Consiglio Nazionale delle Ricerche
Istituto di Matematica Applicata
e Tecnologie Informatiche
“Enrico Magenes”

REPORT SERIES

L. Chiang, F. Giannini, M. Monti

Identification of Patterns of Repeated Parts in Solid Objects

16-14

IMATI REPORT Series

Nr. 16-14 – November 2016

Editorial Office
Istituto di Matematica Applicata e Tecnologie Informatiche "*E. Magenes*"
Consiglio Nazionale delle Ricerche
Via Ferrata, 5/a
27100 PAVIA (Italy)
Email: reports@imati.cnr.it
http://www.imati.cnr.it

IMATI Report Series Nr. 16-14
2nd November 2016

Identification of Patterns of Repeated Parts in Solid Objects

L. Chiang, F. Giannini, M. Monti

Corresponding author: franca.giannini@ge.imati.cnr.it

Abstract.

The feature-based modeling approach for designing a mechanical component has been widely adopted by most CAD modelers to maintain the design intent on models. A set of higher level information is requested to enrich the usual geometrical and topological data, in terms of feature information. A well known example of semantically significant data of this type is represented by the knowledge of the existence of symmetrical arrangements of repeated sub-parts in CAD models. Because of the information transferring, repeated modifications of the models and different tolerances used by the various CAD systems, these specific data can be lost. In this report, after having provided the necessary mathematical background and the state of art of the existing symmetry detection methods, we propose an approach for retrieving symmetries involving a set of user-selected sub-parts in a given BRep part model. The approach is based on a preliminary analysis of the position of points (centroids), each of them representing one repeated feature; a further examination on the faces of the sub-parts is performed to confirm the existence of symmetric arrangements. We have provided the description for the detection of linear translational, reflectional, and circular translational patterns of repeated sub-parts.

Keywords: *Regular pattern detection, BRep models, shape analysis*

Contents

Introduction

Regular geometric structures are ubiquitous in both natural and man-made objects, as a universal concept. In the physical world, geometric symmetries and structural regularity occur at all scales, from crystal lattices and carbon nano-structures to the human body, architectural artifacts, and the formation of galaxies. This abundance of symmetry in the natural world has inspired mankind to incorporate symmetry in fields such as architecture, art, and engineering. To cite a prominent example, repeated structures and motifs are fundamental in almost all design styles used in architecture, as visual regularities is a commonly recognized aesthetic factor since ancient times [23], [24].

In engineering, symmetry in the design of manufactured parts has been gaining increasing interest as a result of economical, manufacturing, functional, or aesthetic considerations [24].

In mathematics, the term "symmetry" constitutes a function that, once applied to a shape, leaves it unchanged. In the context of computer graphics and engineering, the meaning is extended to a wider concept, including not only the classical geometric property referred to a single shape, but also a "regularity" intended as arrangements of repeated sub-parts of the model subjected to geometric transformations as reflections, translations, rotations or combinations thereof [23].

The most popular models employed by CAD (Computer-Aided Design) systems adopted by designers in industrial field are boundary representation (BRep) models.

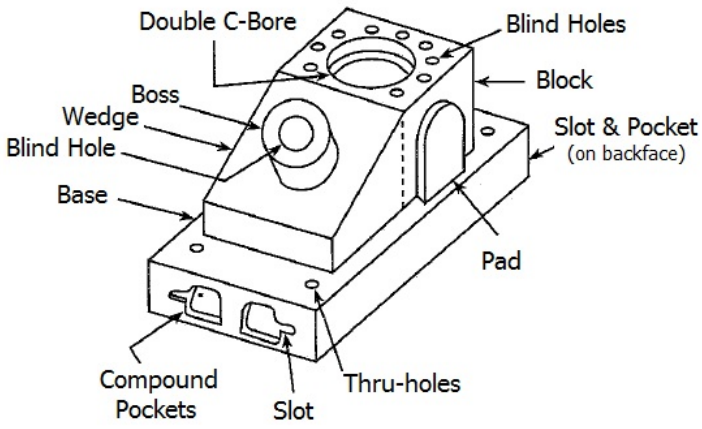


Figure 1: A feature based CAD model (source [27]).

BRep models describe objects in terms of their boundary, which is segmented in terms of a finite number of subset, called faces, which in turn are represented by their bounding edges and vertices [4]. These models are difficult to be directly handled by designers. To overcome such a limitation, CAD systems have been empowered by more advanced modeling capabilities including parametric and feature-based approaches [11], [25], [27].

The feature-based modeling approach for designing a mechanical component has been widely adopted by most CAD modelers to maintain the design intent on BRep models. With the term "*design intent*" we intend the set of detailed requirements, criteria, constraints established by the creator of the shape model, to fulfil his intended initial project requirements and to grant the expected functionalities once the object is manufactured. As a consequence, the design intent establishes constraints and guides on how the model behaves when dimensions are edited [2], [28].

A feature based CAD model is created through the chronological Boolean combination of finite features (pad, shaft, shell, hole, chamfer, and so on, see Figure 1) to generate the final solid. Furthermore, intentional patterns of congruent features can be

adopted by the CAD designer, such as circular patterns, rectangular patterns, reflectional patterns, or even user-defined patterns. In this way, a feature-based CAD model contains not only the BRep representation, but can be enriched by a set of higher level information, in terms of feature information, including feature types, parameters, references, and building histories. which can further exploited either in the object production or for the creation of new parts as a variation of the existing ones. In fact, it has been estimated that around 80% of all design tasks concerns the adaptation of existing designs to new requirements [27].

Therefore, when a designer generates a solid, its model structure may not contain the pattern operations or other symmetry constraints, due to the changes in the functional objectives and the consequent modifications that prevent the designer from generating a well structured representation of the final object. Furthermore, automatic translations between different file formats in CAD data transfer commonly lead to the loss of this information. Another recurring problem derives from the different tolerances used in CAD systems: what is symmetric in one CAD system might be not symmetric in another [21]. Similarly, geometric data acquired by scanning existing physical parts lack high-level information on symmetry by nature, as the acquired data is in form of point cloud structures that may also be incomplete or corrupted by noise [19].

Recovering explicit information on existing regular arrangements of repeated entities in a part can be very useful for the product development activities, from the design to the downstream processing stages. In the following, some examples showing the benefits which may derive from the availability of high-level symmetry information are provided.

- By reorganizing the structure of the building history of a model, intended as an integration of the regular arrangement information to the feature tree structure, future modifications can take advantage of a more explicit and ordered structure. For instance, changes on the characteristics of the identified patterns can automat-

ically change the position of the related elements, e.g. modifying the radius of a circular pattern. Moreover, once added this set of information, the representation of a solid can be further compressed by exploiting the symmetry information in a optimal way. For example, an object characterized by a global reflectional symmetry in a certain plane can be stored by keeping only data related to half an object and then annotating that the entire object can be obtained by replicating with a reflection in the specified plane. Similarly, for repeated elements arranged according to a predefined pattern, only a representative shape element can be stored together with the arrangement rule.

The resulting model deriving from the restructuring operation is called *compressed model*, and it must replicate the input model while preserving the accuracy of the manufacturing process (in terms of tolerances) [16].

- During the product development process, symmetry properties of mechanical components can be used to compute *tool path trajectories* of a machining process and to structure these trajectories in order to optimize the tool displacements. In the practice this helps locating the machining and assembly arms when the product needs to be machined and assembled [15], [16].

- *Reverse engineering* (or back engineering) is the process of extracting knowledge or design information from anything man-made and reproducing it. In some cases, for BReps arising from this process it could be very helpful to recover the design intent [17].

  In general, design intent concerning the shape of a CAD model can be expressed via intentional geometric properties of, and relations between, its vertices, edges, faces and sub-parts.

  In our context, we refer to recover design intent embodied as intentional high-level geometric relations between a CAD model congruent sub-parts or as sym-

metries of the single sub-parts.

- Having at disposal a set of high-level information may provide a substantial support in the retrieval of CAD models [23]. By considering queries that include symmetry constraints, the searching can more effective and selective than choosing more general and less significant searching parameters, such as the number and types of faces.

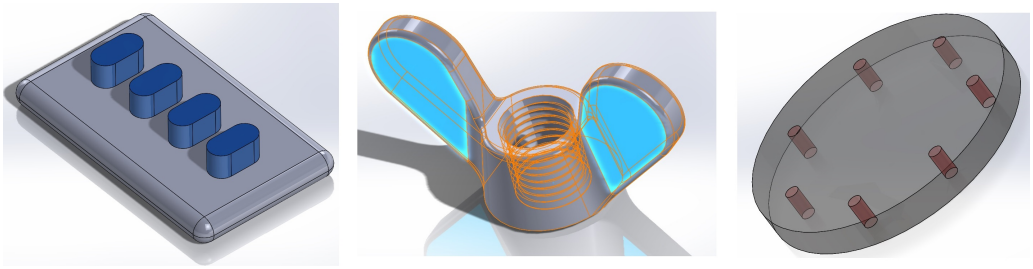
- Another process benefiting from the presence of information about existing regularities in a CAD model is the finite-element analysis (FEA). During this process symmetries can be used to facilitate the simplification of the model, to adapt it for the computational validation stage.

As a consequence of the wide variety of possible employments of the information on regularity, the symmetry concept has received significant attention in computer graphics and computer vision research in recent years. With the advances of computer technology, automatic methods for symmetry analysis have been developed and different approaches proposed. The main currently existing symmetry detection methods can be classified in two categories: *global symmetry detection methods* and *partial symmetry detection methods*. The first category of approaches focuses on the detection of a regularity involving the whole input shape, aiming to characterize the object as intentionally globally symmetrically designed. The detected regularities unnecessarily refer to the whole input model, indeed the second category of approaches aims to detect regularities also limited to a sub-part or a set of sub-parts of the entire model.

This work, as a report of [9], aims to provide a method for detecting symmetries involving a set of user-selected sub-parts in a given BRep model. Therefore, the proposed approach belongs to the partial symmetry detection method class, moreover it focuses on the detection of symmetric arrangements of congruent sub-parts, which constitute intentional patterns of repeated components in the model.

The goal is achieved by providing a preliminary analysis of the centroid location of the selected sub-parts, to select an initial set of repeated sub-parts, as candidates for a possible regular pattern. This analysis detects *all* the existing regular arrangements of the centroids. In the second stage, each candidate sub-part set is geometrically verified, by analyzing the correct position of the faces, with respect to the pattern of the corresponding centroids.

The proposed method is able to detect the following pattern types of congruent selected sub-parts:



a) b) c)

- **linear translational pattern** (Fig. a): the congruent sub-parts are characterized by a repetition along a fixed direction, placed at a constant distance from each other;

- **reflectional pattern** (Fig. b): referring to two of the selected sub-parts, they are interrelated by a reflectional relation in a fixed plane;

- **circular translational pattern** (Fig. c): the congruent sub-parts are related by translation, while they are positioned at a constant distance on a given circumference in the 3D space.

The remainder of this research work is organized as follows.

Chapter 1 provides the mathematical background concepts exploited in this work. In particular, it focuses on the process for modeling and representing solids objects, by listing a set of essential requirements a good model should fulfil to accurately represent the solid, to satisfy the validity conditions and to be computationally usable. The boundary representation scheme, as basic modeling scheme in CAD system, is characterized and analyzed. Mathematical theory related to isometries and symmetries in Euclidean space, with particular attention to $\mathbb{E}^2$ and $\mathbb{E}^3$, is reported. A set of formal definitions inherent to the application context is provided, explaining the meaning of "symmetry" in CAD uses.

Chapter 2 is dedicated to the state of art of the existing symmetry detection methods. We present a possible classification of the approaches developed in recent years as for symmetry detection, and we describe in details some existing algorithms relevant for this work.

Chapter 3 presents the proposed approach for the pattern detection of congruent sub-parts in a given BRep model. The set of entities specifically defined for the purpose is characterized and formally defined. The followed method to the guide the pattern search is step by step analyzed. The proposed algorithms are reported and commented.

A relevant number of tests has been performed to verify the effectiveness of the proposed pattern detection algorithm. A report of the most significant experiments is presented in *Chapter 4*; moreover, this chapter describes the development environment.

Chapter 1

Representation and symmetries in solid model

In this chapter the main notions related to isometries and symmetries in the Euclidean space are reported. Finally, a set of definitions related to the application context and relevant for this work aims are presented.

1.1 Isometries

In this section we will introduce the concept of *isometry*. This set of functions, when applied in 3D case, covers a fundamental role in the solid object modeling and analysis, as these functions preserve the shape of the objects once applied to the corresponding geometric models, and for this reason are relevant for our purposes.

More details about the notions introduced in this section can be found in [6]. In the following, with the term "map" we will refer to a function.

Definition 1.1.1. Let (X, d) be a metric space. A map $T : X \to X$ is an *isometry* if it

is invertible and preserves distances, so

$$d(T(x), T(y)) = d(x, y) \quad \text{for all } x, y \in X.$$

Observation 1.1.2. *The set of isometries of X forms a group $Isom(X)$ under composition.*

In the following, we consider as metric space the Euclidean one, whose metric is defined in terms of the scalar product of the vectors.

Definition 1.1.3. Let $a \in \mathbb{R}^n$. The map defined by

$$T_a : \mathbb{R}^n \to \mathbb{R}^n \quad x \mapsto x + a \tag{1.1}$$

is called the *translation by a*.

Observation 1.1.4. *Observe that*

$$\|T_a(x) - T_a(y)\| = \|x - y\|.$$

So the map T_a is an isometry.

Definition 1.1.5. Let $\lambda \in \mathbb{R}$ and u a unit vector. The set $\pi = \{x : x \cdot u = \lambda\}$ a hyperplane of $\mathbb{R}^n$. The map defined by

$$R_\pi : \mathbb{R}^n \to \mathbb{R}^n \quad x \mapsto x - 2(x \cdot u - \lambda)u \tag{1.2}$$

is called the *reflection in the hyperplane π*.

Observation 1.1.6. *Observe that*

$$\|R_\pi(x) - R_\pi(y)\| = \|(x - y) - 2((x - y) \cdot u)u\| = \|x - y\|.$$

So the map R_π is an isometry.

Observation 1.1.7. *Any vector $x \in \mathbb{R}^n$ can be written as $(x \cdot u)u + x^\perp$, with $x^\perp$ such that $x^\perp \cdot u = 0$, i.e. $x^\perp$ is perpendicular to u. Then, reflection in π leaves $x^\perp$ unaltered but maps $(x \cdot u)u$ to $(2\lambda - x \cdot u)u$.*

Definition 1.1.8. A linear map $f : \mathbb{R}^n \to \mathbb{R}^n$ is *orthogonal* when it preserves the inner product, i.e.

$$f(x) \cdot f(y) = x \cdot y \quad \text{for all } x, y \in \mathbb{R}^n. \tag{1.3}$$

Let B be the matrix associated to the orthogonal linear map f, so such that $f(x) = Bx$ for all $x \in \mathbb{R}^n$.

Since the inner product is given by $x \cdot y = x^t y$, we have that $x^t B^t B y = x^t y$. Hence, we have that

$$B^t B = I.$$

Definition 1.1.9. Let $\mathbf{M}_n(\mathbb{R})$ be the set of $n \times n$ real matrices, the set of matrices

$$\mathbf{O}(n) = \{M \in \mathbf{M}_n(\mathbb{R}) : M^t M = I\}$$

is said to be the *orthogonal group*.

To simplify, in the following, we will indicate as M the linear map associated to the matrix M, for every $M \in \mathbf{M}_n(\mathbb{R})$.

Lemma 1.1.10. *Every orthogonal linear map $B : \mathbb{R}^n \to \mathbb{R}^n$ is an isometry for the Euclidean metric. Conversely, if $B : \mathbb{R}^n \to \mathbb{R}^n$ is a Euclidean isometry that fixes the origin, B is an orthogonal linear map.*

Proof. Let B be an orthogonal linear map. Then for any $x, y \in \mathbb{R}^n$

$$d(Bx, By)^2 = \|Bx - By\|^2 = B(x - y) \cdot B(x - y) = (x - y) \cdot (x - y) = d(x, y)^2,$$

so B is an isometry.

Now suppose that B is an isometry of $\mathbb{E}^n$ that fixes the origin. The *polarization identity*:

$$\begin{aligned} 2x \cdot y &= \|x\|^2 + \|y\|^2 - \|x - y\|^2 \\ &= d(x, 0)^2 + d(y, 0)^2 - d(x, y)^2 \end{aligned}$$

shows that for any $x, y \in \mathbb{R}^n$

$$Bx \cdot By = x \cdot y.$$

So B preserves the inner product.

Let $\{e_1, \ldots, e_n\}$ be the standard orthonormal basis for $\mathbb{R}^n$. Because B preserves the inner product, (Be_i) is an orthonormal basis too. For any vector x we have

$$x = \sum_{i=1}^{n} (x \cdot e_i) e_i$$

and also

$$Bx = \sum_{i=1}^{n} (Bx \cdot Be_i) Be_i \tag{1.4}$$

$$= \sum_{i=1}^{n} (x \cdot e_i) Be_i \tag{1.5}$$

as B preserves the inner product. Hence

$$B : \sum_{i=1}^{n} x_i e_i \mapsto \sum_{i=1}^{n} x_i Be_i,$$

so B is a linear map.

Then B is an orthogonal linear map. □

Proposition 1.1.11. *If $A : \mathbb{R}^n \to \mathbb{R}^n$ is an isometry of the Euclidean space $\mathbb{E}^n$, then there is a vector $v \in \mathbb{R}^n$ and an orthogonal matrix B with*

$$A(x) = Bx + v \quad \textit{for } x \in \mathbb{R}^n.$$

Conversely, a map A defined by $A(x) = Bx + v$, with $v \in \mathbb{R}^n$ and $B \in \mathbf{O}(n)$, is an isometry.

Proof. Let A be an isometry of $\mathbb{E}^n$ and set $v = A(O)$, where O denotes the origin. Then the translation T_v is an isometry and

$$B = T_v^{-1} \circ A : x \mapsto A(x) - A(O)$$

is an isometry that fixes the origin. Hence Lemma 1.1.10 shows that B is the matrix associated to an orthogonal linear map and $A(x) = Bx + v$.

Conversely, suppose that $B \in \mathbf{O}(n)$, $v \in \mathbb{R}^n$ and $A(x) = Bx + v$. Then A is the composition of the isometry associated to B and the isometry translation by v, so it is also an isometry. □

1.1.1 Isometries in $\mathbb{E}^2$

Suppose B to be an isometry that fixes the origin. Then

$$B = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathbf{O}(2).$$

The columns of B are unit vectors orthogonal to one another, hence, by choosing $\theta \in [0, 2\pi)$ and imposing

$$\begin{pmatrix} a \\ c \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix},$$

for the vector $\begin{pmatrix} b \\ d \end{pmatrix}$ there are two possibilities:

$$\begin{pmatrix} b \\ d \end{pmatrix} = \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} b \\ d \end{pmatrix} = \begin{pmatrix} \sin\theta \\ -\cos\theta \end{pmatrix}.$$

The first case gives a *rotation* about the origin through an angle θ (or the identity when $\theta = 0$). The second gives *reflection* in the line $\{(x, y) \in \mathbb{R}^2 : y = (\tan\frac{1}{2}\theta)x\}$ at an angle $\frac{1}{2}\theta$ from the x-axis.

Let $B \in \mathbf{O}(2)$, $p \in \mathbb{R}^2$. If we conjugate B by a translation T_p, then we obtain another isometry $T_p \circ B \circ T_p^{-1}$. This isometry first translates p back to the origin, then applies B, and then translates the origin back to p. When B is a *rotation* through an angle θ, then $T_p \circ B \circ T_p^{-1}$ is a rotation about $T_p(O) = p$ through an angle θ. When B is a *reflection* in a line ℓ, then $T_p \circ B \circ T_p^{-1}$ is reflection in the line $T_p(\ell)$.

Now consider an isometry A defined in $\mathbb{E}^2$ that does not fix the origin. From Proposition 1.1.11, we have that $Ax = Bx + v$ for some $B \in \mathbf{O}(2)$ and some vector $v \in \mathbb{R}^2$. Depending on B we can have the following cases.

- When B is the identity, then A is a *translation* by v.
- When B is a rotation, we can always choose a vector p such that $(I - B)p = v$. This means that

  $$T_p \circ B \circ T_p^{-1}(x) = Bx + (I - B)p = Bx + v.$$

  So the isometry A is a *rotation* about the point p.
- When B is a reflection in a line ℓ trough the origin, then there exist $v_1, v_2 \in \mathbb{R}^2$ such that v_1 is perpendicular to ℓ, v_2 is parallel to ℓ and $v = v_1 + v_2$. The linear map $I - B$ maps onto the vector subspace of vectors perpendicular to ℓ, so we can choose a vector p with $(I - B)p = v_1$. This means that

  $$T_p \circ B \circ T_p^{-1}(x) = Bx + (I - B)p = Bx + v_1.$$

  So $Ax = Bx + v = T_p \circ B \circ T_p^{-1}(x) + v_2$. Furthermore:

  - when $v_2 = 0$, A is *reflection* in the line ℓ *translated* by p;
  - when $v_2 \neq 0$, A is a *glide reflection*, which is a reflection in the line ℓ translated by p followed by a translation parallel to ℓ.

Here follows a proposition distinguishing between *orientation preserving isometries* and *orientation reserving isometries*. The notion of these two categories of isometries is introduced in Subsection 1.1.3.

Proposition 1.1.12. *An orientation preserving isometry of the Euclidean plane $\mathbb{E}^2$ is:*

1. *the identity I;*
2. *a translation T_v, with $v \in \mathbb{E}^2$;*
3. *a rotation about some point $c \in \mathbb{E}^2$.*

An orientation reversing isometry of $\mathbb{E}^2$ is:

1. *a reflection R_ℓ, with ℓ a line in $\mathbb{E}^2$;*
2. *a glide reflection.*

1.1.2 Isometries in $\mathbb{E}^3$

With reasonings analogous to those for $\mathbb{E}^2$, the following is the proposition for isometries of $\mathbb{E}^3$.

Proposition 1.1.13. *An orientation preserving isometry of Euclidean space $\mathbb{E}^3$ is:*

1. *the identity I;*
2. *a translation T_v, with $v \in \mathbb{E}^3$;*
3. *a rotation about some line ℓ;*
4. *a screw rotation, that is a rotation about some line ℓ followed by a translation parallel to ℓ;*

An orientation reversing isometry of $\mathbb{E}^3$ is:

1. *a reflection R_π, with π a plane in $\mathbb{E}^3$;*

2. *a glide reflection, that in $\mathbb{E}^3$ is a reflection in a plane π followed by a translation parallel to π;*

3. *a rotatory reflection, that is a rotation about some axis ℓ followed by reflection in a plane perpendicular to ℓ.*

1.1.3 Other results for $Isom(\mathbb{E}^n)$

Recalling that, for Proposition 1.1.11, every $A \in Isom(\mathbb{E}^n)$ can be written as $A(x) = Bx + v$ with $B \in \mathbf{O}(n)$ and $v \in \mathbb{R}^n$, we define the map

$$\Phi : Isom(\mathbb{E}^n) \to \mathbf{O}(n) \quad \text{by} \quad A \mapsto B. \tag{1.6}$$

Proposition 1.1.14. *The map Φ defined in (1.6) given above is a group homomorphism with kernel equal to the group $Trans(\mathbb{E}^n)$ of all the translations of $\mathbb{E}^n$.*

Proof. Let A_1, A_2 be two isometries such that $A_k x = B_k x + v_k$ for every $x \in \mathbb{R}^n$, with $B_k \in \mathbf{O}(n)$, $v_k \in \mathbb{R}^n$, for $k = 1, 2$. Then

$$(A_2 \circ A_1)(x) = A_2(B_1 x + v_1) = B_2(B_1 x + v_1) + v_2 = (B_2 B_1)x + (B_2 v_1 + v_2).$$

Hence,

$$\Phi(A_2 \circ A_1) = B_2 B_1 = \Phi(A_2)\Phi(A_1)$$

which shows that Φ is a group homomorphism. The isometry $A : x \mapsto Bx + v$ is in the kernel of Φ when $\Phi(A) = B = I$. This means that A is a translation. □

We define another important homomorphism from $Isom(\mathbb{E}^n)$ that tells us whether an isometry preserves or reverses orientation, let it be

$$\varepsilon : Isom(\mathbb{E}^n) \to \{-1, +1\} \quad \text{by} \quad A \mapsto det(B), \tag{1.7}$$

where $A \in Isom(\mathbb{E}^n)$ is such that $A(x) = Bx + v$ with $B \in \mathbf{O}(n)$.

ε is a group homomorphism and the group of the orientation preserving isometries, denoted by $Isom^+(\mathbb{E}^n)$, is the kernel of ε. It is a normal subgroup of $Isom(\mathbb{E}^n)$.

Recall that, for any surjective group homomorphism $\alpha : G \to H$, the inverse images $\alpha^{-1}(h)$ are the *cosets* of $ker\alpha$ in G. The number of cosets is equal to the number of elements in H and is called the *index* of $ker\alpha$ in G. Since $\varepsilon : Isom(\mathbb{E}^n) \to \{-1, +1\}$ is a group homomorphism onto $\{-1, +1\}$, the subgroup $Isom^+(\mathbb{E}^n)$ has index 2 in $Isom(\mathbb{E}^n)$. This means that it has just two cosets $Isom^+(\mathbb{E}^n) = \varepsilon^{-1}(+1)$ and the complement $Isom^-(\mathbb{E}^n) = \varepsilon^{-1}(-1)$. For any orientation reversing isometry J, we have that $Isom^-(\mathbb{E}^n) = J\ Isom^+(\mathbb{E}^n)$.

Let G be a subgroup of $Isom(\mathbb{E}^n)$. The group $G^+ = G \cap Isom^+(\mathbb{E}^n)$ is the set of the orientation preserving isometries in G.

