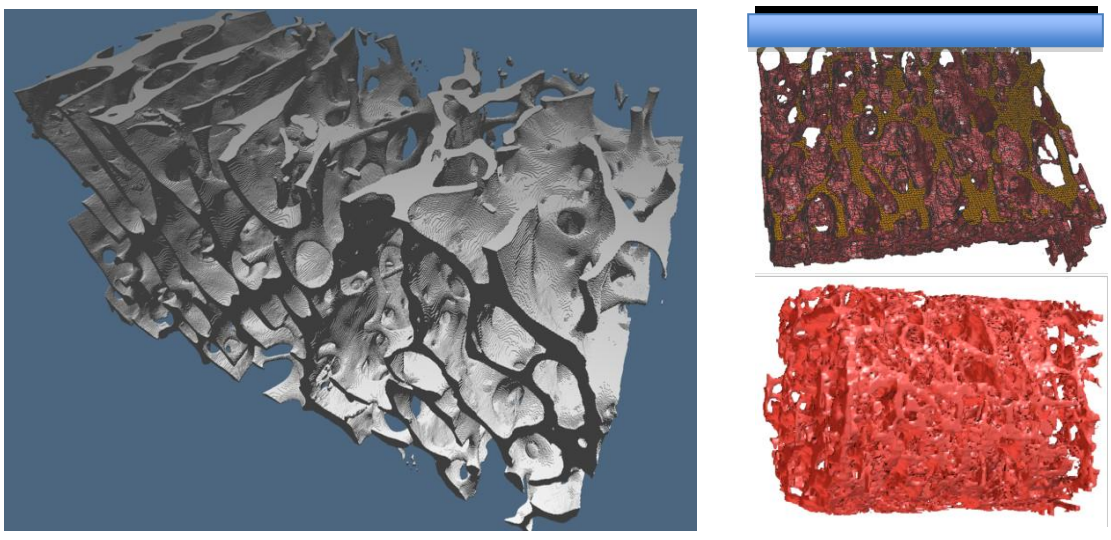


Figure 8: (a) The *solid model*, closed and topologically exact at the resolution of the image, of a sample of spongy bone, as LAR of the boundary of the chain of solid voxels. It is computed by using the GPGPU support provided by *OpenCL*. Let us compare it with (b),(c) the open surfaces generated by marching-cubes, or similar algorithms, over the 3D field defined by image intensity. Non trivial difficulties are encountered in order to close such sets of surfaces without topological errors, that may appear even locally.

Let us remark that the matrix $[\partial_3]$ of the boundary operator $C_3 \rightarrow C_2$, used to compute the boundary of *any* possible subset of voxels, i.e. of any vector in the linear space C_3 of 3-chains of the image, depends *only* on the image *shape*, and may be computed only one time (choosing a set of standard image shapes), and stored or transmitted accordingly. Since the bottleneck of GPGPU implementations is the moving of data from global to local memory, our solution is to store the (sparse) matrix operator $[\partial_3]$ of n^3 voxels, with $n \in \{64, 128, 256\}$, in device's *Constant Memory*, and move the (binary) *coordinate vectors* of chains in *Private Memory*. The boundary computation is therefore done by portions of the image, according to the paradigm *divide et impera*, as shown in Figures 7b and 7c

Grid of hyper-cubes Let $S_i = (0, 1, ..., n_i - 1)$ be ordered sets of integers with n_i elements, and $S := S_0 \times S_1 \times \ldots \times S_{d-1}$ the set of indices of elements of a d-image.

Definition 3 (d-image shape). *The* shape *of a d-image, with* $N = n_0 \times n_1 \times \cdots \times n_{d-1}$ *elements, called* voxels *or d-cells, is the ordered set* $(n_0, n_1, \ldots, n_{d-1})$.

