







## On the Role of Geometric Constraints to Support Design Intent Communication and Model Reusability

Pedro Company<sup>1</sup> , Ferrán Naya<sup>2</sup> , Manuel Contero<sup>2</sup>  and Jorge D. Camba<sup>3</sup> 

<sup>1</sup>Universitat Jaume I, [pcompany@uji.es](mailto:pcompany@uji.es)

<sup>2</sup> Universitat Politècnica de València, [fernasan@upv.es](mailto:fernasan@upv.es), [mcontero@upv.es](mailto:mcontero@upv.es)

<sup>3</sup> Purdue University, [jdorribo@purdue.edu](mailto:jdorribo@purdue.edu)

Corresponding author: Pedro Company, [pcompany@uji.es](mailto:pcompany@uji.es)

**Abstract.** The assurance of model quality in parametric CAD implies that designers must build models that facilitate reuse while retaining their design intent when modified. In this context, a suitable selection of geometric constraints that operate within and between features is key to produce robust-while-flexible models which are a prerequisite for reusability. This paper introduces a new classification of 3D CAD model constraints that builds on the idea that making the meanings, similarities and differences explicit can result in better usage, making models robust to prevent catastrophic failures when edited as well as flexible enough to enable easy editing.

Results from a preliminary user study designed to validate the significance of the new classification are discussed. Simple exposure to the proposed classification appears insufficient to validate effectiveness. Future work on educational approaches that focus on the challenges of selecting an effective set of constraints for particular reuse scenarios is suggested.

**Keywords:** Geometric constraints, Design intent, Model reuse.

**DOI:** <https://doi.org/10.14733/cadaps.2020.61-76>

### 1 INTRODUCTION

This paper revisits the role of constraints, also called relations or bindings, in history-based feature-based parametric models and its implementation in procedural Mechanical CAD systems (MCAD). Representations are parametric (or constraint-based, or variational) if they are solved as geometric figures by defining and positioning a set of geometric elements with respect to each other so that a set of constraints is satisfied [26]. They are procedural (also known as generative or history-based), when they can be described in terms of a sequence of instructions or "procedures" (which may include the solution to constraint sets).

In our view, the historical development of geometric constraints in 3D CAD models has been primarily driven by increasingly more powerful computational resources to support the constraint solvers [2], [3], [7], [21], while the adoption of a proper theoretical framework has not received

the required attention. This has negatively impacted the effective use of CAD systems, which requires both the acquisition of strategic knowledge such as the selection of effective modeling alternatives and the proper use of constraints to capture design intent [6], [13], [28], [31]. In this context, it is important to note that the problem of constraint application is complex. On one hand, the designer must incorporate as many suitable constraints as needed to produce a robust model that can retain its original design intent when modified. On the other hand, the geometric model should pursue maximum flexibility to “permit design changes yet be sufficiently robust to prohibit undesirable changes that alter topology or violate design intent” [5].

Developing proper constraining skills is not an easy task. In this regard, it has been stated that beginner CAD users do not fully understand the semantic differences between constraints by simply being exposed to those concepts [17]. This could be considered an example of “shallow” CAD learning (as opposed to deep CAD learning), according to the distinction by Menary and Robinson [28]. An improvement in the learning of the constraining activity would support that trainees not only learn the steps required to construct CAD models, but also understand the most appropriate strategy of constructing a model for a particular application.

Current commercial CAD applications usually offer “automatic” constraining capabilities that, at best, let users express their preferences regarding what constraints to apply. This approach is distant from the vision of Anderl and Mendgen, where automatic constraint detection mechanisms “should help the designer formulate the constraints which consistently describe the intention of his design” [3]. A recent survey of 3D CAD model quality failures revealed that the automatic detection mechanisms provided by current interactive modeling editors do not prevent all the morphological errors derived from ineffective modeling strategies [15]. Although modern commercial history-based parametric CAD applications provide mechanisms (or at least information) to effectively manage both under and over-constrained sketch profiles, assurance of design intent is hardly supported by current CAD quality testers [15]. In particular, there is no support to detect and improve fully constrained profiles that use low semantic constraints (i.e. constraints that hardly convey the meaning linked to their functionality or design intent). As an example of the negative effects of low semantic constraints, Gonzalez et al. [16] demonstrated that relations that fix the location of geometric entities relative to the reference system (anchor constraints) are inefficient, as they produce valid but inflexible profiles that reduce model reusability.

The problem becomes even more challenging when interoperability forces model exchange by translating product data representation into a different format, as constraints information is lost in the process. Currently, no commercial CAD application exists to support the functionality described by the ISO 10303 application protocol 242 [22] to manage parametric and constrained geometry models.

Furthermore, there is a lack of understanding of the theoretical basis required to build robust-while-flexible CAD models which is a necessary prerequisite for model reuse [31]. Reusability and design intent may require conflicting strategies [14], [29]. It has been recently stated that design intent may be embedded at three different levels: 1) sketch constraints, 2) relationships between modeling operations, and 3) modeling operations themselves [29]. Adding constraints to a sketch may be fast and reliable, but hides the design intent within the sketch, whose internal contents are not directly visible when inspecting the model’s design tree. However, adding constraints at the 3D level can arguably be more limiting than the 2D alternative, since the available constraints are usually limited to bilateral symmetry and rectangular and polar replication patterns.

In order to contribute to the improvement of modeling strategies, this paper introduces a new classification of 3D CAD model constraints that builds on the idea that making the meaning (or “semantics”) of the constraints—as well as their similarities and differences—explicit will result in improved design intent communication. This improvement will ultimately have a significant positive impact on the overall quality of parametric CAD models (particularly in terms of robustness) so catastrophic failures can be minimized when a model is edited, but provide sufficient flexibility to accommodate variations without losing the original design intent.