Observation 1.1.15. *Proposition 1.1.11 is translated in the fact that each $A \in Isom(\mathbb{E}^n)$ is affine, so*

$$A(\sum_{j=1}^{n} \lambda_j x_j) = \sum_{j=1}^{n} \lambda_j A(x_j) \quad \textit{provided that} \quad \sum_{j=1}^{n} \lambda_j = 1.$$

Observation 1.1.16. *Let G be a finite subgroup of $Isom(\mathbb{E}^n)$, let $a \in \mathbb{E}^n$. Considering the action of G on $\mathbb{E}^n$, the centroid of the orbit of a is*

$$c = \frac{1}{|G|} \sum_{T \in G} T(a).$$

Observe that $A(c) = c$ for each $A \in G$. Therefore all of the elements of G fix the point c.

Proposition 1.1.17. *A finite subgroup G of $Isom(\mathbb{E}^2)$ is either a cyclic group consisting of N rotations through angles $2\pi k/N$, $k = 0, 1, 2, \ldots, N-1$, about some point c, or else a group consisting of N rotations through angles $2\pi k/N$, $k = 0, 1, 2, \ldots, N-1$, about some point c and N reflections in lines through c (this group is called* dihedral group*).*

Proof. The centroid c is fixed by all of the isometries in G (Observation 1.1.16).

The subgroup G^+ is also a finite group, suppose it has order N. Then each transformation $A \in G^+$ is a rotation about c with $A^N = I$. Hence A must be a rotation through an angle $2\pi k/N$ for some integer $k \in \{0, 1, 2, \ldots, N-1\}$. There are only N such rotations about c so all of them must lie in G^+. Hence G^+ must be the cyclic group of order N generated by a rotation R about c through an angle $2\pi/N$.

If G consists only of orientation preserving isometries, $G = G^+$ is cyclic. Otherwise, there must be an orientation reversing isometry M in $G \setminus G^+$. This fixes c so it must be a reflection in a line ℓ through c. The homomorphism $\varepsilon : G \to \{-1, +1\}$ maps G^+ onto $+1$ and the coset G^+M onto -1, so $|G| = 2N$.

The products $M, RM, R^2M, \ldots, R^{N-1}M$ are all distinct and are reflections in the line obtained by rotating ℓ about c through angles $\pi k/N$ for $k = 0, 1, 2, \ldots, N-1$ respectively. So we see that G is dihedral of order $2N$. □

1.2 Object symmetries

In the context of geometry modeling, we consider geometric transformations as the "symmetry operations", such as reflections, translations, rotations, or combinations thereof ([23]). Formally in mathematics, referring to the 3D case, symmetries are transformations that, applied to an object intended as its geometric model, preserve its shape.

Definition 1.2.1. We say that a object M is (*globally*) *symmetric* with respect to $T \in Isom(\mathbb{E}^3)$, if $M = T(M)$, i.e., M is invariant under the action of T ([23]).

Definition 1.2.2. The *symmetry group* of a geometric object M is the set $\mathcal{S}_M$ defined as ([23])

$$\mathcal{S}_M = \{T \in Isom(\mathbb{E}^3) : T(M) = M\}. \tag{1.8}$$

$\mathcal{S}_M$ is a subgroup of $Isom(\mathbb{E}^3)$:

- *Closure*. If $T_1, T_2 \in \mathcal{S}$, then $T_2 \circ T_1 \in \mathcal{S}_M$.
- *Identity element*. The identity transformation $I \in \mathcal{S}_M$.
- *Inverse element*. For each $T \in \mathcal{S}_M$ there exists an element $T^{-1} \in \mathcal{S}_M$ such that $T^{-1} \circ T = T \circ T^{-1} = I$.
- *Associativity*. $(T_1 \circ T_2) \circ T_3 = T_1 \circ (T_2 \circ T_3), \forall T_1, T_2, T_3 \in \mathcal{S}_M$

This exact and global symmetry leads to a group structure because after applying a transformation, we end up with the same situation as before, creating a closed algebraic structure.

Definition 1.2.3. Let M_1, M_2 be two geometric objects. We say that M_1 and M_2 are *congruent* if there exist $T \in Isom(\mathbb{E}^3)$ such that $T(M_1) = M_2$ ([15]).

Definition 1.2.4. Let M be a geometric object. M' such that $M' \subseteq M$ is called a *sub-part* of M.

Definition 1.2.5. Let M be a geometric object. We say that M has a *partial symmetry* with respect to an isometry T if there exist two sub-parts M_1, M_2 of M such that they are congruent by T [23].

Observation 1.2.6. *If in the previous definition $M_1 = M_2 = M$, the object is globally symmetric. So global symmetry is a special case of partial symmetry.*

1.2.1 Examples in 2D

Now we show some examples of symmetry groups in the 2D case.

In Figure 1.1 the dihedral group D_3 is graphically represented (see also Proposition 1.1.17): this group of six elements represents the symmetries of the equilateral triangle. Figures 1.1 a), 1.1 b), 1.1 c) indicate the three symmetries representing rotations. Note that a rotation of 360° is equal to the identity transformation. Figures 1.2 d), 1.2 e), 1.2

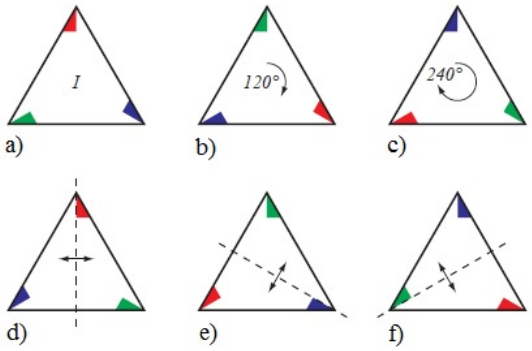


Figure 1.1: The dihedral group D_3 (source [23]).

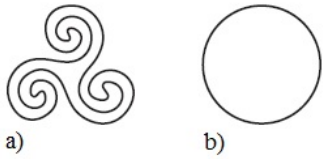


Figure 1.2: a) Triskelion; b) Circle: $\mathbf{O}(2)$ is the symmetry group of this shape (source [23]).

f) refer to the three reflectional symmetries across the lines from each vertex through the triangle center [23].

In general, the *dihedral group* D_n represents the symmetry group of a regular n-gon and the symmetries can be represented as finite combinations of two generating transformations: the rotation through angle $2\pi/n$ and a reflection.

In Figure 1.2 a) there is an example of shape with rotational symmetries but no reflectional symmetries. The name of this shape is *triskelion* and it is characterized by a *cyclic group* (see Proposition 1.1.17), denoted by C_3, whose generator is the rotation through angle $2\pi/3$. C_3 is a finite group and has three elements [23].

In general, the *cyclic group* C_n is generated by the rotation through angle $2\pi/n$.

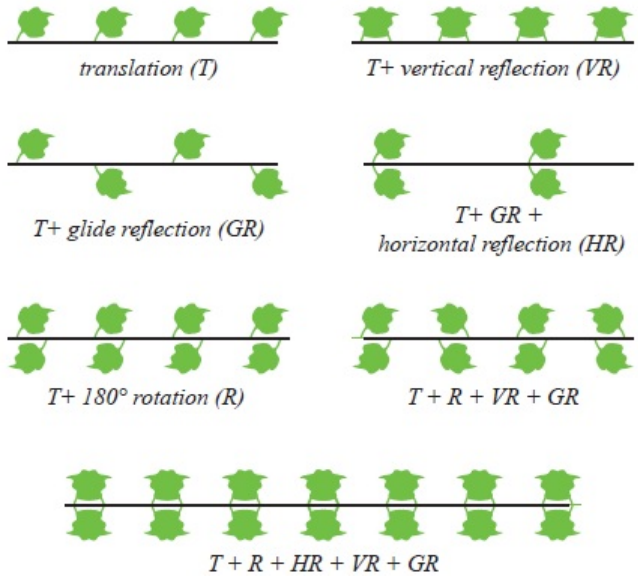


Figure 1.3: Frieze groups: they are composed of translation, rotation by 180°, glide reflection, reflection about a horizontal line, or reflection about a vertical line. (source [23]).

Figure 1.4: Tiling patterns of Alhambra: they are described by wallpaper groups. (source [23]).

The symmetry group of the circle (see Figure 1.2 b)) is represented by all the isometries that leave the origin fixed, so the orthogonal linear maps associated to the matrices in $\mathbf{O}(n)$ (as stated by Lemma 1.1.10). Indeed, the circle is symmetric with respect to rotations of arbitrary angle around its center and reflections across arbitrary lines through its center.

Symmetry groups have been used extensively in the study of decorative art and structural ornaments. The symmetries of a two-dimensional surface that is repetitive in one direction and extends to infinity along that direction can be classified by one of exactly seven *Frieze groups* [20], [23]. Figure 1.3 contains their representation. If repetition occurs in two different directions, seventeen distinct groups are possible, denoted as *wallpaper groups* (from [23], see Figure 1.4).

1.2.2 Symmetries for applications: approximate symmetries

In this subsection we will report a set of definitions strictly related to the applications and to the works described in the Chapter 2. For these reasons the definitions will take into account a *tolerance* factor and will not request exact equality, accepting the correspondence except for a given tolerance constant.

The following is the definition of *approximate symmetry* (from [23]).

Definition 1.2.7. Let d be a function that we intend as the indicator of the "distance" between two geometric object from being congruent. We say that M is ε*-symmetric* with respect to an isometry T if $d(M, T(M)) < \varepsilon$.

Different variants of distance functions have been proposed for the applications, depending on the specific case (see [23] for more details).

Observation 1.2.8. *A difficulty with approximate symmetries is that ε-symmetries are not closed under composition and thus the group structure is not preserved, while exact*

symmetries do (see Section 1.2). Indeed the situation $d(M, T_1(M)) < \varepsilon$, $d(M, T_2(M)) < \varepsilon$ *but* $d(M, T_2 \circ T_1(M)) \geq \varepsilon$ *could occur.*

The references of all the following notions and definitions are [17] and [19].

Suppose to be in a 3-dimensional Euclidean space $\mathbb{E}^3$. We denote the set of all euclidean distances between members of a finite point set $\mathcal{P} \subset \mathbb{E}^3$ as $\mathcal{D}(\mathcal{P})$, in formula:

$$\mathcal{D}(\mathcal{P}) = \{\|P - Q\| : P, Q \in \mathcal{P}\}. \tag{1.9}$$

Furthermore, if a, b are two real numbers we write $a =_\epsilon b$ to say that $|a - b| \leq \epsilon$, i.e. to say that a and b are *approximately equal* with tolerance ϵ.

Definition 1.2.9. Let $\mu : \mathcal{P}_1 \to \mathcal{P}_2$ be a bijection between two point sets $\mathcal{P}_1, \mathcal{P}_2$, and let $\epsilon \geq 0$ be a tolerance. We say $DEC(\mathcal{P}_1, \mathcal{P}_2, \mu, \epsilon)$ is satisfied if $\|P - Q\| =_\epsilon \|\mu(P) - \mu(Q)\|$ for all $P, Q \in \mathcal{P}_1$, and if $=_\epsilon$ is an equivalence relation on $\mathcal{D}(\mathcal{P}_1) \cup \mathcal{D}(\mathcal{P}_2)$. We define two point sets $\mathcal{P}_1$, $\mathcal{P}_2$ to be *approximately congruent* at tolerance ϵ if, for at least one of all possible bijections $\mu : \mathcal{P}_1 \to \mathcal{P}_2$, $DEC(\mathcal{P}_1, \mathcal{P}_2, \mu, \epsilon)$ is satisfied. We say that a point set $\mathcal{P}$ has an *approximate symmetry* (μ, ϵ) for bijection $\mu : \mathcal{P} \to \mathcal{P}$ and tolerance ϵ if $DEC(\mathcal{P}, \mathcal{P}, \mu, \epsilon)$ is satisfied.

Observation 1.2.10. *The condition that* $=_\epsilon$ *forms an equivalence relation means that the set* $\mathcal{D}(\mathcal{P})$ *is grouped into pairwise distinct subsets of approximately equal distances (the equivalence classes).*

Definition 1.2.11. If a set of points $\mathcal{P}$ is approximately symmetric under a bijection $\mu : \mathcal{P} \to \mathcal{P}$, we call *tolerance validity interval* the interval $E_\mathcal{P} = [E_{min}(\mathcal{P}), E_{max}(\mathcal{P}))$ containing the values of ϵ such that (μ, ϵ) is still an approximate symmetry. $E_{min}(\mathcal{P})$ is called the *minimal tolerance* and $E_{max}(\mathcal{P})$ is called the *maximal tolerance*.

Definition 1.2.12. Let $\mathcal{C} = (P_1, \ldots, P_c)$ be a sequence of $c \geq 2$ points from a point set $\mathcal{P}$, which induces a bijection μ mapping P_k to P_{k+1} for $k = 1, \ldots, c-1$. We say that $\mathcal{C}$ is a (*maximal approximate*) *incomplete cycle* at tolerance $\epsilon \geq 0$ if

(1) $DEC(\mathcal{C}, \mathcal{C}, \mu, \epsilon)$ is satisfied;

(2) no point in $\mathcal{P} \setminus \mathcal{C}$ can replace a point in $\mathcal{C}$ while (1) still holds under the same μ ($\mathcal{C}$ must be unambiguous);

(3) no single point in $\mathcal{P} \setminus \mathcal{C}$ can be added to $\mathcal{C}$ for any tolerance ϵ while still satisfying properties (1) and (2).

Observation 1.2.13. *The given definitions cover all the seven elementary isometries in 3D described in [20]:* reflection*,* inversion*,* translation*,* n-fold rotation*,* n-fold rotation-reflection *(reflection followed by* n*-fold rotation about an axis orthogonal to the mirror plane),* glide *(reflection in a line followed by translation parallel to the line), and* screw.

Sometimes only a subset of a set of points could have a symmetry. Here follows a definition of approximate incomplete symmetry of a subset of points in terms of the just defined incomplete cycles.

Definition 1.2.14. Let $\mathcal{P}$ be a set of points. A point subset $\mathcal{S} \subset \mathcal{P}$ has an (*approximate*) *incomplete symmetry* (μ, ϵ) of symmetry type t if

(1) $\mathcal{S}$ is the union of a set of non-intersecting cycles of type t, each having at least $N(t)$ points depending on the given type t: for example 2 points if t corresponds to reflection, inversion, 2-fold rotation; 3 for translation, 3-fold rotation; 4 for n-fold rotation with $n \geq 4$, glide; 5 for n-fold rotation-reflection, screw.

(2) $DEC(\mathcal{S}, \mathcal{S}, \mu^*, \epsilon)$ is satisfied, where μ^* is the concatenation of the individual μs of the cycles from (1);

(3) no cycle $\mathcal{C}^*$ of type t in $\mathcal{P} \setminus \mathcal{S}$ exists such that (2) is true for $\mathcal{C}^* \cup \mathcal{C}$ at tolerance less than ϵ for any cycle $\mathcal{C}$ of $\mathcal{S}$.

Now let us consider a particular type of points. In [17] authors widely use *characteristic points*: they are model's vertices and some other special points which characterize

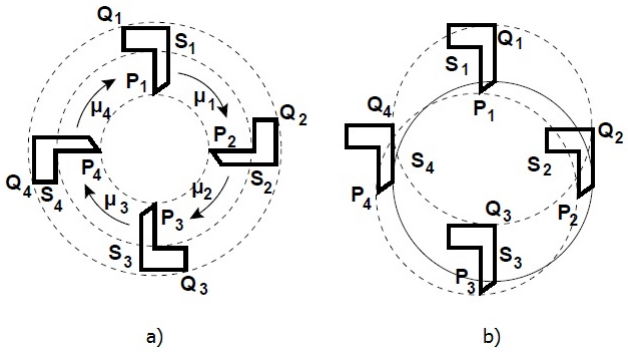


Figure 1.5: a) Type I of symmetric arrangement of sub-parts; b) Type II of symmetric arrangement of sub-parts (source [17]).

curved edges and faces. These points are able, together with topological and face type information, to uniquely characterize the model ([13], [22]). They are introduced to verify the sub-parts of the BRep to be congruent or symmetric in [17]. More precisely, here a *consistency condition* is reported with the aim of ensuring matching of entities of the same *geometric type*.

Definition 1.2.15. Let $\mu : \mathcal{H}_1 \to \mathcal{H}_2$ be a bijection between the characteristic point sets $\mathcal{H}_1, \mathcal{H}_2$ of two sub-parts S_1, S_2 of a given BRep model. We say μ is *consistent* if whenever a subset $\mathcal{H}^* \subset \mathcal{H}_1$ defines an entity of S_1, $\mu(\mathcal{H}^*)$ are corresponding characteristic points of an entity of the same type of S_2.

Furthermore, according to the previous definition: two sub-parts are *approximately congruent* if their characteristic point sets are congruent at tolerance ϵ under a consistent bijection μ; a sub-part is *approximately symmetric* at tolerance ϵ if its characteristic points are symmetric under a consistent bijection μ.

1.3 Symmetric arrangements of sub-parts

Now the definitions used in [17] to formally introduce the concept of symmetric arrangements of congruent sub-parts in a 3D model are given. In short, a symmetric arrangement can be seen as a pattern of congruent sub-parts determined by a symmetry group. The authors of [17] only consider two particularly important types of pattern of interest to mechanical engineering, even if the various possibilities can be determined by 230 *space groups* ([12]). The definitions make use of the concept of *location cycle*: with this term we intend a cycle containing one characteristic point from each congruent sub-parts, such that each of these points is situated in the same location in every sub-part.

Definition 1.3.1. A set of congruent sub-parts $S_1, \ldots, S_s$ with characteristic point sets $\mathcal{H}_1, \ldots, \mathcal{H}_s$ has a *symmetric arrangement* at tolerance ϵ if consistent bijections $\mu_k : \mathcal{H}_1 \to \mathcal{H}_k$, $k = 2, \ldots, s$ exist such that for all $P_1 \in \mathcal{H}_1$ with location cycle $\mathcal{C}(P_1) = (P_1, \mu_2(P_1), \ldots, \mu_s(P_1))$, and

- if all location cycles $\mathcal{C}(P_1)$, $P_1 \in \mathcal{H}_1$ form an incomplete symmetry at tolerance ϵ the symmetric arrangement is of *Type I*;
- if for any other point $Q_1 \in \mathcal{H}_1$, translating cycle $\mathcal{C}(P_1)$ by vector $Q_1 - P_1$ gives cycle $\mathcal{C}(Q_1)$ at tolerance ϵ the symmetric arrangement is of *Type II* .

A symmetric arrangement can be also interpreted as the repeated application of an isometric transformation. In Figure 1.6 there is an illustration of some regular structures. Formally, in [24] a *regular structure* of size n is defined as a tuple $(\mathcal{P}, \mathcal{G})$, where the set $\mathcal{P} = \{P_0, \ldots, P_{n-1}\}$ is a collection of n congruent sub-parts $P_k \subset \mathcal{S}$, and $\mathcal{G}$ is a transformation group acting on $\mathcal{P}$. For instance, let $\mathcal{G}$ be a 1-parameter group with T as generating transformation. Then, the element $P_k \in \mathcal{P}$ can be represented

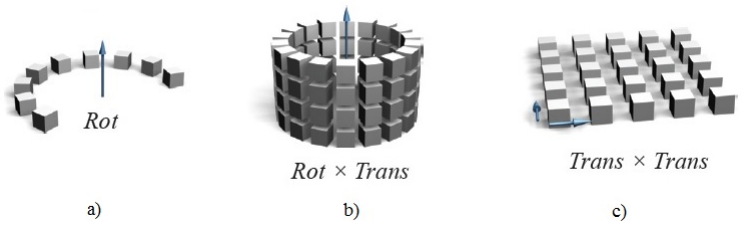


Figure 1.6: a) regular structure with 1-parameter group transformation (rotation); b) regular structure with 2-parameter group transformation (rotation, translation); c) regular structure with 2-parameter group transformation (translation, translation) (source [24]).

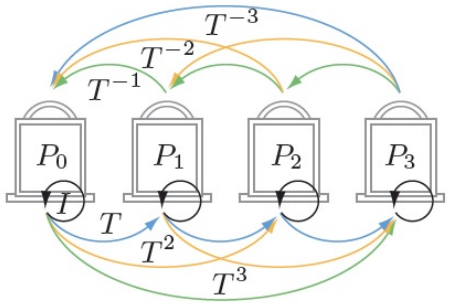


Figure 1.7: A 1D regular structure of size 4 and the representation of all the transformations between the repeated congruent elements (source [24]).

as $P_k = T^k P_0$ for $k = 0, \ldots, n-1$. Furthermore, any element $P_i \in \mathcal{P}$ of a regular structure can be transformed into any other element $P_j \in \mathcal{P}$ by the transformation $T_{ij} = T^{j-i}$. In Figure 1.7 there is an example of a 1D regular structure.

In general, the geometry of a k-parameters regular structure $(\mathcal{P}, \mathcal{G})$ can be compactly represented by a single sub-part $P_0 \in \mathcal{P}$, the $\mathcal{G}$ group generator(s) $T_1, \ldots, T_k$, and the integer dimension(s) $n_1, \ldots, n_k$ with $n_1 \cdot \ldots \cdot n_k = n = |\mathcal{P}|$. We call P_0 the *representative element* and the tuple $(P_0, \{T_i\}, \{n_i\})$ the *generative model* of the regular structure.

We observe that a regular structure defined in this way can be seen as a repeated par-

tial symmetry with respect to the isometry T between the sub-parts in $\mathcal{P}$ (see Definition 1.2.5).

Chapter 2

Related works

The concept of symmetry has received significant attention in computer graphics and computer vision research in recent years. Numerous methods have been proposed to find and extract geometric symmetries and exploit such high-level structural information for a wide variety of geometry processing tasks [23]. In the first section we will provide a classification of the existing symmetry detection approaches for 3D objects, by differentiating them according to the portion extension of the model the detected regularities involve. Finally some existing method of interest for this work are examined in details.

2.1 Classification of symmetry detection methods

Theoretical characterization of symmetry detection algorithms has been a topic of interest in computational geometry. In [23] the main existing symmetry detection algorithms are classified by considering if the resulting symmetries involve the whole input object or only parts of it.

- **Global symmetry detection algorithms.** The global Euclidean symmetries for finite objects can only occur as reflection or rotation (see Definition 1.2.1). This

class of algorithm generally exploit an important property shared by all models exhibiting a global symmetry: the planes of reflection and/or the axes of rotation pass through their center of mass of the object. This property greatly reduces the search field for symmetry extraction.

The computation of global symmetries can be further simplified if reliable global shape descriptors can be computed (as in method described in Section 2.2).

- **Partial symmetry detection algorithms.** As just observed in Observation 1.2.6, global symmetries are particular cases of partial symmetries (see Definition 1.2.5). In general, the algorithm aimed at partial symmetry detection share many similarities in their structure and the main stages of these procedures can be more or less summarized as follows.

  1. *Feature/sub-part selection*: this stage aims to decompose the entire model in a set of smaller subsets of interest for the computation.
  2. *Symmetry collection*: identification of the local symmetries information.
  3. *Extraction*: the final stage consists of the detection of the meaningful partial symmetries obtained from the collection of the detected local symmetries.

  Some partial symmetry detection methods of interest will be described in details in Section 2.3.

2.2 Global symmetry detection algorithms

2.2.1 The Martinet, Soler, Holzschuch, Sillion's method

The global symmetry detection method proposed by Martinet et al. in [21] focuses on 3D mesh based models. The main tools used by this algorithm are the *generalized moment* functions.

The authors intend to find an the global symmetry of the input object as an isometry A represented by a matrix which takes the following form:

$$A(\lambda,\alpha) = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \quad \text{with} \begin{cases} \alpha \in [0, 2\pi[\\ \lambda = \pm 1 \end{cases} \tag{2.1}$$

This corresponds to three different classes of isometries: rotations, reflections, and their compositions, depending whether λ is positive and/or $\alpha = 0(\text{mod}\pi)$. The final output consists of a vector v, called the $axis$ of the isometry, and an angle α, called the *angle* of the isometry, such that $A(\lambda, \alpha)$ maps the shape onto itself.

The approach is deterministic and finds good candidates of v, λ, α by using intermediate functions whose set of symmetries is a superset of the symmetries of the shape it refers to. These functions are defined in the following.

Definition 2.2.1. For a surface $\mathcal{S}$ in a 3D, we define its *generalized moment* of order $2p$ in direction ω by

$$\mathcal{M}^{2p}(\omega) = \int_{s\in\mathcal{S}} \|s \times \omega\|^{2p} ds. \tag{2.2}$$

s identifies a vector linking the center of gravity of the shape (placed at the origin) to a point on the surface. $\mathcal{M}^{2p}$ is itself a directional function.

Observe that the denomination of these functions is due to the fact that, considering $\mathcal{S}$ to have a thickness dt, the expression $\mathcal{M}^2(\omega)dt$ corresponds to the *moment of inertia* of the this shell $\mathcal{S}$ along ω.

In [21] there is the proof of the following fact: any symmetry A of a shape $\mathcal{S}$ also is a symmetry of all its $\mathcal{M}^{2p}$ moment functions; furthermore, if $\mathcal{M}^{2p}$ has a symmetry A with axis ω, then the gradient of $\mathcal{M}^{2p}$ is null at ω. In formula

$$A(\mathcal{S}) = \mathcal{S} \Rightarrow \forall\omega\ \mathcal{M}^{2p}(A(\omega)) = \mathcal{M}^{2p}(\omega) \tag{2.3}$$

$$\mathcal{M}^{2p}(A(\omega)) = \mathcal{M}^{2p}(\omega) \Rightarrow (\nabla\mathcal{M}^{2p})(\omega) = 0. \tag{2.4}$$

So, once the directions of the zeros of the gradients of the moment functions have been found, they must be checked on the shape itself to eliminate those not of interest for the input object, i.e. the false positives.

