

Chapter 2

Theoretical Frame and Previous Work

This chapter will introduce some results and concepts which are related with the study to be described in this work. In **Section 2.1** we will focus in basic definitions related to the dominion of the objects we consider in our research. **Section 2.2** describes some schemes for the Modeling of n-Dimensional Polytopes. We will briefly comment the n-Dimensional Boundary Representations, Hypervoxelizations, 2^n -trees and the n-Dimensional Simplexation of Convex Polytopes (through the Cohen & Hickey's Algorithm). Finally, **Section 2.3** will summarize our previous work which is related to the topological characterization of the elements that compose the boundary of the 4D Orthogonal Pseudo-Polytopes.

2.1. Terminology

2.1.1. The n-Dimensional Euclidean Space

Banchoff points out that the term “dimension” is commonly used for specifying characteristics which are feasible to be measured [Banchoff96]. For example, an object's list of dimensions would include width, height, weight, color, brightness, temperature, etc.

Another perspective is offered by the Albert Einstein's Relativity Theory and the Space-Time Geometry as one of its main contributions. For the relativists, time is considered as the fourth dimension [Russell85] and it is fully linked with space. Einstein proposed that time and space are not independent because an event must be described in terms of the place and the time at which it occurs [Kaku94] (in other words, time and space compose the event's list of dimensions). Consequently, space is a 3D cross section of the 4D Space-Time where 3D objects are moving forward in the direction of the remaining dimension, the time [Rucker77].

In strict terms, the fourth dimension is spatial, represented by a line perpendicular to each of three other perpendicular lines and it leads out of the space defined by the other three and never intersects them [Robbin92]. Coxeter considers **4D Euclidean space** as the space with four coordinates (x, y, z, w) instead of habitual two (x, y) or three (x, y, z) [Coxeter84]. And it is established by him that two distinct points determine a straight line, three vertices of a triangle determine a plane and four vertices of a tetrahedron determine a hyperplane which has only a lineal equation that relates to the four coordinates.

The way that Coxeter builds the definition of 4D Euclidean space can be easily extended in order to define five, six, ..., and n-dimensional spaces [Sommerville58]. However, we can appeal to the theory of metric spaces to define Euclidean spaces by specifying how the points are composed and their associated metric. Therefore, the **n-Dimensional Euclidean Space** is the set of all ordered n-tuples $x = (x_1, x_2, \dots, x_n)$ of real numbers x_1, x_2, \dots, x_n with the distance

$$d(x, y) = \sqrt{\sum_{k=1}^n (x_k - y_k)^2}$$

([Kolmogorov75] & [Shilov73]). In this work we will refer to Euclidean spaces which are based in the above definitions.

2.1.2. Polyhedra

A **polyhedron** is a bounded subset of the 3D Euclidean Space enclosed by a finite set of plane polygons such that every edge of a polygon is shared by exactly one other polygon (adjacent polygons) [Preparata85]. Additionally, it can be established that the polygons that are incident to a vertex must compose a single circuit [Coxeter63].

Coxeter established that the polyhedron's boundary is a simple and closed surface of a volume (therefore, “dangling” faces and edges are not accepted) [Coxeter63]. When the volume's surface (i.e. the polyhedron's boundary) is considered without aspects like areas, distances or angles but only taking the surface's aspects not affected by deformation, then we will work with the surface's topology [Weeks02]. In this context, the whole surface is called a two-dimensional manifold or a 2-manifold [Boltyanskii82].

A 2-manifold has both local and global properties. Local properties are those observable inside a manifold's small region, while global properties require considering the manifold entirely [doCarmo76]. For example, a 2-manifold defines a 2D space with a local topology for a plane; however its global topology can correspond to a sphere's surface or a torus' surface (or any other).

The polyhedron's surface must fulfill two characteristics cited by Coxeter [Coxeter63]: to be simple and closed. The surface is closed, or in other words, the manifold is closed when it decomposes the space where it is embedded (3D space) into exactly two regions: the interior, which is finite [Coxeter63] and the outside. The interior and the outside both are connected sets in the sense that we can get from any point of the interior to any other point by drawing continuously a curve between them which never leaves the interior (and similarly for the outside), but one cannot draw continuously a curve from a point in the interior to a point in the exterior which does not contain at least one point of the polyhedron's surface, which is in fact the boundary between the interior and the outside [Herman98]. A polyhedron that accomplishes this last property is said that satisfies the Jordan's Theorem [Boltyanskii82]. The polyhedron's surface is simple and closed, or the manifold is oriented and closed, when it is possible to distinguish both of its opposite sides in the 3D space, that is to say, it is clearly possible to distinguish between the interior and the outside (Klein's bottle is a classic example of a closed not oriented surface) [Hansen93].

Edges and vertices, as boundary elements for polyhedra, are classified as 2-Manifold (or just Manifold) elements. A Manifold edge is adjacent to exactly two faces, and a Manifold vertex is the apex (i.e., the common vertex) of only one cone of faces (the faces compose a single circuit) [Rossignac91].

2.1.3. Pseudo-Polyhedra

A **pseudo-polyhedron** is a bounded subset of the 3D Euclidean Space enclosed by a finite collection of planar faces such that every edge has at least two adjacent faces, and if any two faces meet, they meet at a common edge [Tang91]. From this definition we have that polyhedra are a special case (a subset) of pseudo-polyhedra when exactly two faces are incident to each of its edges. The boundary's Pseudo-Polyhedra also must fulfill to be a closed surface ("dangling" faces and edges are not accepted).

From the topological's point of view, only some regions of the pseudo-polyhedron's surface are 2-manifold. This is because, contrary to polyhedra, the pseudo-polyhedra's interior is composed by more than two quasi-disjoint regions. An interior region can be seen as limited by a surface. In a pseudo-polyhedron, at least two interior regions' surfaces have common points, which compose the regions, from the surface seen globally (the pseudo-polyhedron's whole surface) that are not 2-manifold, or in other words, these regions are non-manifold.

Edges and vertices, as boundary elements for pseudo-polyhedra, may be either two-manifold (or just manifold) or non-manifold elements. In the case of edges, they are (non) manifold elements when every points of it is also a (non) manifold point, except that either or both of its ending vertices might be a point of the opposite type [Aguilera98]. A manifold edge is adjacent to exactly two faces, and a manifold vertex is the apex (i.e., the common vertex) of only one cone of faces. Conversely, a non-manifold edge is adjacent to more than two faces, and a non-manifold vertex is the apex (i.e., the common vertex) of more than one cone of faces [Rossignac91].

2.1.4. Four-Dimensional Polytopes

We will extend the polyhedron's definition presented by Preparata (see section 2.1.2) for defining the 4D Polytopes: A **4D Polytope** is a closed subset from the 4D Euclidean Space, which is delimited by finite set of three-dimensional cells or volumes (polyhedra) such that every volumes' face is shared only with another volume (adjacent volumes) [Pérez-Aguila03c]. In the same way, it can be established that a 4D Polytope's boundary will be a simple and closed hyper-surface of a hyper-volume, therefore, "dangling" volumes, faces and edges are not accepted.

Authors such as Takala or Weeks point out that the definitions related to the 2-manifolds' topology (surfaces) can be extended for defining the 3-manifolds' topology ([Takala92] & [Weeks02]). A 2-manifold can be defined as a space with a plane's local topology on its boundary, and analogously, a 3-manifold can be defined as a hyperspace with our "ordinary" three-dimensional space's local topology on its boundary [Weeks02].

We can appeal to *Flatland* [Abbott84] for a better understanding. *Flatland* is a two-dimensional Universe, therefore, it is a surface or a 2-manifold, which is inhabited by polygonal beings. *A.Square's* interaction with his universe allowed him to determine that *Flatland* is a plane; however, this conclusion is topologically valid from a local point of view. *Flatland's* global topology could belong to a sphere's surface (as described in *Sphereland* [Burger83]) or a torus' surface, for example. Our three-dimensional universe, which we inhabit, can be seen as *Spaceland* [Abbott84]. However, due to Einstein, our universe is seen as a hyper-surface, or better, a 3-manifold [Sagan80]. Our universe can be the hyper-sphere's boundary, or a 4D torus' boundary. Because we are embedded in a 3-manifold, our universe, we can not perceive the effects by its curvature, or in other words, by its global topology. Since the 4D Polytopes' boundary is composed by three-dimensional cells, its topology will be related with a 3-manifold.

Faces, edges and vertices, as boundary elements for 4D polytopes, will be manifold. Hansen has established that a manifold face is adjacent to exactly two volumes [Hansen93], and Pérez & Aguilera have suggested that a manifold edge is the common edge of only one hyper-cone of volumes [Pérez-Aguila03c].