The paper is organized as follows: first, the scope of the study is described. Then, a review of the classifications of geometric constraints proposed by different authors is provided. Next, we introduce a new classification that is consistent with the existing literature but emphasizes the different types of design intent that constraints convey. Finally, we present the results of an experiment that suggest that CAD trainees comprehend the differences between constraints only partially and discuss how the new classification can help novice users to become more conscious about the differences and similarities between constraints.

## 2 SCOPE

Engineering constraints can be defined as those “relating geometric and other product properties, like material properties or technology and manufacturing properties” [3]. Geometric constraints operate in procedural MCAD systems at three complementary levels: within features, between features and between parts [4]. In the context of our paper, feature is used as a generic term, encompassing from simple modeling operations (also called form features) up to design and manufacturing features (functional features, as defined by Shah and Mantyla [32]). Engineering and geometric constraints are relevant at the detailed design level, as opposed to the specifications that act as requirements at the higher conceptual design level [33]. This paper focuses exclusively on geometric constraints.

Geometric constraints can operate inside parts or between parts. Constraints that operate inside parts are mainly embedded within the profiles used to produce 3D swept shapes. Constraints between parts (also called mating conditions or mates) can be hierarchical or relational. The first “can be modeled as relationships between sets of points (markers) on different bodies, which constrain the rigid bodies to which they are attached” [24]. Alternatively, relational constraints are used to establish relative placements between parts to replicate physical conditions (like the joints used by Autodesk®), design functionalities (by implementing affordances in the assembly process, such as a flap that fits into a groove [29]), or assembly methods (e.g. joining, fastening, bonding, etc., as described by [1]). Constraints that operate between parts are out of the scope of the present study. Additionally, this work does not consider the Semantic Feature Modelling (SFM) paradigm (where “all properties of features, including their geometric parameters and validity conditions, are declared by means of constraints” [9]) or 3D constraints (which were reported as an open problem by Chen [12]). We assume that interaction between features is mainly due to the effect of some constraints that describe validity conditions of the features [8]. Therefore, the so-called attach constraints [9] are also considered here, as they describe where and how features are attached to each other. To a certain extent, they are responsible for the parent-child relationships in the model trees. In the words of Anderl and Mendgen “the first are defined or detected in sketching mode while the second may also be defined as attributes of the geometry generating function” [4]. Attach constraints are considered here only as long as they are introduced into the sketches to link them to previous sketches of features. The problems derived from mutual relationships between constraint-structures in CAD models—which were extensively described by Anderl and Mendgen [4] and continue to be open problems are not considered in this paper.

We assume the so-called constructive approach (where the user adds constraints while defining sketches or features [3]). Automatically detected constraints (rule-based approach) make no difference, provided that the automatic detector preserves the design intent (which is not as frequent as it should, since the rules tend to prioritize constraints that refer to the position and orientation of sketched elements relative to a given coordinate system, instead of relative to one another). Otherwise, the user should be allowed to edit the detected constraints, since “the detected constraints (like parallelism or perpendicularity of lines) may be correct only for the current instance of the model and not for any of its intended variants that may be generated by parameter variation” [4].

Finally, and being consistent with Aldefed [2], the classical distinction between explicit and implicit constraints is not considered, as implicit constraints (those that are not declared because they are assumed to be necessarily perceived by the user) make no sense in parametric CAD. However, we note that some authors use the term “implicit” to refer to structural, associative or geometric constraints [25], [26]. Likewise, other authors use explicit or implicit for constraints that directly or indirectly relate two separate items, respectively. For instance, two lines are said to be explicitly constrained as parallel, or implicitly made parallel by making them share the same direction [33]. Naturally, we assume the direct approach as it is the most appropriate for conveying design intent.

### 3 RELATED WORK

The explicit use of constraints can be traced back to Hilbert and what the author called relations [20]: “We think of these points, straight lines, and planes as having certain mutual relations, which we indicate by means of such words as ‘are situated,’ ‘between,’ ‘parallel,’ ‘congruent,’ ‘continuous,’ etc.” To discuss the relations systematically, Hilbert developed five groups of axioms: connection, order, parallels, congruence and continuity. As an example, the plane axioms of group connection define the coincidence constraint between point and line, as “Two distinct points A and B always completely determine a straight line a,” or “a goes through A and through B.”

Since the beginning of CAD applications, Sutherland defined constraints as basic relationships between picture parts built into the system. This definition implies that constraints must be explicit and that the drawing is constraint driven. According to Sutherland, “The major feature which distinguishes a Sketchpad drawing from a paper and pencil drawing is the user's ability to specify to Sketchpad mathematical conditions on already drawn parts of his drawing which will be automatically satisfied by the computer to make the drawing take the exact shape desired”. [34]. Other relevant authors contributed to express constraints by predicates [11].

The key role of constraints in the process of CAD modeling has been highlighted by other authors such as Anderl and Mengen, who described the architecture of constraint-based CAD systems as consisting of “the hybrid modular for geometry and topology on the one hand and of the components for modelling and solving the constraints on the other hand” [3]. Shih and Anderson [33] defined constraints as “limitations on an item’s permissible values within itself or with respect to other elements in the design.”

Constraints behave differently within the CAD model depending on its type. Aldefeld [2] distinguished between structural and metric constraints. Structural constraints refer to topological and other relationships that “cannot reasonably be modified continuously, as is the case, for example, with point-track incidence, parallelism, tangency, and symmetry.” Conversely, metric constraints fix coordinates, lengths, or angles formally described by numbers, or variables, ranging over numerical intervals. The use of dimensions as constraints can be traced back to the work by Gossard et al. [18], [25]. The most common practice in the use of constraint-based CAD tools still distinguishes between two consecutive steps in the modeling procedure: 1) adding constraints/relations, and 2) adding dimensions [10], [19], [31]. Some authors assume that once a fix subset of constraints is determined, a subset of variable constraints can be used to find a good design, or a family of objects. They also assume that most (if not all) fix constraints are associative, while the metric ones drive the changes [30].