They provided an efficient method to compute the spherical harmonic coefficients of the moment functions, avoiding to sampling the functions. In [21] the passages to decompose $\mathcal{M}^{2p}$ into a finite number of spherical harmonics are shown and they provide an equation to directly compute the corresponding coefficients. The final result is a decomposition of this type:

$$\mathcal{M}^{2p} = \sum_{l=0}^{p} \sum_{m=-2l}^{2l} C_{2l,m}^{2p} Y_{2l}^{m}(\omega). \tag{2.5}$$

The coefficients $C_{2l,m}^{2p}$ are surface integrals and numerical accuracy only concern this computation.

This way of computing $\mathcal{M}^{2p}$ is much cheaper than the alternative method of computing the scalar product as defined by Equation 2.2 with each spherical harmonic basis function.

The proposed algorithm takes as starting point the solving of $(\nabla \mathcal{M}^{2p})(\omega) = 0$. The first step of this computation is done by estimating a number of vectors which are close to the actual solutions, then by refining the sphere of directions starting from an icosahedron, examining the value of $\|(\nabla \mathcal{M}^{2p})(\omega)\|^2$ in each face in several directions. Only faces with small minimum values are refined recursively. A descent minimization on $\|(\nabla \mathcal{M}^{2p})(\omega)\|^2$ is applied starting from the selected candidates, usually converging in few steps, because starting positions are by nature very close to actual minima.

Once found the zero directions of the gradient of the moment functions, the parameters of the corresponding isometric transformations are deterministically extracted by studying the spherical harmonic coefficients of the moment functions themselves. These are the properties used (see [14]):

1. A function has a reflectional symmetry R_z around the $z = 0$ plane if and only if

all its spherical harmonic coefficients for which $l + m$ is even are equal to zero. In the specific case of the moment functions:

$$m \equiv 0(\text{mod}2) \Rightarrow C_{2l,m}^{2p} = 0.$$

2. A function has a revolution symmetry around z axis if and only if

$$\forall l \; \forall m \;\; m \neq 0 \Rightarrow C_l^m = 0.$$

3. A function has a rotational symmetry Rot_α of angle α around z axis if and only if

$$\forall l \; \forall m \;\; C_l^m = \cos(m\alpha)C_l^m - \sin(m\alpha)C_l^{-m}.$$

4. In case of composition of a rotation and a symmetry with the same axis the equation to be checked for is

$$\forall l \; \forall m \;\; (-1)^{l+m}C_l^m = \cos(m\alpha)C_l^m - \sin(m\alpha)C_l^{-m}.$$

These properties are checked by admitting a tolerance, so the final detected symmetries are approximated.

The statement in (2.3) is a necessary condition only. For this reason, the directions and rotation angles obtained from the moment functions must be verified on the shape itself. This is done by using a *symmetry measure* inspired by the work in [29]. Let $\mathcal{S}$ and $\mathcal{R}$ be two tassellated shapes, let $V_\mathcal{S}$ and $V_\mathcal{R}$ respectively be their mesh vertices. Then the *measure* d_M between $\mathcal{S}$ and $\mathcal{R}$ is defined by

$$d_M(\mathcal{S}, \mathcal{R}) = \max_{p \in V_\mathcal{S}}(\min_{q \in V_\mathcal{R}} \|p - q\|).$$

The *symmetric measure* $d_A(\mathcal{S})$ of a shape $\mathcal{S}$ with respect to a symmetry A is then defined by:

$$d_A(\mathcal{S}) = \max(d_M(\mathcal{S}, A\mathcal{S}), d_M(A\mathcal{S}, \mathcal{S})).$$

This is the last step of the algorithm for single shape symmetry detection.

The authors of this work also provided a constructive algorithm for composite shapes built by assembling simpler structures. This algorithm first compute the symmetries of sub-parts by applying the just described algorithm, considering each sub-parts as an independent entity, then iteratively builds the set of symmetries of the composite shape, taking into account both the relative positions of the subparts and their relative orientations.

In conclusion, the presented method is an efficient approach to accurately retrieve symmetries in geometric mesh based models, independently of their tassellation. Computing the generalized moment functions by spherical harmonic coefficients, the algorithm allows to find potential symmetry quickly and with a good accuracy.

We are interested in detecting symmetries and regularities in BRep models, which do not present a sufficient number of vertices to apply the just described approach.

2.3 Partial symmetry detection algorithms

In this section the outstanding steps followed by four symmetry detection methods addressed to partial symmetry retrieval are reported.

For the first detection method, the contents of two reference papers have been reported. Indeed, the first paper describes an approach of symmetry detection referred to a discrete point set, although the shape models this research focuses on are boundary representations, but it is preliminary for the second one, referred to BRep. Furthermore, the just cited discrete point set detection method is very close to the approach proposed in Chapter 3.

These four works have been chosen from the existing ones because they are more relevant for this work subject.

2.3.1 The Li, Langbein, Martin's method

The proposed symmetry detection approach for discrete point set

In this section the symmetry detection approach proposed by Li, Langbein and Martin in 2007 for [19] is summarized. The aim of their method is to detect *local approximate symmetries in a discrete point set* derived from a BRep model. These points are carefully chosen *characteristic points* (introduced in Chap. 1 Subsection 1.2.2) extracted from the input model. The work is a preliminary of the algorithm proposed by the same authors in the reference paper [17] and the following results are strongly exploited in it.

The symmetries will be detected by using *cycles* defined in Chap. 1, Subsec.1.2.2. The symmetry transformations detected by the proposed method are limited to *rotation* and *rotation-reflection*, as both these transformations are detected through *finite cycles* use, i.e. if we apply the symmetry operation a sufficient number of times, the points go back to their original permutation. Rotational symmetry cycles will be represented as vertices of an approximately regular polygon (the description is provided for a 2D point set), and rotation-reflection symmetry (reflection in a plane followed by rotation about an axis perpendicular to that plane, so for a 3D point set) will be seen as vertices of an anti-prism (an n-sided anti-prism is a polyhedron composed of two parallel copies of some particular n-sided polygon, connected by an alternating band of triangles, see Fig. 2.2).

The basic idea of the algorithm can be clarified by this simple description of an example in case of exact symmetry in $\mathbb{R}^2$. Let $P_1, P_2, P_3 \in \mathbb{R}^2$ be the vertices of an isosceles triangle with $\|P_1 - P_2\| = \|P_2 - P_3\|$ and $\|P_1 - P_3\| \geq \|P_1 - P_2\|$ (as in Figure 2.1(a)). These three points partially define a rotation (P_1 moves to P_2 and P_2 moves to P_3) which could potentially represent the symmetry of a regular polygon. The rotation is correctly defined if the image of P_3 is such that

$$\|P_4 - P_3\| = \|P_2 - P_1\|, \quad \|P_4 - P_2\| = \|P_3 - P_1\|. \tag{2.6}$$

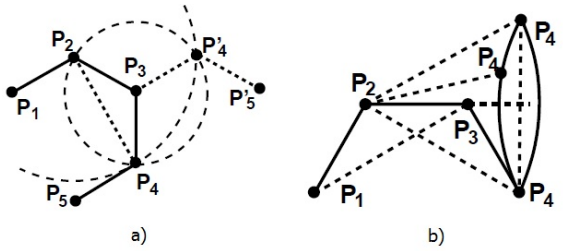


Figure 2.1: (a) Expansion from three points in 2D; (b) expansion from three points in 3D (source [17]).

There are two possible locations for P_4, but only one of these lies on the same side of the line P_2P_3 as P_1 (the one denoted as P_4 in Figure 2.1(a)) and it is the right one to add to the polygon expansion. In the same way, another point P_5 can be chosen as image of P_4, by replacing (P_1, P_2, P_3, P_4) by (P_2, P_3, P_4, P_5) in (2.6). Then, applying this expansion process iteratively, the method could eventually lead to a polygon cycle.

In the approximate case, which is the situation corresponding to the real applications, the difficulties arise because of the possibility of accumulation of errors during the process of expansion and the need of choosing a tolerance.

The first problem is solved by adding further constraints to determine the next expansion point. With reference to the previous example of exact symmetry, in order to determine the expansion point P_5 from the current seed set (P_1, P_2, P_3, P_4), instead of requiring at some tolerance ϵ that $\|P_5-P_4\| =_\epsilon \|P_4-P_3\|$ and $\|P_5-P_3\| =_\epsilon \|P_4-P_2\|$, stronger approximate equality conditions are required involving *all* the distances within the same distance group (coherence with Definition 1.2.9), i.e.

$$\begin{aligned} \|P_5 - P_4\| &=_\epsilon \|P_{k+1} - P_k\| \quad k = 1, 2, 3 \\ \|P_5 - P_3\| &=_\epsilon \|P_{k+2} - P_k\| \quad k = 1, 2 \\ \|P_5 - P_2\| &=_\epsilon \|P_4 - P_1\| \end{aligned} \tag{2.7}$$

Then the reasoning can be easily applied to successive expansion points. All can be

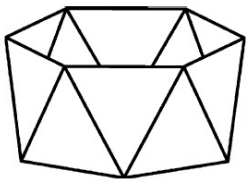


Figure 2.2: The lateral surface of an anti-prism (source [19]).

summarized in the following way: let $\mathcal{C} \subset \mathbb{E}^2$ be a cycle of $c = |\mathcal{C}|$ points at tolerance ϵ. The cycle $\mathcal{C} = \{P_k : 1 \leq k \leq c\}$ can be seen as generated by a permutation that maps P_l to P_k, satisfying for each $1 \leq r \leq c-1$

$$\|P_l - P_{(l+r)\text{mod}c}\| =_\epsilon \|P_k - P_{(k+r)\text{mod}c}\| \quad \text{for } 1 \leq l, k \leq c. \tag{2.8}$$

All the distances between point pairs (P_l, P_k), with $l - k = r$ or $l - k = c - r$, are approximately the same at tolerance ϵ and, since $=_\epsilon$ is an equivalence relation on the set $\mathcal{D}(\mathcal{C})$ (see Definition 1.2.9 in previous chapter), each group of distances between point pairs with index differences r or $c - r$ corresponds to one equivalence class.

As for the tolerance computation, the proposed algorithm finds suitable tolerances during the process instead of using a predetermined value. The basic idea is the following (see Definition 1.2.11 of the previous chapter for notations): if a point P is a valid expansion point during the process for a current set $\mathcal{S}$, the values $E_{min}(\mathcal{S} \cup \{P\})$, $E_{max}(\mathcal{S} \cup \{P\})$ are computed using inter-point distances (these values depend on the difference between the maximum and the minimum distance in each distance class, more details are reported in [19]) and the following relationship must be satisfied

$$E_{min}(\mathcal{S} \cup \{P\}) < E_{max}(\mathcal{S} \cup \{P\}). \tag{2.9}$$

In 3D, the two symmetry transformations considered in the reference paper are rotation and rotation-reflection. The rotation case is analogous to the description of polygon creation given for the 2D case. The rotation-reflection expansion process aims

to the construction of a shape that can be viewed as an *anti-prism* (see Figure 2.2). Supposing to be in the exact case as already done in 2D and supposing to start from a seed set of points $\{P_1, P_2, P_3\}$ (see Figure 2.1(b)), if P_4 lies on the plane P_1, P_2, P_3 the only valid and possible solution the algorithm could lead to is a polygon (rotation), otherwise the described method could only lead to an anti-prism arrangement of points (rotation-reflection) if it exists. A more detailed explanation can be found in [19]. As in 2D, the proposed algorithm computes at every step a suitable tolerance to determine the best unambiguous cycle as output.

In conclusion, with a time complexity of $O(Cn^4)$, where n is the number of points in the input point set and C is the current maximal symmetry order, the presented algorithm can help to detect symmetries in a BRep model by using characteristic points to find unambiguous cycles among them, as will be immediately explained here below.

The principles of a specific stage of the approach proposed in Chapter 3, regarding the path searching of points, are very close to the just described method, anyway we tried to avoid to verify the entire set of equations 2.6 whenever a new candidate point exists. Furthermore, we will find a way to foresee the existence of significant cycles by verifying the existence of a dominant point distances.

The proposed symmetry detection approach for BRep

This work presented in [17] aims mainly to detect *design intent* in boundary representation models, embodied as intentional congruencies, symmetries and symmetric arrangements of the sub-parts deriving from the decomposition of the model.

One of the main characteristics of this approach is the construction of a *regularity feature tree* (*RFT*): it is a structure representing a hierarchical decomposition of the BRep model in simple, closed volumes which in combination describe the original shape and that are selected because more symmetric than other parts, called *regularity features* or *leaf-parts*. More details about RFT construction can be found in [18]. More

than one RFT may exist, each with a different validity interval. Let T be a RFT, the user is asked to make a simple choice of a suitable tolerance interval E_T that is used to establish a specific RFT. The results do not depend on this arbitrary choice, as most tolerances are inferred automatically from the distances present in the model.

This method makes strong use of *characteristic points* (already introduced in Chap. 1, Subsec. 1.2.2). Regularities of the solid are detected using mappings between characteristic point sets.

The second step of the algorithm consists on the detection of the *congruencies* between the sub-parts to partition the regularity features in congruence sets. Given a set $\mathcal{L}$ of leaf-parts and the validity interval E_T from the RFT construction, a hierarchical grouping is applied to the elements of $\mathcal{L}$ and each congruence set contains:

- a set of leaf-parts C_k;
- the point mappings Γ_{C_k}, giving all the pairwise congruency matching between the leaf-parts;
- the computed validity interval E_{C_k} for each set of leaf parts.

Essentially, a *clustering algorithm* (see [17] for details) is applied in this phase to cluster leaf-parts into respective congruence set.

Once this grouping is done, *incomplete symmetries* are searched in every congruence set. First a symmetry is found in an exemplar leaf-part, then this symmetry can be transferred to all the other leaf-parts of the congruence set using the individuated congruence mappings. This phase consists on searching Type I symmetric arrangement of faces of the leaf-part (see Definition 1.3.1). This goal is achieved by subdividing faces in congruence sets (as done with leaf-parts before), by finding incomplete cycles of the centroids of the faces in the leaf-part and by using them to find Type I symmetric arrangement of faces. Finally, these are merged into compatible symmetry transformations.

Incomplete cycles (defined in Chap. 1, Subsec. 1.2.2) in point sets detection are used both in incomplete symmetry and symmetric arrangement detection. The incomplete cycles construction concept has been described in the approach proposed for discrete point sets, even if in this work also cycles are considered. Different type of incomplete cycle could lead to different type of symmetry. Notice that the special case of translational symmetry arises when the first three points composing the initial isosceles triangle are collinear. More precisely, the algorithm considers *approximate* incomplete cycle (see [17] for details) and provides an accurately calculated validity interval.

Symmetric arrangements of the elements of a congruence set are detected by first detecting incomplete cycles formed by the *centroids* of the leaf parts and then selecting those which also match the leaf-parts. The leaf-parts matching is verified by characteristics points parts usage: in case of Type I symmetric arrangement, cycle of leaf-parts centroids must be compatible with location cycles for corresponding characteristics points, one from each leaf-parts, while cycle must be linked by a translation for Type II symmetric arrangement (see Definition 1.3.1). Congruency mappings clarify which characteristic point of a leaf-parts corresponds to a characteristic point of another one. Cycles of centroids induce the search of a precise type of cycle for characteristic points and to achieve it *global symmetries* are necessary, because by combining these symmetries with the congruencies all possible ways of matching the leaf-parts are revealed. More precisely, if P_l^1 is a characteristic point of the leaf-part L_1, we must put it in correspondence with all points of the leaf-part L_k resulting from $\gamma_k(g(P_l^1))$, where $\gamma_k : L_1 \rightarrow L_k$ is the congruence mapping between L_1 and L_k and g a global symmetry in the set of global symmetries detected for the congruence set. These global symmetries are found by filtering the set of all detected incomplete symmetries. The detected cycles between characteristic points are possibly merged to see if they can be represented by a single isometry.

Incomplete cycles merging is used both in incomplete symmetries detection and

symmetric arrangements detection. The essential steps of this stage are:

1. to detect the possible types of symmetry corresponding to the cycles;

2. to cluster the cycles of the same symmetry type to find the final arrangement.

In conclusion, the approach proposed in [17] provides a good and robust algorithm to detect regularities in BRep models, as confirmed by the tests performed by the authors. The main lacks of this method could be the strong unambiguity condition used in cycle building, which could be very sensitive to the presence of noisy points that may disrupt a generally well-defined regularity, and the strict dependence of the regularities detected on the RFT decomposition, which could influence the entire detection process.

2.3.2 The Pauly, Mitra, Wallner, Pottmann, Guibas' method

Pauly, Mitra, Wallner, Pottmann and Guibas [24] introduced in 2008 an algorithm to describe and classify possible regular structures in terms of repeated geometric patterns in 3D shapes coming from point or mesh based models.

In the presented work they observed that repetitive patterns are generated by combining and applying a small number of generative transformations to geometry building blocks. The main phases of the method are:

1. model decomposition into small local surface patches, estimation and analysis of similarity transformation between them;

2. estimation of parameters of the generative model that gives rise to these patterns;

3. spatially adjacent patches aggregation to build larger repetitive elements.

So, formally, given a surface $\mathcal{S}$ defining a shape, the aim of the algorithm in [24] is to find a generative model $(P_0, \{T_i\}, \{n_i\})$ such that the union of the elements in

$\mathcal{P} = \{P_0, \ldots, P_{n-1}$ covers as much of the surface $\mathcal{S}$ as possible and minimizing the number of repetition n at the same time. Actually, the transformation group and $\mathcal{P}$ are unknown and must be estimated simultaneously by the method.

The very first step consists on computing a uniform random sampling of the input model S with average sample spacing h.

A regular structure can be detected by analyzing the similarity transformations that map pairs of these patches onto each other. For this purpose patches are grouped into similarity sets Ω_l using a local shape descriptor that is invariant under similarity transformations. For each sample point the mean and Gaussian curvatures H and K respectively are estimated, then samples are grouped according to the value H^2/K which is invariant under uniform scaling, rotation, and translation. If no scaling is involved, we can further split the similarity sets based on the values of (H, K). The descriptor is presented in details in [8]. They noticed that large collections of similar patches are more likely to contain a regular structure of significant size.

Let Ω be a similarity set selected as just described. In this phase the goal is to understand whether elements of Ω form a regular structure and, if so, estimate the parameters of the underlying transformation group deriving them from the collection $\mathcal{T} = \{T_{ij} | \mathbf{p}_i, \mathbf{p}_j \in \Omega\}$, where T_{ij} is the pairwise transformation mapping sample $\mathbf{p}_i$ to sample $\mathbf{p}_j$. In particular, translation and rotation can be derived by aligning the local frames computed from the surface normal and principal curvature directions. The uniform scaling factor can be calculated from the ratio of corresponding mean curvatures H_i/H_j . *Geometric registration* (the description of the procedure can be found in [26]) is used to improve the accuracy of the estimated similarity transformation and to discard the incorrect matches.

To compute the group generators from the set $\mathcal{T}$ the authors of the paper used the fact that spacial coherence of regular structures leads to strong accumulative patterns in the corresponding transformation space. In case of *translational transformations*,

without scaling and/or rotation, the more noticeable properties of the resulting pattern are arrangements of the transformations of the underlying transformation group on a 2D plane through the origin in the space of transformations and an undeniable accumulation in clusters that form a uniformly spaced grid. In case of patterns involving *scaling* and/or *rotation involving* instead, transformations usually cluster at points on a curved manifold, which is more difficult to detect. For this reason, they introduce a suitable mapping function that allows a similar uniform *2D lattice structure* for non-translational structures. They underline the importance of considering variables of the transformation that are invariant under a change of origin, since the translation component depends on the center of scaling and the axis of rotation. Therefore, the properties of the transformation considered are quantities such as the scaling factor s, the rotation angle θ, and the direction of the axis of rotation that can be extracted from the estimated transformation matrix $\mathbf{H}_{ij}$ of a sample pair $(\mathbf{p}_i, \mathbf{p}_j)$.

The approach to establish which class of commutative 2-parameters structure (see also Chap. 1, Section 1.1) the case falls in is as follows: if a significant fraction of the transformations have zero rotation angle, the model can only contain regular structures of type "$Translation \times Translation$"; else, if there are a significant number of rotations, the method separates all transformations into sets with similar direction of rotation axis (since group transformations of a rotational structure need to have coincident axis directions) and then, if there is a significant variation in scaling factors the model potentially contains regular structures of type "$Rotation \times Scale$", otherwise, if there is a translations parallel to the axis of rotation (which is the only possible to stay in a commutative 2-parameter group), the model contains regular structures of type "$Rotation \times Translation$".

The parameters used in each class of structure are in a form such that each of the three mappings has the crucial property that composition of similarities corresponds to the sum of the 2-dimensional vectors of the parameters in transformation space.

In the *model estimation* step authors try to obtain an estimate of the parameters of the generative model of a regular structure. Here, the two main problems are that there could be spurious clusters that are not part of the regular structure and that clusters that instead should be present are missing or too weak. For these reasons, the authors proposed a *grid fitting* approach able to face also outliers and holes. This method operates on a set of cluster centers $C = \{\mathbf{c}_k\}$ that they extract from the set of transformations mapped to a 2D space (thanks to the regular 2D lattice structure). They use *mean-shift clustering* for this purpose (see [10] for references). Since the identity transformation is always part of a transformation group, the grid must pass through the origin. Let $X = \{\mathbf{x}_{ij}\}$ be the $n_1 \times n_2$ regular grid to fit, the grid location can be represented with two vectors $\mathbf{g}_1, \mathbf{g}_2 \in \mathbb{R}^2$ such that $\mathbf{x}_{ij} = i\mathbf{g}_1 + j\mathbf{g}_2$, $-n_1 < i < n_1$ and $-n_2 < j < n_2$. Then, to find the unknown grid generators $\mathbf{g}_1, \mathbf{g}_2$ they apply an optimization that combines different energy terms (see [24] for more details).

The final step consists of extracting large-scale repetitive elements and optimize the generating transformations. The proposed *aggregation* method alternates between *local region growing* and coupled *registration*. The result is aggregation of spatially adjacent patches of regular structures with compatible group structure, identified more precisely with the aid of the evaluation of the alignment error of the registration.

The just described detection algorithm takes in input a point cloud or a surface mesh, and, after the specification of the sample spacing h, can provide the set of detected regular structure, without any additional user assistance. The main limitation is that currently this method cannot find generative model in case of "warped"sequential repetitive patterns of patches $P_1, \ldots, P_n$, where the transformation $T_{i,i+1}$ mapping P_i to P_{i+1} is changing with the index i.

2.3.3 The Jiang, Chen, He's method

The approach proposed by Jiang, Chen, He [15] aims to detect global exact rotational and reflectional symmetries in feature-based CAD models, differing for the fact that the basic unit of symmetry detection is the *feature*, while usually it is the face. Input considered in this method consists of CAD models created by Boolean combinations of design features. The set of features supported in this work are: Pad, Shaft, Rib, Stiffener, Pocket, Slot, Shell, Groove, Hole, Fillet, Chamfer, Draft, Mirror and Pattern (Circular Pattern, Rectangular Pattern, User-defined Pattern).

The main idea of the approach is to detect as many symmetries in the features of a model M as possible by using only *feature information* (such as feature types, parameters, sketches, references and building histories).

Not all the symmetries in a feature set can be detected by using only feature information. Therefore, in this set of well identified situations (e.g. feature intersection) the geometric shapes of the features, which can be described by its real faces, are used to detect the symmetries of the feature sets. The *real faces* of a feature f are the faces belonging to feature f that, once the feature has been added to the model, they are effectively a face of the model.

Here are reported some definitions necessary for the comprehension:

- Let $\{M_1, M_2, \ldots, M_n\}$ be a finite set of geometric objects in 3D. If there exists a rotational or reflectional transformation T which satisfies the following two conditions:

  1. T is a permutation for the set $\{M_1, M_2, \ldots, M_n\}$;
  2. $T(M_i) \neq M_i$ for $i = 1, \ldots, n$, i.e. T is not a symmetry of any object in $\{M_1, M_2, \ldots, M_n\}$,

  then T is said to be an *inter-symmetry* of the object set $\{M_1, M_2, \ldots, M_n\}$.

The set of all the inter-symmetries of $\{M_1, M_2, \ldots, M_n\}$ is denoted as $\mathcal{S}_{inter}(\{M_1, M_2, \ldots, M_n\})$. An inter-symmetry can exist if and only if M_i and M_j are congruent for every $i, j \in \{M_1, M_2, \ldots, M_n\}$ (see Definition 1.2.3).

- Let M be an object and $\mathcal{S}(M) = \{s_1\}$. If M is symmetric under s_2, then we say s_2 is *no stronger* than s_1, and s_1 is *no weaker* than s_2, which is denoted as $s_2 \leq s_1$ or $s_1 \geq s_2$.

- Given two symmetries s_1, s_2, the strongest symmetry s among the ones that are no stronger than s_1 and s_2, is called the *merging* of s_1 and s_2, denoted as $s = s_1 \cup s_2$. It is the strongest symmetry among the ones under which both M_1 and M_2 are symmetric, so $s_1 \cup s_2 \in \mathcal{S}(M_1 \cup M_2)$.

- Let M_1, M_2 be two objects. If for any $s_i \in \mathcal{S}(M_2)$, M_1 is symmetric under s_i, then we say $\mathcal{S}(M_2)$ is *no stronger* than $\mathcal{S}(M_1)$ and $\mathcal{S}(M_1)$ is *no weaker* than $\mathcal{S}(M_2)$, which is denoted as $\mathcal{S}(M_2) \leq \mathcal{S}(M_1)$ or $\mathcal{S}(M_1) \geq \mathcal{S}(M_2)$.

Suppose that M_1 and M_2 are two volumetric objects in 3D, that they are built by the unions of features, and that for any feature f_i that is used to build M_1 and any f_j that is used to build M_2, f_i and f_j are incongruent. The aim is computing $\mathcal{S}(M_1 \cup M_2)$. The authors observed that $\mathcal{S}(M_1 \cup M_2)$, under the cited hypothesis, depends only on $\mathcal{S}(M_1)$ and $\mathcal{S}(M_2)$ and not on the inter-symmetries between the two objects.