Definition 4 (d-dimensional row-major order). *Given a d-image of shape* $S = (n_0, n_1, ..., n_{d-1})$ *and number of d-cells* $N = \Pi n_i$, *the mapping* $S_0 \times S_1 \times \cdots \times S_{d-1} \to \{0, 1, \ldots, N - 1\}$ *is a linear combination with integer weights* $(w_0, w_1, ..., w_{d-2}, 1)$, *such that:* $(i_0, i_1, ..., i_{d-1}) \to i_0 w_0 + i_1 w_1 + \cdots + i_{d-1} w_{d-1}$, *where* $w_k = n_{k+1} n_{k+2} \cdots n_{d-1}$, $\quad 0 \le k \le d - 2$

From multi-index tuples to chain coordinates A functional implementation of a *tuple* $\to$ *integer* mapping may use a second-order function, that accepts in a first application the *shape* of the image (in order to compute the tuple space of indices of d-cells), and then takes a multi-index tuple as parameter in a second application. Of course, this function will return the cell index in the linear address space associated to the given shape.

The set of address tuples of d-cells (i.e. of d-dimensional image elements) within a given image *mask* is mapped to the corresponding set of (single) integers associated to the low-level image elements (pixels or voxels, depending on the image dimension and shape). Such *total chain* of the mask window is then filtered to contain the only coordinates of image elements of the given *colour* (intensity value) within the considered *image window*, and returned as a list of integer cell indices.

4.2 Imaging Morphology with LAR

It is possible to implement the four operators of mathematical morphology, i.e. the *dilation*, *erosion*, *opening* and *closing* operators, by the way of matrix operations representing a composition of the linear topological operators — *boundary* and *coboundary* — with other incidence relations. According to its multidimensional character, the LAR implementation of morphological operators is dimension- independent.

In few words, it works as follows: (a) the input is (the coordinate representation of) a d-chain γ; (b) compute its boundary $\partial_d(\gamma)$; (c) extract the maximal $(d - 2)$-chain $\varepsilon \subset \partial_d(\gamma)$; (d) consider the $(d-1)$-chain returned from its coboundary $\delta_{d-2}(\varepsilon)$; (e) compute the d-chain $\eta := \delta_{d-1}(\delta_{d-2}(\varepsilon)) \subset C_d$ *without* performing the mod 2 final transformation on the resulting coordinate vector, that would provide a zero result, according to the standard algebraic constraint $\delta \circ \delta = 0$. It is easy to show that

$$\eta \equiv (\oplus\gamma) - (\ominus\gamma)$$

provides the *morphological gradient* operator. The four standard morphological operators are therefore consequently computable.

LAR is being used in biomedical applications which require fast performances with big geometric data for topological tasks such as model extraction from 3D images. Density values in medical images represent scalar fields (cochains) over cubical cellular complexes, and LAR is used to guarantee topologically correct 3D image segmentation as well as to extract (enumerative) solid model from the image1 , which is subsequently smoothed out via Laplacian smoothing, substituting to every vertex position the mean of its neighbors. A good feature of this approach is that the whole image is partitioned into a set of cochains associated to field intervals, including the interstitial space, so providing a well-defined meshing of both the features and their outer space.

In particular, any portion of a d-image $(2 \le d \le 4)$ can be seen as a d-chain in the linear space of chains induced by a regular d-cubical CCC decomposition of the bounded d-cuboidal image space, delimited by two extreme picture elements of minimum and maximum multi-indices. Because of the isomorphism between a d-complex S and its dual S^*, any d-image subset $c \subset S(d)$ can be represented as a 0-chain $c^* \subset S^*(0)$, and stored as a CSC vector of binary coordinate representation CSC(c). Since the structure of the space decomposition does not vary with the image

content, and depends only on the *shape* of the image, the boundary/coboundary matrices of images will be computed once, and stored/transmitted in CSR format for the most used image shapes.

