Geometric & Graphics Programming Lab: Lecture 14

Alberto Paoluzzi

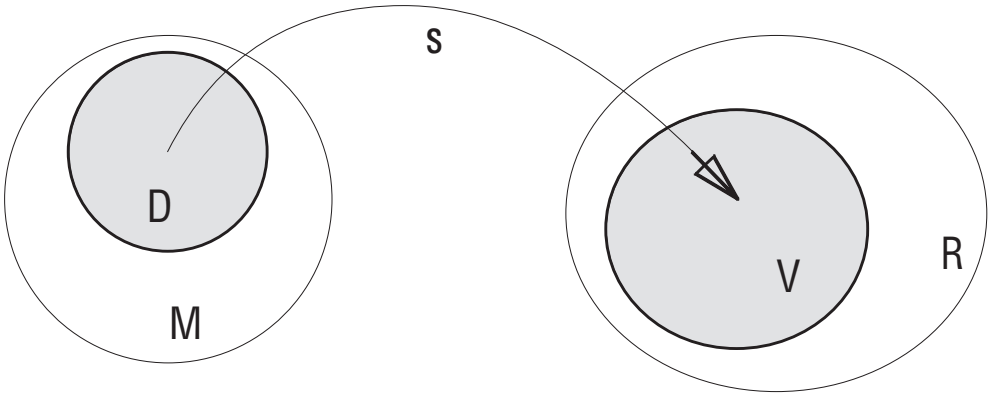
November 18, 2016

Outline: Cell complexes (LAR)

Solid Modeling

Representation scheme: definition

mapping $s : M \rightarrow R$ from a space M of mathematical models to a space R of computer representations



1. The M set contains the mathematical models of the class of solid objects the scheme aims to represent
2. The R set contains the symbolic representations, i.e. the proper data structures, built according to a suitable grammar

A. Requicha, Representations for Rigid Solids: Theory, Methods, and Systems, ACM Comput. Surv., 1980.

V. Shapiro, Solid Modeling, In Handbook of Computer Aided Geometric Design, 2001

Representation schemes

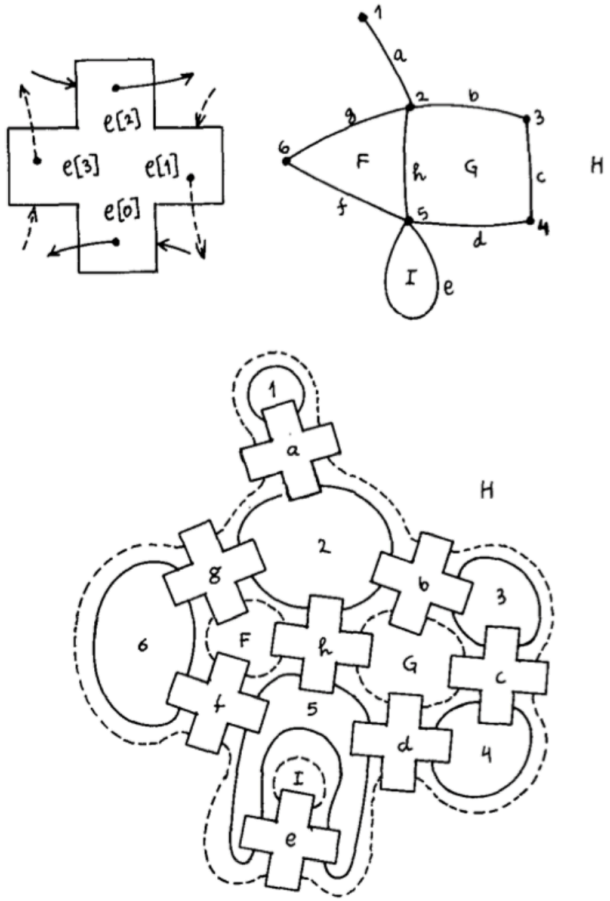
Most of such papers introduce or discuss one or more representation schemes . . .

1. Requicha, ACM Comput. Surv., 1980 [?]
2. Requicha & Voelcker, PEP TM-25, 1977, [?]
3. Rossignac & Requicha, Comput. Aided Des., 1991, [?]
4. Bowyer, SVLIS, 1994, [?]
5. Baumgart, Stan-CS-320, 1972, [?]
6. Braid, Commun. ACM, 1975, [?]
7. Dobkin & Laszlo, ACM SCG, 1987, [?]
8. Guibas & Stolfi, ACM Trans. Graph., 1985, [?]
9. Woo, IEEE Comp. Graph. & Appl., 1985, [?]
10. Yamaguchi & Kimura, Comp. Graph. & Appl., 1995, [?]
11. Gursoz & Choi & Prinz, Geom.Mod., 1990, [?]
12. S.S.Lee & K.Lee, ACM SMA, 2001, [?]
13. Rossignac & O'Connor, IFIP WG 5.2, 1988, [?]
14. Weiler, IEEE Comp. Graph. & Appl., 1985, [?]
15. Silva, Rochester, PEP TM-36, 1981, [?]
16. Shapiro, Cornell Ph.D Th., 1991, [?]
17. Paoluzzi et al., ACM Trans. Graph., 1993, [?]
18. Pratt & Anderson, ICAP, 1994, [?]
19. Bowyer, Djinn, 1995, [?]
20. Gomes et al., ACM SMA, 1999, [?]
21. Raghothama & Shapiro, ACM Trans. Graph., 1998, [?]
22. Shapiro & Vossler, ACM SMA, 1995, [?]
23. Hoffmann & Kim, Comput. Aided Des., 2001, [?]
24. Raghothama & Shapiro, ACM SMA, 1999, [?]
25. DiCarlo et al., IEEE TASE, 2008, [?]
26. Bajaj et al., CAD&A, 2006, [?]
27. Pascucci et al., ACM SMA, 1995, [?]
28. Paoluzzi et al., ACM Trans. Graph., 1995, [?]
29. Paoluzzi et al., Comput. Aided Des., 1989, [?]
30. Ala, IEEE Comput. Graph. Appl., 1992, [?]

and much more . . .

Representation scheme: Quad-Edge data structure

(Guibas & Stolfi, ACM Transactions on Graphics, 1985)



(a) Edge record showing Next links.
(b) A subdivision of the sphere.
(c) Data structure for the subdivision

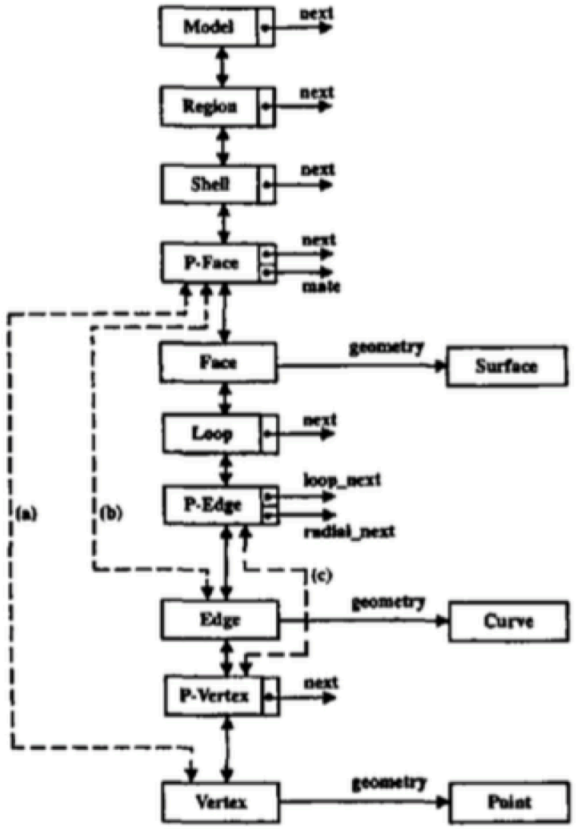
Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams

largely used in computational geometry algorithms and in geometric libraries

Representation scheme: Partial-Entity data structure

(Sang Hun Lee & Kunwoo Lee, ACM Solid Modeling, 2001)

Compact Non-Manifold Boundary Representation Based on Partial Topological Entities

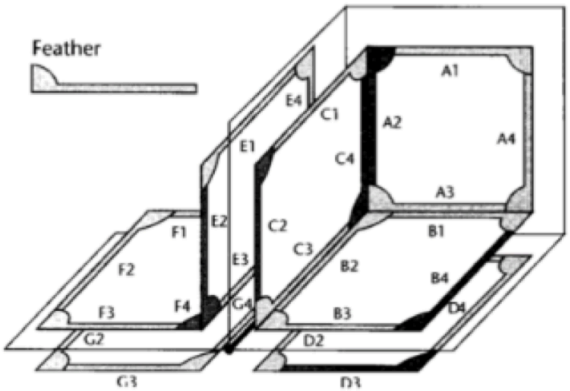



```
class Entity {
   int        _id;
   Attribute  *_attribute;
};
class Model : public Entity {
   Model      *_next;          // next model
   Region     *_region;        // list of regions
};
class Region : public Entity {
   Region     *_next;          // link field of the region list of a model
   Model      *_model;         // parent model
   Shell      *_shell;         // peripheral shell
};
class Shell : public Entity {
   Shell      *_next;          // next void shell
   Region     *_region;        // parent region
   Pface      *_pface;         // partial face
};
class Pface : public Entity {  // partial face (p-face) class
   Pface      *_next;          // next p-face
   Shell      *_shell;         // parent shell
   Entity     *_child;         // child entity: a face, an edge, or a vert
   Orient     _orient;         // orientation flag w.r.t. the face normal
   Pface      *_mate;          // mate p-face
};
class Face : public Entity {
   Pface      *_pface;         // one of two incident p-faces
   Loop       *_loop;          // peripheral loop
   Surface    *_geometry;      // surface
};
```

```
class Loop : public Entity {
   Loop       *_next;          // next hole loop
   Face       *_face;          // parent face
   Pedge      *_pedge;         // a p-edge in a loop
};
class Pedge : public Entity {  // partial edge (p-edge) class
   Loop       *_loop;          // parent loop
   Entity     *_child;         // child entity: an edge or a p-vertex
   Orient     *_orient;        // orientation flag w.r.t. the edge direction
   Pvertex    *_pvertex;       // start p-vertex
   Pedge      *_looped_prev;   // previous p-edge in the loop cycle
   Pedge      *_looped_next;   // next p-edge in the loop cycle
   Pedge      *_radial_prev;   // previous p-edge in the radial cycle
   Pedge      *_radial_next;   // next p-edge in the radial cycle
};
class Edge : public Entity {
   Entity     *_parent;        // parent entity: a p-edge or a p-face
   Pvertex    *_pvertex[2];    // two end p-vertices
   Curve      *_geometry;      // curve
};
class Pvertex : public Entity { // partial vertex (p-vertex) class
   Pvertex    *_next;          // another p-vertex associated with _vertex
   Entity     *_parent;        // parent entity: an edge or a p-edge
   Vertex     *_vertex;        // mother vertex
};
class Vertex : public Entity {
   Entity     *_parent;        // parent entity: a p-vertex or a p-face
   Point      *_geometry;      // position
};
```

Representation scheme: Coupling Entities data structure

(Yamaguchi & Kimura, IEEE Computer Graphics and Applications, 1995)



Fan
B4 & D3 :D3 = FM(B4), B4 = FM(D3)

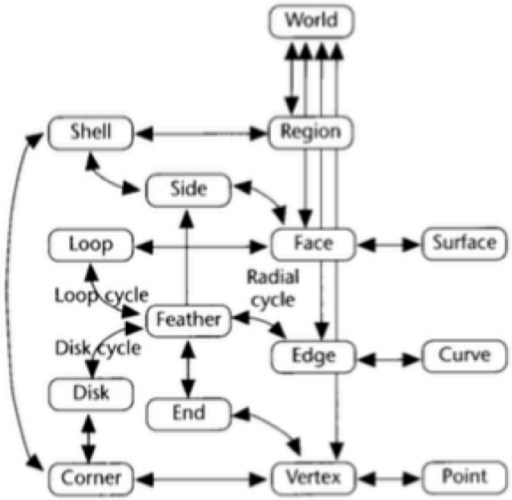
Blade
C2 & E2 :E2 = BM(C2), C2 = BM(E2)