If $\mathcal{S}(M_1) = \emptyset$ or $\mathcal{S}(M_2) = \emptyset$, then $\mathcal{S}(M_1 \cup M_2) = \emptyset$.

If $\mathcal{S}(M_1) \neq \emptyset$ and $\mathcal{S}(M_1)$ is known, suppose that $\mathcal{S}(M_1) = \{s_{11}, s_{12}, \ldots, s_{1n}\}$, $\mathcal{S}(M_2) = \{s_{21}, s_{22}, \ldots, s_{2m}\}$ and for any $s_{1i}, s_{1j} \in \mathcal{S}(M_1)$, $1 \leq i, j \leq n$, $i \neq j$, let $s_{1i} \cup s_{1j}$ be null, the same applies to $\mathcal{S}(M_2)$. Then,

$$\mathcal{S}(M_1 \cup M_2) = \{s_{1i} \cup s_{2j} : s_{1i} \in \mathcal{S}(M_1), s_{2j} \in \mathcal{S}(M_2), i = 1, 2, \ldots, n, j = 1, 2, \ldots, m\}. \tag{2.10}$$

The *result of merging of* $\mathcal{S}(M_1)$ and $\mathcal{S}(M_2)$ is denoted as $\mathcal{S}(M_1) \cup^{\mathcal{S}} \mathcal{S}(M_2)$.

Various cases are now considered.

- $\mathcal{S}(M_2)$ is known. Let A be a null symmetry set, select the first symmetry s_{11} in $\mathcal{S}(M_1)$, compute the merging of s_{11} and the symmetries in $\mathcal{S}(M_2)$ one by one. If s_{11} can be successfully merged by s_{2j} in $\mathcal{S}(M_2)$, then we know there exist no other symmetries in $\mathcal{S}(M_2)$ that can successfully merge s_{11}, thus add $s_{11} \cup s_{2j}$ to A. Then, handle the next symmetry s_{12} in $\mathcal{S}(M_1)$ as we do with s_{11}, and go in this way until the last symmetry in $\mathcal{S}(M_1)$. Finally, $\mathcal{S}(M_1 \cup M_2) = A$.

- $\mathcal{S}(M_2)$ is unknown. From expression (2.10) follows that $\mathcal{S}(M_1) \geq \mathcal{S}(M_1 \cup M_2)$ and $\mathcal{S}(M_2) \geq \mathcal{S}(M_1 \cup M_2)$. So $\mathcal{S}(M_1 \cup M_2)$ can be obtained by verifying whether $M_1 \cup M_2$ satisfies the symmetries which are no stronger than those of $\mathcal{S}(M_1)$.

- $\mathcal{S}(M_2)$ is partly known. The part of $\mathcal{S}(M_2)$ which is known is denoted as $\mathcal{PS}(M_2)$. The approach to obtain $\mathcal{S}(M_1 \cup M_2)$ involves both the above cases.

The just described procedure is applied after a construction of the parent–child relationships of the topological elements in the model and a feature classification in congruent feature sets. First, single *feature symmetries* have to be detected by using only feature information (for example using 2D sketches, by the approach described in [5]). Then *symmetries of feature sets* are detected. Given a feature set **FS** of u elements,the approach of deriving the set of detected symmetries of **FS** by using only feature information, denoted by $\mathcal{S}_d(\mathbf{FS})$, is:

- if $u = 1$, then $\mathcal{S}_d(\mathbf{FS})$ corresponds to the symmetries of the single feature;

- if $u \geq 2$, then the feature in **FS** that belongs to the same Mirror or Pattern feature are grouped together as a single feature. Once it is done, for each group of feature USD (unit for symmetry detection) $\mathcal{S}_d(USD)$ is derived from feature information. Then, if $\mathbf{FS} = \{USD_1, USD_2, \ldots, USD_q\}$, $q \leq u$, the

approach described in "$\mathcal{S}(M_2)$ is known" is applied $q-1$ times to compute $\mathcal{S}_d(USD_i) \cup^{\mathcal{S}} \mathcal{S}_d(USD_{i+1})$ and the computation result is $\mathcal{S}_d(\mathbf{FS})$ ($\subset \mathcal{S}(\mathbf{FS})$).

$\mathcal{S}(\mathbf{FS})$ is tagged as *undetermined* if $\mathcal{S}_d(\mathbf{FS}) \neq \emptyset$, as *partly determined* if part of $\mathcal{S}(\mathbf{FS})$ has been detected, as *determined* if $\mathcal{S}(\mathbf{FS})$ has been totally detected by using only feature information.

Finally the feature sets are sorted into an ordered sequence $(\mathbf{FS}_1, \mathbf{FS}_2, \ldots \mathbf{FS}_m)$ by using criteria to optimize the number of operations (see [15] for more details), the symmetries are merged in each feature sets by using the usual merging process.

The advantage of the choice of considering the feature as the basic unit of symmetry detection is the independence of the efficiency from the number of faces in the input model, and this is a strong point of this method respect to the face-based existing methods. Meanwhile, this approach could detect a limited number of symmetries if the input models contain less feature information than they could because of the followed design approach.

2.3.4 The Li, Foucault, Léon, Trlin's method

The symmetry detection algorithm proposed by Li, Foucault, Léon, Trlin in 2014 [16] aims to identify rotational and planar reflective symmetry considering as input BRep CAD models bounded by a 2-manifold, with the objective of restructuring the object *feature tree* and the relative information. Analyzing the local as well as the global symmetry properties of a solid is useful to reorganize its feature tree so that it becomes more intuitive. The result is a compressed model that replicates the input model and that should maintain the accuracy of the manufacturing process, at least. This fact is formalized in the following way: let M, M' and T be respectively the input object, the compressed one and the geometric transformation expressing the symmetry. Then, the replication error of M with M', conserving the same shape of M, and T can be

measured by the Hausdorff distance:

$$\epsilon = d(M, T(M')) = \max\Big(\sup_{p \in M} (\inf_{p' \in M'} \|p' - p\|), \sup_{p' \in M'} (\inf_{p \in M} \|p' - p\|) \Big) \qquad (2.11)$$

The *condition of validity* of M' so that M and M' can be regarded as identical in the CAD modeler, so that M' cannot be distinguished from the object manufactured from M, is

$$\epsilon \leq \delta,$$

where δ denotes the CAD modeler tolerance expressing the maximal distance between boundaries of adjacent patches. The validity condition can be expressed as a need to obtain symmetry properties as accurate as the modeling kernel of a CAD system since the incorporation of symmetry properties in the feature tree can be considered significant.

The class of faces considered by this approach is composed by planar, cylindrical, conical, spherical, toroidal faces.

In the *preparation phase*, this method takes the boundary of a solid M, denoted as ∂M, and generates a new paving ∂M_{max} referred to a new solid M_{max} such that it contains only maximal faces and edges (see Chap. 1 Section 1.6 for maximal faces and edges definition). It is mandatory that the new paving does not influence the symmetry properties of M, by its decomposition into faces, edges and vertices. Furthermore, it must not change the shape of M, i.e. only modifications to adjacencies between faces, edges and vertices are performed. The paving must meet the validity condition. Another measure is applied to loop adjacent to other loops through a vertex: it is called the *vertex split operation*, which aims to characterize the independence of surfaces around a vertex so that local symmetry properties can be better represented once loop connection are removed by splitting their common vertex. See Figure 2.3: the vertex V (signed in red in Figure 2.3a) is split in three vertices, so that new maximal faces and edges can be created (in Figure 2.3b) there is a zoom of a changes in the V neighbourhood). It is not

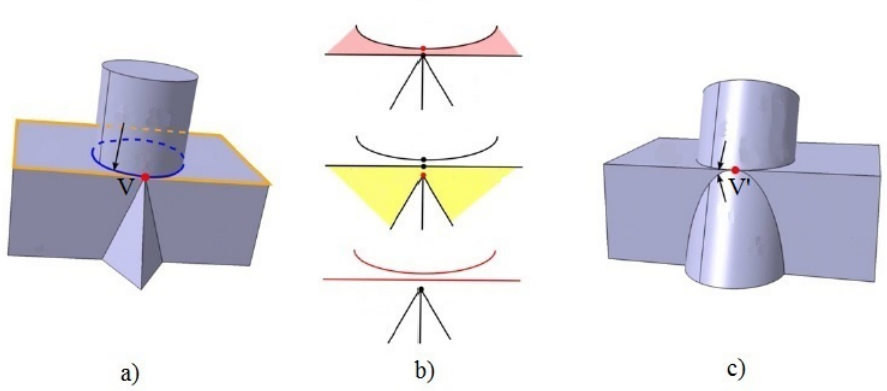


Figure 2.3: a) V is a *regular* vertex and the vertex split operation can be applied to it; c) a zoom of the vertex split operation; c) there are faces adjacent to V' sharing the lying surface that crosses with other faces adjacent to V' sharing another lying surface: in this case the vertex split operation cannot be applied (source [16]).

always possible to apply the vertex split operation to a vertex: the vertex V' in Figure 2.3c is an example of these particular case.

The symmetry analysis process for the identification of Global Symmetry Planes (GSPs) and Global Symmetry Axis (GSAs) is a divide and conquer approach.

The *divide phase* consists of the generation of Candidate Symmetry Planes (CSPs) and Candidate Symmetry Axis (CSAs). This step is carried out by analyzing a set of basic configurations in ∂M_{max} to assign CSPs and CSAs valid only locally for set of one, two or three adjacent faces. Fix the attention on CSPs. The assignment starts from the supposition that when a GSP Π intersects M_{max}, the resulting curve C_c is a closed curve, for the hypothesis of input. The possible configurations of the curve C_c are here summarized:

- C_c contains points P that all lie inside a face of ∂M_{max};
- C_c contains points P that all lie on an edge of ∂M_{max};

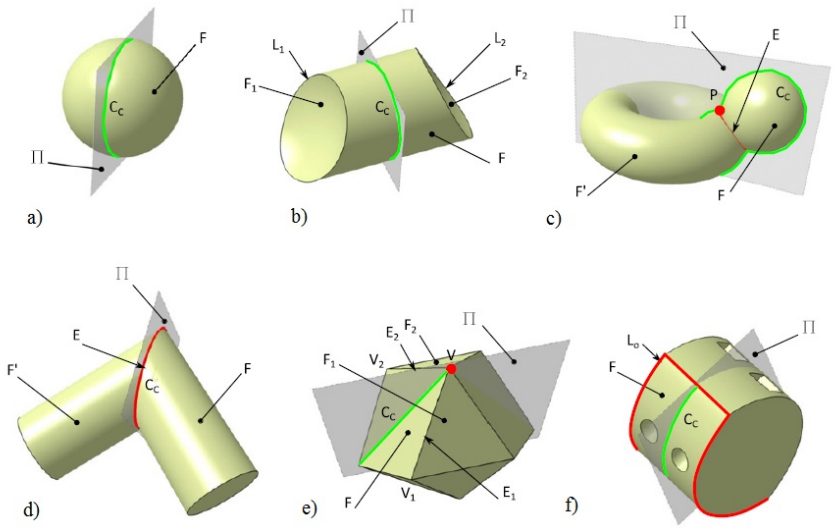


Figure 2.4: Taxonomy of configurations of symmetry planes Π intersecting M_{max} to define C_c (source [16]).

- C_c contains a point P that lies exactly on a vertex of an edge $E \in \partial M_{max}$;
- C_c contains a point P that lies on an edge of ∂M_{max} and it coincides with one of its vertices;
- C_c contains a point P that lies on an edge of ∂M_{max} and it does not coincide with one of its vertices.

Furthermore, given the set of primitives surfaces describing ∂M_{max}, a valid CSP Π is such that the corresponding C_c is any straight line on a plane, any generatrix or circle on a cylinder, any generatrix on a cone, any circle containing the center for a sphere, any small circle or the inner or outer circles on a torus.

The elementary configurations that can define CSPs or CSAs and form a subset of C_c as well as any couple of loops aiming to the definition of a GSP or a GSA are here listed. They are classified by observing the neighbourhood of a point $P \in C_c$.

1. “Surface Symmetry CSP”: it divides a face (the reference examples are Figures 2.4a, 2.4b).
2. “Orthogonal CSP”: it divides an edge and this configuration also produce CSAs (the reference example is Figures 2.4c).
3. “Bisector CSP”: it divides two faces (the reference example is Figures 2.4d).
4. “Loop Bisector CSP”: it divides two edges (the reference example is Figures 2.4e).
5. “Loop Symmetry CSP”: it divides a face with multiple loops (the reference example is Figures 2.4f).

Then, the *conquer phase* is subdivided in two propagation process.

- *First level propagation process*. Starting from an arbitrary CSP (or CSA) Π obtained from the previous step, the algorithm propagates to neighboring entities of ∂M_{max} when one of their associated CSPs coincides with the current Π, thus expanding in the area of ∂M_{max} where Π is valid. So a CSP chain is created. The propagation stops when either a CSP chain forms a loop (i.e. when the set of adjacent faces attached to Π forms a cycle over ∂M_{max} or the coincidence criterion fails (i.e. asymmetric configuration).
- *Second level propagation process*. Starting from a CSP chain, the algorithm evolves from face to face, on both sides of this CSP chain, through adjacency relationships described in the paving of ∂M_{max}. Its purpose is to check that two pairs of faces, each one located on each side of a CSP chain, are symmetrically set with respect to Π, the CSP originating the chain. This process stops either when all the faces of M_{max} have been processed or when asymmetric configurations have been encountered everywhere along the domain boundary expanded from a CSP chain.

At the end of this phase, if all the faces of M_{max} have been covered after the second propagation process, the CSP become officially a GSP. Otherwise, the symmetry is limited to the area of M_{max} covered by the propagation process (partial symmetry).

CSA can be easily obtained following a process analogue to CSP one.

The just described method is independent of the parameterization of the faces and edges of the component boundary, but this work refers to a limited set of predefined types of faces. Furthermore, the objectives of this work are different from ours, as in the Chapter 3 we aim to find regularities in terms of symmetric arrangements of congruent sub-parts while the goal of this approach is to find reflectional and rotational symmetries of the entire input model.

Chapter 3

The proposed approach

In this chapter a method to detect regular arrangements of congruent sub-parts in a given BRep model is proposed and described.

A symmetric regularity is an immediate perception for the human visual sense. The common use of regular arrangements of faces in geometric modeling constituted by the repetition of an exemplar set of faces, made, as time passes, automatic detection of these structures an issue of increasing importance.

This method wants to focus on the symmetric arrangements retrieval of repeated entities in a BRep model, for this reason the starting point of the proposed approach is the strong hypothesis that the analyzed input sets of faces are granted to correspond to sub-parts in the model certainly congruent with each other. So, the goal of this work is the search of a potential regular configurations of given sets of faces, without any further verification of their nature.

The main lying idea is founded on the awareness that *if a set of congruent sub-parts is characterized by a regular arrangement, then of course also the respective centroids do*. This fact could be taken as a slogan for the proposed method, as it constitutes the fundamental basis for the successive regularity detection, and could mathematically be

considered as a necessary condition for the symmetric arrangements existence. The same starting technique was used by Li, Langbein and Martin in [17] for their proposed algorithm aiming to design intent detection.

Another cornerstone fact of this approach is the repetition of the constant distance between two consecutive sub-parts' centroids involved in a regular configuration of those belonging to the set of symmetric arrangements considered in this work.

These two important considerations will be better examined in depth in the following sections.

3.1 Repeated entities

As just outlined in this Chapter introduction, the input data considered by this algorithm is a set of faces representing the repeated sub-parts which we want to examine to discover any regular configuration. In practice, what must be provided by the user is a set of sets of faces and each set of faces must correspond to one of the repeated entities considered. In Chapter 2 a formal definition of *congruency* of two "objects" has been provided. Here, we suppose that each given repeated entity is congruent to every other in the input set. Formally, this discussion can be summarized in the following definition.

Definition 3.1.1. Let $\mathbf{B} = (\mathbf{F}, \mathbf{E}, \mathbf{V})$ the BRep of a solid object. The user must provide a set $\mathbf{A} = \{\mathbf{A}_0, \ldots, \mathbf{A}_{n-1}\}$, where the element $\mathbf{A}_i$ is itself a BRep and $\mathbf{A}_i = (\mathbf{F}_i, \mathbf{E}_i, \mathbf{V}_i)$ with $\mathbf{F}_i \subset \mathbf{F}$, $\mathbf{E}_i \subset \mathbf{E}$, $\mathbf{V}_i \subset \mathbf{V}$, and $\mathbf{F}_i \cap \mathbf{F}_j = \emptyset$ if $i \neq j$ for $i, j = 0, \ldots, n-1$. Let $i \in \{0, \ldots, n-1\}$, the element $\mathbf{A}_i$ is asked to be congruent to $\mathbf{A}_j$ for $j = 0, \ldots,$ $i-1, i+1, \ldots, n-1$. The set of n elements $\mathbf{A}$ is called the *set of repeated entity*, each element $\mathbf{A}_i$ is a *repeated entity* (sometimes we will shortly write RE).

The hypothesis of congruency of the REs requires the input data to grant it and to be definitely reliable, coherent and well-defined.

Similarly to the method proposed in [17], where *characteristic points* cover a fundamental role in the symmetry detection, here vertices of the model and other particular points are fully exploited during the algorithm execution. Firstly, these points are used to compute a representative point for every RE in the set **A** provided as input (the centroid). Secondly, they constitute a way to compare REs once a subset of REs of the initial input set is supposed to be a candidate symmetric arrangement.

3.1.1 Type of faces

The nature of the faces of a boundary representation has already been discussed in Chapter 1. The faces can be classified depending on the corresponding host surface. The set of types of faces this algorithm admits and threats are listed here together with the specifying parameters.

- *Planar*. The lying planar surface is stored by the coefficients a, b, c, d, obtained by considering the general geometric equation of a plane in $\mathbb{R}^3$, $ax+by+cz+d = 0$. The vector (a, b, c) corresponds to the normal vector and so it is normalized.

- *Cylindrical*. The lying cylindrical surface is stored by a point on the axis cylinder (called *origin of the cylinder*), the axis direction vector of the cylinder, the line corresponding to the axis of the cylinder, the radius of the cylinder (radius of a circumference obtained by cutting the cylinder with a plane perpendicular to the axis line).

- *Conical*. The lying conical surface is stored by a point on the axis cone (called *origin of the cone*), the axis direction vector of the cone, the line corresponding to the axis of the cone.

- *Spherical*. The lying conical surface is stored by the coefficients a, b, c, d, obtained by considering the general geometric equation of a sphere in $\mathbb{R}^3$, $x^2 +$

$y^2 + z^2 + ax + by + cz + d = 0$. Furthermore, the sphere radius and the sphere center are stored.

- *Toroidal*. The lying toroidal surface is stored by the torus center, the line corresponding to the axis of the torus, the "major radius" which is the distance between the center of the torus and the center of the revolved circle, and the "minor radius" which is the radius of the revolved circle.

These are the classic types of faces commonly used for mechanical objects in BRep models and that have therefore been considered in this work. The choice of the attributes of the corresponding surfaces have been chosen in such a way because they are information that can easily be acquired by the Application Programming Interface of SolidWorks© (system used in this work, as it will be explained in detail in the next Chapter), and because these surface information are appropriate for our purpose.

We want to grant the congruency not only of the volume included by the faces corresponding to the REs (the resulting REs shape) but also of the boundary decomposition in cells. For example, complete cylindrical faces are usually built by cutting the face with an edge having its extremes on the two base edge of the cylindrical face, or even by cutting the face with two such edges. This is done for BRep validity reasons. On the other hand, maximal edges and faces are uniquely defined (the meaning of "maximal" has been explained in Chapter 1), so we adopt them in the proposed approach, converting them in maximal in phase of acquisition .

3.1.2 Centroid

Now we introduce a formal definition of *centroid* of a RE (the references of the following definition can be found in [7], where the definition is given for a *material point*).

Definition 3.1.2. Let **P** be a set of n material points (a point in $\mathbb{R}^3$ provided with a mass) such that $\mathbf{P} = \{P_1, \ldots, P_{n-1}\}$. Suppose that P_i has coordinates $(x_{P_i}, y_{P_i}, z_{P_i})$

and m_i is the associated mass, for $i = 0, \ldots, n-1$. Then, the *centroid* C of the set of points in **P** is the point defined by the coordinates (x_C, y_C, z_C) resulting from

$$x_C = \frac{\sum_{i=0}^{n-1} m_i x_{P_i}}{\sum_{i=0}^{n-1} m_i}, \quad y_C = \frac{\sum_{i=0}^{n-1} m_i y_{P_i}}{\sum_{i=0}^{n-1} m_i}, \quad z_C = \frac{\sum_{i=0}^{n-1} m_i z_{P_i}}{\sum_{i=0}^{n-1} m_i}. \tag{3.1}$$

In our case, we will apply the definition of centroid of a set of points to the set of vertices of a RE and we will suppose the vertices to have constant mass for every $i = 0, \ldots, n-1$, in other words $m_i = m$, with m constant, for $i = 0, \ldots, n-1$.

So, let $\mathbf{V}_j$ be a set of n vertices corresponding to the RE $\mathbf{A}_j$ belonging to the input set of REs, $\mathbf{V}_j = \{V_{j,1}, \ldots, V_{j,n-1}\}$. Denote the centroid of the set $\mathbf{V}_j$ as C_j with coordinates $(x_{C_j}, y_{C_j}, z_{C_j})$. For instance, x_{C_j} value is defined by

$$x_{C_j} = \frac{\sum_{i=0}^{n-1} m\, x_{V_{j,i}}}{\sum_{i=0}^{n-1} m} = \frac{m \sum_{i=0}^{n-1} x_{V_{j,i}}}{nm} = \frac{\sum_{i=0}^{n-1} x_{V_{j,i}}}{n}$$

which is the arithmetic mean value of the x coordinates of the vertices in $\mathbf{V}_j$. Analogously y_{C_j} and z_{C_j} can be obtained in the same way. So the centroid C_j of the set $\mathbf{V}_j$ is the point with coordinates defined by

$$\left(\frac{\sum_{i=0}^{n-1} x_{V_{j,i}}}{n}, \frac{\sum_{i=0}^{n-1} y_{V_{j,i}}}{n}, \frac{\sum_{i=0}^{n-1} z_{V_{j,i}}}{n} \right) \tag{3.2}$$

For simplicity, we will refer to the centroid of the set of vertices of a RE as the *centroid of the RE*.

As just sketched in the introduction of this Chapter, the first phase of this approach consists on the analysis of the REs' centroids configuration. This need raises from the intuition that managing 0-cells is much more handy than directly compare the REs' faces, without any preliminary selection of the REs from the initial set. To transpose the problem of finding a regular configuration of sub-parts to a problem of finding a regular configuration of points, it is necessary to ensure these points to be computed in the same way for every RE provided by the initial set. In other words it means that if we could overlap two REs (which are congruent) their corresponding computed

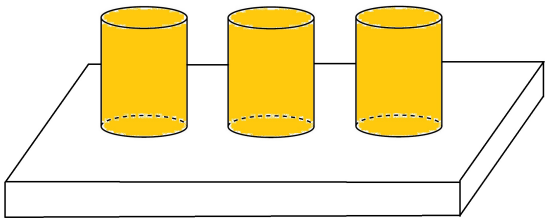


Figure 3.1: An example of possible REs (highlighted in yellow) without vertices.

representative points should overlap too. It becomes natural to take the centroid of the vertices of the RE as this representative point, as it is simple to compute and the RE's vertices are easily accessible.

Now we discuss the elements of the set of vertices $\mathbf{V}_j$ used to compute the centroid C_j of the RE $\mathbf{A}_j$.

Some adjustments must be applied during the vertices acquisition phase, especially in case of curved edges presence. The risk of non homogeneous way of computation of centroids in the various REs raises when there are *closed curved edges* in the REs. See Figure 3.1 and consider one of the highlighted REs: closed edges do not have any vertex lying on them, so in case of exclusive presence of closed edges the set of vertices of the RE would be void and this would lead to the impossibility of computing a centroid. The proposed solution is the insertion of two new vertices on each closed edge of the REs if the lying curve of those edges is a circumference or an ellipse, which are the only possible planar closed curve in the set of types of faces considered for the models in this method.

Also not planar closed edges could occur in a BRep, but we do not worry about them, because the tests demonstrated that considering only the BRep natural vertices and the new vertices put on closed circular or elliptical edges are sufficient to obtain the required results. The proposed method does not treat REs without original vertices and provided with only closed non planar edges, because in this case it is unable to properly

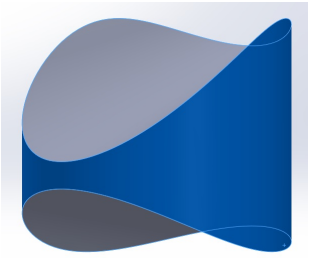


Figure 3.2: A BRep constituted by the cylindrical face (in blue) and two conical base faces. This model is an example not treated by the proposed approach.

add vertices on the edges (in Figure 3.2 there is an example of a not considered case).

The new vertices are assigned in the following way.

Let $\mathbf{A}_j$ be a RE with a closed planar edge. The parametric form of the lying curve of the edge can be easily obtained by the specific functions provided by the development environment, that will be described in details hereinafter. In this way we can consider the image of the application $\gamma : [t_0, t_1] \to \mathbb{R}^3$ to be an arc of regular curve, with $I = [t_0, t_1]$ its parametric interval of definition, $\gamma(t) = (\gamma_1(t), \gamma_2(t), \gamma_3(t))$, $\gamma_i : [t_0, t_1] \to \mathbb{R}$ for $i = 1, 2, 3$. As this must represent a closed curve, $\gamma(t_0) = \gamma(t_1)$. Let $\mathbf{V}'_j$ be the set of vertices provided by the BRep. The two new vertices added to $\mathbf{V}'_j$ have coordinates:

$$\Big(\gamma_1(t_0), \gamma_2(t_0), \gamma_3(t_0)\Big), \quad \Big(\gamma_1\Big(\frac{t_1 - t_0}{2}\Big), \gamma_2\Big(\frac{t_1 - t_0}{2}\Big), \gamma_3\Big(\frac{t_1 - t_0}{2}\Big)\Big).$$

Two new vertices must be added to the set of vertices contributing to the centroid computation for a RE whenever a closed edge occurs.