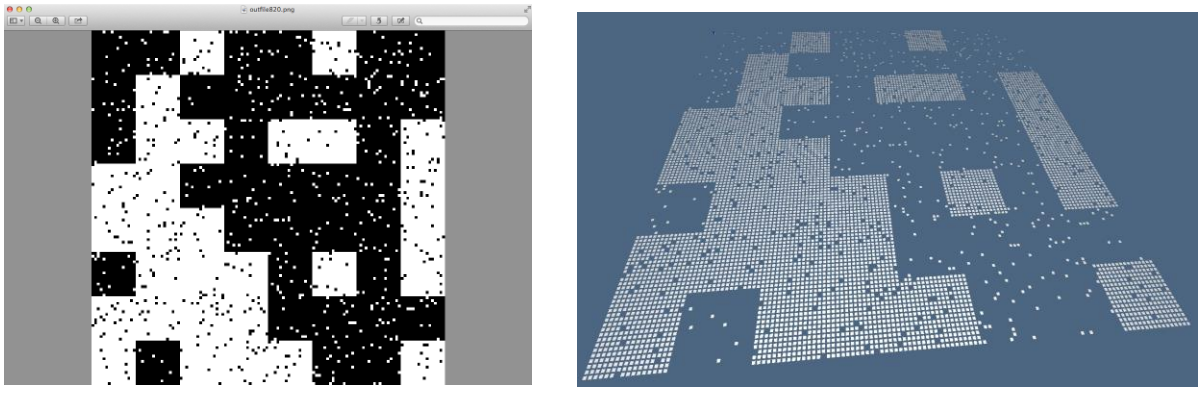


Figure 9: Let us consider the chain $\gamma \in C_2$ of white pixels: (a) original PNG image; (b) exploded solid model of chain $\gamma \in C_2$ as $|\gamma| \in E^3$

Figure 10: Subimage 128 ×128 example: (a) initial (white) chain $\gamma \in C_2$; (b) extraction of boundary chain $\beta = \partial_2(\gamma) \in C_1$ of the chain γ; (c) the down(boundary) chain $\eta = VE(\partial_2(\gamma)) \in C_0$; (d) the up(up(down(boundary))) chain $\beta_2 = FV(VE(\partial_2(\gamma))) \in C_2 \equiv (F\ V \circ VE \circ \partial_2)(\gamma)$; (e) from the chain $\beta_2 = (F\ V \circ VE \circ \partial_2)(\gamma)$ we derive the dilation chain = $\beta_2 - \gamma$ (yellow colour), and the erosion chain = $\beta_2 \cap \gamma$ (cyan colour).

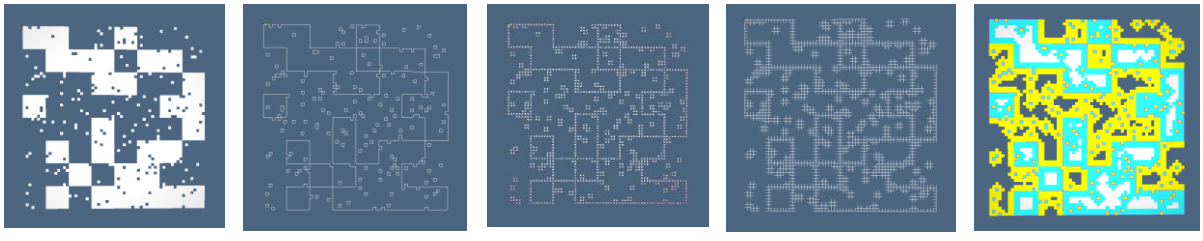


Figure 11: Subimage 64×64: (a) chain $\gamma \in C_2$; (b) chain $\beta = \partial_2(\gamma) \in C_1$; (c) chain $(VE \circ \partial_2)(\gamma) \in C_0$; (d) $(EV \circ VE \circ \partial_2)(\gamma) \in C_1$; (e) $\beta_2 = (FE \circ EV \circ VE \circ \partial_2)(\gamma) \in C_2$, giving the *dilation* chain $DIL(\beta_2)(\gamma) = \beta_2 - \gamma$ (yellow colour), and the *erosion* chain $ERO(\beta_2)(\gamma) = \beta_2 \cap \gamma$ (cyan colour).

The stored content of any image chain (subset of image elements – either pixels or voxels) shall be seen as a cochain associated to the given chain, and its discrete integrals (e.g. the volume, or surface area, or inertia moments) or other chains to be computed by means of discrete differential operators, shall be computed accordingly, by the proper SpMspV multiplication, taking appropriate benefits by advanced computational hardware, e.g., by GPGPU methods.