2.1.5. Four-Dimensional Pseudo-Polytopes

The pseudo-polyhedron's definition presented by Tang is extended, in [Pérez-Aguila03d], for defining the 4D Pseudo-Polytopes: A **4D Pseudo-Polytope** is a bounded subset of the 4D Euclidean Space enclosed by a finite collection of volumes such that every face has at least two adjacent volumes, and if any two volumes meet, they meet at a common face. From this definition we have that 4D Polytopes are a special case (a subset) of Pseudo-Polytopes when exactly two volumes are incident to each of its faces. The boundary's Pseudo-Polytopes also must fulfill to be a hyper-volume's closed hyper-surface ("dangling" volumes, faces and edges are not accepted), i.e. polytopes with non-manifold faces, edges or vertices.

Basically, the topological differences between 4D Polytopes and Pseudo-Polytopes are analogous to the Polyhedra and Pseudo-Polyhedra's case. Certain regions from the 4D Pseudo-Polytopes' boundary can be considered as not belong exclusively to just one 3-manifold, because, as analogously to Pseudo-Polyhedra, these regions (faces, edges or vertices) can be seen as shared by several hyper-surfaces. Therefore, faces, edges and vertices, as boundary elements for 4D polytopes, may be either manifold or non-manifold elements. We mentioned that a manifold face is adjacent to exactly two volumes, and a manifold edge is the common edge of only one hyper-cone of volumes. Pérez & Aguilera have proposed that a non-manifold face is adjacent to more than two volumes, and a non-manifold edge is the common edge of more than one hyper-cone of volumes [Pérez-Aguila03c].

2.1.6. The n-Dimensional Polytopes and Pseudo-Polytopes

Coxeter defines an **n-Dimensional Euclidean Polytope** Π_n as a finite region of n-dimensional Euclidean space enclosed by a finite number of (n-1)-dimensional hyperplanes [Coxeter63]. The finiteness of the region implies that the number N_{n-1} of bounding hyperplanes satisfies the inequality $N_{n-1} > n$. The part of the polytope that lies on one of these hyperplanes is called a cell. Each cell of a Π_n is an (n-1)-dimensional polytope, Π_{n-1} . The cells of a Π_{n-1} are Π_{n-2} 's, and so on; we thus obtain a descending sequence of elements Π_{n-3} , Π_{n-4} , ..., Π_3 (a volume), Π_2 (a polygon), Π_1 (an edge), Π_0 (a vertex).

The way that the cells Π_{n-1} , Π_{n-2} , Π_{n-3} , Π_{n-4} , ..., Π_1 , Π_0 are related is given by the following Sommerville's observations [Sommerville58]:

- The Π_{n-1} 's share Π_{n-2} 's, in that way, it is defined a Polytope Π_n when two and only two Π_{n-1} 's share a Π_{n-2} ; when more than two Π_{n-1} 's share a Π_{n-2} then it is defined an **n-Dimensional Pseudo-Polytope** Π_n (the notation Π_n is common to polytopes and pseudo-polytopes).
- Three or more Π_{n-1} 's will have a common Π_{n-3} .
- p or more Π_{n-1} 's will have a common Π_{n-p} .
- n or more Π_{n-1} 's will have a common Π_0 (a vertex).

From the topological's point of view, n-dimensional Polytopes are considered by Hansen as a closed set of (n-1)-manifolds, one for each cell Π_{n-1} [Hansen93]. In the previous sections, the 3D and 4D Polytopes' boundary was entirely considered as a manifold. For example, a cube's boundary is topologically equivalent to a sphere's surface, but from Hansen's point of view, each face of the cube will be topologically equivalent to a plane, or in other words,

each one will be a 2-manifold [Hansen93]. Furthermore, Hansen states that each element on a cell Π_{n-1} 's boundary will have its respective topologic equivalence [Hansen93]. In this way, the edges of a cube's face will be topologically equivalent to a 1-manifold (a space with the local topology of a line [Weeks02]) and so forth. By representing the nD Polytopes by this way, Hansen presents the following properties [Hansen93]:

1. A 0-manifold is a point, and it has no boundary.
2. All boundary elements of an n-manifold are (n-1)-manifold elements.
3. All (n-1)-dimensional elements belong to exactly two n-manifold elements (or twice to the same element).
4. Manifold elements may not intersect each other except at common boundary elements.

2.2. Schemes for the Modeling of n-Dimensional Polytopes

2.2.1. Polytopes Modeling

Solid Modeling is an area of wide development in several applications as the Computer Aided Design and Manufacturing (CAD/CAM), electronic prototypes, animation planning, etc. If a solid object can be modeled in a way that its geometry is appropriately captured, then it will be possible to apply, on such object, a range of useful operations. Due to the need of modeling objects as solids, the development of a variety of specialized mechanisms to represent them has arisen. The representation schemes for solid objects are frequently divided in some large categories (although not all the representations are completely inside in one of them): Boundary Representations, Spatial Partitioning Representations, Constructive Solid Geometry, etc.

The extensions of the solid modeling schemes, by considering their application to spaces beyond the three-dimensional, have allowed the modeling of n-dimensional polytopes [Paoluzzi93]. Previously we mentioned the grouping of the representation schemes for solid objects: Boundary Representations, Spatial Partitioning Representations, Constructive Solid Geometry, etc. Related to the schemes for the Polytopes Modeling, in this work we will concentrate on two categories:

- The n-Dimensional Boundary Representations: They describe an nD polytope in terms of the elements that compose its boundary: vertices, edges, faces, ..., Π_{n-2} 's and Π_{n-1} 's. Moreover, these representations have the information about the connectivity between these elements.
- Hyperspatial Partitioning Representations: Where a polytope is decomposed in a collection of attached n-dimensional cells, without intersections, and more primitive than the original polytope, although they are not necessarily of the same kind. Inside this category we can find schemes as the n-Dimensional Cell Decompositions, the Hypervoxelizations and the 2^n -trees (hyperoctrees).

In the following sections we will describe the fundamentals behind the nD boundary representations (2.2.3), the n-Dimensional Simplexation of Convex Polytopes (a specific cell decomposition; section 2.2.4), the hypervoxelizations (2.2.5) and, finally, the 2^n -trees (2.2.6).

2.2.2. Regularized Boolean Operations

Independently of the polytopes' representation, it should be feasible to combine them to compose new objects [Foley95]. One of the most common methods to combine polytopes are the set theoretical Boolean operations, as the union, difference, intersection and exclusive OR. However, the application of an ordinary set theoretical Boolean operation on two polytopes doesn't necessarily produce a polytope. For example, the ordinary intersection between two cubes with only a common vertex is a point.

Instead of using ordinary set theoretical Boolean operators, The **Regularized Boolean Operators** ([Putnam86] & [Requicha77]) will be used. The practical purpose of regularization of polytope models is to make them dimensionally homogeneous [Takala92]. The regularization operation can be defined as

$$\text{Regularized}(S) = \text{Closure}(\text{Interior}(S))$$

which results in a closed regular set [Arbab90].

Each regularized Boolean operator is defined in function of an ordinary operator in the following way:

$$A \text{ op}^* B = \text{Closure}(\text{Interior}(A \text{ op} B))$$

In such way we will have:

$A \cup^* B = \text{Closure}(\text{Interior}(A \cup B))$	Regularized Union
$A \cap^* B = \text{Closure}(\text{Interior}(A \cap B))$	Regularized Intersection
$A \otimes^* B = \text{Closure}(\text{Interior}(A \otimes B))$	Regularized Exclusive OR
$A -^* B = \text{Closure}(\text{Interior}(A - B))$	Regularized Difference

These operators are defined as the closure of the interior of the corresponding set theoretical Boolean operation ([Requicha77] & [Mäntylä86]). In this way, the regularized operations between polytopes always will generate polytopes [Takala92]. Recapturing the previous example, the regularized intersection between two cubes with a common vertex is the null object (the empty set).

2.2.3. The n-Dimensional Boundary Representations

A boundary model for a three-dimensional solid object is a description of the faces, edges and vertices that compose its boundary together with the information about the connectivity between those elements [Requicha80]. However, the boundary representations can be recursively applied not only to solids or surfaces or segments, but to hyperdimensional objects, or in other words, n-dimensional Polytopes [Hansen93].

Hansen describes that this type of representation is equivalent to a graph structure, called **Incidence Graph** [Hansen93], whose nodes belong to the cells $\Pi_n, \Pi_{n-1}, \dots, \Pi_1, \Pi_0$ on the polytope's boundary. The edges between the graph's nodes express the information about the connectivity. Together, they constitute the combinatorial structure (the topology) of the representation. The vertices' n coordinates contain the metric information (the geometry) associated with the representation. **Figure 2.1.a** presents the incidence graph of a 4D simplex.