As for *open curved edges*, both planar and not planar, even if the BRep provides for them two vertices (the starting point and the ending point) and so the set of vertices of that RE is surely not void, we add a new vertex in the exact middle of the arc. This operation is necessary for the geometric comparison of the REs in the final step of this proposed method, as we will see hereinafter.

So if the RE $\mathbf{A}_j$ has a open curved edges, considering its host open arc of regular

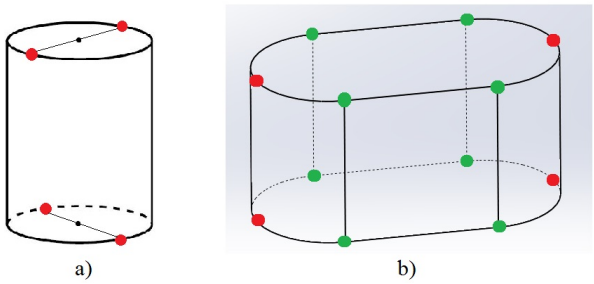


Figure 3.3: a) An example of RE with two closed circular edges; b) the cylindrical faces of the RE have two open circular edges. The vertices highlighted in red have been added for the centroid computation, those in green are original vertices of the BRep.

curve $\gamma : [t_0, t_1] \to \mathbb{R}^3$, $\gamma(t_0) \neq \gamma(t_1)$, the new vertex added to $\mathbf{V}'_j$ has coordinates:

$$\Big(\gamma_1\Big(\frac{t_1 - t_0}{2}\Big), \gamma_2\Big(\frac{t_1 - t_0}{2}\Big), \gamma_3\Big(\frac{t_1 - t_0}{2}\Big)\Big).$$

A new vertex must be added to the set of vertices contributing to the centroid computation for a RE whenever a curved edge occurs.

Figure 3.3 shows two examples of vertex adding on curved planar edges.

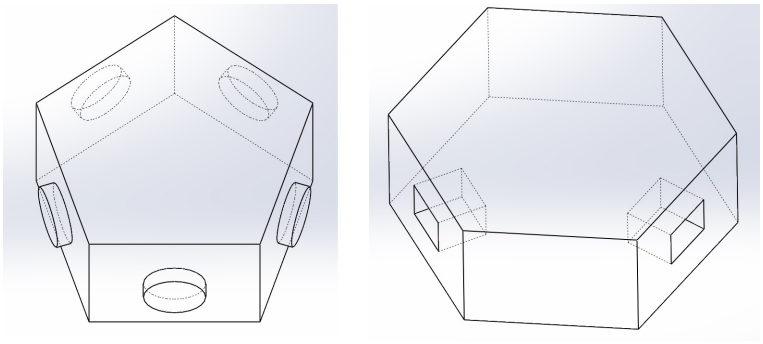
Summarizing, let $\mathbf{V}'_j$ be the set of vertices of $\mathbf{A}_j$ coming directly from the BRep and $\mathbf{V}''_j$ the set of vertices added on curved closed planar edges and on curved open edges of $\mathbf{A}_j$, then the set $\mathbf{V}_j$ containing the vertices used for the centroid computation is:

$$\mathbf{V}_j = \mathbf{V}'_j \cup \mathbf{V}''_j$$

As the REs are congruent with each other, at the end of this analysis $|\mathbf{V}_j| = |\mathbf{V}_k|$ for every $j, k \in \{0, \ldots, n-1\}$.

3.2 Grouping surfaces

If we consider the RE as a whole entity (precisely it is a set of more face entities itself) we could say the RE is adjacent to other faces of the examined BRep. In this method



a) b)

Figure 3.4: a) The five cylindrical faces lying of the lateral faces of the prism constitute a rotational symmetric arrangements, but the regularity is not detected because each RE lies on a different surface; b) The two pockets are related by reflection but they are associated to different GSs.

we decide to group the REs by the lying surfaces of the faces they are adjacent to: if the RE $\mathbf{A}_i$ is adjacent to the face F of the BRep and the host surface of F is S, we associate $\mathbf{A}_i$ to the surface S. In this context, S is called a *grouping surface* (it will be briefly noted by GS). In this way, we will find a set of GSs and each of them is associated to a set of REs. Intersecting the set of REs associated to a GS with the set of REs associated to another GS we could find a not void set, as a RE $\mathbf{A}_i$ could be adjacent to different faces with different host surfaces, associating it to more than one GS. At the end of this procedure of association, we will find a list of GSs $S_0, \ldots, S_{p-1}, p \geq 1$, with respectively associated $LRE_0, \ldots, LRE_{p-1}$ lists of the REs, where LRE_j is the list of REs adjacent to a face with S_j as host surface.

The proposed approach aims to detect symmetric arrangements of REs lying on the same GS. This restriction could prevent to detect symmetric arrangements like those represented in Figure 3.4: the first image represents a rotation, the second one a reflection, but both are not detected because the REs lie on different GS. However, this

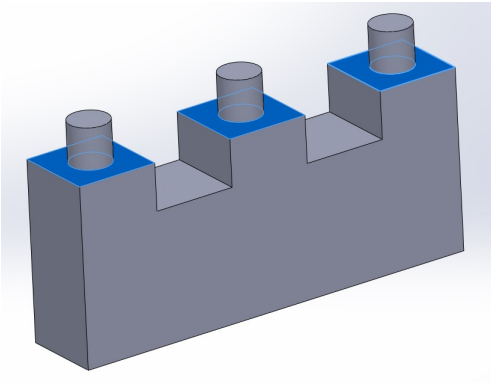


Figure 3.5: A symmetric arrangement of REs (each RE is composed by a cylindrical face and a planar base face): each of them lies of a different face but with same lying surface.

procedure is adopted to do an initial selection of the REs provided as input and to make faster the detection, as the field of searching is reduced if compared to the whole initial set of REs of the most probable arrangements.

Observation 3.2.1. *Notice that a symmetric arrangements like those represented in Figure 3.5 is included in the set of the symmetric arrangements considered by this algorithm.*

Denoting as **S** the list of all GSs deriving from the REs' adjacency analysis, we order **S** by *decreasing ordering respect to the number of REs* in the list associated to the GS. This is done because the most the number of REs on a GS is high and the most there is probability to find numerous and meaningful symmetric arrangements.

3.2.1 Adjacency matrices at constant distance

Let S be a GS, consider the REs in the list associated to it. Observing that a symmetric regularity exists if and only if the distance between two centroids of two consecutive REs involved in the configuration is constant, another selection level can be applied

inside S to the REs . In formulas, the constant distance between two consecutive centroids can be formalized as follows: let $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ be the set of REs constituting a symmetric arrangement, $\{C_0, \ldots, C_{q-1}\}$ the set of the respective centroids, then the Euclidean distance $d(C_i, C_j) = \alpha$, with $\alpha > 0$, $i, j \in \{0, \ldots, q-1\}$ such that $i \neq j$.

The idea is to store, for every value of distance d occurred, information about centroids being distant at that value d in a "*adjacency matrix*".

Definition 3.2.2. Let $P_0, \ldots, P_{n-1}$ a set of n points in $\mathbb{R}^3$ and d a real number, $d > 0$. We call *adjacency matrix at constant distance* d (or simply d*-adjacency matrix*) of the points $P_0, \ldots, P_{n-1}$ the $n \times n$ matrix $\mathbf{M}_d$ such that:

$$\mathbf{M}_d(i,j) = \begin{cases} 1 & \text{if } d(P_i, P_j) = d \\ 0 & \text{if } d(P_i, P_j) \neq d, \end{cases} \tag{3.3}$$

where $\mathbf{M}_d(i,j)$ denotes the entry of $\mathbf{M}_d$ at position (i,j), $i, j \in \{0, \ldots, n-1\}$.

Observation 3.2.3. *Since* $d(P_i, P_j) = d(P_j, P_i)$ *then* $\boldsymbol{M}_d(i,j) = \boldsymbol{M}_d(j,i)$. *In other words,* $\boldsymbol{M}_d$ *is a symmetric matrix.*
Observe that $\boldsymbol{M}_d(i,i) = 0$ *for every* $i \in \{0, \ldots, n-1\}$*, so* $\boldsymbol{M}_d$ *is a square matrix whose diagonal elements are all equal to* 0*.*

A d-adjacency matrix can be viewed as a network of points in $\mathbb{R}^3$ each of them connected to one or more points of the network by a straight arc of length d.

So this step of the algorithm consists on the creation, for every GS found, of a list of adjacency matrices at constant distance of the centroids $C_0, \ldots, C_{n-1}$, one for each d occurrence. For each of these matrices the number of occurrences of the distance d between the involved points is computed: this attribute for each matrix is called *nOccur*. Its value is computed by summing the entries in the inferior triangle or indifferently in the superior triangle of the matrix.

The adjacency matrices construction is mainly made to find "long" patterns of REs, which means that these constructed data will be used to find paths of equidistant centroids with a number of involved centroids as high as possible. Paths of two centroids are banal and not significant. For these reasons the d-adjacency matrices with *nOccur*$= 1$ are deleted from the list of adjacency matrices to examine.

Another important observation, confirmed by the various tests performed in the application context, is that proximity between REs has to be considered as a relevant factor to indicate possible pattern candidates. when REs are close to each other they are more probable to be in a verified pattern. For this reason the remaining elements in the list are reordered by creasing order respect to the repeated distance of d.

3.2.2 Centroids classification

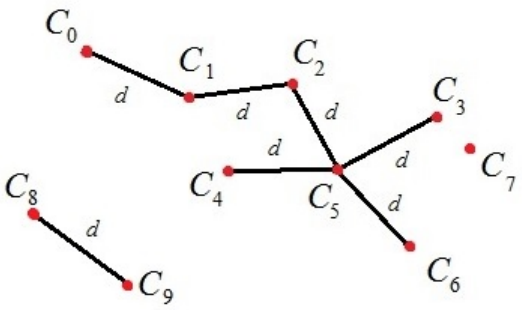
In the next section we will describe the proposed algorithm part dedicated to find all the possible sequences of centroids (what we will call *paths*) to give an outline of all the possible existing symmetric arrangements of REs.

This discussion about the centroids classification is preliminary for the path detection algorithm.

At a fixed d-adjacency matrix, we want to classify a centroid considering the number of centroids indicated at distance d from it in the matrix. We call *branch* of a centroid C_j a centroid C_k, $k \neq j$, such that $\mathbf{M}_d(i, j) = 1$. We will also say that C_j and C_k are *connected at distance* d. To classify C_j it is sufficient to extract information from the matrix, by observing the row corresponding to C_j: summing the 1 entries in that row we can obtain the number of branches of C_j.

Denote the resulting number of branches as s_j, then:

- if $s_j = 1$, C_j is called *extreme point*;
- if $s_j = 2$, C_j is called *simple point*;



a)

	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
C_0	0	**1**	0	0	0	0	0	0	0	0
C_1	**1**	0	**1**	0	0	0	0	0	0	0
C_2	0	**1**	0	0	0	**1**	0	0	0	0
C_3	0	0	0	0	0	**1**	0	0	0	0
C_4	0	0	0	0	0	**1**	0	0	0	0
C_5	0	0	**1**	**1**	**1**	0	**1**	0	0	0
C_6	0	0	0	0	0	**1**	0	0	0	0
C_7	0	0	0	0	0	0	0	0	0	0
C_8	0	0	0	0	0	0	0	0	0	**1**
C_9	0	0	0	0	0	0	0	0	**1**	0

b)

Figure 3.6: a) the graphic representation of the connections at distance d among the centroids $C_0, \ldots, C_9$; b) the corresponding d-adjacency matrix.

- if $s_j > 2$, C_j is called *multibranch point*.

In Figure 3.6 there is a simple example situation: Figure 3.6a) graphically represents the relations among centroids $C_0, \ldots, C_9$, signed in red, and the arcs connecting them represent the constant distance d from a centroid to another one; in Figure 3.6b) there is the associated d-adjacency matrix. In the example $C_0, C_3, C_4, C_6, C_8, C_9$ are extreme points, C_1, C_2 are simple points, C_5 is a multibranch point, while C_7 is not classified in this adjacency matrix because there not exist a centroid C_j, $j \neq 7$, on the current GS such that the Euclidean distance $d(C_7, C_j) = d$.

3.3 Paths of centroids

In this section we will describe the *path detection algorithm* which carries out the process of detecting all the possible sequences of centroids whose REs could possibly represent a symmetric arrangement. These sequences of centroids are made of at least

three centroids satisfying specific geometric conditions.

In this proposed method we decide to focus the attention on symmetric arrangements of REs whose centroids lie all on a *line* or on a *circumference*, so on two planar curves.

Definition 3.3.1. Let $\mathbf{C} = \{C_0, \ldots, C_{n-1}\}$ be a set of centroids of REs. We call *path* of length l, with $l \geq 3$, an ordered sequence of l centroids $(C_{j_0}, \ldots, C_{j_{l-1}})$, with $C_{j_h} \in \mathbf{C}$ for $h = 0, \ldots, l-1$, such that:

- $C_{j_0}, \ldots, C_{j_{l-1}}$ all lie on the same *line* (path of type "*line*")

  or

- $C_{j_0}, \ldots, C_{j_{l-1}}$ all lie on the same *circumference* (path of type "*circumference*")

and such that $d(C_{j_h}, C_{j_{h+1}}) = d$, with $d > 0$ and for $h \in \{0, \ldots, l-2\}$.

The line or the circumference containing all centroids of the path is called *curve associated to the path*.

The following described steps aim to find *all* the existing *maximal* paths of centroids given as input, which is to say that we want to find *all* the paths $(C_{j_0}, \ldots, C_{j_{l-1}})$ such that there cannot exist a path $(C_{k_0}, \ldots, C_{k_{p-1}})$ with $p > l$ such that $\{C_{j_0}, \ldots, C_{j_{l-1}}\} \subset \{C_{k_0}, \ldots, C_{k_{p-1}}\}$.

Conducting the detection within the network built by an adjacency matrix (and so at a fixed distance d) we can avoid to verify during the process the constant distance between the centroids, and so focusing the searching on the detection of those centroids satisfying the required geometric equations. The process will be described can be seen as an "expansion process" in a network of centroids.

3.3.1 Path building

A path is built step by step, by first choosing an initial seed set of three centroids $(seed1, seed2, seed3)$ and, once the type of the path is going to be constructed has been established , by adding every time a new centroid to the current path if it possible. If the three initial points are aligned it will be a path of type line, otherwise it will be a path of type circumference. In both cases, the first time the attempt of expansion is done in the "$seed1$ to $seed2$" direction, when the expansion towards that direction is no more possible is made a second expansion attempt in the "$seed2$ to $seed1$" direction.

Let discuss when a new point can be added to the current path and so when the expansion is possible. Suppose the current path in expansion to be $(C_{j_0}, \ldots, C_{j_{l-1}})$, with $l > 2$, the associated curve to be $\mathcal{C}$, and suppose we are attempting to expand in the "C_{j_0} to C_{j_1}" direction. The expansion in that direction is possible if and only if a branch of $C_{j_{l-1}}$ lying on $\mathcal{C}$ and different from $C_{j_{l-2}}$ there exist. If such a point does not exist the expansion in that direction is no more possible.

The expansion from a seed set ends when the maximum expansion is reached in both the cited directions. Then the new found path is added to a *list of paths* associated to the current adjacency matrix, hanging on the next verification step (described in the next section).

3.3.2 The starting point choice

The seed points determine the nature of the path that is to be built, as they allow to establish the equations of the associated curve $\mathcal{C}$.

The choice of the starting point ($seed1$) is an important issue that must be discussed.

As just stated, this path detection algorithm aims to detect *all* the existing paths of centroids in the given set. If we follow the strategy of considering every centroid as a starting point for the path creation, we can easily see that many unnecessary operations

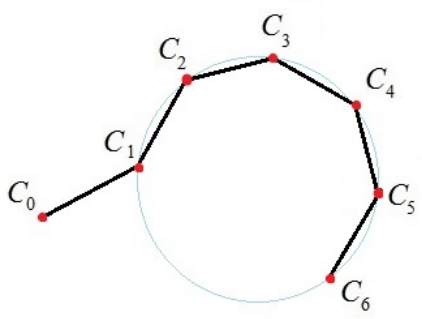


Figure 3.7: A set of centroids at a constant distance: $C_1, C_2, C_3, C_4, C_5, C_6$ lie on the circumference (in blue) and constitute a path, while C_0 does not lie on the circumference.

are done. Considering Figure 3.7 it is clear that assuming as a starting point every centroids leads to find many times the same path. In the example, moving towards the "C_1 to C_2" direction, taking C_1, C_2, C_3, C_4, C_5 as starting point for the expansion process one always obtain the same path of type circumference $(C_1, C_2, C_3, C_4, C_5, C_6)$. So, we must exclude some centroids from being a starting point.

Furthermore, some starting points could lead to many different paths: this is the case of the multibranch points and of the simple points. To find all the existing paths it is necessary to explore every branch of the starting point ($seed1$) and so to consider every branch as second point ($seed2$).

In the same way, independently from the starting point class, a second point could have more than one branch and we must consider every possible third point ($seed3$) not to prevent ourselves from finding every possible path.

Once the seed set of three elements is established, it is necessary to verify that the three centroids are not contained in an already existing path in the list of already found paths. If such a path has already been created the seed set is discarded. Otherwise, the seed set fixes the type of the path and the associated curve, expecting the algorithm to expand the seed set along the curve as much as possible.

To ensure the covering of all the existing paths in the network of centroids while avoiding the re-identification of the same, a set of supporting lists of centroids are created and updated each time a new path is found. Let **SP** be the set of the simple points, **MBP** the set of the multibranch points in a given d-adjacency matrix. Every time a new path pass through a simple or a multibranch point the corresponding list must updated deleting the element. A list of extreme points **EP** is created exclusively to save centroids of that type and no update of it is necessary. Furthermore, for every new path found $(C_{j_0}, \ldots, C_{j_{l-1}})$ if C_{j_1} and/or $C_{j_{l-2}}$ are not multibranch points and respectively C_{j_0} and/or $C_{j_{l-1}}$ are not extreme points, C_{j_1} and/or $C_{j_{l-2}}$ are added to a list called *list of penultimate points* while C_{j_0} and/or $C_{j_{l-1}}$ are added to a list called *list of last points*. C_{j_0} is the "last point" of C_{j_1}, while $C_{j_{l-1}}$ is the $C_{j_{l-2}}$ one.

Observation 3.3.2. *Notice that we will never choose an extreme point as a starting point: choosing only simple and multibranch points as starting point ensure to find a path every time we start an expansion process, while choosing an extreme point we cannot guarantee to find a path if it is connected to another extreme point because the seed set would not be defined (as in Figure 3.8)*

Here follows the sequence of starting points the algorithm chooses to optimize the number of attempts of expansion process from seed sets of centroids.

1. The first step of the path detection algorithm starts by considering all the multibranch points as starting point for the expansion process, if **MBP** is not void. Suppose **MBP** not void, let C_0 be a multibranch point, $\{C_{0,0}, \ldots, C_{0,q-1}\}$ be the set of branches of C_0, with $q \geq 3$ (because C_0 has at least three branches).

   The algorithm need to find at least a path to fulfill the next steps. So, if **MBP** is void, we check if **SP** is not void. If it is not we take the first simple point of the list as starting point.

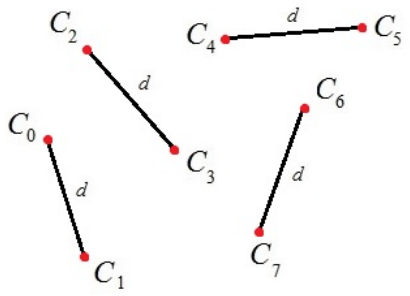


Figure 3.8: The graphic representation of the d-adjacency matrix relations between the centroids $C_0, C_1, C_2, C_3, C_4, C_5, C_6, C_7$: they are all extreme points.

If also **SP** is void it means that all the centroids involved in the current adjacency matrix are extreme points (like in Figure 3.8) and so it is impossible to find a path of centroids in this network of points. The current d-adjacency matrix cannot lead to a result in terms of paths.

2. Suppose that at least a path has been found, so exclude the case **MBP** $= \emptyset$ and **SP** $= \emptyset$.

   The second step of the path detection algorithm consists on considering all the centroids in the list of penultimate points as starting points. Let C_{pen} be a penultimate point and C_{last} the respective last point: we force the expansion direction, imposing expansion attempt in the "C_{pen} to C_{last}" direction.

3. Once all the penultimate points have been considered as starting point and the respective list is void, we take the first simple point available in the list dedicated to the simple points not yet in a path, if that list is not void. Supposing that the list of available simple points is not void, if new paths are built and the list of penultimate points is now no more void, we go back to the Point 2. The process ends when both the list of penultimate points (and so the last points one) and the

list of simple points not yet in a path are void.

4. The last level of starting points choice consists of a final check of some multibranch points: they are the multibranch points lying on an already found path, let it be $(C_{j_0}, \ldots, C_{j_{l-1}})$, where there exist $i \in \{j_0, \ldots, j_{l-1}\}$ such that C_i is not a multibranch point.

   Let denote as **MBPC** the set of the multibranch points satisfying the just cited characteristics, let $C_0 \in$ **MBPC**, let $\{B_0, \ldots, B_{r-1}\}$ be the set of the branches of C_0. We check if there already exists a path in the list of found paths such that it contains $\{B_j, C_0, B_k\}$, for every $j, k \in \{0, \ldots, r-1\}$, with $j \neq k$. If the check gives negative verdict and there is not a path containing $\{B_j, C_0, B_k\}$ for some j and k, a new expansion process must start from the seed set (B_j, C_0, B_k).

At the end of all these expansion processes *all* the existing paths in the network of centroids deriving from a fixed adjacency matrix are found.

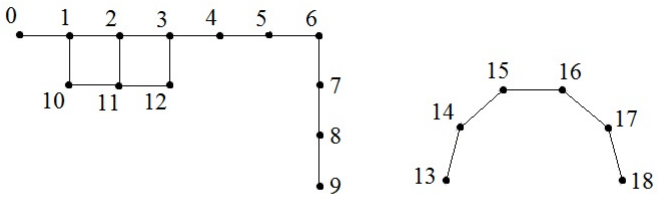
Observation 3.3.3. *The path detection algorithm considers an early interruption in case of a specific situation. If $\{C_0, \ldots, C_{n-1}\}$ is the set of n centroids referred to the REs associated to a GS and during the algorithm a path of length n or $n-1$ is found, the geometric verification process of REs is immediately applied (described in next section), without waiting for the natural end of the path detection algorithm. This exception is applied because a path of maximum length (or almost maximum length, as the exception occurs also in case of path of length $n-1$) could be in general a strong signal as it represents a path involving the centroids of all the REs on the current GS. It let us hope to find an entire and compact symmetric arrangement on that GS.*

Observation 3.3.4. *The choice of using penultimate points as starting point forcing the expansion direction towards the respective last point follows from the intuition that, when a path cannot no more expand itself, this may be a signal of the beginning of*

new ones, one for each extremity of the interrupted path. New paths rising from the extremity of an old path exist only if there are other centroids connected (by the d-adjacency matrix information) to the extremity centroid of the old path (what we called "last" point), and so if the centroid at the extremity is not an extreme point. Moreover, multibranch points are already taken as starting points, so we exclude a penultimate point to be a multibranch point.

3.3.3 An example of paths detection

Suppose that a fixed d-adjacency matrix in a fixed GS gave the situation represented in the following figure:



The centroids are identified by an index from 0 to 18 and the connection segment between two centroids represents the constant distance d.

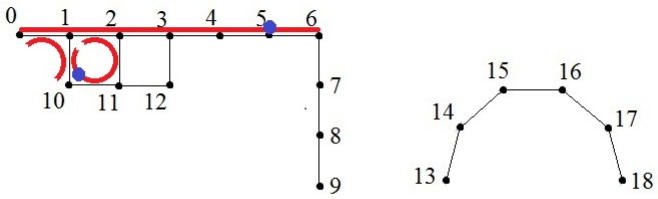
Counting the branches of each centroid, we see that

- centroids 0, 9, 13, 18 are extreme points;
- centroids 4, 5, 6, 7, 8, 14, 15, 16, 17 are simple points;
- centroids 1, 2, 3, 11 are multibranch points.

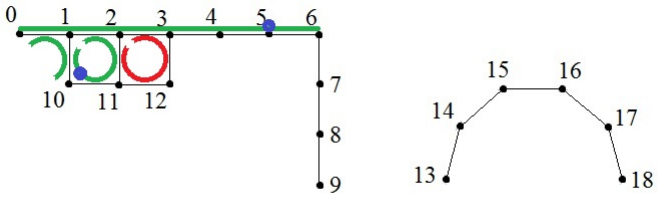
Suppose to apply the path detection algorithm on this d-adjacency matrix. We will highlight in red the new found paths with the current starting point considered, in blue the penultimate points not yet used, in green the paths found in the previous steps. At each step, suppose to delete a centroid from the corresponding set when it is used as

starting point and to update the set of simple points whenever a path containing it is created, as arranged by the path detection algorithm.