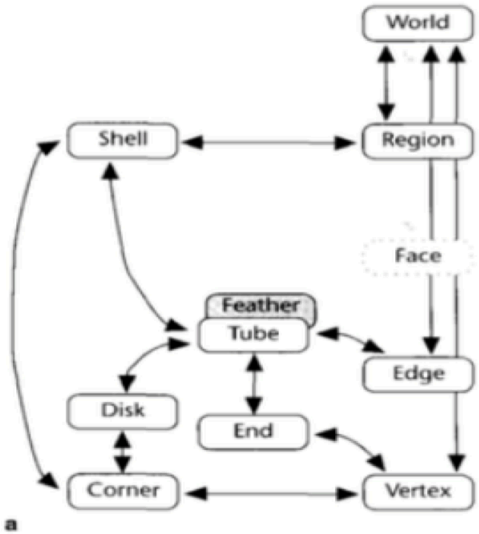
Wedge
A2 & C4 :C4 = WM(A2), A2 = WM(C4)

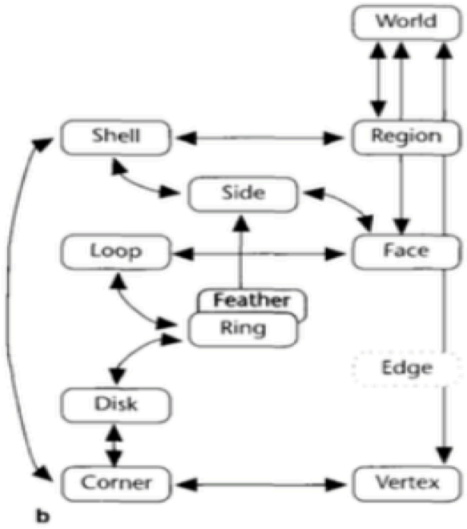
Loop cycle
A1>>A2>>A3>>A4 :A2 = CCL(A1), A3 = CCL(A2)
A4>>A3>>A2>>A1 :A1 = CL(A2), A2 = CL(A3)

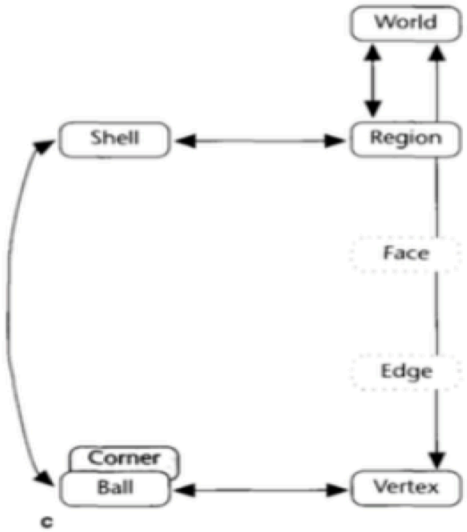
Radial cycle
D2>>C3>>F4 :C3 = CCR(D2), F4 = CCR(C3)
F4>>C3>>D2 :D2 = CR(C3), C3 = CR(F4)

Disk cycle
A3>>C4>>B2 :C4 = CCD(A3), B2 = CCD(C4)
B2>>C4>>A3 :A3 = CD(C4), C4 = CD(B2)









Linear Algebraic Representation (LAR)

LAR: Mathematical models

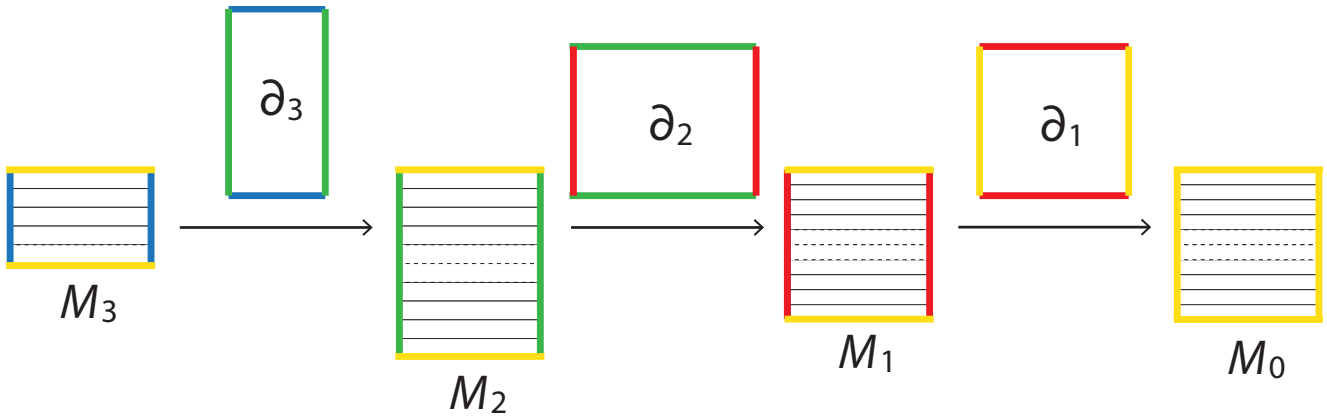
Space of CW-complexes (cellular complexes)

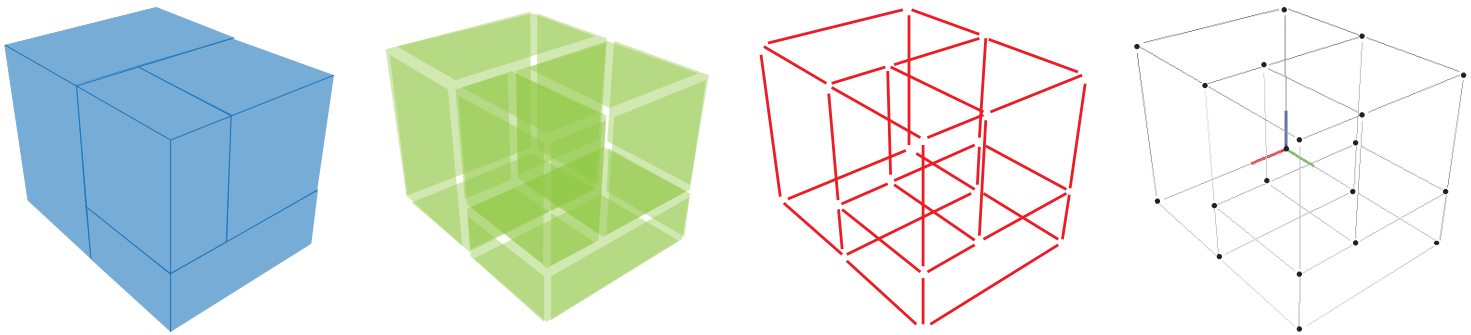
Chain complex (of chain spaces)

Sequence of linear spaces (over $\mathbb{Z}_2$) of d-cell **subsets**

A chain complex C is a complex of chain spaces and boundary maps:

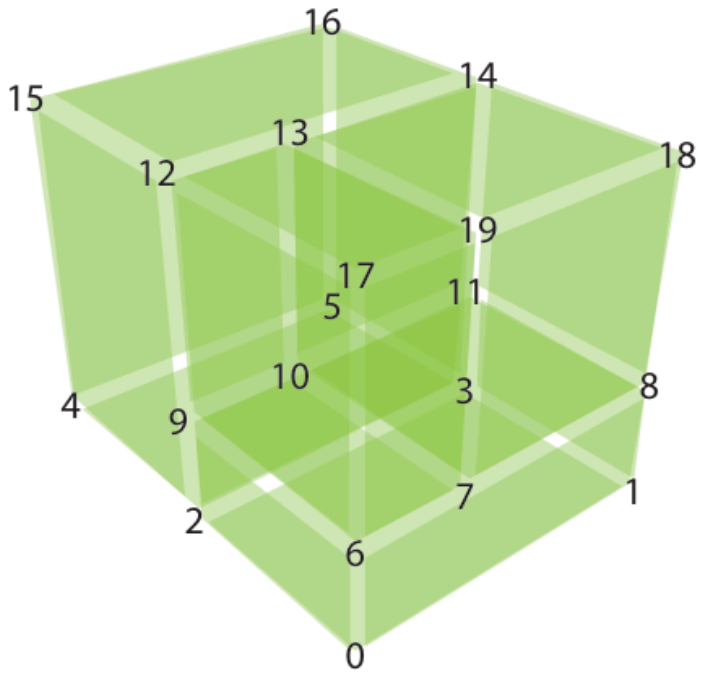
Unit d-chains (single d-cell subsets), give the standard bases (M_d rows) of d-chain spaces





Characteristic matrices of d-chain spaces

Matrix representation of the basis — d-**cells as subsets of vertices**



$$M_3 = \begin{pmatrix} 0&0&1&1&1&1&0&0&0&1&1&1&1&1&1&1&1&0&0&0 \\ 1&1&1&1&0&0&1&1&1&1&1&1&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&1&0&1 \\ 0&0&0&0&0&0&0&1&1&0&1&1&0&1&1&0&0&0&1&1 \end{pmatrix}$$

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

LAR: computer representation

Sparse matrix representations

Basic representations

A few basic representation of topology are used in LARCC.

They include some common sparse matrix representations:

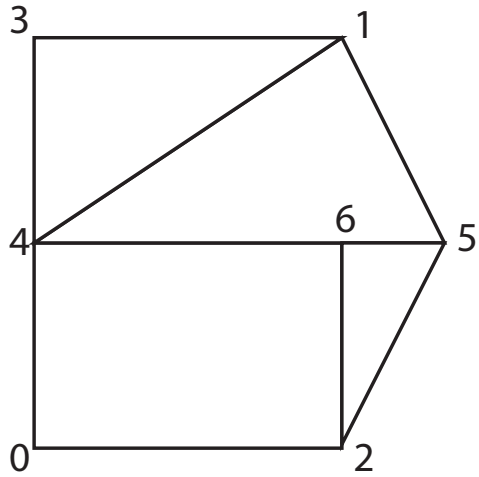
- CSR (Compressed Sparse Row),
- CSC (Compressed Sparse Column),
- COO (Coordinate Representation),
- and BRC (Binary Row Compressed)

Data definition

- We can identify each cell with its set of vertices.
- Thus, to define S, we start from the vertex set V, and specify the vertex subsets which correspond to the cells, and the rank of each cell.
- The partial order is induced by set inclusion.

Characteristic matrix $M_2 : C_0 \to C_2$

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



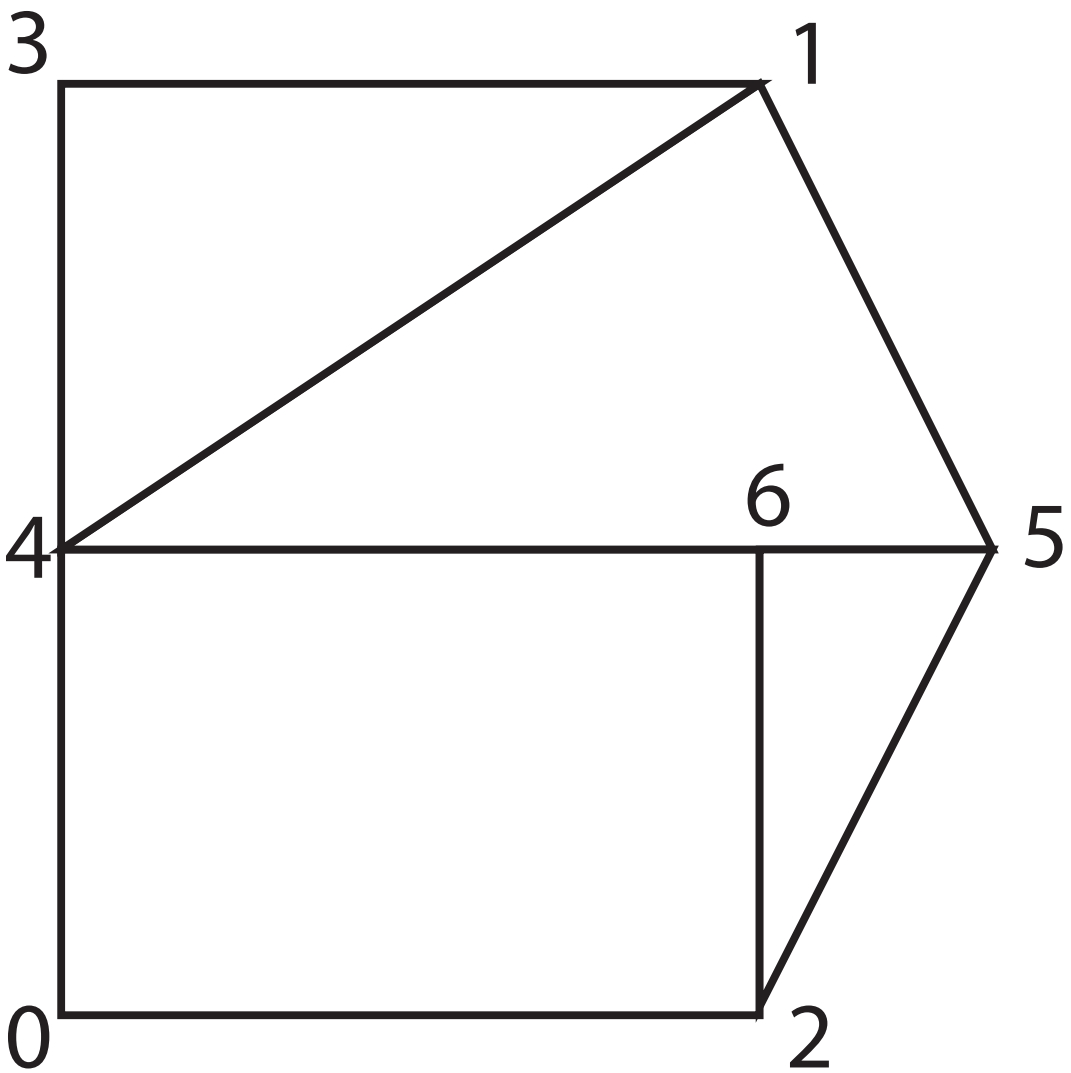
BRC (Binary Row Compressed) representation