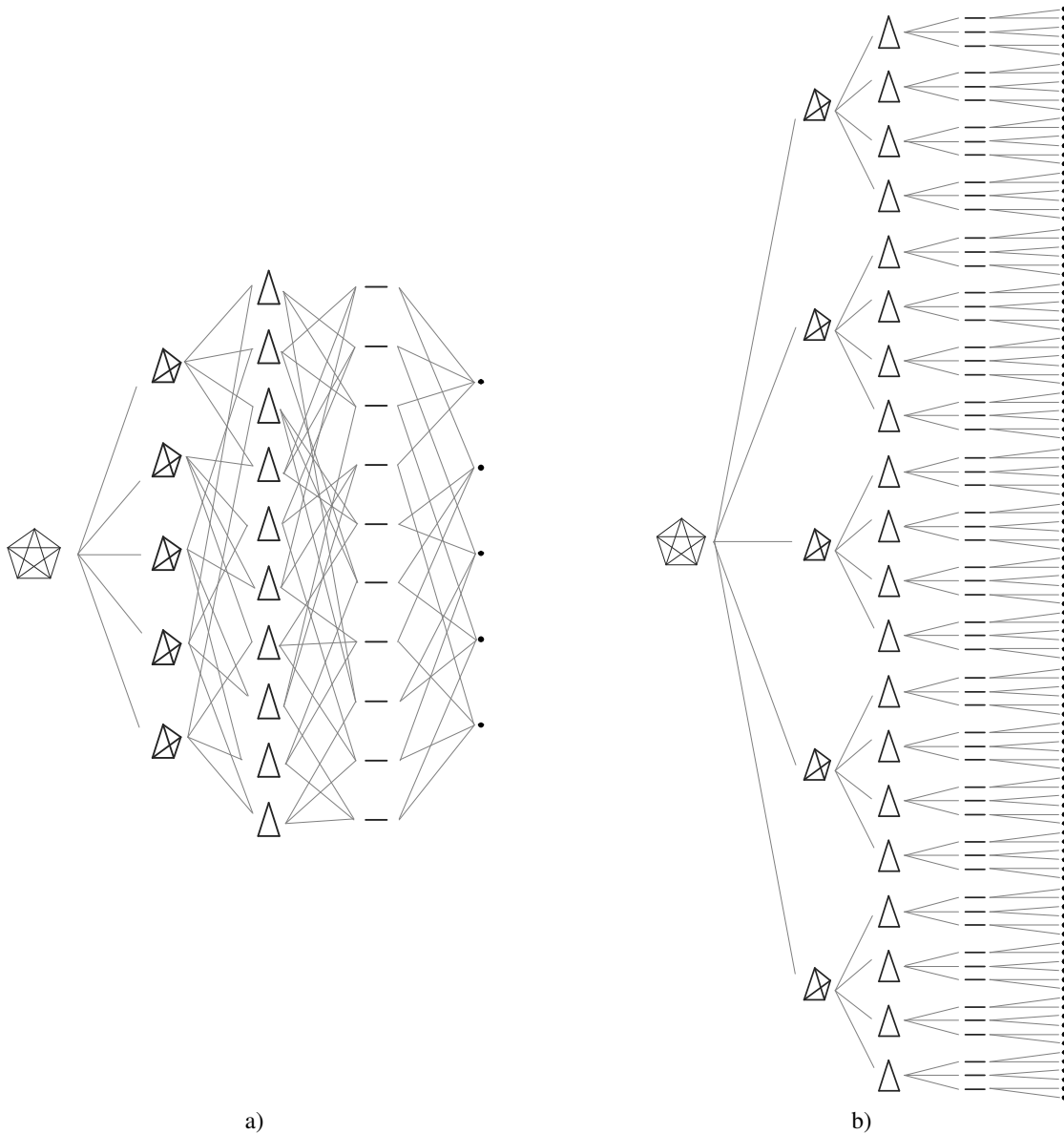


Figure 2.1. The incidence graph (a) and the boundary tree (b) for the elements on the boundary of a 4D simplex.

Another point of view is provided by Putnam & Subrahmanyam, where a boundary representation can be seen as a **Boundary Tree** [Putnam86]. In the tree, each node is split into a component for each element that it bounds. An element (vertex, edge, etc.) will be represented several times inside the tree, one for each boundary that it belongs to. See **Figure 2.1.b** for a 4D simplex's boundary tree.

Independently of the representation to use, we must consider the hyperdimensional entities to be modeled. For example, the boundary models defined in [Hansen93] or [Gomes99] allow the representation of n-dimensional objects whose boundary can be orientable or not orientable or even incomplete. Therefore, in our context we must consider restrictions for guaranteeing that a represented object is a valid polytope. For example, a restriction that can be considered is the Euler's formulae. Every one-component convex nD polytope must fulfill that [Sommerville58]:

$$\sum_{k=1}^n (-1)^{k-1} \cdot N_{n-k} = N_{n-1} - N_{n-2} + N_{n-3} - N_{n-4} + \dots + (-1)^{n-1} N_0 = 1 - (-1)^n$$

where N_{n-1} is the number of (n-1)-dimensional cells (the Π_{n-1} 's), N_{n-2} is the number of (n-2)-dimensional cells (Π_{n-2} 's) and so forth until N_0 which is the number of vertices in the polytope.

A Boolean operation between two n-dimensional polytopes represented under a boundary representation scheme can be performed, according to the procedure described in [Hansen93] and [Luo92], through two general main steps, that is, "cuts" and "sewings":

- The polytopes are subdivided (or "cut") in their intersecting boundary elements.
- Later on, the polytopes' subdivided elements are alternated and "sewn", after the consideration of which of them are preserved (according to the Boolean operation), to compose the new boundary or boundaries.

A wide range of algorithms to perform Boolean operations, under 2D or 3D boundary representations, can be found in the literature (see [Turner84], [Requicha85], [Putnam86] and [Laidlaw86]). Some of these algorithms consider only convex polygons, in the 2D case, because of their simplicity; see for example [Toussaint85]. There exist algorithms that consider non-manifold polygons, see [Greiner98] or [Maillot92], however, some of these procedures provide solutions under specific situations and some of them include mechanisms to avoid extreme cases. In the 3D case, we can find very difficult situations to handle. Some algorithms consider brute force methods by detecting the intersections between all the faces of a polyhedron with all the faces of another polyhedron. Through this detection, a procedure could classify the boundaries of the solids, and according to the Boolean operation to perform, to determine faces, edges or vertices to preserve, to add to the representation, or to "cut" and "sewn". However, there are many extreme and degenerate cases to consider and sometimes there is supported only a limited domain. For example, [Mäntylä86] provides algorithms on boundary representations of planar polyhedra, i.e., polyhedra topologically equivalent to 2-manifolds. The complexity of the problem is substantially elevated when we consider non-convex and/or non-manifold polyhedra. In order to provide partial solutions there are procedures that consider decomposition of non-convex polyhedra into sets of convex polyhedra, see for example [Rupper92] and [Edelsbrunner95], however, these procedures sometimes only consider a specific number of situations. In the nD boundary representations, some algorithms are given assuming that polytopes are subdivided into sets of primitive simplexes previous to perform the corresponding Boolean operation, see for example [Luo92]; and there are many general descriptions to perform these operations but whose specific details are not given.

2.2.4. The n-Dimensional Simplexation of Convex Polytopes

In this section we will describe the Cohen & Hickey's algorithm for the n-dimensional Simplexation of convex polytopes [Cohen79]. In this scheme, an nD polytope is subdivided in a set of nD quasisdisjoint simplexes (i.e., those that do not intersect between them. In fact, a polygon's 2D simplexation is a triangulation; and a polyhedron's 3D simplexation is a tetrahedrization. Being quasisdisjoint imply that these simplexes can share some of their boundary elements, that is, we can find vertex, edge, face, etc. adjacencies). In first place we will consider some definitions:

- **Generic Cell:** A generic cell is denoted by $\Pi_{\text{dimensions}}$
- **Specific Cell:** A specific cell is denoted by $\Pi_{\text{dimensions}}^{\text{index}}$.

Then we have that: Π_0^i denotes the i-th vertex of a polytope; Π_1^j denotes the j-th edge of a polytope; Π_2^k denotes the k-th face of a polytope, ..., Π_n^1 denotes to the polytope itself.

- **The function ψ :** Let $\psi(\Pi_d^i)$ be the set of vertices in the i -th cell of d dimensions, i.e. every Π_0 's in Π_d^i .
- **The function η :** The mapping of a cell to a vertex is given by the function

$$\eta(\Pi_d^k) = \Pi_0^j \text{ where } j = \min\{i \mid \Pi_0^i \in \psi(\Pi_d^k)\}, \text{ i.e. the vertex with the minimum index}$$
- **The function F_i :** Let F_i be:

$$F_i = \psi(\Pi_{n-1}^i)$$

In other words, F_i is the set of vertices in a $(n-1)$ -dimensional cell i . See **Figure 2.2** for the application of the above definition over a polygon.