1. Consider the first available multibranch point, which is the centroid 1. In the following figure we have represented the found paths and the new penultimate points:

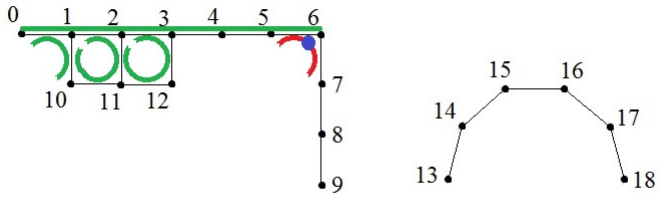


Now consider the next available multibranch point, which is centroid 2:



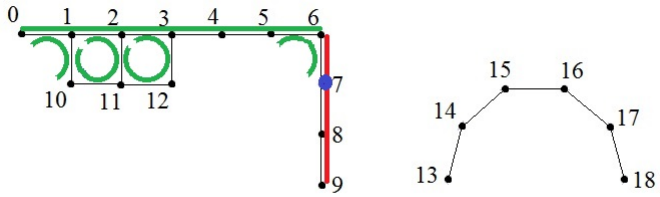
At this stage multibranch centroids 3 and 11 are considered as well but do not return any new path.

2. Once all multibranch points have been studied, we consider the first available penultimate point, which is centroid 5. Moving towards the respective last point (centroid 6) we find the new path of type circumference highlighted in the following figure, which carries also a new penultimate point:



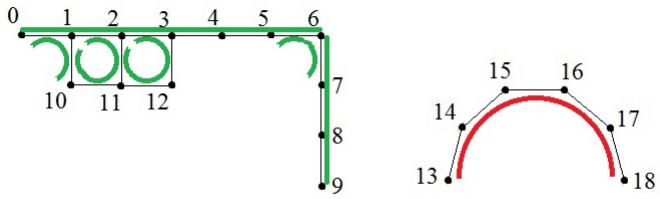
The penultimate point 10 does not return any new path.

Using penultimate point 6 we find a new path of type line, which gives back a new penultimate point:



However, the last found penultimate point 7 does not lead to a new path.

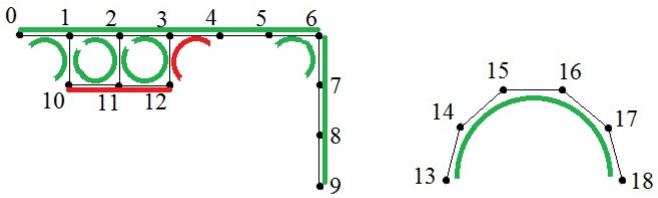
3. Since the set of penultimate points is currently void, we take as starting point the first available simple point in the list of simple points not yet belonging to a path. It is centroid 14 and it returns a path of type circumference of length 6:



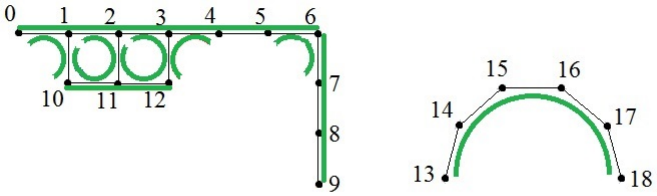
None new penultimate point is found. The set of penultimate points and the

set of simple points not yet in a path are now void.

4. The final verification of the multibranch points let us find two new paths from the branch exiting from centroids 3 and 11:



The process ends. The final situation is here represented:



In this example all the existing paths of centroids are therefore detected following the steps of the path detection algorithm proposed.

3.4 Patterns of Repeated Entity

In the previous section we have discussed how to detect all the paths of centroids of REs existing in a network of centroids. The network is built following the relations contained in a d-adjacency matrix.

In this section we intend to exploit the identified paths to verify if a regular arrangement of centroids can be translated in a regular arrangement of the corresponding repeated entities. A path of centroids gives an outline of the REs placement, so, ver-

ifying the correct orientation only relatively to the REs whose centroids are regularly organized, we avoid to examine all the REs associated to a GS. This approach optimizes the number of controls.

This phase of the algorithm has been developed for REs containing exclusively *planar* and *cylindrical* faces.

3.4.1 Patterns

Now we give the list of the symmetric arrangements of REs the proposed method is able to detect, classifying them by the type of the planar curve the REs' centroids lie on: line or circumference.

The symmetric arrangement types are defined through the translation and reflection notions, that have been introduced in Section 1.1. We will use the same notations of the section in which they are defined.

Let $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ be a set of REs and $\{C_0, \ldots, C_{q-1}\}$ be the set of the respective centroids, $q \geq 2$. Suppose $(C_0, \ldots, C_{q-1})$ to be a path.

If $(C_0, \ldots, C_{q-1})$ is a path of type **line**, the REs in $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ could constitute a pattern of type:

- *linear translational*.
- *reflectional*. This type of pattern can exist only if $q = 2$.

If $(C_0, \ldots, C_{q-1})$ is a path of type **circumference**, the REs in $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ could constitute a pattern of type:

- *circular translational*.

In this step of the approach we will fully exploit the vertices of the REs. The set of vertices of the REs checked during this phase does not correspond to the set of vertices used for the centroid computation (see Subsection 3.1.2): let $\mathbf{V}_j$ the set of vertices used

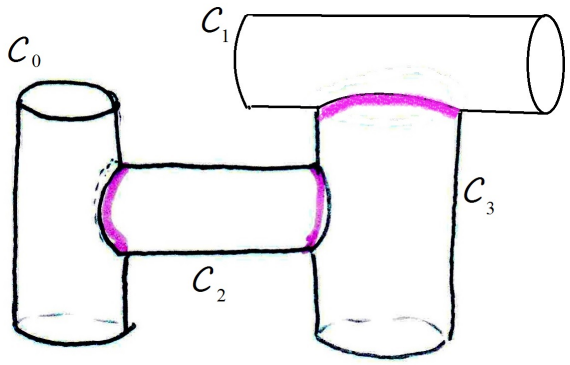


Figure 3.9: The representation of an example of RE with four cylindrical faces: $\mathcal{C}_0$ and $\mathcal{C}_1$ have two closed planar base edges, $\mathcal{C}_3$ has one of them, while $\mathcal{C}_2$ does not have closed planar base edges.

to compute the centroid C_j of the RE $\mathbf{A}_j$, then the set $\widetilde{\mathbf{V}}_j$ of the vertices of $\mathbf{A}_j$ we will use in this section is defined by $\widetilde{\mathbf{V}}_j = \mathbf{V}_j \setminus \mathbf{H}_j$, where $\mathbf{H}_j$ is the set of the vertices of $\mathbf{A}_j$ added on closed curved edges of $\mathbf{A}_j$ for the centroid computation.

We justify the exclusion of these vertices from the set of vertices by considering that the two points added on a closed curve edge of a RE and the ones added to the corresponding closed curve edge of another RE are not granted to be inserted in the same position. Indeed, to determine these points we used the lying curve parametrization and different ways of designing two repeated entities could lead to different parametrizations.

In the next subsection we will suppose $\widetilde{\mathbf{V}}_j = \{V_{j,0}, \ldots, V_{j,p_j-1}\}$ for $j = 0, \ldots, q-1$. We have $p_j = p_k = p$ for $k = 0, \ldots, q-1$, $p > 1$, as the REs are congruent by definition.

In the next subsections we will suppose $\mathbf{F}_j^{\mathbf{p}}$ and $\mathbf{F}_j^{\mathbf{c}}$ to be respectively the set of planar faces and the set of cylindrical faces of $\mathbf{A}_j$, for $j = 0, \ldots, q-1$. The congruency of the REs implies that $|\mathbf{F}_j^{\mathbf{p}}| = |\mathbf{F}_k^{\mathbf{p}}|$ and $|\mathbf{F}_j^{\mathbf{c}}| = |\mathbf{F}_k^{\mathbf{c}}|$ for $j, k = 0, \ldots, q-1$.

Observation 3.4.1. *Cutting a cylindrical face with a planar face (with lying plane not parallel to the axis of the cylinder), we can obtain two types of closed planar edges corresponding to the base edges of the cylindrical face. We obtain a closed edge with a circumference as hosting curve if the plane has normal corresponding to the axis direction of the cylinder, and the circumference has center lying oh the cylinder axis. Otherwise, we obtain a closed edge with an ellipse as hosting curve with center lying oh the cylinder axis.*

A cylindrical face can have none, one or two planar closed base edges (see Figure 3.9).

3.4.2 Linear translational pattern

Suppose $(C_0, \ldots, C_{q-1})$ be a path of type line. To conclude that the set of REs $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ is a *linear translational pattern* it is necessary to examine the REs in pairs.

Consider for a while two REs $\mathbf{A}_i, \mathbf{A}_{i+1}$ with consecutive centroids C_i, C_{i+1} in the given path , $i \in \{0, \ldots, q-2\}$. To understand if the relation between the two REs is a translation, a check of faces and vertices is provided.

Let $C_i = (x_{C_i}, y_{C_i}, z_{C_i})$ and $C_{i+1} = (x_{C_{i+1}}, y_{C_{i+1}}, z_{C_{i+1}})$. Then, the candidate *translational vector* is $v = (x_v, y_v, z_v)$ computed as

$$(x_{C_{i+1}} - x_{C_i}, y_{C_{i+1}} - y_{C_i}, z_{C_{i+1}} - z_{C_i}). \tag{3.4}$$

Observe that, as $(C_0, \ldots, C_{q-1})$ is a path of type line, the candidate translational vector v is the same for every pair of consecutive REs.

To conclude that $\mathbf{A}_i, \mathbf{A}_{i+1}$ are related by translation by the candidate translational vector v we provide two levels of check, the first referred to the vertices, the second exploits the surface information of the faces. The given REs are congruent by definition and in some cases the correspondence of the vertices would be sufficient to establish is

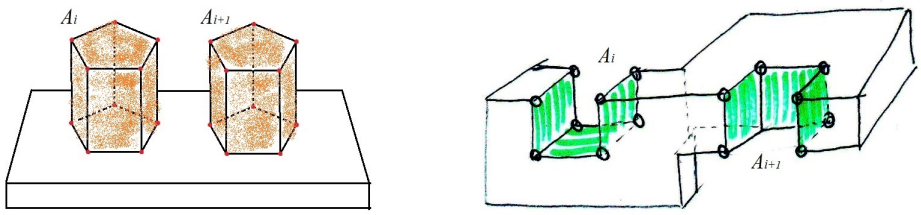


Figure 3.10: The REs are highlighted with different colors respect the remainder BRep: a) $\mathbf{A}_i, \mathbf{A}_{i+1}$ are related by translation and a vertices check completely determines the faces position; b) $\mathbf{A}_i, \mathbf{A}_{i+1}$ are not related by translation, even if an only vertices check gives a positive answer.

the two REs are correlated by translation, but there are cases in which also the correspondence of the faces is necessary, especially when there are symmetries in the set of vertices. For example see Figure 3.10: in the situation represented in a) the vertex check allows to conclude that it is a translation, while in b) the vertex check gives a positive answer but it is not a translation, so a check of the face arrangement is necessary.

Denote the origin of $\mathbb{R}^3$ as O.

1. *Vertex check*. We check if for $j = 0, \ldots, p-1$ there exist $k \in \{0, \ldots, p-1\}$ such that $V_{i+1,k}$ has coordinates $(x_{i,j} + x_v, y_{i,j} + y_v, z_{i,j} + z_v)$, which is to say

$$\overrightarrow{OV_{i+1,k}} = T_v(\overrightarrow{OV_{i,j}}).$$

   If the condition is satisfied, we pass to the next step.

2. *Face check*. Different check are provided for planar faces and for cylindrical faces.

   - *Planar faces*. For each planar face we compare the normals. We simply verify if for each planar face $F_k \in \mathbf{F}_i^{\mathbf{p}}$ there exist a planar face $F_h \in \mathbf{F}_{i+1}^{\mathbf{p}}$

such that $n_k = n_h$, where n_k, n_h are the normal vectors of the planes corresponding respectively to F_k, F_h.

- *Cylindrical faces*. For each cylindrical face in $\mathbf{A}_i$ such that it has at least a closed planar base edge we check if there exist a corresponding translated cylindrical face in $\mathbf{A}_{i+1}$ by exploiting axis and edges information.

  Let $F_k \in \mathbf{F}_i^{\mathbf{c}}$, $\mathcal{C}_k$ the lying cylinder, a_k the direction vector of the axis of $\mathcal{C}_k$, r_k its radius, O_k its origin, $\mathbf{W}_k$ the set of the circumferences corresponding to its closed planar base edges, $\mathbf{Z}_k$ the set of the ellipses corresponding to its closed planar base edges (information provided by the adopted CAD system, see also Subsection 3.1.1), such that $|\mathbf{W}_k| + |\mathbf{Z}_k| > 0$.

  We select the set of cylindrical faces

  $$\mathbf{C}_{i+1} = \{F_h \in \mathbf{F}_{i+1}^{\mathbf{c}} : r_h = r_k, |\mathbf{W}_h| = |\mathbf{W}_k|, |\mathbf{Z}_h| = |\mathbf{Z}_k|\},$$

  where r_h, $\mathbf{W}_h$, $\mathbf{Z}_h$ respectively denote the radius, the set of circumferences corresponding to the closed planar base edges, the set of ellipses corresponding to the closed planar base edges of the cylindrical face F_h.

  We select from $\mathbf{C}_{i+1}$ the set of cylindrical faces with axis corresponding to the line computed in the following way. Let G the point whose coordinates are defined by $\overrightarrow{OG} = T_v(\overrightarrow{OO_k})$, it should be on the axis line of the candidate cylinder. The cylindrical face we are looking for has axis corresponding to the line computed by considering the line passing through G with direction a_k, so we select from in $\mathbf{C}_{i+1}$ the cylindrical faces satisfying this condition.

  When a face F_h satisfying these conditions is found, we finally examine the base edges correspondence. For each circumference in $\mathbf{W}_k$ with center C_k^c, we check if there exist in $\mathbf{W}_h$ a circumference with center C_h^c such that $\overrightarrow{OC_h^c} = T_v(\overrightarrow{OC_k^c})$; for each ellipse in $\mathbf{Z}_k$ with center C_k^e, we check if there

exist in $\mathbf{Z}_h$ an ellipse with center such that $\overrightarrow{OC_h^e} = T_v(\overrightarrow{OC_k^e})$. Furthermore, for the ellipses we have to check also the major and minor radius equality and the correct orientation of the minor and major axis.

If it exists, only one cylindrical face F_h satisfies all the cited conditions and so is the corresponding translated of the cylindrical face F_k.

If both vertex and face checks give a positive answer, we can state that $\mathbf{A}_i$ and $\mathbf{A}_{i+1}$ are related by a translation with translational vector v. In formula

$$\mathbf{A}_{i+1} = T_v(\mathbf{A}_i)$$

Finally, we can say that the set of REs $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ constitute a *linear translational pattern* if $\mathbf{A}_i$ and $\mathbf{A}_{i+1}$ are related by a translational function with the translational vector v, for $i = 0, \ldots, q-2$.

3.4.3 Reflectional pattern

Let $q = 2$. To verify if two REs $\mathbf{A}_0$ and $\mathbf{A}_1$ are related by *reflection* we apply a vertex and a face check scheme, analogously as the translation on line case.

Let $C_0 = (x_{C_0}, y_{C_0}, z_{C_0})$ and $C_1 = (x_{C_1}, y_{C_1}, z_{C_1})$. Consider the normalized vector $n_m = (x_{n_m}, y_{n_m}, z_{n_m})$ defined by

$$\left(\frac{x_{C_0} - x_{C_1}}{\alpha}, \frac{y_{C_0} - y_{C_1}}{\alpha}, \frac{z_{C_0} - z_{C_1}}{\alpha} \right)$$

where $\alpha = \sqrt{(x_{C_0} - x_{C_1})^2 + (y_{C_0} - y_{C_1})^2 + (y_{C_1}, z_{C_1})^2}$, and the point

$$M_{C_0,C_1} = \left(\frac{x_{C_0} + x_{C_1}}{2}, \frac{y_{C_0} + y_{C_1}}{2}, \frac{z_{C_0} + z_{C_1}}{2} \right)$$

midpoint of C_0, C_1. We compute the plane passing through M_{C_0,C_1} and with n_m as normal vector: it is the candidate *reflection plane* of the REs $\mathbf{A}_0$ and $\mathbf{A}_1$ and we denote it as $\mathcal{P}_r$.

Here follows the description of check of the vertices and of the faces.

1. *Vertex check.* For each vertex $V_{0,k}$, $k \in \{0, \dots, p-1\}$, we must check if there exist $h \in \{0, \dots, p-1\}$ such that $V_{1,h}$ is the reflected of $V_{0,k}$ by the plane $\mathcal{P}_r$, in formula

$$\overrightarrow{OV_{1,h}} = R_{\mathcal{P}_r}(\overrightarrow{OV_{0,k}}).$$

   If the check of all the vertices in $\widetilde{\mathbf{V}}_0$ gives a positive answer, the face check is performed.

2. *Face check.* As for the linear translational pattern case, we discuss how to treat planar and cylindrical faces.

   - *Planar faces*.

     We simply verify if for each planar face $F_k \in \mathbf{F}_0^{\mathbf{p}}$ there exist a planar face $F_h \in \mathbf{F}_1^{\mathbf{p}}$ such that

$$n_h = R_{\mathcal{P}_r}(n_k),$$

     where n_k, n_h are the normal vectors of the planes corresponding respectively to F_k, F_h.

   - *Cylindrical faces*. For each cylindrical face in $\mathbf{A}_0$ such that it has at least a closed planar base edge, the following check must be done. Consider the same notations used for cylindrical faces in the translational pattern description. Let F_k be a a cylindrical face in $\mathbf{F}_0^{\mathbf{c}}$ such that $|\mathbf{W}_k| + |\mathbf{Z}_k| > 0$ and let $\mathbf{C}_1 = \{F_h \in \mathbf{F}_1^{\mathbf{c}} : r_h = r_k, |\mathbf{W}_h| = |\mathbf{W}_k|, |\mathbf{Z}_h| = |\mathbf{Z}_k|\}$. Then we consider the reflection of the point O_k by the plane $\mathcal{P}_r$, we denote it as G and it is defined by $\overrightarrow{OG} = R_{\mathcal{P}_r}(\overrightarrow{OO_k})$. We look for a cylindrical face in $\mathbf{C}_1$ such that its axis line corresponds to the line passing through G and with a_k as direction vector.

     Then we examine the closed planar base edges reflection correspondence. If a face F_h satisfying the above conditions is found, for each circumference

in $\mathbf{W}_k$ with center C_k^c, we check if there exist in $\mathbf{W}_h$ a circumference with center corresponding to C_h^c, a point such that $\overrightarrow{OC_h^c} = R_{\mathcal{P}_r}(\overrightarrow{OC_k^c})$, while for each ellipse in $\mathbf{Z}_k$ with center C_k^e we check if there exist in $\mathbf{Z}_h$ an ellipse with center corresponding to C_h^e, a point such that $\overrightarrow{OC_h^e} = R_{\mathcal{P}_r}(\overrightarrow{OC_k^e})$. Furthermore, for each ellipse in $\mathbf{Z}_k$ with v_{major} and v_{minor} as direction vector of respectively the major and the minor ellipse axis, the corresponding ellipse in F_h must have $R_\pi(v_{major})$ and $R_\pi(v_{minor})$ respectively as direction vector of respectively the major and the minor ellipse axis.

If it exists, only one cylindrical face F_h satisfies all the cited conditions and so is the corresponding reflected of the cylindrical face F_k.

If both vertex and face check give a positive answer, we can state that $\mathbf{A}_0$ and $\mathbf{A}_1$ are related by a *reflectional pattern*, with reflection plane $\mathcal{P}_r$. In formula

$$\mathbf{A}_1 = R_\pi(\mathbf{A}_0)$$

3.4.4 Circular translational pattern

Suppose $(C_0, \ldots, C_{q-1})$ to be a path of centroids of type circumference. To conclude that the set of REs $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ is a *circular translational pattern* it is necessary to examine the REs in pairs, analogously to the translational pattern on line case.

The difference in this situation is that the candidate translational vector changes direction for each pair of consecutive REs, while the euclidean norm of the vector is a constant value. Let $i, j \in \{0, \ldots, q-2\}$ and let $\mathbf{A}_i, \mathbf{A}_{i+1}$ and $\mathbf{A}_j, \mathbf{A}_{j+1}$ be two pairs of consecutive REs. Then the candidate translational vector of the pair $\mathbf{A}_i, \mathbf{A}_{i+1}$, noted by v_i, is computed as $(x_{C_{i+1}} - x_{C_i}, y_{C_{i+1}} - y_{C_i}, z_{C_{i+1}}) - z_{C_i}$, and analogously the candidate translational vector of the pair $\mathbf{A}_j, \mathbf{A}_{j+1}$, noted by v_j, is computed as $(x_{C_{j+1}} - x_{C_j}, y_{C_{j+1}} - y_{C_j}, z_{C_{j+1}} - z_{C_j})$ (see Formula 3.4). Denoting the euclidean norm of a vector $v \in \mathbb{R}^3$ as $\|v\|$, we have $\|v_i\| = \|v_j\| = d$, where d is the repeated

distance between centroids involved in the path of type circumference.

To conclude that the set of REs $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ constitute a *circular translational pattern* we must verify that for $i = 0, \ldots, q-2$ the REs $\mathbf{A}_i$ and $\mathbf{A}_{i+1}$ are related by translational function with the translational vector v_i.

3.5 The patterns detection algorithm

In this section we neatly illustrate the sequence of steps of the *pattern detection algorithm*. We provided an outline in the previous sections and the *path detection algorithm* is recalled in the main algorithm, constituting a fundamental part of the process.

This is the pseudo-code that summarizes the entire algorithm:

Input: RE_list, list of REs

Output: Pattern_list, list of patterns of REs

Pattern_list = *empty*

Pattern_length2_list = *empty*

GS_list = *empty*

for each *($\mathcal{R}$ in* RE_list*)* ***do***

\\ *Building and managing of REs information*

$\mathcal{V}$ = *VERTICES_ADDED_TO_CURVED_EDGES($\mathcal{R}$)*

C($\mathcal{R}$) = *CENTROID_COMPUTATION(*vertices*($\mathcal{R}$)*$\cup$ $\mathcal{V}$*)*

\\ *Creation of GSs*

GS = *GET_GS($\mathcal{R}$)*

GS_list = GS_list $\cup$ GS

end for

THIN_OUT_AND_ORDERING(GS_list)

for each *($\mathcal{G}$ in* GS_list*)* ***do***

if List_of_REs($\mathcal{G}$).*count* > *2* ***then***

List_adjacency_matrices = *COMPUTE_ADJACENCY_MATRICES* ($\mathcal{G}$)

THIN_OUT_AND_ORDERING(List_adjacency_matrices)

for each *(*$\mathcal{M}$ *in* List_adjacency_matrices*)* ***do***

List_of_paths = *FIND_ALL_PATHS* ($\mathcal{M}$)

ORDERING(List_of_paths)

for each *(*$\mathcal{P}$ *in* List_of_paths*)* ***do***

\\ *Verification of the identified paths*

VERIFY_PATTERNS_AND_UPDATE_LIST_OF_PATTERNS(

$\mathcal{P}$, Pattern_list, Pattern_list_length2)

UPDATE_OTHER_DATA(

List_of_paths,List_adjacency_matrices, GS_list, Pattern_list_length2)

Remove $\mathcal{P}$ *from* List_of_paths

end for

Remove $\mathcal{M}$ *from* GS_list

end for

Remove $\mathcal{G}$ *from* List_adjacency_matrices

end if

Remove $\mathcal{M}$ *from* GS_list

end for

\\ *Final attempt to find patterns of length 2*

ARRANGEMENTS_IN_PATTERNS_OF_LENGTH2 (

Pattern_list_length2, GS_list, Pattern_list)

return Pattern_list

In details, the operations implemented to reach the detection of symmetric arrangements of REs are the followings.

1. *Acquisition of faces of REs*. The REs are acquired as input by selecting lists of faces and each list of faces correspond to a RE.

2. *Building and managing of REs information*. The input faces and the respective vertices are stored, the necessary vertices for curved (open and closed) edges are added, the centroids are computed.

3. *Creation of GSs*. For each RE, the set of surfaces corresponding to a face adjacent to the RE is extracted. For every surface of this set we verify if the corresponding GS has already been created in a previous step or not: if the GS exists we add the current RE to the REs list of the GS, otherwise the new GS is created and the current RE is added as first element of the corresponding associated REs list.

4. *Thin out and ordering of GSs list*. All the GS with associated a list of REs containing only one element are deleted from the list of GSs. Then, the remaining GSs are ordered by descending criteria respect to the number of elements in the associated REs list.

5. *Examination of GSs list and patterns verification*. Then, taking the first available GS from the list of GSs, the operations described in details in Subsection 3.5.1 are applied to it.

   The current GS is then deleted from the list of GSs.

6. *Final attempt to find patterns of length* 2. At the end of GSs analysis there could remain a set of initially selected REs not yet in a detected symmetric arrangements of length at least 3. For these REs we decided to associate them, if it exists, to an already found symmetric arrangement of length 2 containing them (these patterns are found "by chance" while we were looking for longer patterns, as we will explain below). If such a pattern of length 2 does not exist, a possibility (not implemented yet) is to try to couple the remaining REs belonging to

the same GS list two by two, verifying if they are related by a regular arrangements (translation, reflection, rotation), starting the coupling attempting from the nearest couples of REs.

3.5.1 Grouping surface examination

More specifically, the following operations are considered for each GS such that the number of REs in the associated list in greater than 2:

1. *Creation of the list of d-adjacency matrices*. Every combination of centroids distance is computed: when the corresponding d-adjacency matrix already exists, it is updated, otherwise a new d-adjacency matrix is created.

2. *Thin out and ordering of the list of d-adjacency matrices* . All the d-adjacency matrices with *nOccur* value equals to 1 are deleted from the list of d-adjacency matrices. Then, the remaining matrices are ordered by creasing value of d.

3. *Getting patterns of REs from d-adjacency matrices*. Considering every time the first available d-adjacency matrix from the list of remaining d-adjacency matrices, the following operations are performed for each d-adjacency matrix.