List of list of integer typically used for input of cell complexes

Let $A = (a_{i,j} \in \{0,1\})$ be a characteristic matrix.

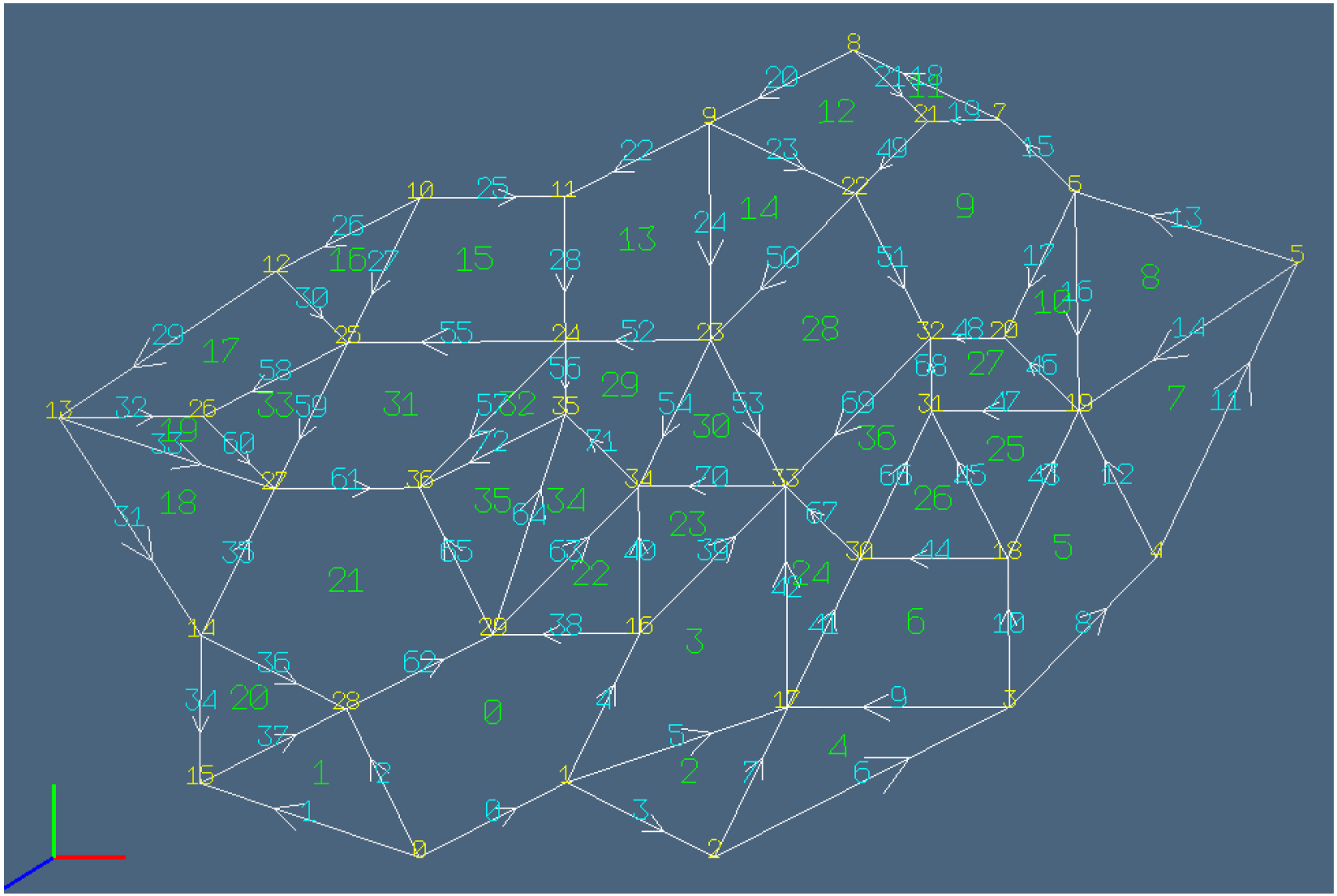
$$A = \begin{pmatrix} 0,1,0,0,0,0,0,1,0,0 \\ 0,0,1,0,0,0,0,0,0,0 \\ 1,0,0,1,0,0,0,0,0,1 \\ 1,0,0,0,0,0,1,0,0,0 \\ 0,0,0,0,0,1,1,1,0,0 \\ 0,0,1,0,1,0,0,0,1,0 \\ 0,0,0,0,0,0,0,0,0,0 \\ 0,1,0,0,0,0,0,1,0,1 \\ 0,0,0,1,0,0,0,0,1,0 \\ 0,1,1,0,1,0,0,0,0,0 \end{pmatrix} \quad \mapsto \quad \texttt{BRC}(A) = \begin{array}{l} \texttt{[[1,7],} \\ \texttt{[2],} \\ \texttt{[0,3,9],} \\ \texttt{[0,6],} \\ \texttt{[5,6,7],} \\ \texttt{[2,4,8],} \\ \texttt{[],} \\ \texttt{[1,7,9],} \\ \texttt{[3,8],} \\ \texttt{[1,2,4]]} \end{array}$$

The idea: generalisation of abstract simplicial sets



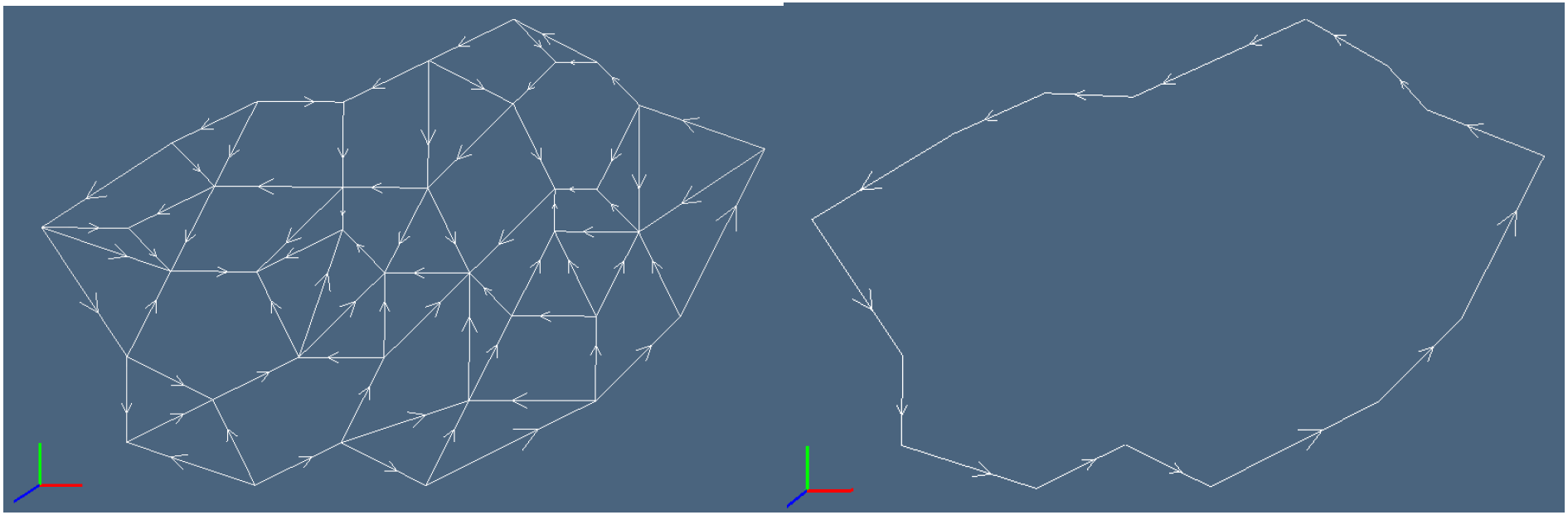
n-cells by
their vertices:

```
[[0,2,4,6],
[1,4,5,6],
[1,5,6],
[2,5,6]]
```



The bare minimum data

with full awareness of topology



```
V=[[5,0],[7,1],[9,0],[13,2],[15,4],[17,8],[14,9],[13,10],[11,11],[9,10],[5,9],[7,
9],[3,8],[0,6],[2,3],[2,1],[8,3],[10,2],[13,4],[14,6],[13,7],[12,10],[11,9],[9,7],
[7,7],[4,7],[2,6],[3,5],[4,2],[6,3],[11,4],[12,6],[12,7],[10,5],[8,5],[7,6],[5,5]]
```

```
FV = [[0,1,16,28,29],[0,15,28],[1,2,17],[1,16,17,33],[2,3,17],[3,4,18,19],[3,17,
18,30],[4,5,19],[5,6,19],[6,7,20,21,22,32],[6,19,20],[7,8,21],[8,9,21,22],[9,11,
23,24],[9,22,23],[10,11,24,25],[10,12,25],[12,13,25,26],[13,14,27],[13,26,27],
[14,15,28],[14,27,28,29,36],[16,29,34],[16,33,34],[17,30,33],[18,19,31],[18,30,
31],[19,20,31,32],[22,23,32,33],[23,24,34,35],[23,33,34],[24,25,27,36],[24,35,36],
[25,26,27],[29,34,35],[29,35,36],[30,31,32,33]]
```