The Cohen & Hickey's algorithm performs the simplexation of a polytope p by choosing any vertex $v \in p$ as an apex and connecting it with the $(n-1)$ -dimensional simplexes that are the result of the simplexation of all the cells in p that do not contain v . Then, the pyramids with apex $\eta(\Pi_n)$ (remember that function η returns the vertex with the least index) and the bases among the cells Π_{n-1} with $\eta(\Pi_n) \notin \Pi_{n-1}$ will compose a dissection of the polytope [Büeler00].

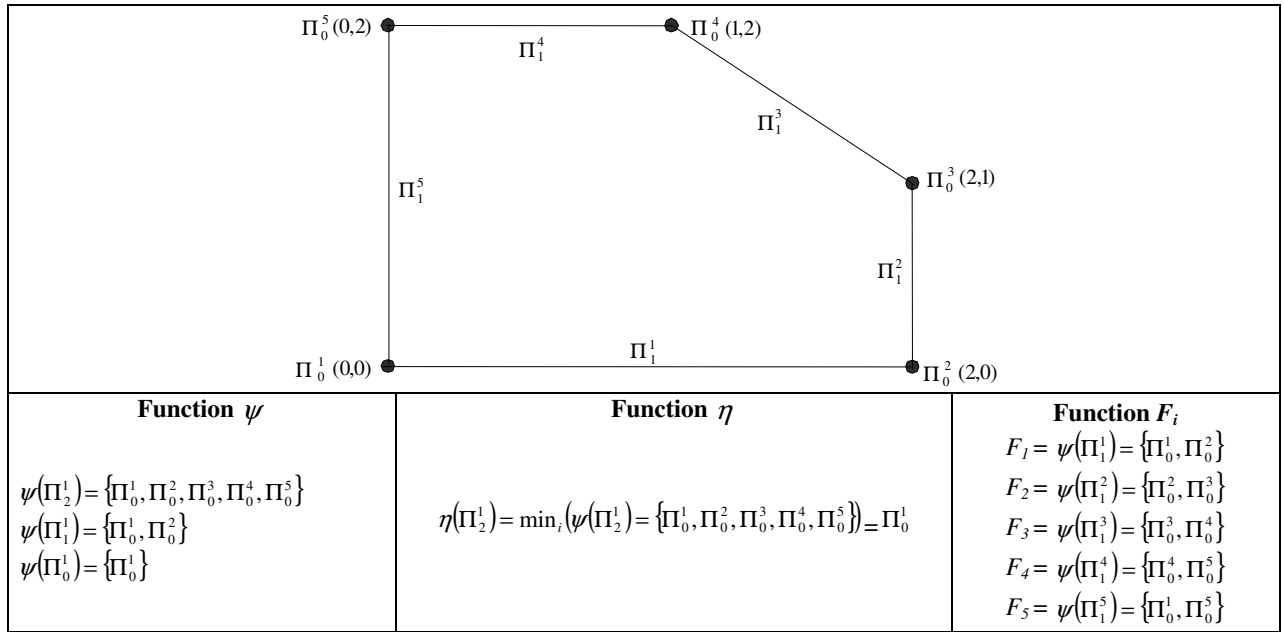


Figure 2.2. Definition of the functions ψ , η and F_i over a polygon Π_2^1 and its cells Π_1^1 and Π_0^1 .

The recursive application of this procedure on all the Π_{n-1} will form a set of decreasing cells $\Pi_n \supset \Pi_{n-1} \supset \dots \supset \Pi_1 \supset \Pi_0$ such that $\eta(\Pi_i) \neq \eta(\Pi_j)$ for $i \neq j$ and $1 \leq i, j \leq n$. Then, the corresponding set of simplexes $S = \{\eta(\Pi_0), \dots, \eta(\Pi_n)\}$ defines a simplexation of p . See in **Figure 2.3** how this process composes a tetrahedron inside a cube.

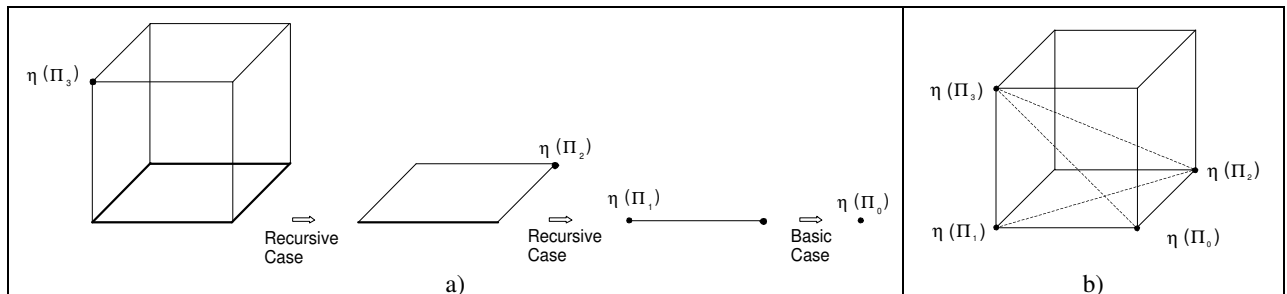


Figure 2.3. Forming a tetrahedron inside a cube.

The implementation of this recursive procedure requires that the cells Π_{n-1} be represented as sets of vertices, i.e. through function F_i . By starting from this representation we pass from a cell Π_k to Π_{k-1} by intersecting the set of vertices in Π_k with the cells Π_{n-1} from p that not contain the vertex $\eta(\Pi_k)$. To avoid the multiple generation of a cell we maintain a list that contains all the cells Π_k earlier generated; only the cells not included in the list are processed [Büeler00].

The algorithm will require initially three input parameters:

- d: Number of dimensions.
- last: A set that contains all the vertices from the polytope, i.e. $\psi(\Pi_n^1)$.
- S: The set that contains the vertices of the nD simplex in construction. In the algorithm's main call $S = \{\Pi_0^1\}$.

simplexation (d, last, S)

```
{
  // m: the number of (n-1)-dimensional cells in the original polytope.
  //  $F_k$ : The set of vertices of each (n-1)-dimensional cell in the original polytope.
  // L: A list of sets.
  If (d > 0)
  {
    L = { { } }
    For k = 1 until m
    {
       $I = last \cap F_k$  // I is a candidate set to represent  $\psi(\Pi_d^j)$ .
      If ( $I \notin L$ ) // It is evaluated if the set was not earlier obtained.
      {
         $L = \{I\} \cup L$  // The set  $\psi(\Pi_d^j)$  is added to the list L.
        If ( $\eta(I) \notin S$ ) /* Verifying if vertex  $\eta(I)$  is not contained in current simplex. */
          simplexation (d-1, I,  $\eta(I) \cup S$ )
      }
    }
  }
  else
    S contains the vertices of an n-dimensional simplex.
}
```

The time complexity of the Cohen & Hickey's Algorithm applied to an n-Dimensional hyper-box is $O(n^3 n!)$ [Büeler00].

2.2.5. Hypervoxelizations

The representation of a polytope through a scheme of Hyperspatial Occupancy Enumeration is essentially a list of identical hyperspatial cells occupied by the polytope. Specific types of cells, called hypervoxels [Jonas95] are hyper-boxes (hypercubes, for example) of a fixed size that lie in a fixed grid in the n-dimensional space. Jonas defines two kinds of hypervoxels [Jonas95]:

- Centered Hypervoxel: an n-dimensional hyper-box whose dimensions are given by $x_1Side, x_2Side, \dots, x_nSide$ and it is represented by the coordinates of its centroid.
- Shifted Hypervoxel: whose characteristics are same that those for the centered hypervoxel, except that its representation is given by some of its 2^n vertices.

By instantiation, we know that a 2D hypervoxel is a pixel while a 3D hypervoxel is a voxel; the term *rexel* is suggested for referencing a 4D hypervoxel [Jonas95].

The collection of hyperboxes can be codified as an n -dimensional array C_{x_1, x_2, \dots, x_n} of binary data. The array will represent the coloration of each hypervoxel:

- If $C_{x_1, x_2, \dots, x_n} = 1$, the black hypervoxel C_{x_1, x_2, \dots, x_n} represents an occupied region from the n -dimensional space.
- If $C_{x_1, x_2, \dots, x_n} = 0$, the white hypervoxel C_{x_1, x_2, \dots, x_n} represents an empty region from the n -dimensional space.