In conclusion, we would like to remark that any model mesh, either of the internal or the external surface, using either unstructured (triangle, tetrahedra) or structured (quadrilaterals, hexaedra) or more general convex cells, can be stored on computer media, or transmitted on communication networks, using LAR as representation of topology and as support for curved geometry.

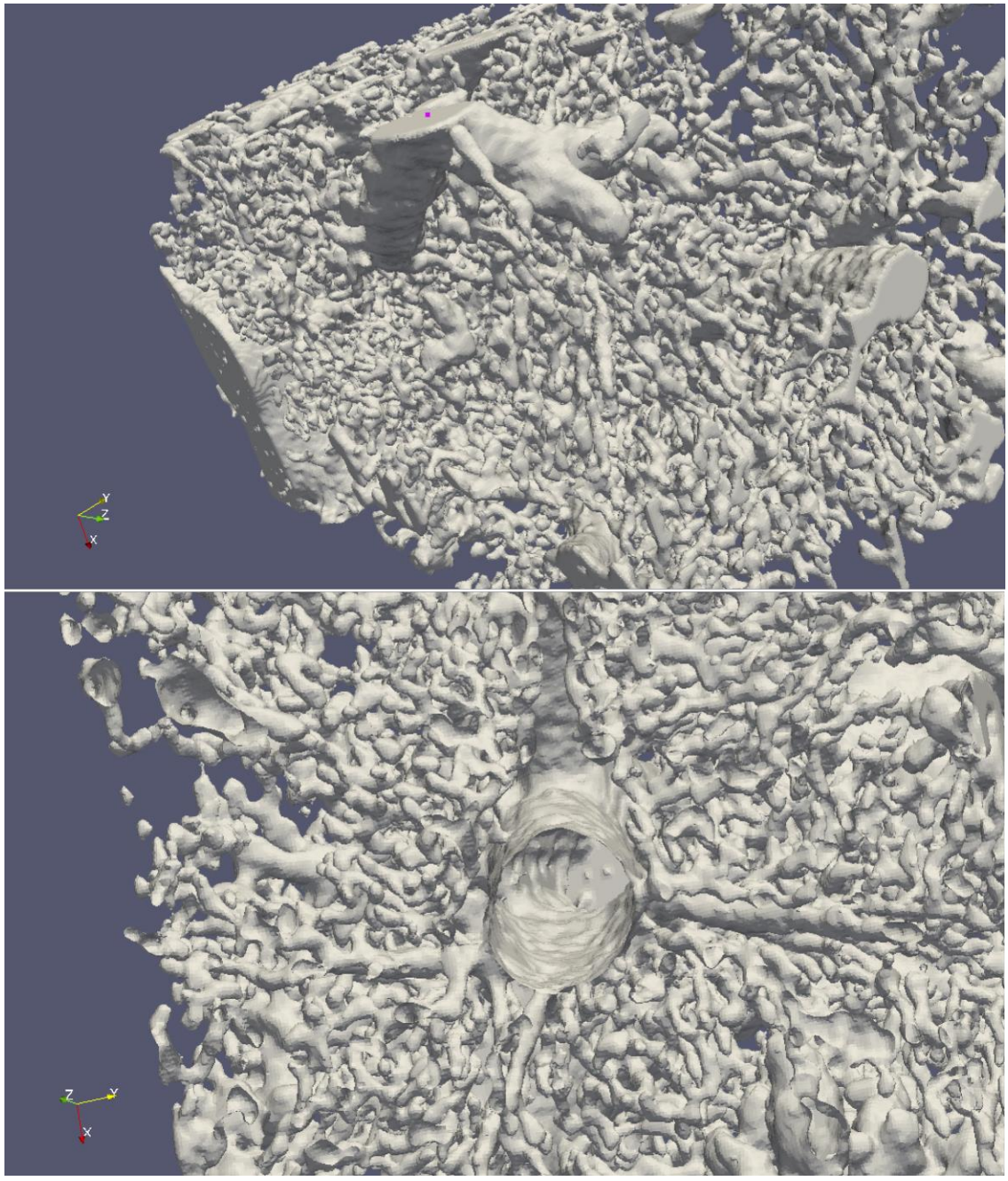


Figure 12: The (a) exterior and (b) a section of the vein system within the liver sample.