Compressed Sparse Row (CSR) matrix storage

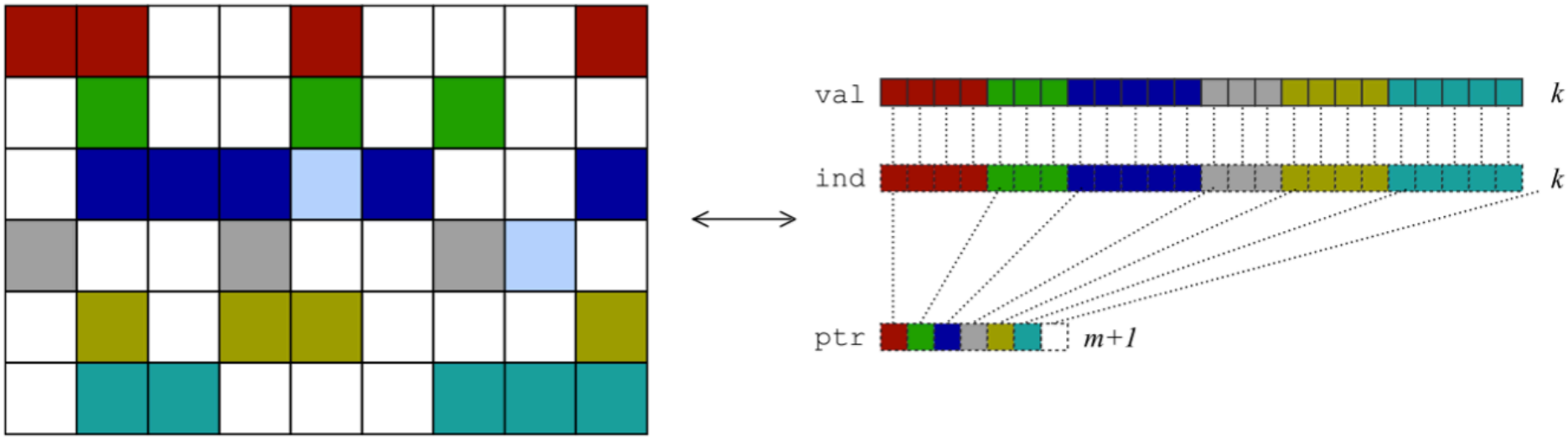
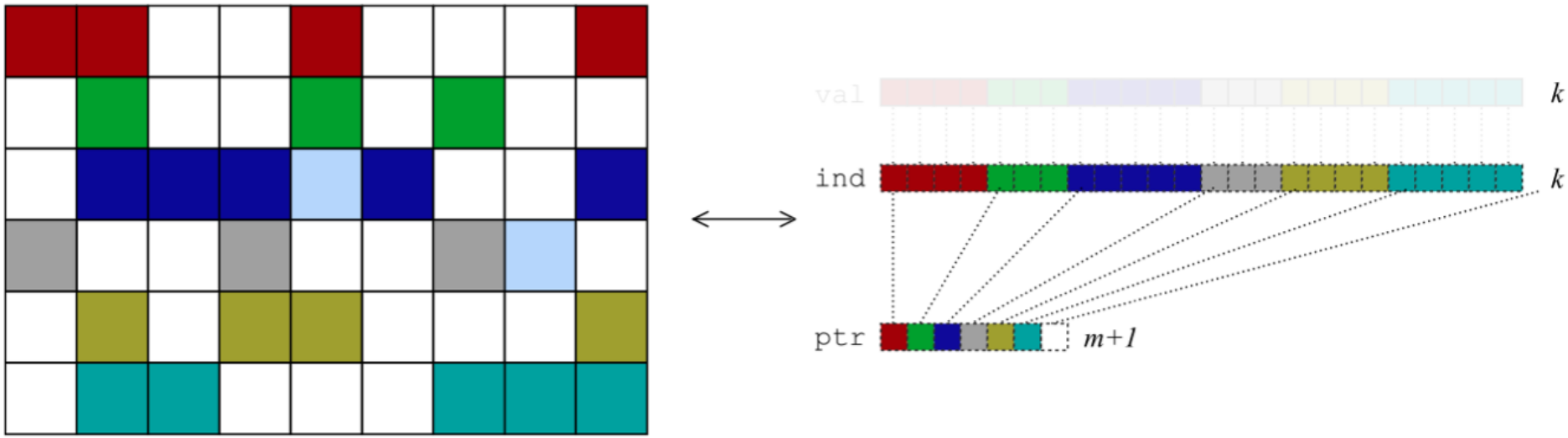


image from

Samuel Williams, Leonid Oliker, Richard Vuduc, John Shalf, Katherine Yelick, and James Demmel, Optimization of sparse matrix-vector multiplication on emerging multicore platforms, Proceedings of the 2007 ACM/IEEE conference on Supercomputing (New York, NY, USA), SC '07, ACM, 2007, pp. 38:1–38:12.

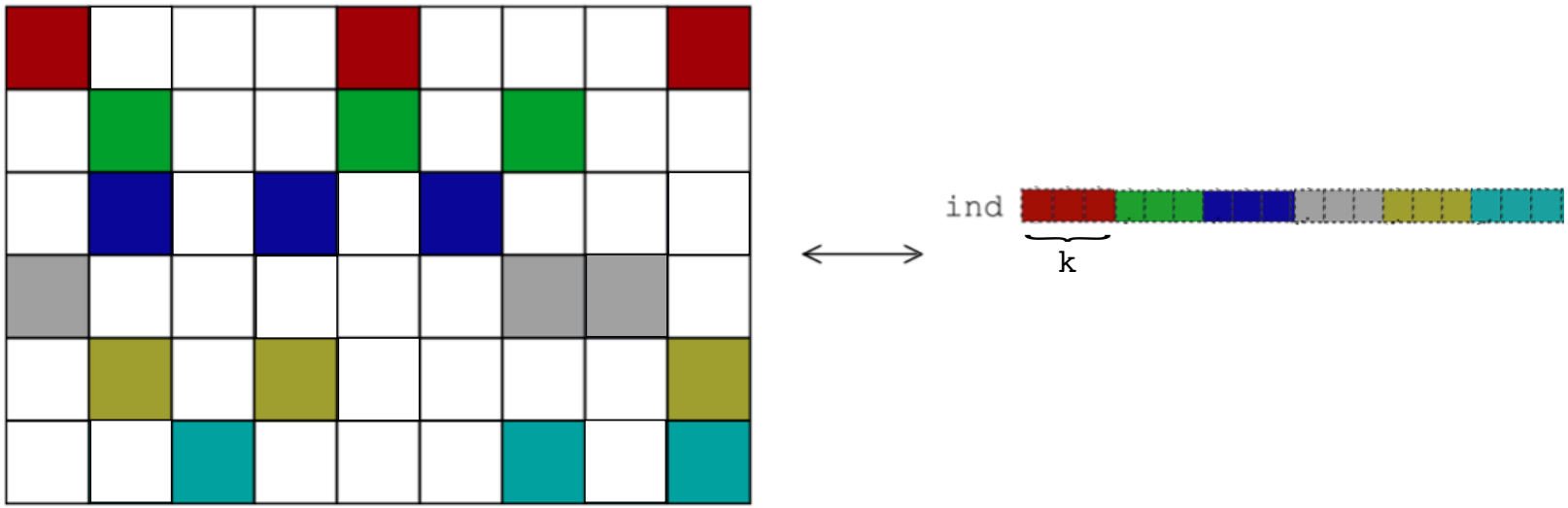
CSR storage of a BINARY matrix



of course, non-zero values (all ones) do not require storage

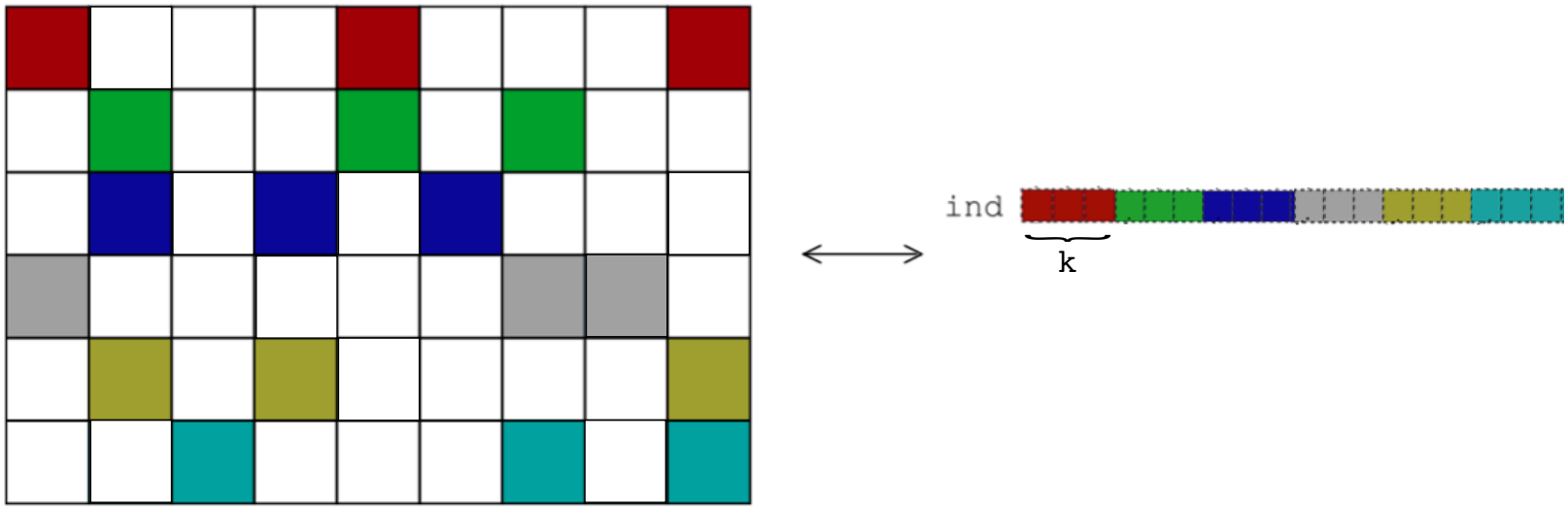
CSR storage of a binary matrix M_d such that

$$M_d \mathbf{1} = \mathbf{k}$$



CSR storage of a binary matrix M_d such that

$$M_d \mathbf{1} = \mathbf{k}$$



important cases:

- regular simplicial d-complexes: $k = d + 1$
- regular cuboidal d-complexes: $k = 2^d$

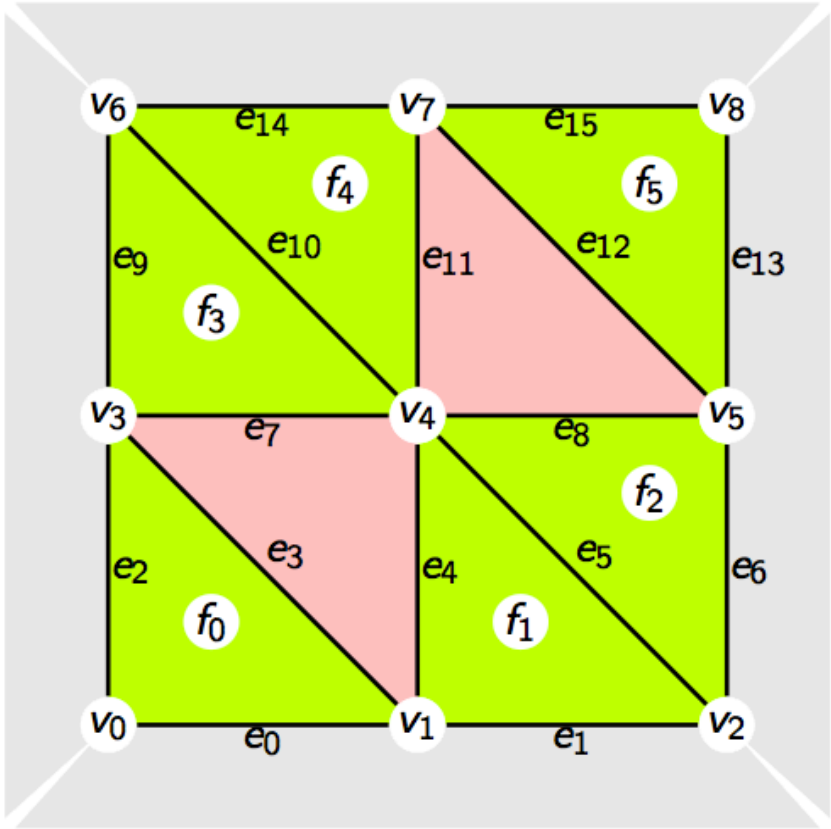
Operations

Facet extraction (1/5)

use the characteristic matrix M_d of a cellular partition Λ_d of $\mathbb{E}^d$, empty cells included

$$2D \Rightarrow d = 2$$

$$M_2 = \begin{pmatrix}
1\,1\,0\,1\,0\,0\,0\,0\,0 \\
0\,1\,1\,0\,1\,0\,0\,0\,0 \\
0\,0\,1\,0\,1\,1\,0\,0\,0 \\
0\,0\,0\,1\,1\,0\,1\,0\,0 \\
0\,0\,0\,0\,1\,0\,1\,1\,0 \\
0\,0\,0\,0\,0\,1\,0\,1\,1 \\
0\,1\,0\,1\,1\,0\,0\,0\,0 \\
0\,0\,0\,0\,1\,1\,0\,1\,0 \\
1\,1\,1\,0\,0\,0\,0\,0\,0 \\
0\,0\,1\,0\,0\,1\,0\,0\,1 \\
0\,0\,0\,0\,0\,0\,1\,1\,1 \\
1\,0\,0\,1\,0\,0\,1\,0\,0
\end{pmatrix}$$