By using the representation through a binary array, the computation of the Boolean set operations just control the operations between bits for all the elements. Let C^1 and C^2 be two n -dimensional grids of hypervoxels, then the Boolean operations between their respective cells, $C^1_{x_1, \dots, x_n}$ op $C^2_{x_1, \dots, x_n}$, are defined as shown in the **Figure 2.4**.

$C^1_{x_1, \dots, x_n} \cup C^2_{x_1, \dots, x_n}$	1	0
1	1	1
0	1	0

$C^1_{x_1, \dots, x_n} \cap C^2_{x_1, \dots, x_n}$	1	0
1	1	0
0	0	0

$C^1_{x_1, \dots, x_n} \otimes C^2_{x_1, \dots, x_n}$	1	0
1	0	1
0	1	0

$C^1_{x_1, \dots, x_n} - C^2_{x_1, \dots, x_n}$	1	0
1	0	1
0	0	0

C_{x_1, \dots, x_n}	$\overline{C_{x_1, \dots, x_n}}$
1	0
0	1

Figure 2.4. Boolean operations between two hypervoxels' grids C^1 and C^2 .

As can be seen, Boolean set operations are trivial under this scheme, however, the spatial complexity of an hypervoxelization is $\prod_{i=1}^n m_i$ where m_i , $1 \leq i \leq n$, is the length of the grid along the x_i -axis. For example, a three-dimensional grid with $m_1 = m_2 = m_3 = 1000$ requires to store 1 billion (1×10^9) voxels.

2.2.6. The 2^n -trees (Hyperoctrees)

It is well known that the Octrees are composed starting from the recursive subdivision of a 3D cubic space in eight octants until each octant is reduced in the possible simplest way. The consideration of this method of recursive subdivision was originated as an extension of Quadrees for the 2D spaces and this leads us to the definition of a *hextree* of 16 hyper-octants in the 4D space. Moreover, the generalization of this hierarchical tree structure leads us to the recursive division of an n -dimensional space in 2^n hyper-octants which is called a 2^n -tree or hyperoctree [Srihari83].

Analogously as the quadrees and the octrees, the 2^n -tree will have three types of nodes:

- Black Nodes: The nodes that correspond to hyper-octants completely occupied by a polytope.
- White Nodes: The nodes that correspond to hyper-octants completely empty.
- Gray Nodes: The nodes that correspond to hyper-octants not completely occupied nor completely empty. These nodes must be subdivided.

The root node of the 2^n -tree corresponds to an n -dimensional hypercube that contains (or encloses) an n -dimensional polytope. The conceptual procedure for the building of the tree is the same that is applied to the quadrees or octrees: If a cell is full or empty, then it must be marked as black or white, respectively; otherwise it must be marked as gray and subdivide it recursively [Requicha00].

By representing a 2^n -tree through a Tree Codification with Pointers we would have to consider the following characteristics [Srihari83]:

- Each node of the tree will contain $2^n + 1$ or $2^n + 2$ fields.
- One of the fields will indicate the kind of node (white, black or gray).
- 2^n fields will be pointers to the hyper-octants in which the given node is divided. If the node is a leaf then these 2^n pointers will be nil.
- It is possible to have an additional field that is a pointer to the parent node.

The achievement of Boolean operations between two 2^n -trees follows the same procedures that are applicable to quadrees or octrees [Srihari83]. Only one consideration must be observed, that is, the initial nD hypercubic universe from both trees to operate must have the same size and location. The achievement of the complement operation consists in traverse the codification of a 2^n -tree changing the white nodes by black nodes and vice versa. Now, the procedure for the computing of the union or intersection T_3 between two trees T_1 and T_2 will be described (specifically the union's case):

1. A synchronous descending traverse in both trees is performed.
2. Each corresponding homologous pair of nodes (that is, with the same size and location) is examined. If some of the nodes in the pair is black, then it is added a corresponding black node in T_3 .
3. If one of the nodes in the pair is white, then it is created in T_3 the corresponding node with the value of the other node in the pair.
4. If both nodes in the pair are gray, then it is added a gray node in T_3 and the algorithm is recursively applied to the pair's sons. In this case the sons of the new node in T_3 must be inspected after the application of the algorithm. If all are black, then they are eliminated and its father in T_3 changes from gray to black.

The intersection between two trees follows the same procedure before described only considering the criteria to apply according to the corresponding pair of nodes. In the **Figure 2.5** are presented the results of the operations of union and intersection between two nodes and the complement for a node.

$T_1 \cap T_2$	B	W	G
B	B	W	G
W	W	W	W
G	G	W	G*

$T_1 \cup T_2$	B	W	G
B	B	B	B
W	B	W	G
G	B	G	G*

T	\bar{T}
B	W
W	B
G	G*

Figure 2.5. The operations of union, intersection and complement for the nodes from 2^n -trees
(B: Black node, W: White node, G: Gray node, G*: recursive case).

Other operations such as the Difference or Exclusive OR can be easily derived through the three operations before described (union, intersection and complement). By this way, the difference can be determined starting from the well known expression:

$$A - B = A \cap \bar{B}$$

While the Exclusive OR is calculated through:

$$A \otimes B = (A \cup B) - (A \cap B)$$

In **Figure 2.6** are presented the results of the Difference and Exclusive OR operations between two nodes.

$T_1 - T_2$	B	W	G
B	W	W	W
W	W	B	\bar{G}
G	W	G	G*

$T_1 \otimes T_2$	B	W	G
B	B	W	G
W	W	B	\bar{G}
G	\bar{G}	G	G*

Figure 2.6. The operations of difference and Exclusive OR for the nodes from 2^n -trees
(B: Black node, W: White node, G: Gray node, \bar{G} : Gray node's complement, G*: recursive case).

Consider an n-dimensional hypercube whose sides have length 2^m and it is positioned in the hyper-octant in \mathbb{R}^n defined by the positive part of the axes X_1, \dots, X_n . In this case variable m refers to the resolution to consider. The number of nodes in a complete hyperoctree, whose universe is the hypercube we have defined, is given by [Srihari83]:

$$\frac{2^{(m+1)n} - 1}{2^n - 1}$$

For example, if $m = 100$ and $n = 3$ then we have approximately 2.328×10^{90} nodes in the corresponding octree. Although this is an upper-bound, the implementations of algorithms related with hyperoctrees must consider this worst case for spatial complexity.

2.3. Topological Properties of 4D Orthogonal Pseudo-Polytopes

This section will describe some of our previous results related to the 4D Orthogonal Pseudo-Polytopes. We will cover the analysis related to the configurations that can represent the 4D-OPP's. Moreover, the procedures for classifying edges and faces as manifold or non-manifold elements in 4D-OPP's will be described. We will quote our

conditions for classifying faces in the 4D-OPP's as manifold or non-manifold. For the edges' analysis in the 4D-OPP's our two approaches will be described: 1) The analogy between incident (manifold and non-manifold) edges to a vertex in 3D-OPP's with incident (manifold and non-manifold) faces to an edge in 4D-OPP's; and 2) The extension of Rossignac's "Cones of Faces" to "Hypercones of Volumes" for classifying an edge as manifold or non-manifold in 4D-OPP's. Finally, we will describe the characterization of edges in the 4D-OPP's as Extreme or Non-Extreme.

2.3.1. Dimensional Analogies

Our previous work has its procedural foundations in the Method of the "Dimensional Analogies" (Sagan called them "Interdimensional Contemplations" [Sagan80]). When we are trying to visualize and understand the 4D space, the situation is similar for *Flatland*'s inhabitants (*flatlanders*) trying to visualize and understand the 3D space. Due to this, it results very useful to consider the analogous situations with a reduced number of dimensions [Zhou91]. For example, try to answer the following question: What is a 4D being able to see in the 3D beings? In order to get the answer, first it must be referenced the interaction between a 3D being with a 2D being. *A.Sphere* is the 3D being that makes contact with *A.Square* in *Flatland*. From his 3D space, *A.sphere* can visualize the *Flatland* polygons' boundary, but additionally, he is able to see their interior (and therefore, their internal organs, if they have them). But in *Flatland* it is also referred *Lineland*, a one-dimensional universe. *Lineland*'s inhabitants were segments whose interior was visualized by *A.Square*. By analogy, we can expect that a 4D being, interacting with our 3D universe, could visualize our "boundary" (the skin), but furthermore, he could visualize our internal organs (in other words, the 4D being's vision could work as the systems of X rays, tomography or magnetic resonance [Pickover99]).

Fundamentally, the method of the analogies considers the contemplation of an analogy between 1D and 2D spaces, as well as between 2D and 3D spaces, then (through some extrapolation) between 3D and 4D spaces; and so forth. In this way the expected results can be suggested (a hypothesis is established) [Coxeter63]. Once the hypothesis is demonstrated, it is possible to suggest a generalization of the characteristic that has been demonstrated in n-dimensional space.