   - *Path detection algorithm application*. The algorithm described in 3.3 is executed on the current d-adjacency matrix network of centroids.
   - *Results of the path detection algorithm evaluation*. If the applied algorithm gives back a void list of detected paths, we consider the next d-adjacency matrix, if there are available matrices. If no other matrices are available for this GS, the examination of the current GS ends.

     If the list of detected paths is not void, such a list is ordered by descending ordering respect to the path length. Furthermore, we choose to give precedence to the paths of type line of length 3 respect to the paths of type

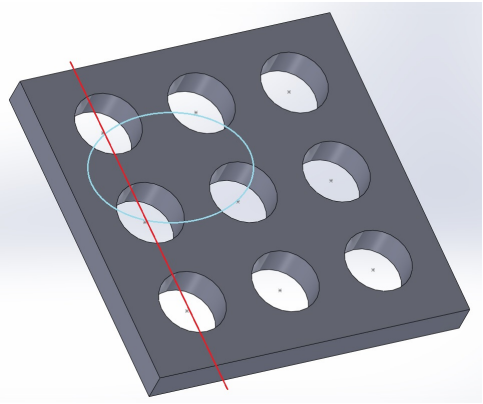


Figure 3.11: "Grid" case: we consider the path corresponding to the red line more significant than the path of type circumference in blue.

circumference of length 4 (see Observation 3.5.1). Then, the first available path of centroids is examined to verify if it is effectively a symmetric arrangements of the respective REs (more details are given in Subsection 3.5.2) and it is deleted from the list of found paths.

This continues until the list of paths is void.

4. *Current d-adjacency matrix deleting*. The just treated matrix if deleted from the list of the adjacency matrices of the current GS.

If the number of REs in the list associated to the GS is 2, the path detection algorithm cannot be applied to the set of centroids, but such a GS indicates that the two REs associated to the GS lie on the same surface and so they can be considered in the final attempt of finding symmetric arrangements of length 2.

Observation 3.5.1. *The choice of giving precedence to paths of type line of length 3 respect to paths of type circumference of length 4 derives from the application context. As an example see Figure 4.2.2: we consider a path of type line of length $l = 3$ more significant respect to a path of type circumference of length $l = 4$ in this case.*

It is not a restrictive choice: intersecting path of type line of length 3 *and of type circumference of length* 4 *are common especially in grid cases. Also in general we have found valid output results in the performed tests.*

3.5.2 Verifying patterns

The type of path of centroids to verify determines the type of candidate patterns it could lead to: if the path is of type *line* we will look for linear translational pattern, if the path is of type *circumference* we will look for a circular translational pattern.

When a path of centroids is verified, the REs are compared in pairs of two. The comparison could give a negative answer at any time during the process. Let $(C_0, \ldots, C_{q-1})$ the path of centroids to verify, let $\{\mathbf{A}_0, \ldots, \mathbf{A}_{q-1}\}$ be the set of the corresponding REs. Suppose the that the algorithm gives the first negative answer while comparing $\mathbf{A}_i$ and $\mathbf{A}_{i+1}$, with $i \in \{0, \ldots, q-3\}$, so before the end of the path. If $i = 0$ it means that the first interruption occurred comparing the first to REs, then no results are stored. If $i > 0$, we consider the pattern built up to the interruption and we store the symmetric arrangement composed by $\{\mathbf{A}_0, \ldots, \mathbf{A}_i\}$. The process starts again from the interruption point: a new comparison between $\mathbf{A}_{i+1}$ and $\mathbf{A}_{i+2}$ takes place and goes on as much as possible, up to a new interruption or up to the end of the path. If a new interruption occurs, the behavior to keep is analogous to the first interruption.

Remember that patterns of length 2 and patterns of length higher than 2 are stored in different lists: the patterns of length 2 are not definitive and are not confirmed up to the end of the GSs analysis. Indeed, until all the GSs are not examined yet, there is still the possibility to arrange the REs composing a pattern of length 2 in a longer and more significant pattern.

3.5.3 New pattern: data updating

When a candidate path of centroids corresponds to a verified symmetric arrangement of length $l \geq 3$, the pattern of REs is stored to be provided as output, while a series of data must be updated to avoid patterns intersection (we decide to assign a RE to a single pattern, which is the first detected containing it) and data incongruence.

- *Update of the list of paths*. A remaining path containing a centroid C_i corresponding to a RE already arranged in a pattern must be split. Let

$$(C'_0, \ldots, C'_{i-1}, C_i, C'_{i+1}, \ldots, C'_{q-1})$$

  be a path in the list of paths not yet verified and suppose it contains C_i. Consider the two "sons paths" $(C'_0, \ldots, C'_{i-1})$ and $(C'_{i+1}, \ldots, C'_{q-1})$ obtained by removing C_i from the path (notice that there could be only one son path if C_i is an extreme of the path). Each son path must have length $l \geq 3$: if it has such a length it is stored in the list of paths to verify, otherwise it is discarded. After this step the list of path must be reordered by descending length of the paths.

- *Update of the list of d-adjacency matrices*. For each centroid C_i corresponding to an just arranged RE we must update all the adjacency matrices of the current GS, by deleting all the relations with other centroids in every remaining d-adjacency matrix. After this step, we must check the updated matrices to have a sufficient $nOccur$: if $nOccur < 2$ the corresponding adjacency matrix is deleted from the list.

- *Update of the list of grouping surfaces*. The REs composing the found verified pattern are deleted from the lists of REs associated to GSs not yet examined, if they are contained in them. If after this updating a GS has an associated REs list containing only one element, the GS is deleted.

- *Update of the list of patterns of length* 2. Every pattern of length 2 intersecting a longer found pattern is deleted.

This data updating ensures not to find a pattern contrasting with the already stored ones.

Indeed, when a pattern of length 2 is found no data are updated, because we need to keep the corresponding REs available for possible new longer patterns.

Chapter 4

Implementation and experiments

This chapter addresses the implementation of the approach proposed in the previous chapter. The first section describes the development environment, giving some details related to the CAD system and the programming languages adopted. The second section reports some of the results obtained applying the method on a set of CAD models.

4.1 Development environment

The developed algorithm requires as input a set of faces selected from the BRep model. Moreover, it is necessary to visually highlight the results of the algorithm, by indicating the detected patterns directly on the original model. To this aim, a tool allowing directly accessing and modifying a BRep model has been developed.

The adopted CAD system is SolidWorks©, and it is the application programming environment in which BRep models can be created, visualised, accessed and processed. The algorithm has been developed in the $C\#$ programming language, using Visual Studio as development environment. The generated methods have been integrated as a plug-in for the CAD system. A plug-in (or plugin, extension) is a software component that adds a specific feature to an existing software application, if it supports them. This

integration enables customization of the CAD system.

Microsoft Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft and provides the user interface (UI) for standard components, such as compilers, editors, and debuggers. Features like Visual C++ and Visual Basic that are included with Visual Studio are themselves extensions of the IDE. It is used to develop console and graphical user interface applications. Visual Studio supports a range of programming languages, such as $C/C++$, $VB.NET$, $C\#$ and $F\#$. It also supports $XML/XSLT$, $HTML/XHTML$, *JavaScript* and *CSS*. By separately installing language services, other languages such as M, *Python*, and *Ruby* can be treated by Visual Studio environment. It accepts plug-ins that enhance the functionality at almost every level, including adding support for source-control systems (like Subversion and Visual SourceSafe). Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger.

SolidWorks© CAD System

Solidworks© is a Parasolid-based solid modeler and utilizes a parametric feature-based approach to create models and assemblies.

Parameters refer to constraints whose values determine the shape or geometry of the model or assembly. Parameters can be either numeric parameters, such as line lengths or circle diameters, or geometric parameters, such as tangent, parallel, concentric, horizontal or vertical, etc. Numeric parameters can be associated with each other through the use of relations, which allows them to capture design intent. With the term “design intent”we intend high-level geometric relations which can be seen as objectives and requirements between a CAD model sub-parts, established by the designer to predeter-

mine how to respond to changes and updates.
Finally, the parametric nature of SolidWorks© means that the dimensions and relations drive the geometry.

Features are the shapes and operations that construct the part. Shape-based features typically begin with a 2D or 3D sketch of shapes such as bosses, holes, slots, etc. This shape is then extruded or cut to add or remove material from the part. Operation-based features are not sketch-based, and include features such as fillets, chamfers, shells, applying draft to the faces of a part, etc.
Building a model in SolidWorks© usually starts with a 2D sketch, which consists of geometry such as points, lines, arcs, conics (except the hyperbola), and splines. Dimensions are added to the sketch to define the size and location of the geometry. Relations are used to define attributes such as tangency, parallelism, perpendicularity, and concentricity. The dimensions in the sketch can be controlled independently, or by relationships to other parameters inside or outside of the sketch.

Additionally, SolidWorks© can import plain BRep models created by other CAD systems.

We obtained the BRep information by exploiting the *Application Programming Interface* (API) of this CAD system: the API consist of a wide range of functions that the developer can recall from Visual Basic for Applications (VBA), VB.NET, Visual $C\#$, Visual $C++$ 6.0, and Visual $C++/CLI$, to access SolidWorks© functionalities. In particular, we used API to achieved the information about faces, edges, vertices, loops and all other entities constituting the boundary of an object. API can also allow operations such as creating a line, changing attributes of the faces, or verifying the parameters of a face surface.

Our plugin is registered and starts up when SolidWorks© starts, becoming visible in the Taskpane tab and available for the user.

4.2 Experiments

We have tested our algorithm on various BRep models and in this section a number of experiments are reported together with output details. Some models were created using the commercial CAD system SolidWorks©, and others were collected from the public repositories GrabCAD [1] and TraceParts [3].
In the following graphical representations we will color in:

- *violet* the faces corresponding to the selected repeated entities (input);
- *red* nuances a detected linear pattern of repeated entities (output);
- *green* nuances a detected circular pattern of repeated entities (output).

If more than one linear (circular) pattern is detected in the experiment, a different nuance of red (green) will be associated to each linear (circular) pattern.

Furthermore, we will highlight the centroids of the selected repeated entities by *blue* circles in correspondence of their positions.

4.2.1 Cylindrical mechanical component

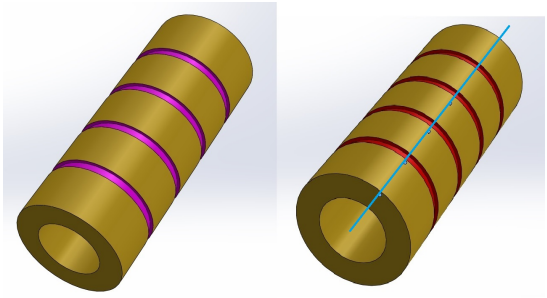


Figure 4.1: Cylindrical mechanical component. a) Selected REs; b) detected pattern.

We first consider the cylindrical mechanical component represented in Figure 4.1. Figure 4.1 a) highlights the selected faces as input, constituting the REs whose any regular arrangement is searched. Each RE is composed by a cylindrical face and two planar faces adjacent to it.

As expected, the algorithm detects a single translational linear pattern (Figure 4.1 b), constituted by all the four initially selected REs. This is the maximum pattern case described in Chapter 3.

4.2.2 Grid

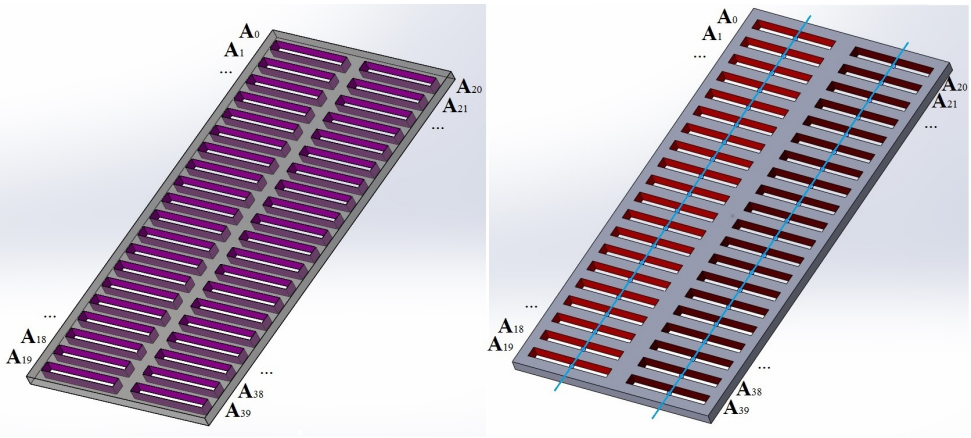


Figure 4.2: Grid. a) Selected REs; b) detected patterns.

The set of selected REs $\{\mathbf{A}_0, \ldots, \mathbf{A}_{39}\}$ is constituted by 40 holes, each of them represented by four planar faces. The input faces are colored in violet in Figure 4.2 a).

The algorithm detects two distinct linear translational patterns, each of them containing 20 REs as shown in Figure 4.2 b). The first is composed by $\{\mathbf{A}_0, \ldots, \mathbf{A}_{19}\}$, the second by $\{\mathbf{A}_{20}, \ldots, \mathbf{A}_{39}\}$. The lines referred to the paths of centroids of each pattern are parallel. All the input repeated entities are included in the detected patterns.

Notice that also 20 parallel linear translational patterns of length 2 exist in the model, if we consider the patterns constituted by pairs of REs $\{\mathbf{A}_i, \ldots, \mathbf{A}_{i+20}\}$, for $i = 0, \ldots, 19$, but the algorithm opts for the longest existing patterns.

4.2.3 Repeated cube

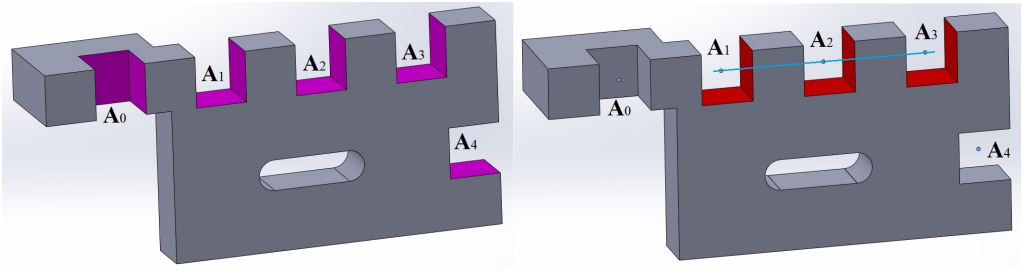


Figure 4.3: Repeated cube. a) Selected REs; b) detected pattern.

The input set in this example is constituted by a set of 5 digs, each of them composed by 3 squared faces (the selection is represented in Figure 4.3 a). The initial input set is therefore $\{\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4\}$.

The algorithm detects 1 linear translational pattern of length 3 whose components are the elements of the set $\{\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3\}$. The REs $\mathbf{A}_0$ and $\mathbf{A}_4$ are not arranged in any pattern, as represented in Figure 4.3 b).

We observe that, during the centroid analysis, a path of type line involving the centroids of $\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3$ is detected. Anyway, the following face check excludes $\mathbf{A}_0$ as its orientation is clearly different respect to $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3$.

4.2.4 Pad with blind-holes

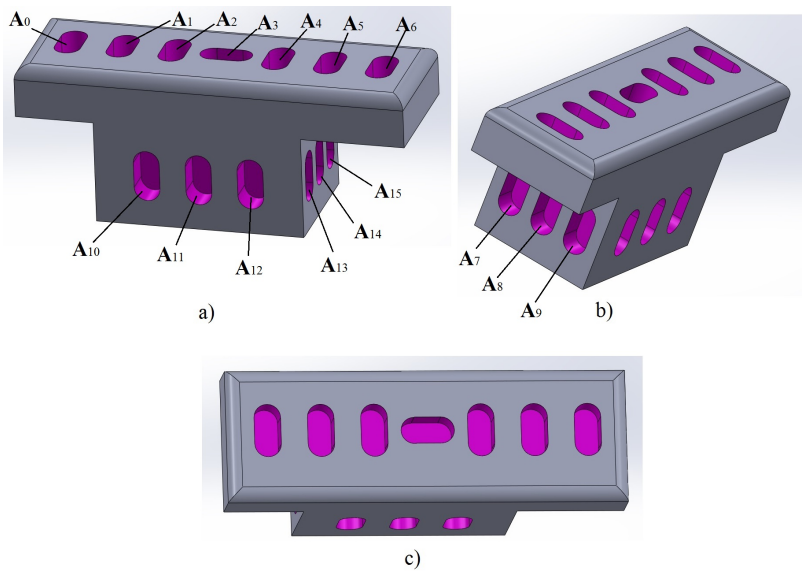


Figure 4.4: Pad with blind-holes: selected REs. a) first view; b) second view; c) third view.

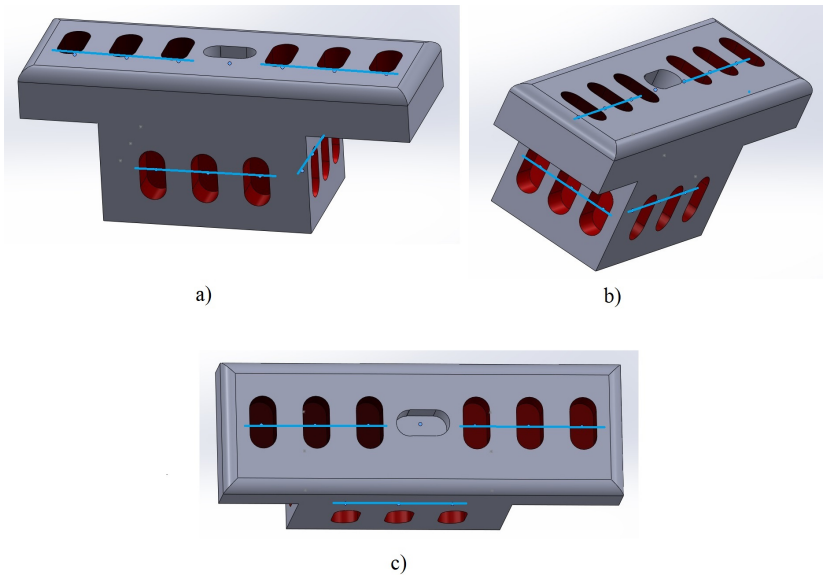


Figure 4.5: Pad with blind-holes: detected patterns. a) first view; b) second view; c) third view.

Consider as input the set of 16 pockets highlighted in Figures 4.4 a), 4.4 b), 4.4 c), noted by $\mathbf{A}_0, \ldots, \mathbf{A}_{15}$. Each RE is composed by 2 planar faces and 2 cylindrical faces. As shown by the reference figures, the REs are placed on different sides of the "T" base component, in particular on the top, on the left, on the front, on the right of it.

The algorithm detects 5 linear translational patterns of length 3, as represented in Figures 4.5 a), 4.5 b), 4.5 c). The first two patterns are detected on the top side of the base component: they are the sets $\{\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2\}$ and $\mathbf{A}_4, \mathbf{A}_5, \mathbf{A}_6$. These two patterns come from a single detected centroid path of length 7, involving the centroids of $\mathbf{A}_0, \ldots, \mathbf{A}_6$. However, $\mathbf{A}_3$ is differently arranged respect to the other 6 REs on the path, despite its centroid is positioned on the same line path of the others. For this reason, the algorithm detects two distinct translational patterns, breaking the detection in correspondence of $\mathbf{A}_3$ and excluding it from the patterns. Then, the other 3 linear translational patterns are positioned on the left, front, and right side of the base component. They are the sets $\{\mathbf{A}_7, \mathbf{A}_8, \mathbf{A}_9\}$, $\{\mathbf{A}_{10}, \mathbf{A}_{11}, \mathbf{A}_{12}\}$, and $\{\mathbf{A}_{13}, \mathbf{A}_{14}, \mathbf{A}_{15}\}$.

4.2.5 Multiplier with gear pump

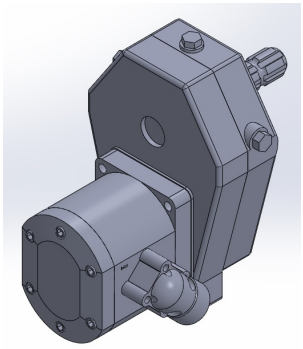


Figure 4.6: Multiplier with gear pump.

We analyze the mechanical component model showed in Figure 4.6. Different sets of congruent sub-parts are present in this model. We performed two separate symmetric

arrangement detection processes, each of them referred to a different set of congruent sub-parts.

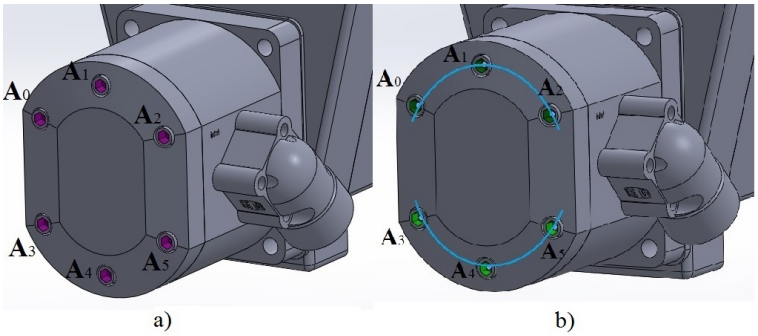


Figure 4.7: Multiplier with gear pump: first selection. a) Selected REs; b) detected patterns.

The first input set, as Figure 4.7 a) shows, is composed by 6 REs, each of them constituted by 7 planar faces.

The algorithm detects two circular translational patterns (see Figure 4.7 b), each of them containing 3 REs from the input set: the first one is composed by the set $\{\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2\}$, the second one by the set $\{\mathbf{A}_3, \mathbf{A}_4, \mathbf{A}_5\}$. The two circumferences resulting from the detected centroid paths have same radius. Furthermore the two circumferences share the constant centroid distance.

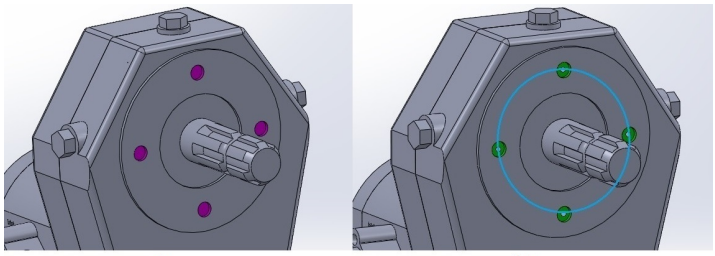


Figure 4.8: Multiplier with gear pump: second selection. a) Selected REs; b) detected pattern.

The second detection process receive as input a set of 4 congruent sub-parts, each of them composed by a cylindrical face and a circular planar face (Figure 4.8 a)).

As Figure 4.8 b) shows, the output is constituted by a single circular translational pattern containing all 4 REs of the initial set (maximum pattern case).

4.2.6 Castle

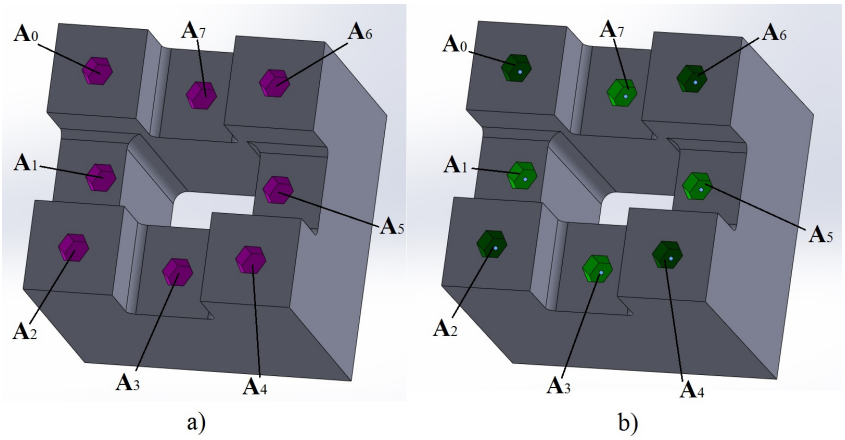


Figure 4.9: Castle. a) Selected REs; b) detected patterns.

The selected input set (see Fig. 4.9 a) is composed by 8 REs, each of them representing a pocket composed by 7 planar faces.

The output result of the detection algorithm is the following (see Fig. 4.9 b):

- the set $\{\mathbf{A}_0, \mathbf{A}_2, \mathbf{A}_4, \mathbf{A}_6\}$ constitutes a first closed circular translational pattern;
- the set $\{\mathbf{A}_1, \mathbf{A}_3, \mathbf{A}_5, \mathbf{A}_7\}$ constitutes a second closed circular translational pattern.

The two patterns are detected on two different grouping surfaces and no RE is excluded from the set of detected patterns.

4.2.7 Circular plate 1

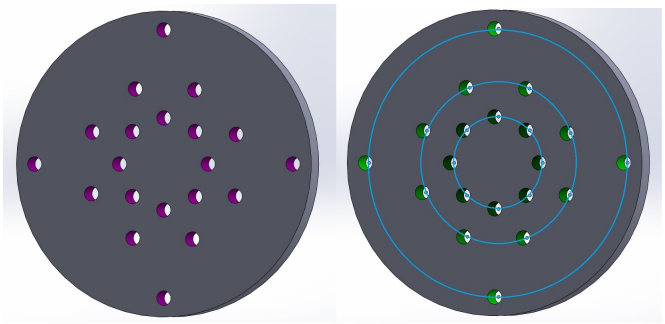


Figure 4.10: Circular plate 1. a) Selected REs; b) detected patterns.

The input set is composed by 20 thru-holes on a circular plate, each of them composed by a single cylindrical face (Fig. 4.10 a). The algorithm detects the REs to be subdivided on 3 different circular translational patterns (Fig. 4.10 b), each of them having in common with the others the center of the reference circumference. No RE from the input set in excluded from the detected patterns.