4.3 Liver's vein portal system extraction

In human anatomy, the hepatic portal system is the system of veins comprising the hepatic portal vein and its tributaries. It is also called the portal venous system. A portal venous system occurs when a capillary bed pools into another capillary bed through veins, without first going through the heart. When unqualified, "portal venous system" often refers to the hepatic portal system. For this reason, "portal vein" most commonly refers to the hepatic portal vein. Some images of the topologically exact and geometrically accurate exporting of the portal vein system as a solid model, using a closed and valid *boundary representation* generated from a 3D image of a sample of pig liver are shown in Figures 12a, 12b, and 13.

Figure 13: Some images of a vein interior. Notice the mesh of quadrilaterals

5 CONCLUSION

We have shown in this paper the generation of *topologically valid* and geometrically accurate boundary models of the vein system inside the liver, using LAR, a general-purpose framework for solid and geometric modeling. LAR is an ongoing initiative to rethink the foundations of solid modeling, with the aim of simplifying and generalizing its data representation and to disentangle its main algorithms—like a sort of *MapReduce* for geometric computing—so providing a computational framework for the "new world" of big geometric data over cloud- and web-based infrastructures.

This project is currently showing its usefulness in model extraction from 3D medical imaging, and in supporting the simplified generation of building models for *indoor mapping* and the *Internet of Things* [37]. The LAR prototype implementation in Python is work-in-progress. We are currently engaged with a novel simplified approach to Boolean operations with cellular complexes, that is both multidimensional, variadic and suitable of a distributed implementation over big geometric datasets. Next stage will be the LAR consolidation as a *Haskell* library, in order to give a strongly-typed reference implementation with well-defined *API* and function signatures, to be subsequently compiled to *C* and to *JavaScript*, the language of the modern web platforms.

ACKNOWLEDGEMENTS

The authors A.P. and A.DC. would gratefully thank the IEEE-SA WG P3333.2 — Standard for Three-Dimensional Model Creation Using Unprocessed 3D Medical Data: Young Lae Moon (president), Kazuomi Sugamoto (vice president), Jaekeun Kwak (segretary), Dae Won Yoon (former segretary), Dong Sun Shin, Dae Ok Kim, Dae Hyun Lee, Jooyoung Kim, Valerio Pascucci, and several other collegues, for their interest and encouragement to develop the LAR scheme. They also acknowledge the help of Vadim Shapiro for inspiration and long-standing joint work on this topic, and Federico Spini, Enrico Marino, VittorioCecchetto for contributions with the initial experiments and implementation. The LAR development was supported in part by grants from SOGEI, the ICT company of the Italian Ministry of Economy and Finance.

6 REFERENCES

[1] Ala, S. R.: Performance anomalies in boundary data structures, IEEE Computer Graphics & Applications, 12(2), 1992, 49–58. http://dx.do.org/10.1109/38.124288

[2] Bajaj, C.; DiCarlo, A.; Paoluzzi, A.: Proto-plasm: A parallel language for scalable modeling of biosystems, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1878), 2008, 3045–3065. http://dx.doi.org/10.1098/rsta.2008.0076

[3] Bajaj, C; Paoluzzi, A.; Scorzelli, G.; Progressive conversion from B-rep to BSP for streaming geometric modeling, Computer-Aided Design and Applications, 3(5-6), 2006.

[4] Basak, T.: Combinatorial cell complexes and Poincaré duality, Geometriae Dedicata, 147(1), 2010, 357–387. http://dx.doi.org/10.1007/s10711-010-9458-y

[5] Baumgart, B. G.: Winged edge polyhedron representation. Technical Report Stan-CS-320, Stanford, CA, USA, 1972.

[6] Bowyer, A.: SvLis Set-theoretic Kernel Modeller: Introduction and User Manual, Information Geometers, 1995.