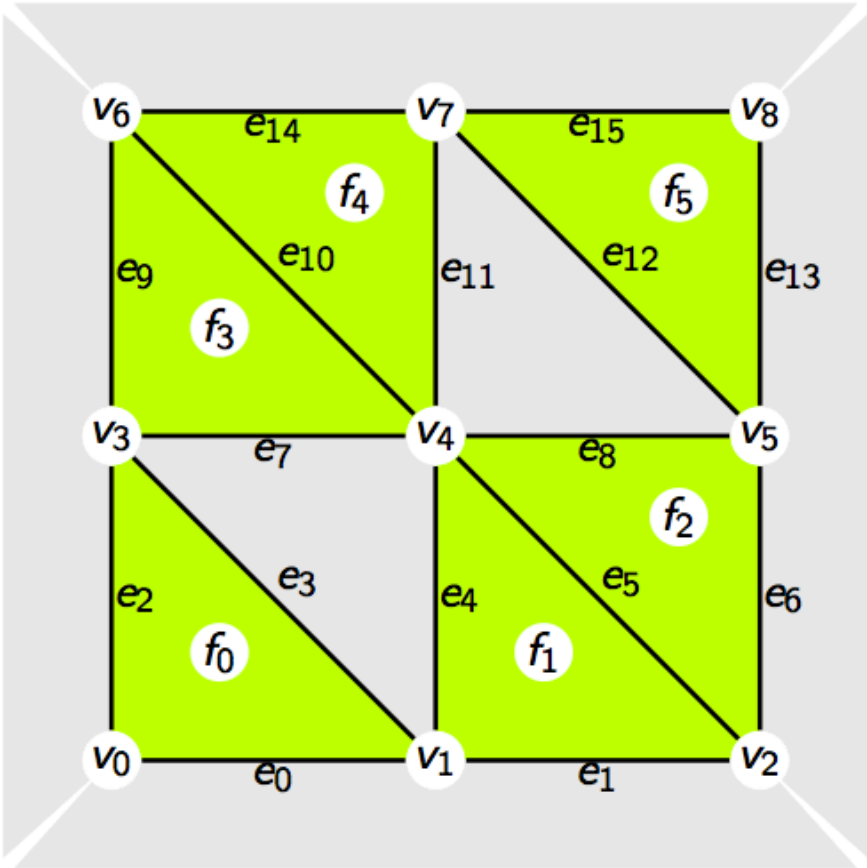


Facet extraction (2/5)

use the characteristic matrix M_d of a cellular partition Λ_d of $\mathbb{E}^d$, empty cells included

$$2D \Rightarrow d = 2$$

$$M_2 = \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,1 \\ 0\,1\,0\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0 \\ 1\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,1 \\ 1\,0\,0\,1\,0\,0\,1\,0\,0 \end{pmatrix}$$



Facet extraction (3/5)

use the characteristic matrix M_d of a cellular partition Λ_d of $\mathbb{E}^d$, empty cells included

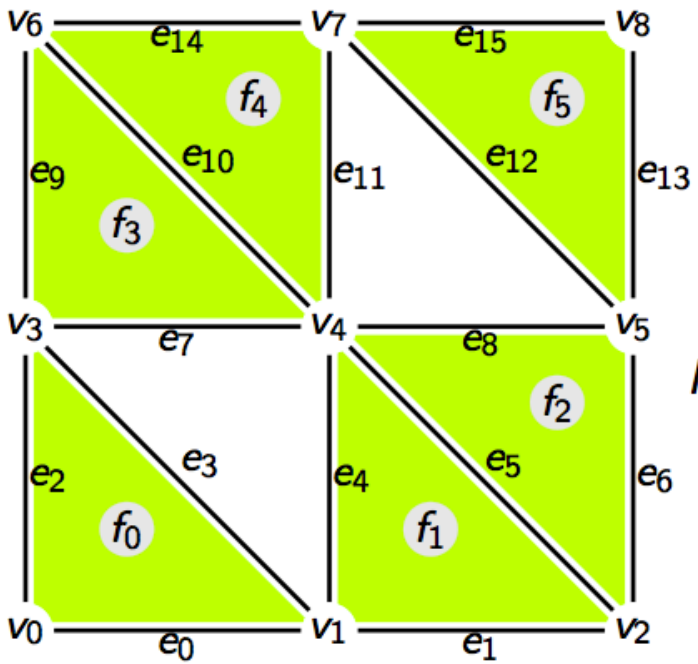
$$A = M_2 M_2^t = \begin{pmatrix}
3 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 2 & 2 & 0 \\
 & 3 & 2 & 1 & 1 & 0 & 2 & 1 & 0 & 0 & 2 & 1 \\
 & & 3 & 1 & 1 & 1 & 1 & 2 & 0 & 0 & 1 & 2 \\
 & & & 3 & 2 & 0 & 0 & 0 & 1 & 2 & 2 & 1 \\
 & & & & 3 & 1 & 0 & 0 & 2 & 1 & 1 & 2 \\
 & & & & & 3 & 0 & 2 & 2 & 0 & 0 & 2 \\
 & & & & & & 3 & 1 & 0 & 1 & 1 & 0 \\
 & & & & & & & 3 & 1 & 0 & 0 & 1 \\
 & & & & sym & & & & 3 & 1 & 0 & 1 \\
 & & & & & & & & & 3 & 1 & 0 \\
 & & & & & & & & & & 3 & 1 \\
 & & & & & & & & & & & 3
\end{pmatrix}$$

$$\sharp(\lambda_d^i \cap \lambda_d^j) = A(i,j) \geq d \ (i \neq j) \ \Rightarrow \ \exists \, \lambda_{d-1} = M_d(i) \wedge M_d(j)$$

Facet extraction (4/5)

use the characteristic matrix M_d of a cellular partition Λ_d of $\mathbb{E}^d$, empty cells included

$$A(0,6) = 2 \quad \Rightarrow \quad (\,110100000\,) \wedge (\,010110000\,) = (\,010100000\,) \quad \Rightarrow \quad e = (v_1, v_3)$$

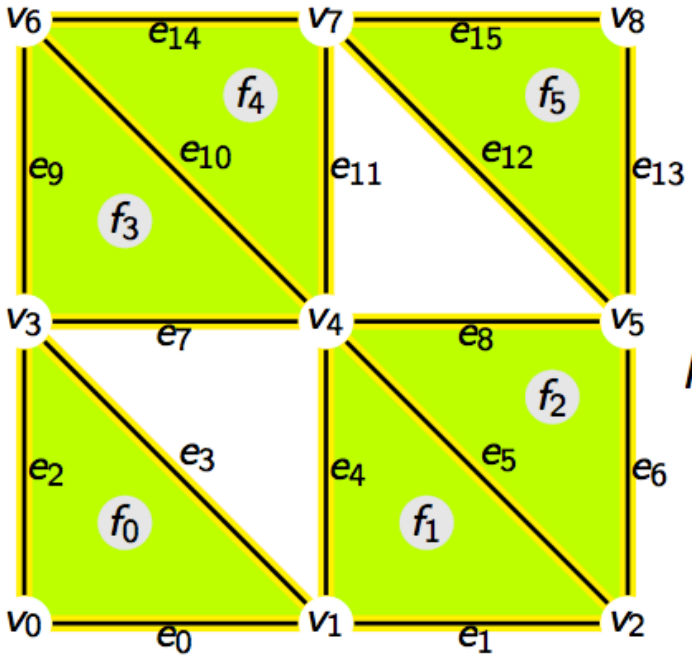


$$M_2 = \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,1 \\ 0\,1\,0\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0 \\ 1\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,1 \\ 1\,0\,0\,1\,0\,0\,1\,0\,0 \end{pmatrix} \Rightarrow M_1 = \begin{pmatrix} 1\,1\,0\,0\,0\,0\,0\,0\,0 \\ 1\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1 \end{pmatrix}$$

Facet extraction (5/5)

use the characteristic matrix M_d of a cellular partition Λ_d of $\mathbb{E}^d$, empty cells included

$$A(0,6) = 2 \quad \Rightarrow \quad (\,110100000\,) \wedge (\,010110000\,) = (\,010100000\,) \quad \Rightarrow \quad e = (v_1, v_3)$$



$$M_2 = \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,1 \\ 0\,1\,0\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0 \\ 1\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,1 \\ 1\,0\,0\,1\,0\,0\,1\,0\,0 \end{pmatrix} \Rightarrow M_1 = \begin{pmatrix} 1\,1\,0\,0\,0\,0\,0\,0\,0 \\ 1\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1 \end{pmatrix}$$

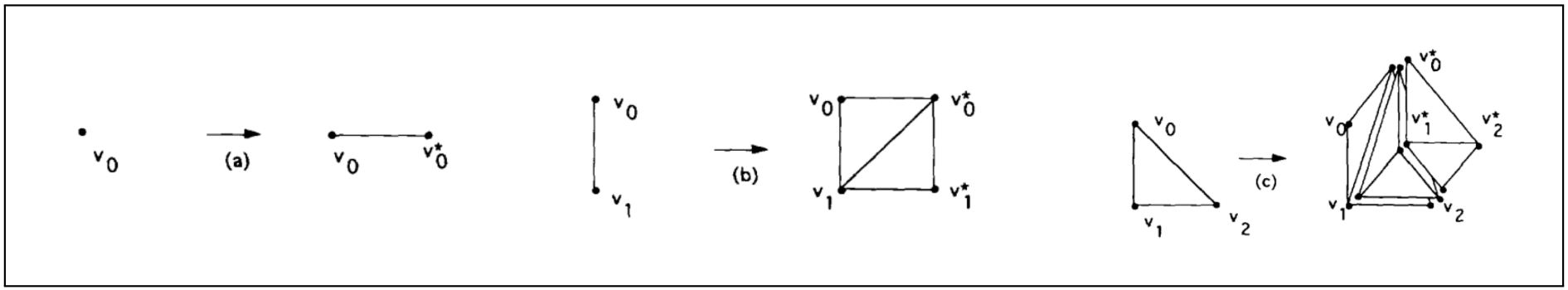
Simplicial extrusion (1/3)

Optimal combinatorial algorithm [Ferrucci & Paoluzzi, CAD, 1991]

for each d-simplex

$$\sigma^d = (v_0, v_1, \ldots, v_d)$$

in the input complex $S(d)$, we generate combinatorially a chain of $d+1$ coherently-oriented simplexes of dimension $d+1$:



So we have, with $|\gamma^{d+1}| = \sigma^d \times I$, and $I = [0, 1]$:

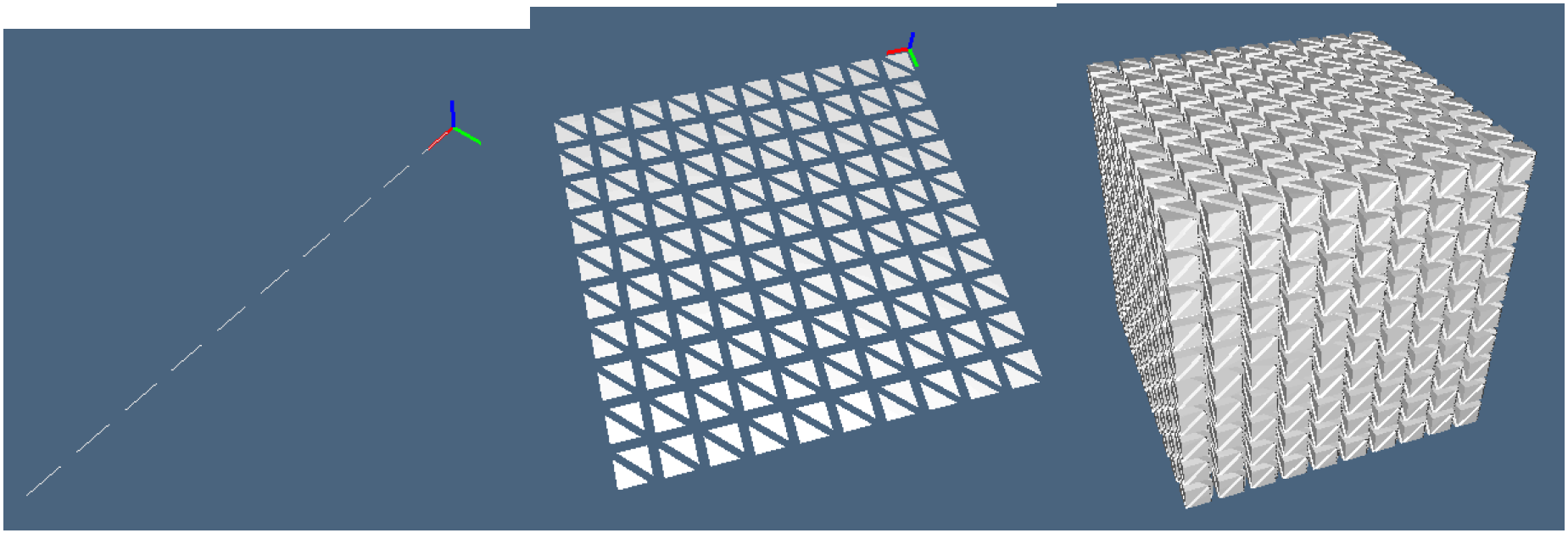
$$\gamma^{d+1} = \sum_{k=0}^{d} (-1)^{kd} \langle v_k, \ldots v_d, v_0^*, \ldots v_k^* \rangle$$

with $v_k \in \sigma^d \times \{0\}$ and $v_k^* \in \sigma^d \times \{1\}$,

Simplicial extrusion (2/3)