4.2.8 Circular plate 2

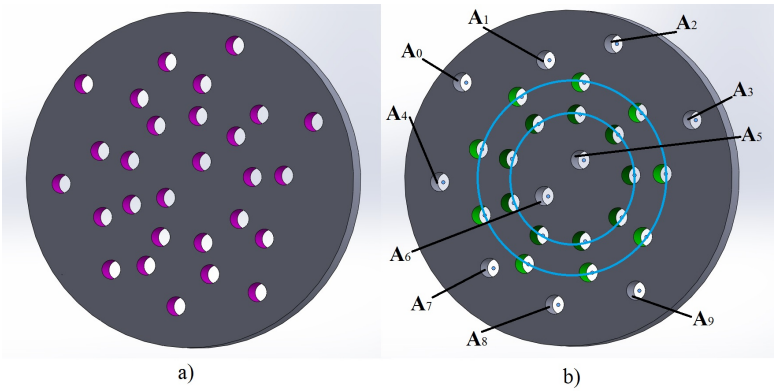


Figure 4.11: Circular plate 2. a) Selected REs; b) detected patterns.

In this example 28 thru-holes constitute the input set. In this case it is not visually evident a pattern involving the whole set of part of the set of REs (Fig. 4.11 a). Anyway, the algorithm detects two concentric circular translational patterns of length 9 (Fig. 4.11 b). The REs noted by $\mathbf{A}_0, \ldots, \mathbf{A}_9$ are excluded from the detected patterns.

4.2.9 Circular plate 3

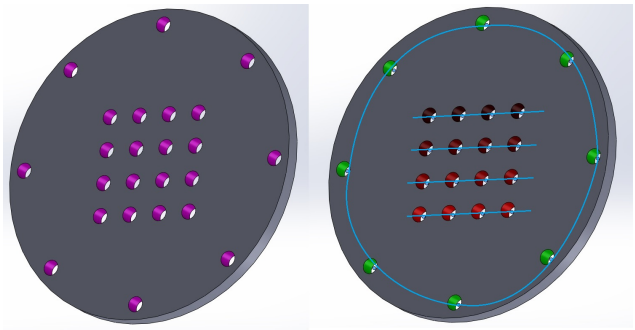


Figure 4.12: Circular plate 3. a) Selected REs; b) detected patterns.

We consider the circular plate represented in Figure 4.12 a). We select all the 24 cylindrical faces corresponding to the holes.

The algorithm in this case detects 1 circular translational patterns of 8 REs from the initial set, which are those located on the border of the plate. Furthermore, 4 parallel linear translational patterns are detected, each of them composed by 4 REs. The lines passing through the linear patterns are parallel and the distance between centroids is equal in each of them.

An alternative and equivalent detection may be the identification of 4 parallel linear translational patterns along the vertical direction, identical in type and in centroid distance to the 4 detected along the horizontal direction. The horizontal direction is also confirmed by the red nuances changing in Fig. 4.12 b). The result depends on the order

in which the REs are selected to create the input set.

Suppose to modify the original model as shown in the following figures:

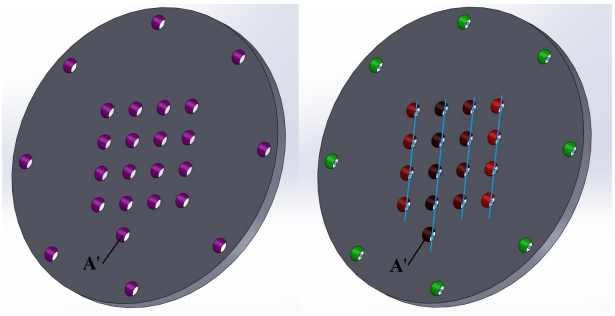


Figure 4.13: Circular plate 3 modified. a) Selected REs; b) detected patterns.

Furthermore, suppose not to change the selecting order of the REs and to select the added RE $\mathbf{A}'$ as last (Fig. 4.13 a). The adding of a $\mathbf{A}'$ forces a change in direction in linear pattern detection, as highlighted in 4.13 b). The pattern containing $\mathbf{A}'$ is detected first as corresponding to the longest path of centroids (length 5), then the other 3 linear translational patterns of length 4 follow the just fixed vertical direction.

4.2.10 Circular plate 4

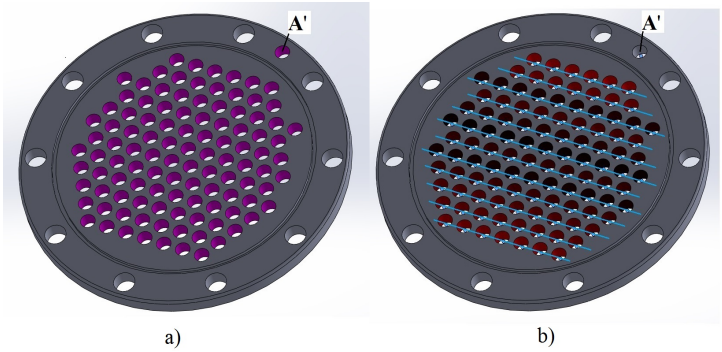


Figure 4.14: Circular plate 4. a) Selected REs; b) detected patterns.

The circular plate model in Fig. 4.14 a) has 119 thru-holes. We select the set of 109 congruent holes, having smaller diameter respect the 10 holes arranged on the on the board of the model, also congruent to each other. The selection is represented in Fig. 4.14 a).

The algorithm detects 12 parallel linear translational patterns:

- 1 pattern of length 6;
- 2 patterns of length 7;
- 1 patterns of length 8;
- 2 patterns of length 9;
- 4 patterns of length 10;
- 2 patterns of length 11.

The RE $\mathbf{A}'$ is excluded from the identified patterns, as shown in Fig. 4.14 b).

4.2.11 Electric component

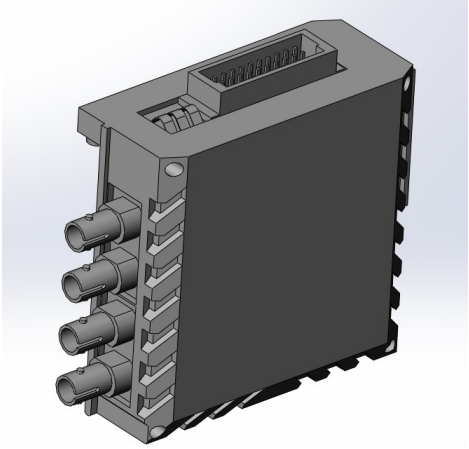
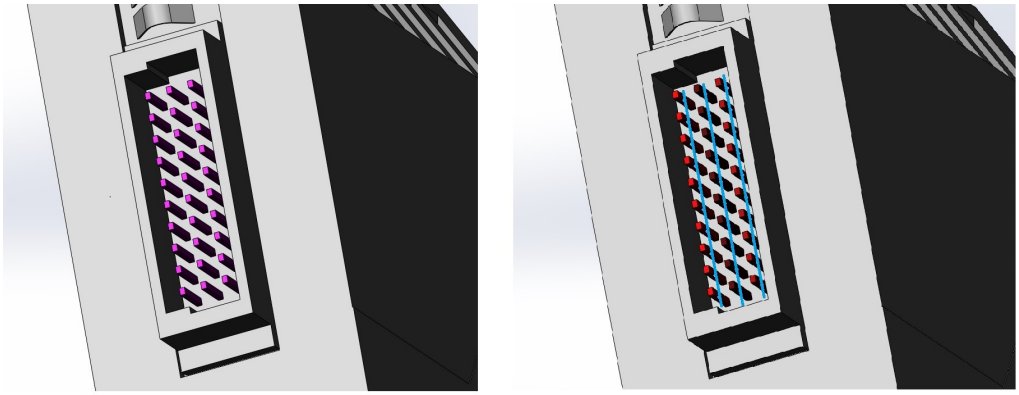


Figure 4.15: Electric component.

The model considered in this test is represented in Fig. 4.15. We perform 4 different detection processes, in correspondence of 4 sets of congruent sub-parts.

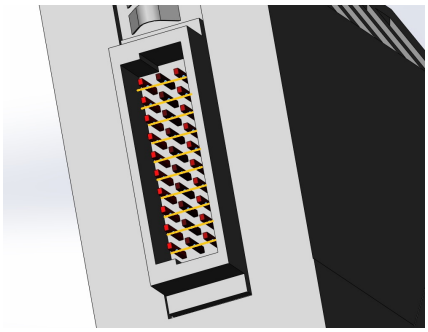


a) b)

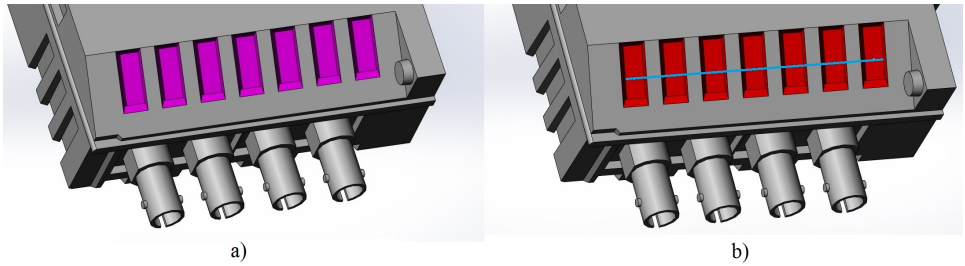
Figure 4.16: Electric component: first selection. a) Selected REs; b) detected patterns.

In Fig. 4.16 a) the first input selection is highlighted: the input is constituted by a set of 30 congruent extrusions, each of them represented by a set of 5 planar rectangular faces.

The detection algorithm returns the information relative to the presence of 3 linear translational patterns of length 10 (Fig. 4.16 b). So, no RE from the initial set is discarded by the detection process.



We observe that an alternative valid result exists if we consider 10 linear translational patterns of length 3 along horizontal direction instead of vertical direction, as shown in the figure on the left. This option is not detected by the algorithm because it gives precedence to longest existing patterns in the input set.

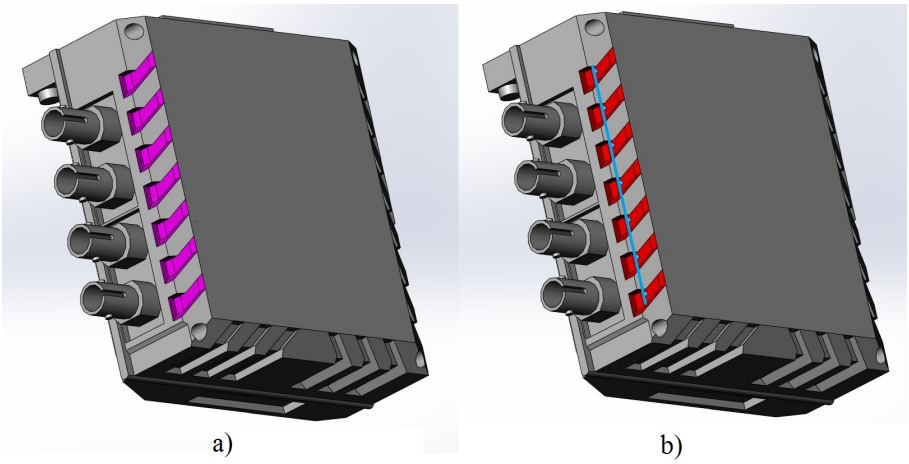


a) b)

Figure 4.17: Electric component: second selection. a) Selected REs; b) detected patterns.

The second selection is referred to the set of 7 pockets represented in Fig. 4.17 a). They are congruent and each of them is composed by 5 planar faces.

The detection result is a linear translational pattern of maximum length, as shown in Fig. 4.17 b).

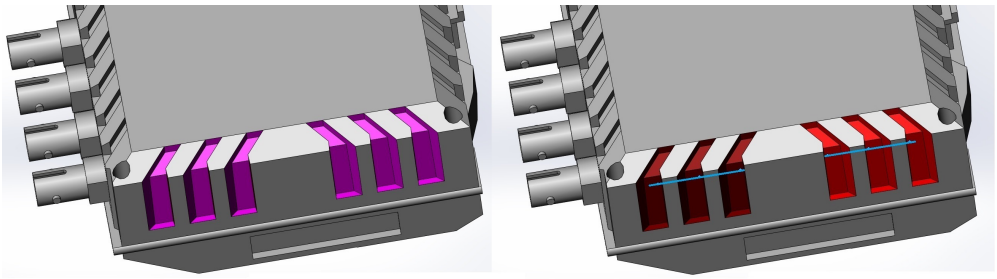


a) b)

Figure 4.18: Electric component: third selection. a) Selected REs; b) detected patterns.

The third input set is constituted by 7 pockets (represented in Fig. 4.18 a), analogously to the previous selection. In this case REs are composed by 7 planar faces.

The algorithm detects a linear translational patterns involving all the REs in the input set (see Fig. 4.18 b).



a) b)

Figure 4.19: Electric component: fourth selection. a) Selected REs; b) detected patterns.

The set of 6 congruent pockets represented in Figure 4.19 a) are selected as input set. Each pocket is composed by 7 planar faces.

The output of the algorithm in this case is a pair of linear translational patterns, each of them containing 3 REs (Fig. 4.19 b). The centroid distance is the same in both the detected patterns.

Chapter 5

Conclusions and future work

This research proposes an approach to detect regular patterns of congruent sub-parts in an solid object, aiming at recovering the high-level information embodied in intentionally incorporated symmetries in a given BRep model.

The approach has been implemented in a commercial CAD (Computer-Aided Design) system and detects linear, circular translational and reflectional patterns of congruent user-selected sub-parts of the model.

Given an input constituted by a set of faces corresponding to the set of selected repeated entities (see Definition 3.1.1), the algorithm applies a progressive grouping to the initial set, up to reach the sets of sub-parts corresponding to verified regular patterns.

The first grouping is based on the consideration that patterns are usually applied on a face at some creation stage when designing a model. For this reason, it is appropriate to think that patterns lie on the same face or, in alternative, on more than one face lying on the same host surface. Thus, the first grouping operation classifies repeated entities putting together those lying on faces with the same host surface, giving rise to the *grouping surface* concept (defined in Section 3.2).

To speed up the detection process, we translate the problem in a simplified one: to

detect symmetric arrangements of repeated entities we first search symmetric arrangements of their corresponding centroids. It is clear that the problem of treating points rather than a set of faces is handier. This consideration arises from the awareness that if a set of repeated entities are symmetrically arranged, then also their centroids do. A further restriction of the search field can be applied by taking into account that a necessary condition for the existence of symmetric arrangement of sub-parts is that the respective centroids must be arranged at a constant distance from each other. The repeated entities can therefore be grouped coherently to this criterion. This last refinement of the input set is implemented by using adjacency matrices, which store the relations between the repeated entities whose centroids are equally spaced according to a fixed distance value (see Subsection 3.2.1).

Starting from the smaller distance value, the algorithm finds *all* the existing paths of equally distant centroids of length at least 3. The paths of centroids we consider can be linear or circular. The method ensures the detection of paths as long as possible (details in Section 3.3).

The found sets of paths is further verified to state the existence of the pattern of the corresponding repeated entities, indeed the well-positioning of the centroids cannot grant the well-positioning of the associated repeated entities. The orientation of the sub-parts need to be verified. It is done by analyzing their vertices and faces, aiming to detect if there exist an isometry that correlates them. If an isometry representing the repetition exists, the pattern is verified and the found transformation characterizes it. If such an isometry covers only parts of the set, then we consider the partial patterns, if their length is at least 3.

Once all the identified candidate patterns are checked, the conceived method intends to analyse the repeated entities not verifying the isometry conditions suggested by their centroid arrangement, and so not belonging to a effective pattern yet. To detect if other relations exist, the method verifies if they can be coupled two by two in translational or

reflectional patterns of length 2, starting with the nearest ones, and then continuing for creasing distance.

The proposed approach focuses on the identification of regular patterns of sub-parts, supposing to receive as input a set of effectively congruent sub-parts. No further information related to their arrangement or to their congruency mappings (intended as mappings expressing how each sub-part can be overlapped to the others) is provided.

This method has many analogies with the one proposed in [17], as both are based on a marked vertices' treatment and use centroids well-positioning as a necessary condition for the existence of a symmetric arrangement. The advantage of this work is the possibility to avoid the time consuming searching of the congruency mappings between repeated sub-parts for the detection of a possible symmetric arrangement, while in [17] they are necessary. Furthermore, symmetries (intended as those transformations that, once applied to a shape, leave it unchanged) of a single sub-part do not need to be computed for the detection of patterns, unlike [17]. This allows to possibly apply this algorithm to data coming from any process of congruent sub-part detection, without demanding further information related to how the congruency has been verified.

The described grouping operations are built to reach the detection of the longest and dominant patterns as soon as possible. To make the process faster, the earlier interruptions of the path search is applied in case of detection of a pattern of maximum length on a grouping surface. In fact, this solution allows switching to the next grouping surface when it becomes clear that no more patterns can be found in the current one. In this way we avoid to wait until the natural end of the process.

The proposed approach gives precedence to the detection of longest patterns, linear then circular. Furthermore, it identifies first patterns with smaller centroid distances, considering the proximity of the centroids a valid signal for the pattern existence. Anyhow, in some situations other patterns can be preferred. For instance, in some cases, it is preferable to consider patterns covering the entire set of repeated entities, instead

of the longer ones but not including all the sub-parts. Thus, a possible extension of the method could be its parametrisation, allowing to indicate which criteria have to be privileged. Another option could be to extract all the possible alternative patterns and let the user choose among them.

Another possible improvement is the possibility of removing the limitedness of the pattern search only within a grouping surface, including also symmetric arrangements of congruent sub-parts lying on faces with different host surfaces.

A further generalization of the implemented procedures could admit also conical, spherical and toroidal faces for the input sub-parts. Freeform surfaces are widely used in industry, so it could be useful an extension of the method to faces defined by this class of surfaces as well. Furthermore, it is significant to extend the class of symmetric arrangements to detect, including for example rotational, glide reflection and screw patterns.

The tolerance level chosen in the proposed approach has been established according to the size of the test CAD models. A possible improvement of the method could involve the automatic adjustment of the tolerance level according to the order of size of the input repeated entities.

Finally, a further relevant aspect for detecting all possibly relevant relationships in the model is the identification of the correlation between the detected patterns. For instance, it would be interesting to put in correspondence two circular translational patterns sharing the same circumference center and lying on the same plane (as in Subsection 4.2.7), or to relate two parallel linear translational patterns of the same length (as in Subsection 4.2.2). This further development involves the analysis of the curves the centroids of the sub-parts lie on.

Bibliography

[1] http://www.grabcad.com/.

[2] http://help.solidworks.com/

[3] http://www.tracepartsonline.net/

[4] S. Ansaldi, L. De Floriani, B. Falcidieno. *An Edge-Face Relational Scheme for Boundary Representations*. Computer Graphics Forum, 4(4). 319-332. The Eurographics Association and Blackwell Publishing. 1985.

[5] M. J. Atallah. *On symmetry detection*. IEEE Transactions on Computers. C-34(7). 633–6. 1985.

[6] T. K. Carne. *Geometry and Groups*. Department of Pure Mathematics and Mathematical Statistics, University of Cambridge. Notes Michaelmas. 2012

[7] G. Caviglia. *Sistemi dinamici e Meccanica analitica*. Università degli Studi di Genova, course notes. 2010.

[8] F. Cazals, M. Pouget. *Estimating differential quantities using polynomial fitting of osculating jets*. Proc. Symp. Geometry Processing. 177–187. 2003.

[9] L. Chiang. *Identification of Patterns of Repeated Parts in Solid Objects*. Thesis, Master Degree in Mathematical Science, University of Genoa. Academic year 2013/2014.

[10] D. Comaniciu, P. Meer. *Mean shift: A robust approach toward feature space analysis*. IEEE Trans. PAMI. 24(5). 603–619. 2002.

[11] T. De Martino, B. Falcidieno, F. Giannini, S. Hassinger, J. Ovtcharova. *Feature-based modeling by integrating design and recognition approaches*. Computer-Aided Design. 26(8). 646 - 653. Modelling in computer graphics. Elsevier. 1994.

[12] J. Gallian. *Contemporary abstract algebra*. D.C. Heath and Company, Lexington, Mass. 1986.

[13] C. Gao, F. Langbein, A. Marshall, R. Martin. *Approximate congruence detection of model features for reverse engineering*. Proc. Int. Conf. Shape Modeling and Applications. 69-77. 2003.

[14] E. W. Hobson. *The Theory of Spherical and Ellipsoidal Harmonics*. Cambridge University Press, Cambridge, UK. 1931.

[15] J. Jiang, Z. Chen, K. He. *A feature-based method of rapidly detecting symmetries in CAD models*. Computer-Aided Design. 45(8-9). 1081–1094. 2013. DOI: 10.1016/j.cad.2013.04.005.

[16] K. Li, G. Foucault, J.-C. Léon, M. Trlin. *Fast global and partial reflective symmetry analyses using boundary surfaces of mechanical components*. Computer-Aided Design. 53. 70–89. 2014. DOI: 10.1016/j.cad.2014.03.005.

[17] M. Li, F. C. Langbein, R. Martin. *Detecting design intent in approximate CAD models using symmetry*. Computer-aided Design. 42(3). 183-201. 2010. DOI: 10.1016/j.cad.2009.10.001.

[18] M. Li, F. Langbein, R. Martin. *Constructing regularity feature trees for solid models*. Proc. Geometric Modeling and Processing, LNCS, Springer. 267-286. 2006.

[19] M. Li, F. Langbein, R. Martin. . *Detecting approximate symmetries of discrete point subsets*. Computer-Aided Design 40(1). 76-93. 2008. DOI: 10.1016/j.cad.2007.06.007.

[20] E. Lockwood, R. Macmillan. *Geometric Symmetry*. Cambridge University Press. 1978.

[21] A. Martinet, C. Soler, N. Holzschuch, F. X. Sillion. *Accurate detection of symmetries in 3D shapes*. ACM Trans. Graph. 25(2). 439-464. DOI: 10.1145/1138450.1138462. 2006.

[22] B. Mills, F. Langbein, A. Marshall, R. Martin. *Approximate symmetry detection for reverse engineering*. Proc. 6th ACM Symp. Solid and Physical Modeling. 241-248. 2001.

[23] N. J. Mitra, M. Pauly, M. Wand, D. Ceylan. *Symmetry in 3D Geometry: Extraction and Applications*. Comput. Graph. Forum 32(6). 1-23. 2013. DOI: 10.1111/cgf.12010.

[24] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, L. J. Guibas. *Discovering Structural Regularity in 3D Geometry*. ACM Transactions on Graphics. 27(3). 2008. DOI:10.1145/1399504.1360642.

[25] J. P. Pernot, B. Falcidieno, F. Giannini, J. C. León. *Incorporating free-form features in aesthetic and engineering product design: State-of-the-art report*. Computers in Industry. 59(6). 626-637. Elsevier. 2008.

[26] H. Pottmann, Q.-X. Huang, Y.-L. Yang, S.-M. Hu. *Geometry and convergence analysis of algorithms for registration of 3D shapes*. Int. J. Computer Vision. 67(3). 277–296. 2006.

[27] J. J. Shah, M. Mantyla. *Parametric and Feature-Based CAD/CAM: Concepts, Techniques, and Applications*. Wyley-Interscience. 1995.

[28] K.Stum. *Design Intent and Basis of Design: Clarification of Terms, Structure, and Use*. ASHRAE Transactions.

[29] H. Zabrodsky, S. Peleg, D. Avnir. *Symmetry as a continuous feature*. IEEE Trans. Patt. Analy. Mach. Intell. 17(12). 1154–1166. 1995.

Recent titles from the IMATI-REPORT Series:

16-01: *Optimal strategies for a time-dependent harvesting problem*, G.M. Coclite, M. Garavello, L.V. Spinolo.

16-02: *A new design for the implementation of isogeometric analysis in Octave and Matlab: GeoPDEs 3.0,* R. Vázquez.

16-03: *Defect detection in nanostructures*, D. Carrera, F. Manganini, G. Boracchi, E. Lanzarone.

16-04: *A study of the state of the art of process planning for additive manufacturing*, M. Livesu, M. Attene, M. Spagnuolo, B. Falcidieno.

16-05: *Calcolo di descrittori ibridi geometria-colore per l'analisi di similarità di forme 3D,* A. Raffo, S. Biasotti.

16-06: *An appointment scheduling framework to balance the production of blood bags from donation*, S. Baş, G. Carello, E. Lanzarone, S. Yalçındağ.

16-07: *From lesson learned to the refactoring of the DRIHM science gateway for hydro-meteorological research,* D. D'Agostino, E. Danovaro, A. Clematis, L. Roverelli, G. Zereik, A. Galizia.

16-08: *Algorithms for the implementation of adaptive isogeometric methods using hierarchical splines*, E.M. Garau, R. Vázquez.

16-09: *Feature curve identification in archaeological fragments using an extension of the Hough transform,* M.L. Torrente, S. Biasotti, B. Falcidieno.

16-10: *Comparing methods for the approximation of rainfall fields in environmental applications,* G. Patané, A. Cerri, V. Skytt, S. Pittaluga, S. Biasotti, D. Sobrero, T. Dokken, M. Spagnuolo.

16-11: *Persistence-based tracking of rainfall field maxima*, S. Biasotti, A.Cerri, S. Pittaluga, D. Sobrero, M. Spagnuolo.

16-12: *Studio e sviluppo di componenti semantiche riusabili per la progettazione di serious game. Applicazioni all'area edutainment,* A. Repetto, C.E. Catalano.

16-13: *Optimal-order isogeometric collocation at Galerkin superconvergent points,* M. Montardini, G. Sangalli, L. Tamellini.

16-14: *Identification of patterns of repeated parts in solid objects*, L. Chiang, F. Giannini, M. Monti.

Istituto di Matematica Applicata e Tecnologie Informatiche ``*Enrico Magenes*", CNR
Via Ferrata 5/a, 27100, Pavia, Italy
Genova Section: Via dei Marini, 6, 16149 Genova, Italy • **Milano Section:** Via E. Bassini, 15, 20133 Milano, Italy

http://www.imati.cnr.it/