[7] Bowyer A. & Geometric Modelling Society: Introducing Djinn A Geometric Interface for Solid Modelling. Information Geometers [for] the Geometric Modelling Society, 1995.

[8] Braid, I. C.: The synthesis of solids bounded by many faces, Communications of the ACM, 18(4), 1975, 209–216. http://dx.doi.org/10.1145/360715.360727

[9] Buluç, A.; Gilbert, J. R.: Parallel sparse matrix-matrix multiplication and indexing: Implementation and experiments, SIAM Journal of Scientific Computing (SISC), 34(4), 2012, 170–191. http://dx.doi.org/10.1137/110848244

[10] DiCarlo, A.; Milicchio, F.; Paoluzzi, A.; Shapiro, V.: Chain-based representations for solid and physical modeling, IEEE Transactions on Automation Science and Engineering, 6(3), 2009, 454–467.

[11] DiCarlo, A.; Paoluzzi, A.; Shapiro, V.: Linear algebraic representation for topological structures. Computer-Aided Design, 46, 2014, 269–274. http://dx.doi.org/10.1016/j.cad.2013.08.044

[12] Dobkin, D. P.; Laszlo, M. J.: Primitives for the manipulation of three-dimensional subdivisions, Proc. of the third annual symposium on Computational geometry, SCG '87, 86–99, ACM, 1987.

[13] Furiani, F.; Paoluzzi, C.; Paoluzzi, A.: Algebraic mining of solid models from images. ESA-EUSC-JRC: Image Information Mining 2014 Conference, Bucharest, ROMANIA, 2014.

[14] Gomes, A.; Middleditch, A.; Reade, C.: A mathematical model for boundary representations of *n*-dimensional geometric objects, Proc. of the fifth ACM symposium on Solid modeling and applications, 270–277. ACM, 1999.

[15] Guibas L.; Stolfi, J.: Primitives for the manipulation of general subdivisions and the computation of Voronoi, ACM Transactions on Graphics, 4(2), 1985, 74–123. http://dx.doi.org/10.1145/282918.282923

[16] Hoffmann, C. M.; Kim, K. J.: Towards valid parametric cad models, Computer- Aided Design, 33(1), 2001, 81–90. http://dx.doi.org/10.1016/S0010-4485(00)00073-7

[17] Hunter, P.; Coveney, P. V.; de Bono, B.; Diaz, V.; Fenner, J.; Frangi, A. F.; Harris, P.; Hose, R.; Kohl, P.; Lawford, P.; McCormack, K.; Mendes, M.; Omholt, S.; Quarteroni, A.; Skår, J.; Tegner, J.; Thomas, S. R.; Tollis, J.; Tsamardinos, J.; van Beek, J. H. G. M.; Viceconti, M.: A vision and strategy for the virtual physiological human in 2010 and beyond, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 368(1920), 2010, 2595–2614. http://dx.doi.org/10.1098/rsta.2010.0048

[18] Jiřík M.; Ryba T.; Svobodová M.; Mírka H.; Liska V.: Lisa: Liver surgery analyser software development, 11th World Congress on Computational Mechanics (WCCM XI), Barcelona, SPAIN, 2014.

[19] Lee, S. H.; Lee, K. W; Partial entity structure: a compact non-manifold boundary representation based on partial topological entities, Proc. of the sixth ACM Symposium on Solid modeling and applications, 159–170, ACM, 2001.

[20] Moon, Y. L.; Sugamoto, K.; Paoluzzi, A.; DiCarlo, A.; Kwak, J.; Shin, D.S.; Kim, D. O.; Lee, D. H.; Kim. J. Y.: Standardizing 3d medical imaging, IEEE Computer, 47(4), 2014, 76–79. http://dx.doi.org/10.1109/MC.2014.103