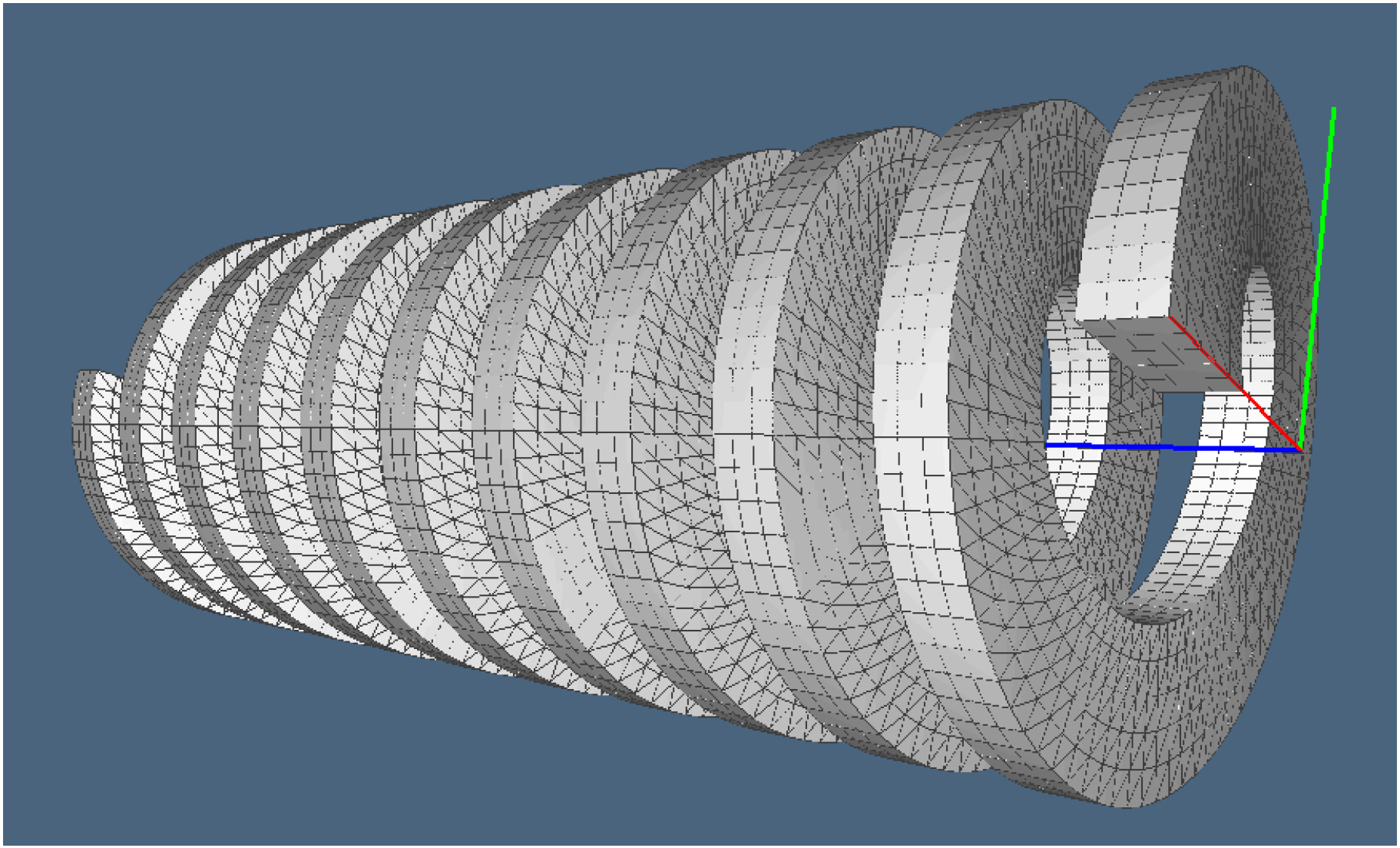
Let $\mathfrak{S}$ be the set of simplicial c.c.c., and $S_d, S_1 \in \mathfrak{S}$:

$$S_d \times S_1 =: S_{d+1} \in \mathfrak{S}$$



```
model1 = larExtrude( VOID, 10*[1] )
model2 = larExtrude( model1, 10*[1] )
model3 = larExtrude( model2, 10*[1] )
```

Simplicial extrusion (3/3)



Cartesian product of complexes (1/4)

Let $\mathfrak{C}$ be the set of c.c.c's. $X, Y \in \mathfrak{C} \Rightarrow X \times Y \in \mathfrak{C}$ [Basak, Geom. dedicata, 2010]

Let

$$X = S(m) \in \mathfrak{S}, \quad \text{and} \quad Y = S(n) \in \mathfrak{S};$$

Then

$$X \times Y = S(X \times Y) \in \mathfrak{S}$$

with

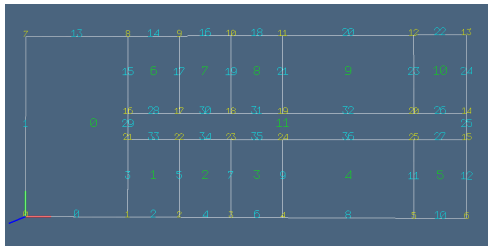
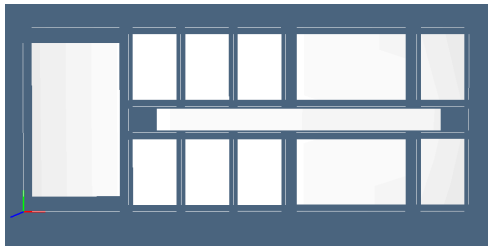
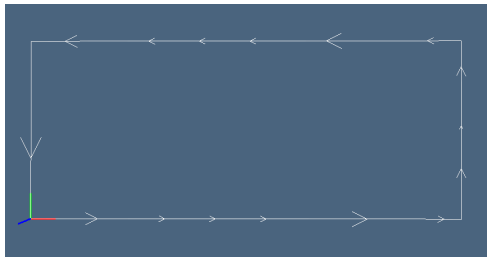
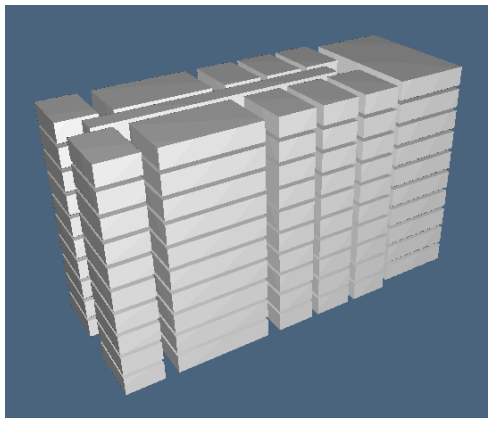
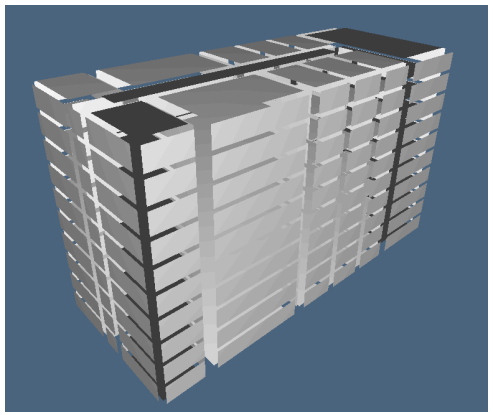
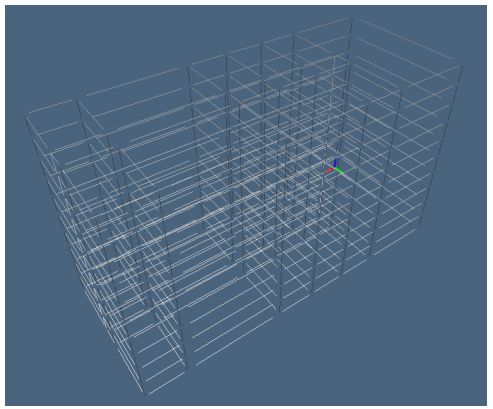
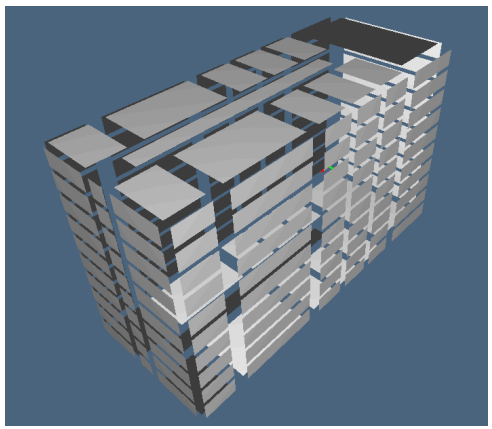
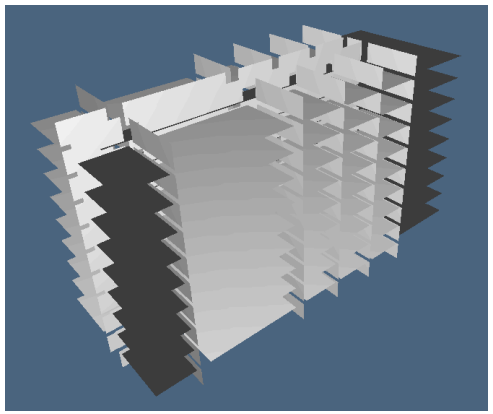
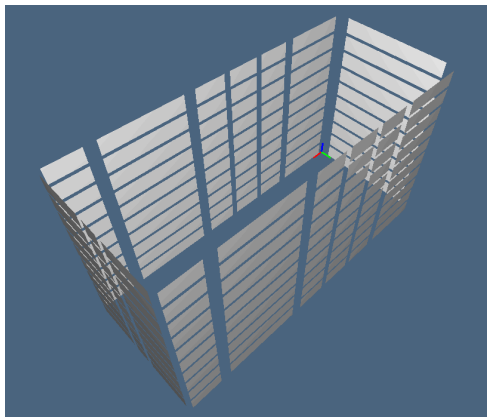
$$x \times y = (x_1 y_1, \ldots, x_1 y_n, \ldots, x_m y_1, \ldots, x_m y_n) \in S(m+n).$$

Cartesian product of complexes (2/4)

Implementation

```
def larModelProduct(twoModels):
    (V, cells1), (W, cells2) = twoModels
    vertices = collections.OrderedDict(); k = 0
    for v in V:
        for w in W:
            id = tuple(v+w)
            if not vertices.has_key(id):
                vertices[id] = k
                k += 1
    cells = [ [vertices[tuple(V[v] + W[w])] for v in c1 for w in c2 ]
             for c1 in cells1 for c2 in cells2]
    model = [list(v) for v in vertices.keys()], cells
    return model
```

Cartesian product of complexes (4/4)

Larlib Examples ()