At this point, the relation between the method of the analogies and the scientific method results obvious [Pérez-Aguila03d]:

- Analysis: Observation of the analogies between 1D and 2D spaces; and between 2D and 3D spaces.
- Hypothesis: Proposal of an analogy between 3D and 4D spaces.
- Synthesis: Selection of a mechanism to demonstrate the analogy.
- Validation: The process of demonstration.
- Argumentation: The proposal of an n-dimensional generalization based in the analogies previously observed and the demonstration already achieved.

Along the remaining sections in this chapter, which resume our previous work, the continuous application of the Method of the Analogies could be contemplated.

2.3.2. Definitions

Orthogonal Polyhedra (3D-OP) are defined as polyhedra with all their edges (Π_1 's) and faces (Π_2 's) oriented in three orthogonal directions ([Juan-Arinyo88] & [Preparata85]). Orthogonal Pseudo-Polyhedra (3D-OPP) will refer to regular and orthogonal polyhedra with non-manifold boundary [Aguilera98].

Similarly, **4D Orthogonal Polytopes (4D-OP)** are defined as 4D polytopes with all their edges (Π_1 's), faces (Π_2 's) and volumes (Π_3 's) oriented in four orthogonal directions and **4D Orthogonal Pseudo-Polytopes (4D-OPP)** will refer to 4D regular and orthogonal polytopes with non-manifold boundary [Aguilera02].

2.3.3. The Π_{n-2} Analysis for 2D, 3D and 4D-OPP's

2.3.3.1. The Π_0 Analysis for 2D-OPP's

A set of quasi-disjoint rectangles determines a 2D-OPP whose vertices must coincide with some of the rectangles' vertices [Aguilera98]. Each of these rectangles' vertices can be considered as the origin of a 2D local coordinate system, and they may belong to up to four rectangles, one for each local quadrant. The two possible

adjacency relations between the four possible rectangles can be of edge or vertex. There are $2^4 = 16$ possible combinations which, by applying symmetries and rotations, may be grouped into six equivalence classes, also called configurations [Srihari81].

Because we are interested in the vertex analysis, we will consider only those configurations where all their rectangles are incident to the origin. According to the configurations' nomenclature presented in [Aguilera98], the studied configurations are b, c, d, e and f (see **Table 2.1**). There are only two types of vertices in the 2D-OPP's: **the manifold vertex with two incident edges** (configurations b and e), and **the non-manifold vertex with four incident edges** (configuration d) [Aguilera98]. The remaining configurations represent no vertex because configuration c has only two incident and collinear edges, and in configuration f there are no incident edges.

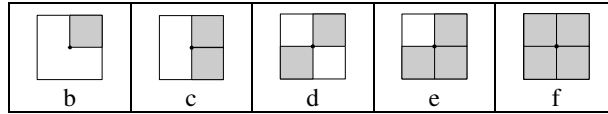


Table 2.1. The 2D configurations with all their rectangles incident to the origin.

2.3.3.2. The Π_1 Analysis for 3D-OPP's

A set of quasi-disjoint boxes determines a 3D-OPP whose vertices must coincide with some of the boxes' vertices [Aguilera98]. Each of these boxes' vertices can be considered as the origin of a 3D local coordinate system, and they may belong to up to eight boxes, one for each local octant. There are $2^8 = 256$ possible combinations which, by applying symmetries and rotations, may be grouped into 22 equivalence classes [Lorensen87], also called configurations [Srihari81]. Each configuration has its complementary configuration which is the class that contains the complementary combinations of all the combinations in the given class [Aguilera98]. Grouping complementary configurations leads to the 14 major cases [VanGelder94]. The configurations with 5, 6, 7 and 8 surrounding boxes are complementary, and thus analogous, to combinations with 3, 2, 1 and 0 surrounding boxes, respectively [Aguilera98]. Finally, each configuration, with four surrounding boxes is self-complementary.

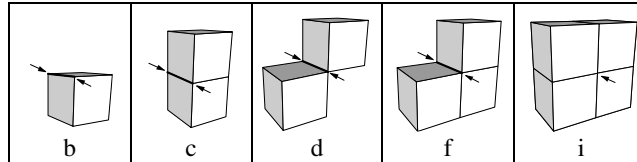


Table 2.2. The 3D configurations where all their boxes are incident to a same edge (the arrows show the analyzed edge).

Because we are interested in the edge analysis, we will consider only those configurations where all their boxes are incident to a same edge. According to the configurations' associated nomenclature presented in [Aguilera98], the studied configurations are b, c, d, f and i (see **Table 2.2**). In [Aguilera98] is concluded that there are only two types of edges in the 3D-OPP's:

- **The manifold edge with two incident faces.** This type of edges is found in configurations b and f. The edge's two incident faces in configuration b belong to one cube's boundary and they are perpendicular to each other. The edge's two incident faces in configuration f belong to two different cubes with edge adjacency and they result perpendicular to each other.
- **The non-manifold edge with four incident faces.** This type of edges is found in configuration d, where two of its faces belong to a cube and the remaining belong to a second cube with edge adjacency.
- The remaining configurations represent no edge because in configuration c there are only two incident and coplanar faces, and in configuration i there are no incident faces.

2.3.3.3. The Π_2 Analysis For 4D-OPP's

A set of quasi-disjoint *hyper-boxes* (i.e., hypercubes) determines a 4D-OPP whose vertices must coincide with some of the *hyper-boxes*' vertices. We will consider the *hyper-boxes*' vertices as the origin of a 4D local coordinate system, and they may belong to up to 16 *hyper-boxes*, one for each local *hyper-octant*. The 4D-OPP's vertices are determined according to the presence or absence of each of these 16 surrounding *hyper-boxes*. The four possible adjacency relations between the 16 possible *hyper-boxes* can be of volume, face, edge or vertex. There are

$2^{16}=65,536$ possible combinations of vertices in 4D-OPP's which can be grouped, applying symmetries and rotations, into 402 equivalence classes, also called configurations [Hill98]. Each configuration has its complementary configuration which is the class that contains the complementary combinations of all the combinations in the given class. Grouping complementary configurations leads to the 222 major cases [Hill98]. The combinations with 9, 10, 11, 12, 13, 14, 15 and 16 surrounding hyper-boxes are complementary, and thus analogous, to combinations with 7, 6, 5, 4, 3, 2, 1 and 0 surrounding hyper-boxes, respectively. Finally, each configuration, with eight surrounding hyper-boxes is self-complementary [Pérez-Aguila01].

We will consider only those configurations whose *hyper-boxes* are incident to a same face. According to the configurations' associated nomenclature presented in [Pérez-Aguila01], the studied configurations are 2, 3, 4, 7 and 13 (Table 2.3). In [Aguilera02] is concluded that there are only two types of faces in the 4D-OPP's:

- **The manifold faces with two incident volumes.** The face's two incident volumes in configuration 2 belong to the boundary of only one hypercube and they are perpendicular to each other. While in configuration 7, The face's two incident volumes belong to two different hypercubes with face adjacency and they result perpendicular to each other.
- **The non-manifold faces with four incident volumes.** This type of faces is found in configuration 4, where two of its incident volumes belong to a hypercube and the remaining two belong to a second hypercube with face adjacency.
- The remaining configurations represent no face because in configuration 3 there are only two incident and *co-hyperplanar* volumes, and in configuration 13 there are no incident volumes (analogous to 3D configurations c and i in Table 2.2).

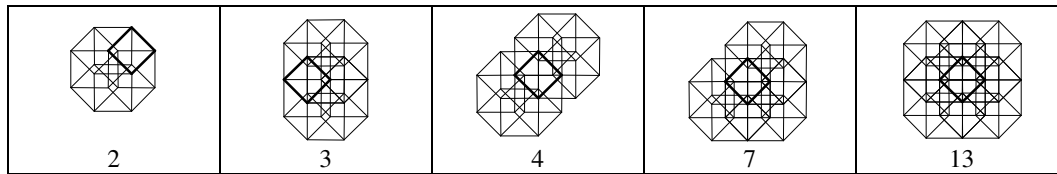


Table 2.3. Configurations 2, 3, 4, 7 and 13 for the 4D-OPP's.

2.3.4. The Π_{n-3} Analysis for 3D and 4D-OPP's

2.3.4.1. The Π_0 Analysis for 3D-OPP's

There are eight types of vertices (also two non valid vertices are identified) for 3D-OPP's [Aguilera98]. These vertices can be classified depending on the number of two-manifold and non-manifold edges incident to them and they are referred as V3, V4, V4N1, V4N2, V5N, V6, V6N1 and V6N2 [Aguilera98] (Table 2.4). In this nomenclature "V" means vertex, the first digit shows the number of incident edges, the "N" is present if at least one non-manifold edge is incident to the vertex and the second digit is included to distinguish between two different types that otherwise could receive the same name. In [Pérez-Aguila03c] the properties for each vertex are described with detail.