[21] Paoluzzi, A.: Bernardini, F.; Cattani, C.; Ferrucci, V.: Dimension - independent modeling with simplicial complexes. ACM Transactions on Graphics, 12(1), 1993, 56–102. http://dx.doi.org/10.1145/169728.169719
[22] Paoluzzi, A.: Geometric Programming for Computer Aided Design, John Wiley & Sons, 2003.
[23] Paoluzzi, A.; Ramella, M.; Santarelli, A.: Boolean algebra over linear polyhedral, Computer-Aided Design, 21(10), 1989, 474–484. http://dx.doi.org/10.1016/0010-4485(89)90055-9
[24] Paoluzzi, A.; Pascucci, V.; Vicentino, M.: Geometric programming: a programming approach to geometric design, ACM Transactions on Graphics, 14(3), 1995, 266–306. http://dx.doi.org/10.1145/212332.212349
[25] Pascucci, V.; Ferrucci, V.; Paoluzzi, A.: Dimension-independent convex- cell based HPC: representation scheme and implementations issues. Proc. of the third ACM Symposium on Solid Modeling and Applications, 163–174, ACM, 1995.
[26] Pratt, M. J.; Anderson, B. D.: A shape modelling API for the STEP standard, Fourteenth International Conference on Atomic Physics, 1–7, 1994.
[27] Raghothama, S.; Shapiro, V.: Boundary representation deformation in parametric solid modeling, ACM Transactions on Graphics, 17(4), 1998, 259–286. http://dx.doi.org/10.1145/293145.293148
[28] Raghothama, S.; Shapiro, V.: Consistent updates in dual representation systems, Proc. of the fifth ACM symposium on Solid modeling and applications, 65–75, ACM, 1999.
[29] Requicha, A. A. G.; Voelcker, H. B.: Constructive solid geometry, Technical Report TM-25, Production Automation Project, Univ. of Rochester, 1977.
[30] Requicha, A. A. G.: Representations for rigid solids: Theory, methods, and systems, ACM Computing Surveys, 12(4), 1980, 437–464. http://dx.doi.org/10.1145/356827.356833
[31] Ricken, T.; Dahmen, U.; Dirsch,O.: A biphasic model for sinusoidal liver perfusion remodeling after outflow obstruction, Biomechanics and Modeling in Mechanobiology, 9(4), 2010, 435–450. http://dx.doi.org/10.1007/s10237-009-0186-x
[32] Rossignac, J. R.; O'Connor, M. A.: SGC: a dimension-independent model for pointsets with internal structures and incomplete boundaries, Geometric modeling for product engineering, North-Holland, 1990.
[33] Rossignac, J. R.; Requicha, A. A. G.: Constructive non-regularized geometry. Computer-Aided Design, 23(1), 1991, 21–32. http://dx.doi.org/10.1016/0010-4485(91)90078-B
[34] Shapiro. V.; Representations of semi-algebraic sets in finite algebras generated by space decompositions, PhD thesis, Cornell University, 1991. UMI Order No. GAX91-31407.
[35] Shapiro, V.; Vossler, D. L.: What is a parametric family of solids?, Proc. of the third ACM symposium on Solid modeling and applications, 43–54, ACM, 1995.
[36] Silva, C. E.; Alternative definitions of faces in boundary representations of solid objects, Technical Report TM-36, Production Automation Project, Univ. of Rochester, 1981.
[37] Virgadamo, M.; Sportillo, M.; Spini, F.; Paoluzzi, A.; Marino, E.; Bottaro, A.: Cartographic documents for interactive web modeling of indoor mapping (submitted paper) 2015.
[38] Weiler, K. J.: Edge-based data structures for solid modeling in curved-surface environments, IEEE Computer Graphics and Applications, 5(1), 1985, 21–40. http://dx.doi.org/10.1109/MCG.1985.276271
[39] Williams, S.; Oliker, L.; Vuduc, R.; Shalf, J.; Yelick, K.; Demmel, J.: Optimization of sparse matrix-vector multiplication on emerging multicore platforms, Proc. of ACM/IEEE conference on Supercomputing, 38:1–38:12, ACM, 2007.
[40] Woo, T.: A combinatorial analysis of boundary data structure schemata, IEEE Computer Graphics & Applications, 5(3), 1985, 19–27. http://dx.doi.org/10.1109/MCG.1985.276337