Multiple classifications for constraints have been proposed. Mantyla and Shah classified modeling constraints in four groups: ground, dimensional, geometric, and algebraic [27]. Ground constraints provide references between the part and the global coordinate system. A geometric constraint imposes relationships between geometric entities such as tangency, collinearity, parallelism, perpendicularity, coincidence of points, symmetry, etc. Dimensional constraints may be used to indicate the size of specific entities or the relative location between different entities. Algebraic constraints impose restrictions on design dimensions in the form of mathematical equations. Ault [5] also supported the classification by Mantyla and Shah. Shih and Anderson [33]

distinguished different types of limitations on an item's permissible values, such as size, shape, position, orientation, dimension and logical relationships (such as identical, distinct, and set relation). Klein [23] defined five categories of constraints: (1) geometric and arithmetic constraints, (2) Topological conditions, (3) Type constraints, (4) Existence requirements, and (5) Relational conditions. Geometric constraints include tangentiality or orthogonality, incidence relations, etc. Topological relations include adjacency, incidence or inside relations, and topological connectedness or disconnectedness.

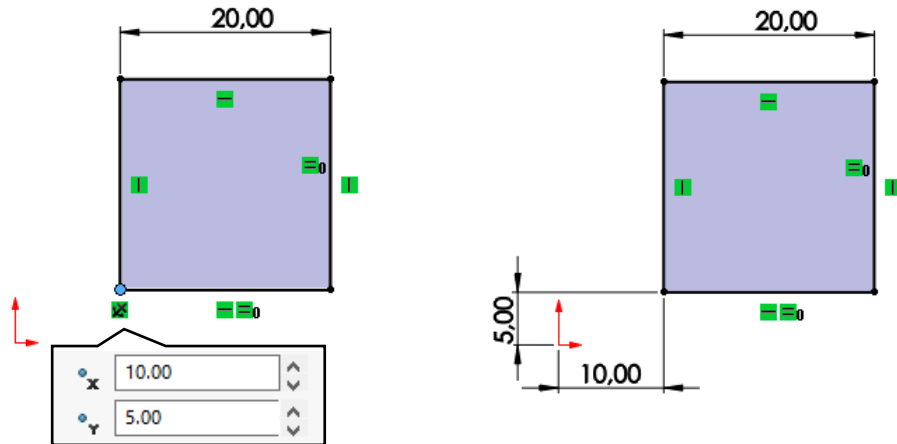
Most of the above classifications were mainly developed from the point of view of geometry and/or computer science. Consequently, they do not put the focus on conveying design intent to enhance model quality. As an exception, we can cite an early approach by Shih and Anderson [33], who described three effects of constraints on design: (1) as restrictions and limitations on parameters that define the design, (2) as freedom for modification of the design, and (3) as a design generating tool subject to #1 and #2.

In this paper, we propose a new classification of geometric constraints that is consistent with the existing literature and takes into consideration the quality and reusability effects of the constraints on the end model.

#### **4 A NEW CLASSIFICATION OF CONSTRAINTS**

Instead of the multiple peer types of constraints proposed by some authors, we advocate for combining two main criteria. First, we distinguish between discrete (or digital) and continuous (or analog) constraints, which are then further subdivided into intrinsic and extrinsic. In our view, the distinction between structural and metric [2] (also called associative and metric) is still the primary classification for constraints. Yet, the constraints are renamed for two reasons: (1) to emphasize their different behaviors, as structural constraints act in a binary manner (active/inactive), while metric constraints vary continuously within a range, and (2) to clarify their differences, as most CAD users tend to classify the constraints according to the symbols used to depict them. Users clearly distinguish between constraints depicted by dimensions and constraints depicted by symbols that are similar to one another (such as those represented by small icons enclosed in squares in many commercial CAD packages). A simple example is the fix constraint, typically represented with an anchor symbol, which conveys a metric constraint that could be continuously modified just like dimensions. In other words, although this constraint could be managed as a dimension, as opposed to an associative constraint, it is "perceived" as a binary mechanism (fixed/unfixed) in which the variability of its coordinates is typically unused. For example, the location of the square in Figure 1-left is perceived as fixed, while the square in Figure 1-right is perceived as variable, despite being both equally variable. Consequently, classifications of constraints should not ignore the semantics of the symbols used to represent them, because they convey a particular meaning that can be misconstrued.

The distinction between discrete (associative) and continuous (metric) constraints is based on computational reasons. Primarily, the two types of constraints are better solved separately, as solving discrete constraints may result in algebraic singularities that are different from those that may arise from solving continuous ones. In addition, for practical reasons, it is usually a good habit to add associative constraints before adding metric ones. However, there are also conceptual reasons that need to be considered, since discrete constraints are usually related to permanent—or, at least, more restrictive—design criteria. In this regard, it is recommended to avoid the use of continuous constraints when a discrete equivalent exists. For example, using an angular dimension of 90 degrees is usually interpreted as a variable constraint that occasionally equals to 90 degrees. However, using a discrete constraint of perpendicularity conveys the clear design intent of maintaining orthogonality in future design changes.



**Figure 1:** Fixed vs. variable perception of fixed constraint.

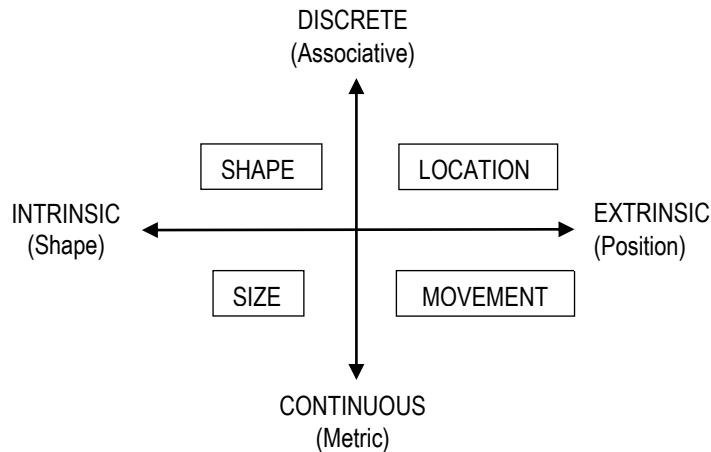
Our second subdivision derives from the classification by Mantyla and Shah [27] which, in addition to distinguishing between associative and metric constraints, also differentiates between constraints that relate geometrical components of the figure to each other (intrinsic), from constraints that relate the figure to the scene (extrinsic). We use the terms shape vs. position to emphasize this distinction. There are practical reasons to distinguish between these types of constraints, as designers typically prefer to control the shape of an object separately from its position. There is also a theoretical background, since geometrical transformations traditionally distinguish between those that maintain the rigid-body properties by producing movements (rigid transformations), from those that deform the original shape or size. Therefore, making this distinction explicit facilitates the perception of the invariances, i.e. the properties that will not change when the model is edited. To note that extrinsic constraints are typically lost when a profile is duplicated (copy/paste), while intrinsic ones are retained.