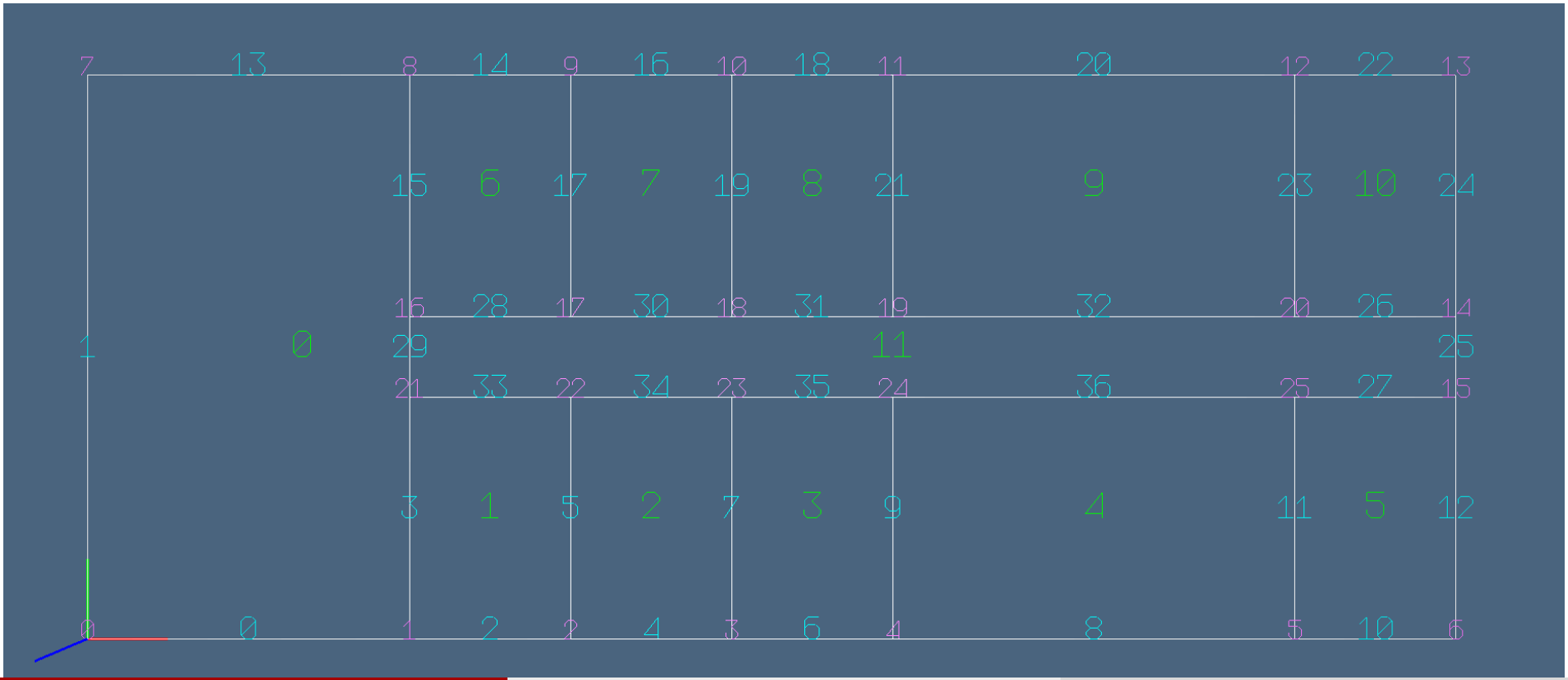
```
from larlib import *

V = [[0,0],[4,0],[6,0],[8,0],[10,0],[15,0],[17,0],[0,7],[4,7],[6,7],[8,7],[
[15,7],[17,7],[17,4],[17,3],[4,4],[6,4],[8,4],[10,4],[15,4],[4,3],[6,3],[8,
[10,3],[15,3]]

FV = [[0,1,7,8,16,21],[1,2,21,22],[2,3,22,23],[3,4,23,24],[4,5,24,25],[5, 6
[8,9,16,17],[9,10,17,18],[10,11,18,19],[11,12,19,20],[12,13,14,20],range(14
range(7),range(7,14),[0,7],[6,13,14,15]]
```

Larlib Examples (1/8)

```
submodel = SKEL_1(STRUCT(MKPOLS((V,FV))))
V,EV = larFacets((V,FV),2,4)
VV = AA(LIST)(range(len(V)))
VIEW(larModelNumbering(1,1,1)(V,[VV,EV,FV[:-4]],submodel,3))
```



Larlib Examples (2/8)

```
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V,FV[:-4]))+MKPOLS((V,EV))+AA(MK)(V)))
```

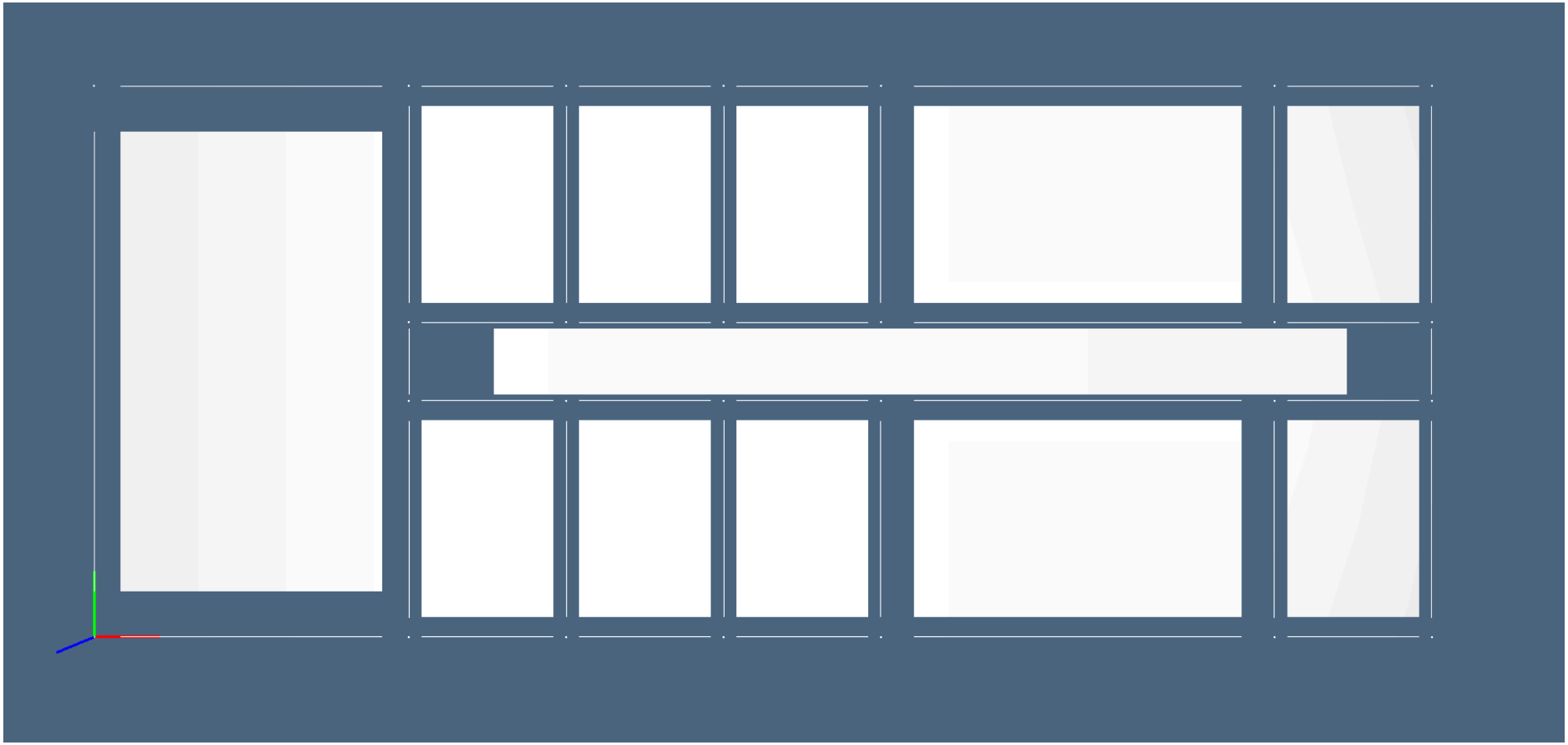


Figure 2:

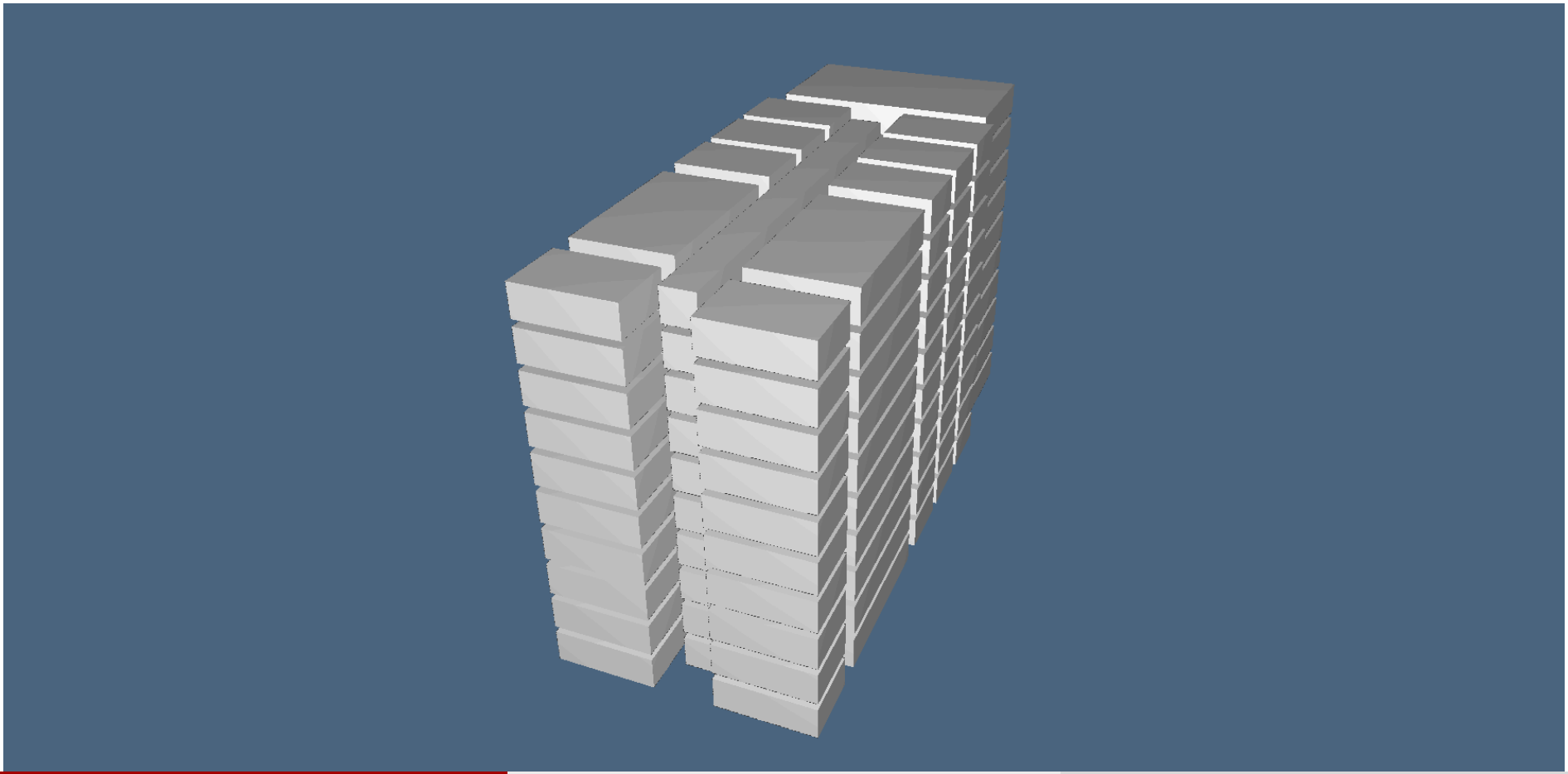
Larlib Examples (3/8)

```
BE = boundaryCells(FV[:-4],EV)
VIEW(EXPLODE(1.2,1.2,1)(MKPOLS((V,[EV[e] for e in BE]))))
```