V3	V4	V4N1	V4N2	V5N	V6	V6N1	V6N2	Non valid vertex 1	Non valid vertex 2

Table 2.4. Vertices present in 3D-OPP's (dotted lines indicate non-manifold edges and continuous lines indicate manifold edges).

2.3.4.2. The Π_1 Analysis for 4D-OPP's

Vertices can be defined in terms of the manifold or non-manifold edges that are incident to these vertices in 3D-OPP's [Aguilera98]. The same process will be extended to describe edges in terms of the manifold or non-manifold faces that are incident to those edges in 4D-OPP's. In this way, we have identified eight types of edges and two non valid edges. We will also extend the nomenclature used by Aguilera & Ayala to describe them [Aguilera98]. Such edges will be referred as E3, E4, E4N1, E4N2, E5N, E6, E6N1 and E6N2 (Table 2.5) [Pérez-Aguila03d]. The only difference with the nomenclature used to describe the vertices is that "E" means edge instead of "V" that means vertex. In [Pérez-Aguila03c] the properties for each edge are described.

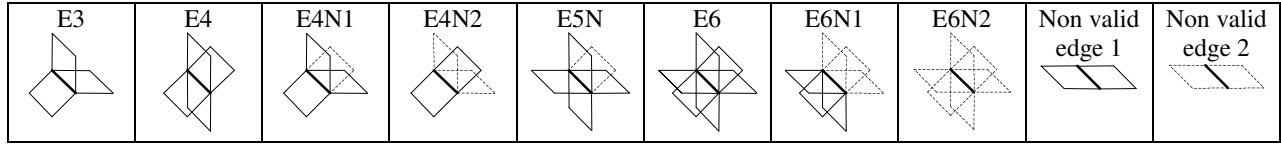


Table 2.5. Edges present in 4D-OPP's (dotted lines indicate non-manifold faces and continuous lines indicate manifold faces).

2.3.4.3. Classifying the Π_0 's in Polyhedra through its Cones of Faces

As seen in section 2.1.2, a **polyhedron** is a bounded subset of the 3D Euclidean Space enclosed by a finite set of plane polygons such that every edge of a polygon is shared by exactly one other polygon (adjacent polygons) [Preparata85]. A **pseudo-polyhedron** is a bounded subset of the 3D Euclidean Space enclosed by a finite collection of planar faces such that every edge has at least two adjacent faces, and if any two faces meet, they meet at a common edge [Tang91] (section 2.1.3). Edges and vertices, as boundary elements for polyhedra, may be either two-manifold (or just manifold) or non-manifold elements. In the case of edges, they are (non) manifold elements when every points of it is also a (non) manifold point, except that either or both of its ending vertices might be a point of the opposite type [Aguilera98]. A manifold edge is adjacent to exactly two faces, and a manifold vertex is the apex (i.e., the common vertex) of only one **cone of faces**. Conversely, a non-manifold edge is adjacent to more than two faces, and a non-manifold vertex is the apex (i.e., the common vertex) of more than one **cone of faces** [Rossignac91].

Using the concept of cones of faces it is easy to construct an algorithm to determine the classification of a vertex as manifold or non-manifold in any polyhedron or pseudo-polyhedron. Using this algorithm over the possible vertices in 3D-OPP's we have the results presented in **Table 2.6** which coincide with those presented in [Aguilera98].

3D vertex	Classification
V3	Manifold
V4	Manifold
V4N1	Non-manifold
V4N2	Non-manifold
V5N	Non-manifold
V6	Non-manifold or manifold according to its geometric context.
V6N1	Non-manifold
V6N2	Non-manifold

Table 2.6. 3D-OPP's vertices classification.

2.3.4.4. Classifying the Π_1 's in 4D Polytopes through its Hyper-Cones of Volumes

Due to the analogy between 3D-OPP's vertices described in terms of their incident manifold or non-manifold edges, and 4D-OPP's edges described in terms of their incident manifold or non-manifold faces, the next logical step is to extend the concept of cones of faces presented in the previous section to classify 4D polytopes' edges as manifold or non-manifold [Pérez-Aguila03].

Faces, edges and vertices, as boundary elements for 4D polytopes, may be either manifold or non-manifold elements. Coxeter and Hansen have stated that a manifold face is adjacent to exactly two volumes ([Coxeter63] & [Hansen93]), and Aguilera & Pérez [Pérez-Aguila03d] have suggested that a manifold edge is the common edge (apex) of only one **hyper-cone of volumes**. Conversely, we have suggested that a non-manifold face is adjacent to more than two volumes, and now we suggest that a non-manifold edge is the common edge (apex) of more than one **hyper-cone of volumes** [Pérez-Aguila03d].

Using the concept of hyper-cones of volumes, it is easy to extend the algorithm for obtaining the vertex classification for 3D-OPP's used for previous section, to allow us classifying an edge, as manifold or non-manifold, in any 4D polytope or 4D pseudo-polytope. The algorithm is defined with the following steps [Pérez-Aguila03c]:

- 1 Get the set of Π_3 's that are incident to edge A (a Π_1).
- 2 From the set of Π_3 's select one of them.
- 3 The selected Π_3 has two Π_2 's that are incident to A , get one of them and label it as *START* and *ANOTHER*.
- 4 Repeat
 - 4.1 If the number of Π_3 's to *ANOTHER* is more than one, then A is a non-manifold Π_1 . End.
 - 4.2 The *ANOTHER* Π_2 is common to another Π_3 , find it.
 - 4.3 The Π_3 has another Π_2 that is common to A , find it and label it as *ANOTHER*.
 - 4.4 Until *START* = *ANOTHER* (it has been found a hyper-cone of volumes).
- 5 If there are more Π_3 's to analyze then A is non-manifold (there are more hyper-cones of volumes). End.
- 6 Otherwise, A is manifold (A is the common edge of only one hyper-cone of volumes). End.

Using the algorithm over the possible edges in 4D-OPP's we have that the edges' classifications are analogous to the 3D-OPP's vertices' classifications [Pérez-Aguila03c]. **Table 2.7** shows the edges' classifications given by the extended algorithm and their analogous 3D results.

4D edge	Classification through hyper-cones of volumes	3D vertex	Classification through cones of faces
E3	Manifold	V3	Manifold
E4	Manifold	V4	Manifold
E4N1	Non-manifold	V4N1	Non-manifold
E4N2	Non-manifold	V4N2	Non-manifold
E5N	Non-manifold	V5N	Non-manifold
E6	Non-manifold or manifold according to its geometric context.	V6	Non-manifold or manifold according to its geometric context.
E6N1	Non-manifold	V6N1	Non-manifold
E6N2	Non-manifold	V6N2	Non-manifold

Table 2.7. 4D-OPP's edges classifications and their analogy with 3D-OPP's vertices.

2.3.5. Extreme Edges in the 4D-OPP's

In this section we will show how we have proceeded, as we have seen in the previous analogies between vertices in the 3D-OPP's and edges in the 4D-OPP's, to define the Extreme Edges. It will be described how this characterization is the result of a 3D analysis over the possible configurations for the 4D-OPP's.

2.3.5.1. The 2D Analysis for Vertices in the 3D-OPP's

We know that there are 22 configurations which determine a 3D-OPP through a set of quasi-disjoint boxes [Aguilera98]. Each of these boxes' vertices can be considered as the origin of a 3D local coordinate system. In such 3D local coordinate system can be identified three main planes: X_1X_2 , X_1X_3 and X_2X_3 . If the faces that are coplanar to such main planes are grouped, ignoring those faces that are shared by two cubes (face adjacency), they compose three 2D configurations (one for each main plane). For these 2D configurations the vertex can be classified as manifold or non-manifold [Pérez-Aguila03b]. By applying this analysis over the 22 configurations for the 3D-OPP's, it results that for those configurations whose number of boxes is odd, the three vertex analysis for their 2D configurations classify the 2D vertex as manifold (see for example, in **Figure 2.7**, configuration f) [Pérez-Aguila03b]).

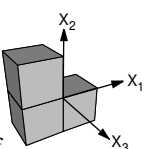
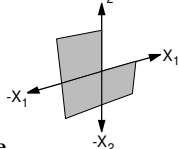
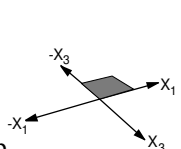
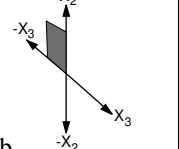
3D configuration	2D configuration on X_1X_2 Plane	2D configuration on X_1X_3 Plane	2D configuration on X_2X_3 Plane	Analysis for 2D vertex
				X_1X_2 : <i>Manifold</i> X_1X_3 : <i>Manifold</i> X_2X_3 : <i>Manifold</i>

Figure 2.7. Vertex analysis for 2D configurations on the main planes in 3D configuration f.