In our classification (shown in Figure 2), we define four main types of constraints: shape, size, location and movement. These four types are organized around two orthogonal axes to visualize their suitability. For example, the proper way to fix the shape is by using constraints that are simultaneously discrete and intrinsic (such as parallelism, perpendicularity, tangency, etc.), whereas fixing the size requires intrinsic-continuous constraints (such as dimensions that fix the magnitude of the elements of the shape). Alternatively, constraining the location or the movement requires external references, which requires the use of extrinsic constraints, such as the discrete constraints that locate lines parallel to the references axes, or dimensions that continuously fix the distance between the figure and the reference system.

The anchor constraint used in Figure 1-left is an example of location (as it is “perceived” as discrete), whereas the two dimensions that fix the distance from the lower left vertex of the Figure 1-right up to the origin convey the idea of movement.

The movement of rigid figures includes displacement/translation and rotation, which are respectively controlled by two dimensions that fix the position of one point in the rigid figure, plus one angle that controls the rotation of the figure as a whole. Note that plane figures are geometrically congruent if they can overlap after being transformed by a movement (or isometry, which includes translations and rotations). Since congruence is similar to equality of numbers, it is sometimes referred to as such. Likewise, figures that require a mirror or specular symmetry to overlap are not congruent (unless we distinguish between direct and inverse congruence). Outside the sketch plane, congruence allows to distinguish between plane figures located in, or viewed from, either side of the plane.





**Figure 2:** Classification of constraints.

















Distinctions like dimensional vs. algebraic constraints (valuated vs. symbolic in [21]) are considered secondary in our classification, since both are metric and may be intrinsic or extrinsic depending on the elements they relate [see the bottom half of Figure 2]). Therefore, the difference is on the way the changes they produce are controlled. A valuated dimension conveys the idea of an independent constraint, whereas a parametrized dimension denotes an external relationship or control.

Mapping the common constraints available in parametric CAD applications into the framework described in Figure 2 is complex. First, each CAD application provides its own set of constraints, which are not always equivalent to those in other applications. As an example, the set of basic constraints provided by SolidWorks® are shown in Figure 3. Second, and more importantly, the same constraint may serve different purposes depending on the elements it connects (see explanation of dimension K in Figure 4 for an example). Furthermore, changes may propagate in a way that “even changes to dimension values can entail substantial topological changes” [12].

Despite the difficulty in defining a general taxonomy of constraints, the above classification is important, since there are common bad habits that users could avoid:

- CAD users tend to distinguish between shape and size, but fail to avoid common modeling mistakes "
- CAD users hardly pay attention to the distinction between location and movement.
- CAD users only pay attention to the distinction between intrinsic and extrinsic constraints when they are unforgettably aware of the need of facilitating future changes in shape that do not affect position, or vice versa.

The proposed classification is intended to emphasize the selection of the most suitable set of constraints for a particular design intent. If the shape of a model is fixed but the size needs to change to produce a family of parts, then the distinction between shape and size becomes key. Alternatively, if the shape and size of the parts are fixed but the parts need to experience transformations (for instance, to assemble them), then the focus must switch to control movements and location. Nevertheless, the classification should not be understood as a rigid framework where the different types of constraints must fit. The constraints are defined based on the geometric effect they produce whereas the classification is aimed at conveying the design intent.

Name	Icon	Relation
Merge Points		Two sketch points or endpoints merge into a single point.
Coincident		One point, or endpoint, lies on a line-segment, arc, or ellipse.
Intersection		One point remains at the intersection of two lines.
Parallel		Two or more lines are parallel to each other.
Perpendicular		Two lines are perpendicular to each other.
Tangent		An arc, ellipse, or spline, and a line or arc remain tangent.
Horizontal or Vertical		One or more lines or two or more points become respectively parallel to the horizontal or vertical axes, or aligned horizontally or vertically.
Fix		The size and location of any entity are fixed. However, the endpoints of a fixed line, arc or elliptical segment are free to move.
Midpoint		One point remains at the midpoint of a line-segment. Two line-segments share their centerpoints.
Collinear		Two or more line-segments lie on the same infinite line.
Concentric		Two or more arcs share the same centerpoint, or a point shares the centerpoint of an arc.
Coradial		Two or more arcs share the same centerpoint and radius.
Equal		The line lengths of two or more line-segments remain equal. Or two or more arcs radii remain equal.
Equal Curvature		The radius of curvature and the vector (direction) matches between the two splines
Symmetric		Two items of the same type (points, lines, arcs, or ellipses) remain equidistant from the centerline, on a line perpendicular to the centerline.
Dimension		Fixes the size of an element, or the distance/angle between two elements.

**Figure 3:** Common sketch constraints in DS Solidworks®.

## 5 EXPERIMENTAL VALIDATION

As part of this work, we designed an experiment to determine whether the proposed classification and the introduction of the concept, types and usefulness of constraints can improve the adoption of suitable constraining strategies by CAD trainees.