Figure 3:

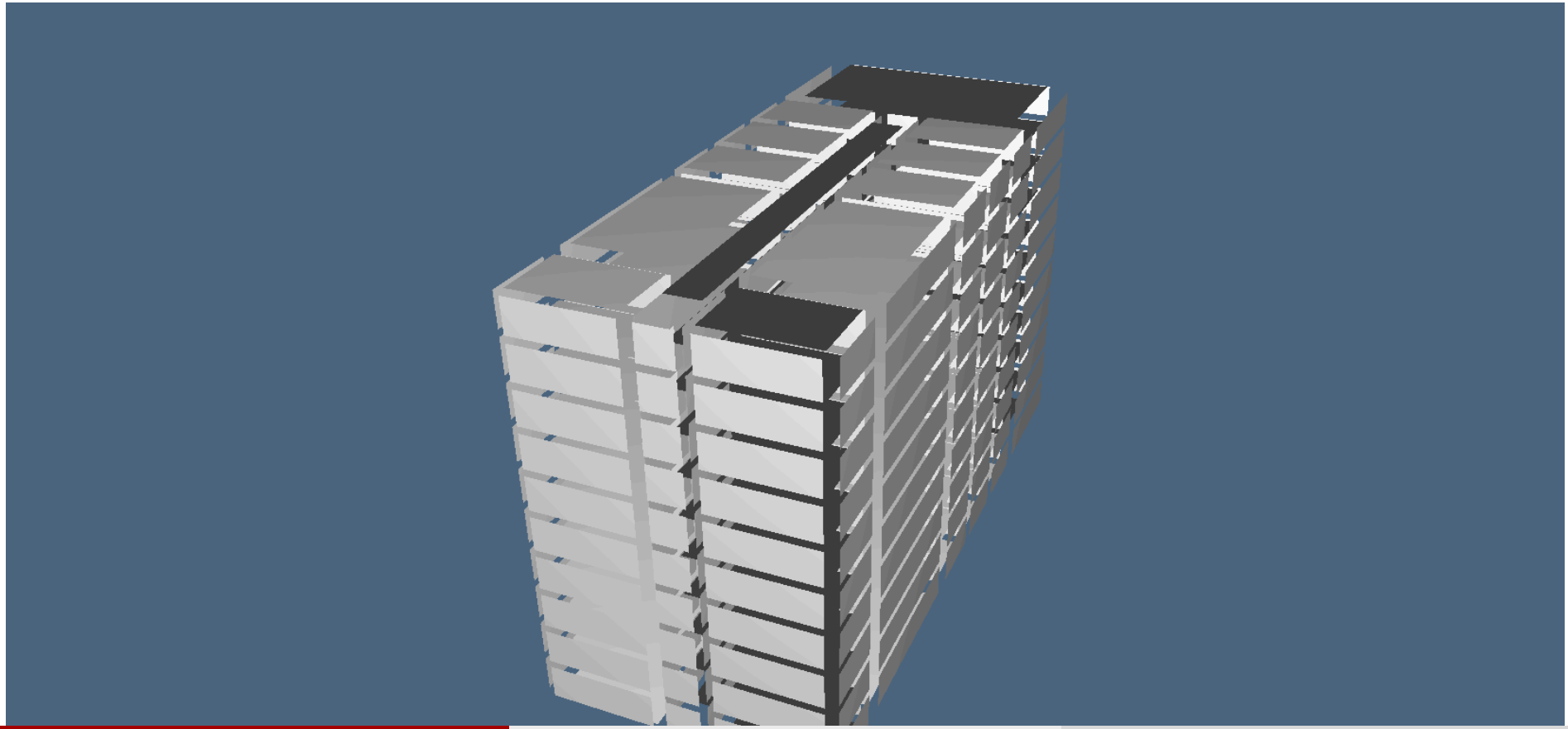
Larlib Examples (4/8)

```
C1 = larCuboids([10])
V3,CV = larModelProduct([(V,FV[:-4]),C1])
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,CV))))
```



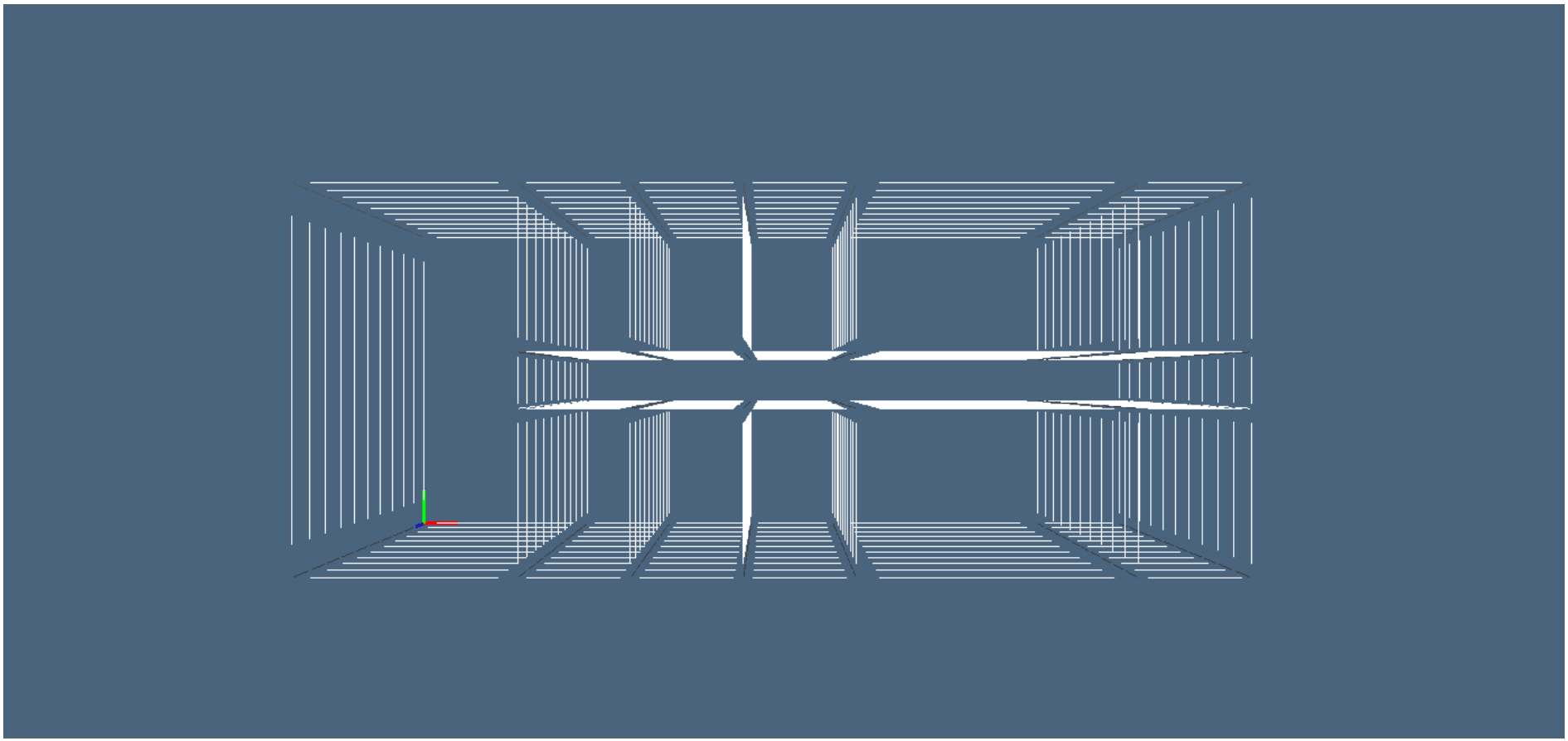
Larlib Examples (5/8)

```
full3D = CV + exteriorCells((V3,CV))
V3,FV3 = larFacets((V3,full3D),3,6)
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,FV3))))
```



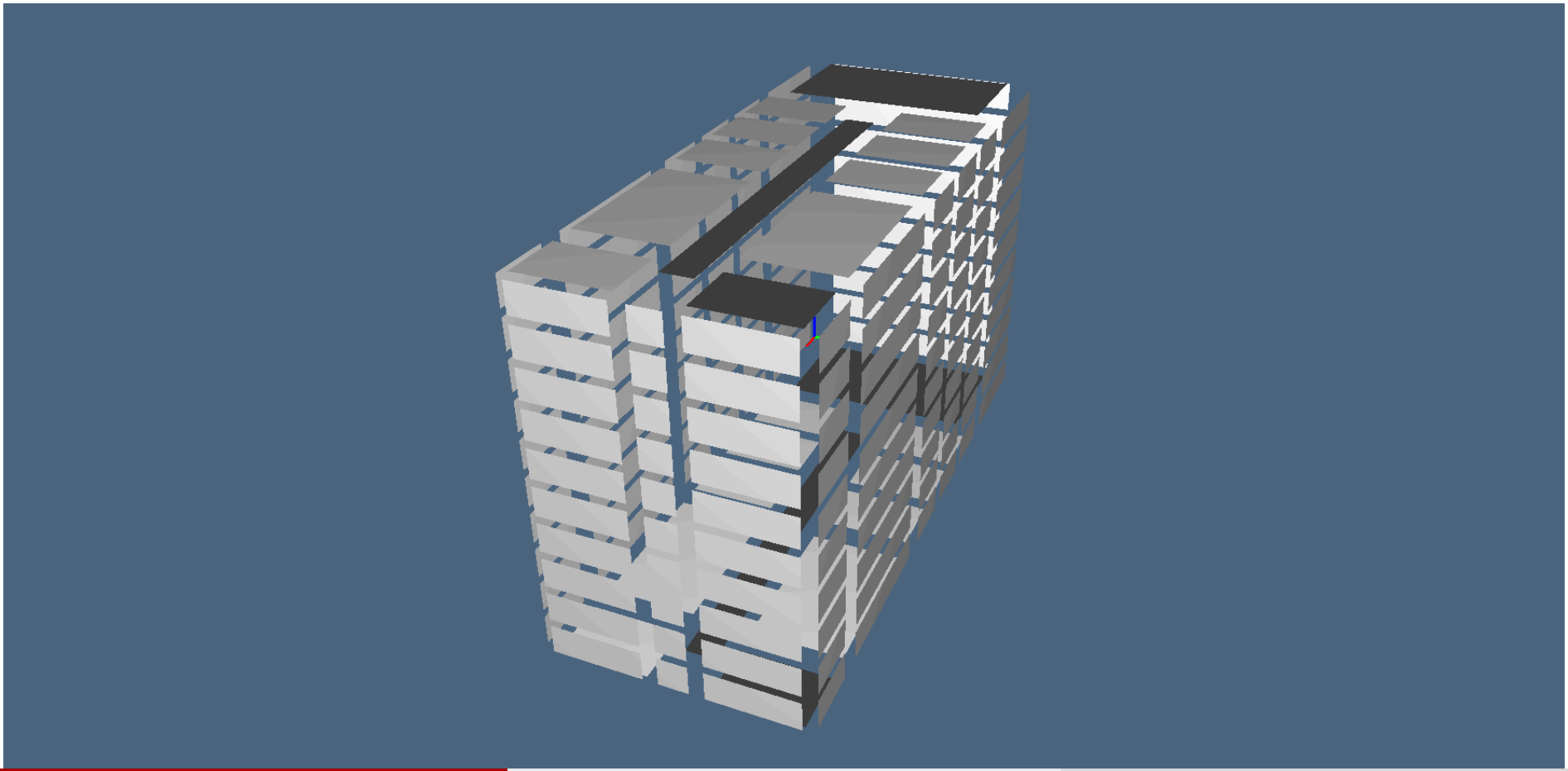
Larlib Examples (6/8)

```
V3,EV3 = larFacets((V3,FV3),2)
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,EV3))))
```



Larlib Examples (7/8)

```
VV3 = AA(LIST)(range(len(V3)))
boundary3D = boundaryCells(CV,FV3)
VIEW(EXPLODE(1.25,1.25,1.25)(MKPOLS((V3,[FV3[f] for f in boundary3D]))))
```



Larlib Examples (8/8)

```
V3,F3 = larFacets((V3,CV))
VIEW(EXPLODE(1.2,1.2,1.2)(MKPOLS((V3,F3))))
```