2.3.5.2. The 3D Analysis for Edges in 4D-OPP's

The vertex analysis for 2D configurations embedded in the main planes of a 3D configuration (previous section) classifies the 2D vertex as manifold or non-manifold. For consequence, in analogous way, we can assume that the edges analysis for 3D configurations embedded in the main hyperplanes of a 4D configuration will classify to 3D edges as manifold or non-manifold, and through these 3D analysis we can infer, due to the analogy with 3D vertex, if the 4D edges are characterized, and named as suggested by Aguilera & Pérez, as **Extreme** or **Non-Extreme** edges [Pérez-Aguila03b].

We know that there are 402 configurations which determine a 4D-OPP through a set of quasi-disjoint hyper-boxes [Hill98]. Each of these hyper-boxes' vertices can be considered as the origin of a 4D local coordinate system. In such 4D local coordinate system can be identified four main hyperplanes: $X_1X_2X_3$, $X_1X_2X_4$, $X_1X_3X_4$ and $X_2X_3X_4$. If the volumes that are *co-hyperplanar* to such main hyperplanes are grouped, ignoring those volumes that are shared by two hypercubes (volume adjacency), they will compose four 3D configurations (one for each main hyperplane) [Pérez-Aguila03b]. In **Table 2.8** there are shown the four 3D configurations that are present in some 4D configurations.

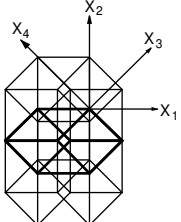
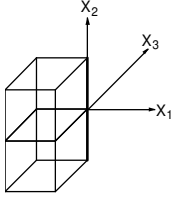
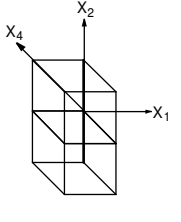
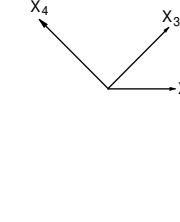
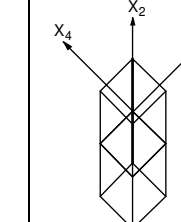
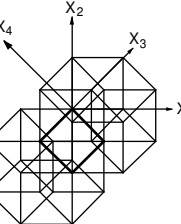
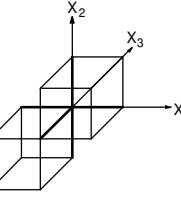
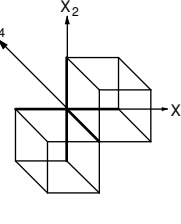
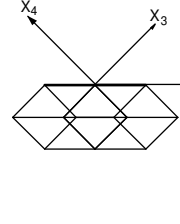
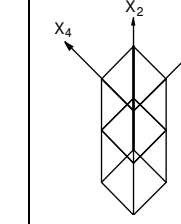
4D configuration	3D configuration on $X_1X_2X_3$ hyperplane	3D configuration on $X_1X_2X_4$ hyperplane	3D configuration on $X_1X_3X_4$ hyperplane	3D configuration on $X_2X_3X_4$ hyperplane
3 	b 	b 	a 	b 
4 	d 	d 	b 	b 

Table 2.8. Determining the 3D configurations on the main hyperplanes in 4D configurations 3 and 4.

For the 3D configurations that are embedded in the main hyperplanes it is possible to analyze their edges and classify them as manifold or non-manifold. In **Table 2.9** are shown the edges analysis for the 3D configurations that are present in 4D configurations 3 and 4.

4D Configuration	3D Edges Analysis			
	Configuration on $X_1X_2X_3$ hyperplane	Configuration on $X_1X_2X_4$ hyperplane	Configuration on $X_1X_3X_4$ hyperplane	Configuration on $X_2X_3X_4$ hyperplane
3	X_1 : Non edge $-X_1$: Non edge X_2 : Manifold $-X_2$: Manifold X_3 : Non edge $-X_3$: Non edge	X_1 : Non edge $-X_1$: Non edge X_2 : Manifold $-X_2$: Manifold X_4 : Non edge $-X_4$: Non edge	X_1 : Non edge $-X_1$: Non edge X_3 : Non edge $-X_3$: Non edge X_4 : Non edge $-X_4$: Non edge	X_2 : Manifold $-X_2$: Manifold X_3 : Non edge $-X_3$: Non edge X_4 : Non edge $-X_4$: Non edge
4	X_1 : Manifold $-X_1$: Manifold X_2 : Manifold $-X_2$: Manifold X_3 : Non edge $-X_3$: Non edge	X_1 : Manifold $-X_1$: Manifold X_2 : Manifold $-X_2$: Manifold X_4 : Non manifold $-X_4$: Non edge	X_1 : Manifold $-X_1$: Manifold X_3 : Non edge $-X_3$: Non edge X_4 : Non edge $-X_4$: Non edge	X_2 : Manifold $-X_2$: Manifold X_3 : Non edge $-X_3$: Non edge X_4 : Non edge $-X_4$: Non edge

Table 2.9. Edges analysis for 3D configurations on the main hyperplanes in 4D configurations 3 and 4.

By applying these analyses over the 402 configurations for the 4D-OPP's can be verified that [Pérez-Aguila03b]:

- An edge in a 4D-OPP can be classified by three 3D analysis (a 4D edge can only be present in three of the four main hyperplanes) as:
 - 3 times as manifold and 0 times as non-manifold, or
 - 0 times as manifold and once as non-manifold, or
 - 0 times as manifold and 3 times as non-manifold, or
 - 0 times as manifold and 0 times as non-manifold.
- The above patterns can be found in any 4D configuration because it can have from 0 to 8 incident edges to the origin.
- Following the analogy with the vertex analysis for 2D configurations embedded in the main planes of a 3D configuration (previous section), it can be proposed that if a edge in a 4D-OPP has been classified in the 3D analysis three times as manifold, then it can be considered, and named as suggested by Aguilera & Pérez, as an **Extreme Edge**, and any other result will classify it as a **Non-Extreme Edge** [Pérez-Aguila03b].
- The manifold or non-manifold classification for a edge in a 4D-OPP is independent of its classification as extreme or non-extreme.

Table 2.10 summarizes the identified characterizations for edges in the 4D-OPP's.

4D edge	Classification (manifold or non-manifold)	Classification (extreme or non-extreme)
E3	Manifold	Extreme
E4	Manifold	Non extreme
E4N1	Non-manifold	Extreme
E4N2	Non-manifold	Non extreme
E5N	Non-manifold	Non extreme
E6	<i>Non-manifold</i> <i>Manifold</i>	Non extreme Non extreme
E6N1	Non-manifold	Extreme
E6N2	Non-manifold	Non extreme

Table 2.10. The 4D-OPP's edges classifications.

2.4 Conclusions

We summarize the drawbacks identified in the previously analyzed schemes for modeling nD polytopes:

- The n-Dimensional Boundary Representations: The complexity imposed by geometry and topology to the algorithms that perform regularized Boolean operations.
- The n-Dimensional Simplexation of Convex Polytopes (as a specific cell decomposition): The time complexity imposed by the conversion, of an nD hypercube for example, to a simplexation: $O(n^3n!)$. Regularized Boolean operations have also a high cost because the way they are performed is related with boundary representations.
- The n-Dimensional Hypervoxelizations: Although regularized Boolean operations are trivial under this scheme,

the memory complexity requires having space for storing $\prod_{i=1}^n m_i$ hypervoxels, where m_i is the length of the hypervoxelization along X_i -axis.

- The 2^n -trees: In this scheme, regularized Boolean operations have a cost of linear time. However, when resolution's value m increases a better approximation of the polytope is obtained, but with a memory complexity increasing considerably according to the formula $\frac{2^{(m+1)n} - 1}{2^n - 1}$ which provides us an upper bound.

In conclusion, we can observe that as the number of dimensions increases, when we deal with the representation of n-Dimensional Polytopes we must consider:

- The domain of polytopes must be taken in account by the algorithms associated with the representation.
- The time complexity of the algorithms associated to the representation, and
- The memory complexity of the representation.

In the last section of this Chapter (section 2.3) we presented some geometrical and topological properties that characterize to the 2D, 3D and 4D-OPP's. Moreover, this analysis summarizes our previous work. These properties associated with the drawbacks previously discussed, lead us to conclude that representing nD-OPP's through these schemes could be problematic due to all the characteristics involved in the dominion of polytopes we are considering in this work.