A group of 35 undergraduate CAD students (all mechanical engineering majors) and 15 graduate students in Industrial Design and Manufacturing were lectured on the importance of fully constraining profiles to favor model reusability. They were also exposed to the different types of constraints, emphasizing the differences between associative and metric, and between intrinsic and extrinsic. No explicit references were made to the relationship between the two classifications as shown in Figure 2.

Participants were required to complete four tasks. First, they were asked to briefly explain the different types of constraints and highlight their differences and similarities. The responses were ranked by the instructor as null, poor, fair, good, or excellent as shown in the "General knowledge about types of constraints" column in Table 1 (undergraduates) and Table 2 (graduates). As expected, results revealed that the theoretical differences between constraints are not fully understood after a simple exposure to those concepts (the average rank was just fair).



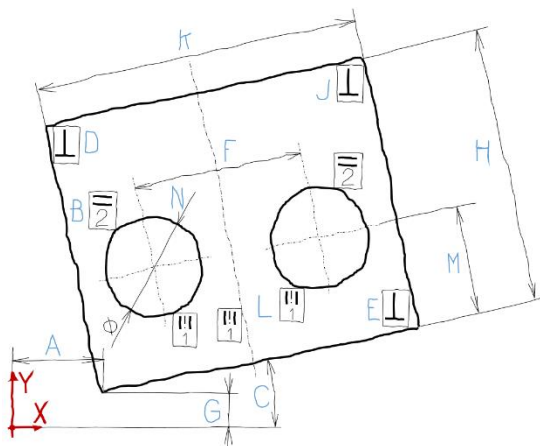
SUBJECT	General knowledge about types of constraints	Under-Over constrained	Equality B is redundant	Symmetry axis not constrained	One perpendicular is redundant (D.E., J)	Symmetry is redundant	Fourth perpendicular is missing	Symmetric distance F is redundant	Fixin one vertex at origin is missing	A	B	C	D	E	F	G	H	J	K	L	M	N	∅	Size can change	Location can change	Rotation can change
1	Good	U								P	F	P	F	F	P	P	T	F	T	F	T	T		No	Removing 1	No
2	Good	O			*					P	F	P	F	F	T	P	T	F	T	F	T	T		Yes	No	No
3	Poor	O		*						P	F	P	F	F		P	T	F	T	O	T	T		Yes	Floating	No
4	Fair	O		*						P	F	P	F	F	T	P	T	F	T	F	T	T		No	No	No
5	Poor	O	*	*						P	P	F	F	P	P	T		T	O	P	T			Yes	Removing 2	No
6	Good	O		*						P	F	P	F	F	P	P	T	F	T	P	P	T		Yes	Removing 1	No
7	Fair	O					*			P	F	P	F	F	F	P	T	F	T	F	T			Yes	Removing 1	No
8	Good	U	*							P	T	P	F	F	P	P	T	F	T	P	P	F		Yes	Removing 1	No
9	Good	O	*							P	T	P	F	F	P	P	T	F	T	T/P	P	T		Yes	Removing 1	No
10	Poor	O	*							P	O	P	F	F	P	P	T	F	T	O	P	T		Yes	Yes	Yes
11	Null	F								P	F	P	F	F	P	P	T	F	T	F	P	T		Yes	Removing 2	No
12	Excellent	U	*							P	T/F	P	F	F	T	P	T	F	T	F	T	T		Yes	Removing 1	No
13	Null	O		*						P	F	P	F	F	T	P	T	F	T	O	T	T		No	No	No
14	Fair	F								P	F	P	P	P	P	P	T	P	T	P/C	P	F/T		Yes	Removing 2	No
15	Good	O	*							P	T	P	F	F	F	P	T	F	T	O	T	T		No	Removing 1	No
16	Fair	O			*					P	F	P	F	F	T	P	T	F	T	O	T	T		Yes	Removing 1	No
17	Poor	O	*	*						P	O	P	F	F	P	P	T	F	T	O	P	T		Yes	Removing 1	No
18	Good	F								P	F	P	F	F	F	P	T	F	T	F	F	T		Yes	No	No
19	Excellent	O	*							P	F	P	F	F	F	P	T	F	T	F	F	T		Yes	Yes	Yes
20	Good	F								P	F	P	F	F	F	P	T	F	T	F	F	T		Yes	Yes	Yes
21	Fair	U				*		*		P	F	P	F	F	T	P	T	F	T	F	T	T		Yes	Removing 1	No
22	Good	O	*							P	F	P	F	F	P	P	P	F	P	F	P	P		No	No	No
23	Fair	O	*	*						P		P	F	F	P	P	T		T	O	P	T	T	Yes	No	No
24	Null	O		*						P	F	P	F	F	O	P	T	F	T	F	O	T		Yes	No	No
25	Poor	O		*						P		P	F	F		P	T	F	T	F				Yes	Removing 1	No
26	Null	F								P	F	P	F	F	T	P	T	F	T	O	T	T		Yes	Yes	No
27	Fair	F								P	T	P	F	F	T	P	T	F	T	L	T	T		Yes	Removing 1	No
28	Poor	U				*					T		F	F				F		P				No	No	No
29	Null	F								P	F	P	F	F	T	P	T	F	T	F	T	F		Yes	Removing 1	No
30	Fair	O	*							P		P	F	F	P	P	T	F	T	F	T	T		Yes	No	No
31	Good	O	*							P		P	F	F	T	P	T	F	T	F	T	T		Yes	Removing 1	No
32	Fair	O	*							P	F	P	F	F	F	P	T	F	T	F	F	T		No	No	No
33	Fair	F								P	T	P	F	F	T	P	T	F	T	O	T	T		No	No	No
34	Fair	O	*							P	T	P	F	F	F	P	T	F	T		F	T		Yes	Removing 1	No
35	Excellent	O	*							P	O	P	F	F	O	P	T	F	T	O	O	O		Yes	Removing 1	No

**Table 1:** Summary of participants' responses (undergraduate).

SUBJECT	General knowledge about types of constraints	Under-Over constrained	Equality B is redundant	Symmetry axis not constrained	One perpendicular is redundant (D, E, J)	Symmetry is redundant	Fourth perpendicular is missing	Symmetric distance F is redundant	Fix in one vertex at origin is missing	A	B	C	D	E	F	G	H	J	K	L	M	N	Ø	Size can change	Location can change	Rotation can change
1	Good	O	*							P	O	P	F	F	P	P	T	F	T	P	P	T		Yes	Removing 1	Floating
2	Fair	O	*	*						P	O	P	F	F	P	P	T	F	T	O	P	T		Yes	Removing 2	No
3	Good	O	*							P	F	P	F	F	P	P	T	F	T	F	P	T		Yes	Yes	Yes
4	Fair	O		*						P	F	P	F	F	P	P	T	F	T	P	P	T		Yes	Removing 3	No
5	Good	U	*							P	F	P	F	F	O	P	T	F	T	O	O	O		Yes	Removing 2	No
6	Fair	O		*							T		F	F				F		F				Yes	Removing 1	No
7	Excellent	U	*							P	T		F	F	P	P	T	F	T	P	P	T		Yes	Removing 1	Removing 2
8	Good	U	*							P	T	P	F/P	F/P	P	P	T	F/P	T	P	P	T		Yes	Yes	Yes
9	Fair	F								P	F	P	F	F	T	P	T	F	T	F	T	T		Yes	Removing 1	Removing 1
10	Good	U	*							P	O	P	F	F	T	P	T	F	T	O	T	T		Yes	Yes	Yes
11	Good	O	*	*						P	F	T/P	F	F	P	P	T	F		F	P	T		Yes	No	No
12	Good	O	*							P	F	P	F	F	T	P	T	F	T	F	T	T		Yes	Yes	Removing 1
13	Good	O	*							P	O	P	F	F	P	P	T	F	T	F	P	T		Yes	Removing 1	No
14	Poor	O			*					P	O	P	F	F	F	P	T	F	T	O	F	T		Yes	Removing 1	Yes
15	Good	U	*							P	F	P	F	F	T	P	T	F	T	F	T	T		No	No	Yes

**Table 2:** Summary of participants’ responses (graduate).

Next, students were asked to analyze a sketch of a simple shape (Figure 4) to determine whether the sketch represented a fully constrained, over-constrained or under-constrained profile. The use of a computer to test and solve the problem experimentally was not allowed. Participants were asked to explain and justify their answers. Explanations about the meaning of the symbols were provided: the two axes labelled X and Y represent the local coordinate system, and the symbols enclosed within rectangles act on the elements they are in contact with, representing symmetry, equality or perpendicularity constraints. It can be easily verified that the profile is both over-constrained (the equal constraint in the two circles is redundant with their symmetry condition) and under-constrained (the symmetry axis is not fixed to the midpoints of the long sides of the rectangle). A fully constrained solution is shown in Figure 5.



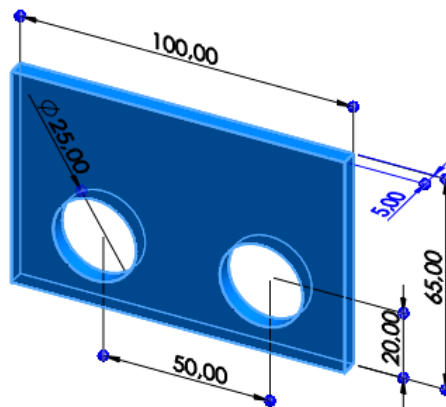
**Figure 4:** Under/over constrained profile.



column ( $\emptyset$ ) was included to report responses from participants who apparently misinterpreted the diameter symbol  $\emptyset$  in the figures as another label.

For undergraduates, our results show that most constraints are always, or almost always, perceived in the same way by the participants (A, C, D, E, G, H, J, K, N), but others are unclear for many (B, F, L, M). More specifically, position constraints represented by dimensions are visibly perceived as such (A, C, G). Perpendicularity constraints (D, E, J) are consistently perceived as describing shape. Finally, dimensions that describe the plate and the holes (H, K, N) are clearly perceived as size constraints (although N was perceived only by 27 subjects). Similarly, graduate students consistently perceived some constraints as belonging to a certain category (A, C, D, E, G, H, J, K, N), while others remained dubious (B, F, L, M). The only significant difference with the undergraduate student group was that constraint N was more often identified as a size constraint.

Finally, we asked participants to model the shape depicted in Figure 6 (dimensions were in mm and symmetry, perpendicularity and equality were conveyed implicitly). Participants were explicitly instructed to produce a 3D model by extruding a single profile that could be easily modified by independently changing its size, position and orientation.



**Figure 6:** Plate to model.

The three rightmost columns of Table 1 (for undergraduates) and Table 2 (for graduates) summarize the participants' performance. In the undergraduate student group, a total of 27 students submitted a model whose size could be easily altered as expected. Only 4 models could be displaced (although 20 participants constrained the shape in a way that could be easily unconstrained to allow for free movement). However, only 3 models could be easily rotated.

The graduate student group performed slightly better than novices. Nevertheless, significant difference exists between size, and translation and orientation. 14 graduates submitted a model whose size could be predictably changed, but only 4 models could be directly displaced (although 10 students constrained the shape in a way that could be easily unconstrained to allow for free movement) and 5 models could be easily rotated (5 models could be rotated after removing one or two constraints).

We conclude that novice CAD users lack the expertise required to produce models that are easily editable, even if they are committed to produce such models. In the context of constraint application, a clear difference exists between size (which seems it can be controlled by size dimensions in a "natural" way), and translation and orientation (location vs. movement constraints). This mental effort was particularly difficult in the case of rotation.

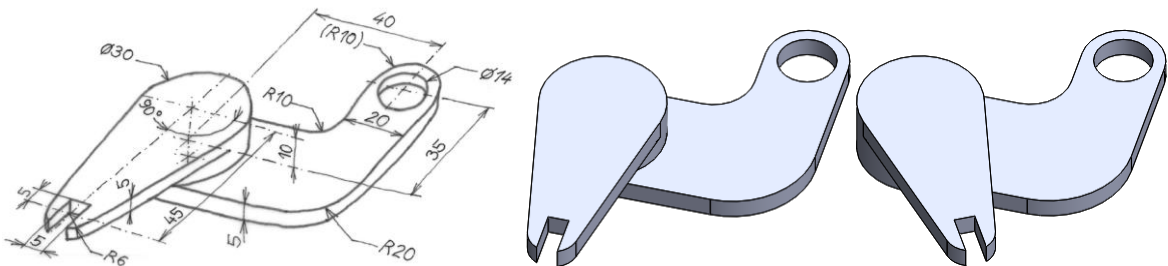
In a complementary experiment to determine whether the lack of practical training in using the classification was critical, a different group of 37 undergraduate engineering students enrolled

in a CAD course repeated the first two tasks. However, this new group had previously analyzed the fully constrained version of the sketch shown in Figure 5 during instruction. In addition, participants in this group were not asked to complete the third task, since they were already familiar with the exercise before they took part in the experiment (it was assigned to them as part of the course and the procedure on how to solve it was available to them in the online learning platform used to support the course). Despite the practical exposure to the classification in general, and the problem in particular, results (shown in Table 3) were consistent with those shown on the left side of Table 1.

SUBJECT	General knowledge about types of constraints	Under-Over constrained	Equality B is redundant	One perpendicular is redundant (D,E,-J)	Constraining the symmetry axis is missing	Smmetry is redundant	Symetric distance F is redundant	Fourth perpendicular is missing	Fixin one vertex at origin is missing	Symmetry is overconstrained	Other, incomplete expla
1	Poor	O	*								
2	Fair	O	*								
3	Fair	-									
4	Excellent	U					*			*	The subject fails to understand that a dimension between lines implies its parallelism
5	Null	O	*								
6	Fair	U		*							
7	Good	-									
8	Good	O	*								
9	Poor	O								*	Poor explanation
10	Excellent	O	*								
11	Fair	O	*								
12	Fair	O	*								
13	Fair	U								*	Circular holes are underconstrained
14	Null	-									
15	Fair	O	*								
16	Null	O	*								
17	Good	O	*								
18	Fair	O	*	*							
19	Good	O							*		
20	Fair	-									
21	Excellent	O		*							
22	Fair	O	*								
23	Null	O		*							
24	Excellent	-									
25	Poor	O	*								
26	Good	O	*							*	Dimension F is redundant
27	Poor	O	*								
28	Excellent	-									
29	Fair	O	*								
30	Good	-									
31	Fair	O	*								
32	Null	O							*		
33	Poor	U		*							
34	Good	-									
35	Fair	-									
36	Null	O							*		
37	Poor	-									

**Table 3:** Summary of participants’ responses (undergraduate engineering students with practical exposure to the classification).

With our current data, we speculate that neither simple exposure to the theoretical aspects of the proposed classification nor the practice of such concepts in generic modeling tasks appear to be sufficiently effective. Therefore, as future development, we suggest exploring new teaching approaches that focus on redesigning models, thus forcing the selection of an effective set of constraints for particular reuse scenarios. For instance, modeling the lever shown in Figure 7 is a simple task, where the type of constraints selected do not seem to be critical. However, asking the trainees to produce a redesigned version of the lever with the upper arm rotated at an angle different from 90° would probably require revisiting the sketch of the upper arm to detect and remove inappropriate location constraints (such as horizontal and vertical ones), which should be replaced by constraints that keep the intrinsic shape and size, while allow the rotation (such as local symmetry related to an axis controlled by and angular dimension).



**Figure 7:** Example of model redesign to force suitable selection of constraints.

## 6 CONCLUSIONS

CAD users are usually encouraged to fully constraint their models in order to ensure robustness. To this end, commercial CAD applications include tools that can help users determine whether their models are fully constrained. These mechanisms are typically based on simple metrics that compare degrees-of-freedom (DOFs) in the sketch against the constraints. Additionally, most CAD packages can also constrain CAD models automatically (often not efficiently) and interactively show the user how the DOFs act on specific geometric elements of the model. However, current CAD systems provide no support to help the user to express properly the design intent through a meaningful set of constraints. For example, the system should give warnings if a symmetric profile is created but no symmetry constraint is applied. All this means, that more intelligent parametric editors should be developed to make easier the constraining tasks to users.

Regarding reusability, parametric CAD models must also be flexible to allow for model reuse. However, flexibility does not only depend on the amount of constraints, but on their semantic level, i.e. the model's capability to convey design intent. In this regard, a review of the relevant literature revealed that constraints have never been characterized according to their semantic level. In this paper, we proposed a new classification of geometric constraints, which distinguishes between discrete and continuous constraints, acting at intrinsic or extrinsic levels. This results in four main types of constraints: shape, size, location and movement.

An experimental study was designed to demonstrate the significance of the new classification. Our study showed that a simple exposure to the proposed classification is not effective, but it seems to be different when users are faced with the challenge of selecting an effective set of constraints for particular reuse scenarios. Evidences in favor of alternative teaching strategies are not conclusive, but they are sufficiently strong to recommend exploring the latter instead of refining the validity of the classification through the former.

Additionally, our results revealed that novice CAD users tend to classify constraints based on the appearance of the symbols used to represent them instead of their semantic level. This is true even after being exposed to the theoretical foundations of design intent, its importance, and how it

is conveyed by constraints. Therefore, a natural next step of this study is the development of new training strategies that emphasize the differences and similarities between the various types of constraints and their usefulness to convey design intent and facilitate model editing and reuse.

## 7 ACKNOWLEDGEMENTS

This work was partially supported by the Spanish grant DPI2017-84526-R (MINECO/AEI/FEDER, UE), project "CAL-MBE, Implementation and validation of a theoretical CAD quality model in a Model-Based Enterprise (MBE) context."

## 8 ORCID

*Pedro Company*, <http://orcid.org/0000-0001-6399-4717>

*Ferrán Naya*, <http://orcid.org/0000-0002-6135-3024>

*Manuel Contero*, <http://orcid.org/0000-0002-6081-9988>

*Jorge D. Camba*, <http://orcid.org/0000-0001-5384-3253>

## REFERENCES

- [1] Abdullah, T. A.; Popplewell, K.; Page, C. J.: A review of the support tools for the process of assembly method selection and assembly planning, *International Journal of Production Research*, 41(11), 2003, 2391-2410. <https://doi.org/10.1080/002075431000087265>.
- [2] Aldefeld, B.: Variation of geometries based on a geometric-reasoning method. *Computer-aided Design*, 20(3), 1988, 117-126.
- [3] Anderl, R.; Mendgen, R.: (1996) Modelling with constraints: theoretical foundation and application, *Computer-Aided Design*, 28(3), 1996, 155-168.
- [4] Anderl, R.; Mendgen, R.: Analyzing and Optimizing Constraint-Structures in Complex Parametric CAD Models, in: Brüderlin, B.; Roller, D.: (eds) *Geometric Constraint Solving and Applications*. Springer, Berlin, Heidelberg, 1998, 58-81
- [5] Ault, H. K.: Using Geometric Constraints to Capture Design Intent. *Journal for Geometry and Graphics*, 3(1), 1999, 39-45.
- [6] Ault, H. K.: Solid Modeling Strategies – Analyzing Student Choices, 121st ASEE Annual Conference & Exposition. Paper ID #9242, 2014.
- [7] Bettig, B.; Hoffmann, C. M.: Geometric Constraint Solving in Parametric Computer-Aided Design, *Journal of Computing and Information Science in Engineering*, 11(2), 2011, 1-9. <https://doi.org/10.1115/1.3593408>.
- [8] Bidarra, R.; Bronsvoort, W. F.: Towards classification and automatic detection of feature interactions. *Proceedings of 29th International Symposium on Automotive Technology and Automation; Dedicated Conference on Mechatronics - Advanced Development Methods and Systems for Automotive Products*, 1996, 99–108.
- [9] Bidarra, R.; Bronsvoort, W. F.: Semantic feature modelling, *Computer-Aided Design* 32, 2000, 201–225. [https://doi.org/10.1016/S0010-4485\(99\)00090-1](https://doi.org/10.1016/S0010-4485(99)00090-1).
- [10] Bodein, Y.; Rose, B.; Caillaud, E.: Explicit reference modeling methodology in parametric CAD system, *Computers in Industry*, 65, 2014, 136–147. <https://doi.org/10.1016/j.compind.2013.08.004>.
- [11] Brüderlin, B.: Constructing Three-Dimensional Geometric Objects Defined by Constraints, *Proceeding I3D '86, Proceedings of the 1986 workshop on Interactive 3D graphics*, 1984, 111-129.
- [12] Chen, X.: Representation, evaluation and editing of feature-based and constraint-based design, PhD. Thesis Purdue University, 1995.
- [13] Chester, I.: Teaching for CAD expertise, *Int J Technol Des Educ*, 17, 2007, 23–35. <https://doi.org/10.1007/s10798-006-9015-z>.



- [14] Company, P.; Contero, M.; Otey, J.; Plumed, R.: Approach for developing coordinated rubrics to convey quality criteria in MCAD training, *Computer-Aided Design*, 63, 2015, 101–117. <https://doi.org/10.1016/j.cad.2014.10.001>.
- [15] González-Lluch, C.; Company, P.; Contero, M.; Camba, J. D.; Plumed, R.: A Survey on 3D CAD Model Quality Assurance and Testing Tools, *Computer-Aided Design*, 83, 2017, 64–79. <https://doi.org/10.1016/j.cad.2016.10.003>.
- [16] González-Lluch, C.; Company, P.; Contero, M.; Pérez, D.; Camba, J. D.: On the effects of the fix geometric constraint in 2D profiles on the reusability of parametric 3D CAD models, *International Journal of Technology and Design Education*, First online. 2018. <https://doi.org/10.1007/s10798-018-9458-z>.
- [17] González-Lluch, C.; Plumed, R.: Are we training our novices towards quality 2D profiles for 3D models? *Lecture Notes in Mechanical Engineering, Advances on Mechanics, Design Engineering and Manufacturing II*, 2018, 714–721. [https://doi.org/10.1007/978-3-030-12346-8\\_69](https://doi.org/10.1007/978-3-030-12346-8_69).
- [18] Gossard, D. C.; Light, R. A.; Meyfarth, P. F.; Lin, V. C.; Von Turkovich, B.: The Use of Symbolic Dimensioning in Computer-Aided Design. *CIRP Annals*, 29(2), 1980, 567–569. [https://doi.org/10.1016/S0007-8506\(16\)30154-8](https://doi.org/10.1016/S0007-8506(16)30154-8).
- [19] Hartman, N. W.: Defining Expertise in the Use of Constraint-based CAD Tools by Examining Practicing Professionals, *Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition*, 2004, Session 2238, 9.369.1–13.
- [20] Hilbert, D.: *The Foundations of Geometry*, Authorized translation by E.J. Townsend, The Open Court Publishing Company, La Salle Illinois, 1902 (Reprint edition 1950).
- [21] Hoffmann, C. H.; Joan-Arinyo, R.: A brief on constraint solving, *Computer-Aided Design & Applications*, 2(5), 2005, 655–663. <https://doi.org/10.1080/16864360.2005.10738330>.
- [22] ISO 10303-242:2014 *Industrial automation systems and integration—Product data representation and exchange—Part 242: Application protocol: Managed model-based 3D engineering*, International Organization for Standardization, 2014
- [23] Klein, R.: *The Role of Constraints in Geometric Modelling*, In *Geometric Constraint Solving and Applications*, B. Grüberlin & D. Roller (Eds). Springer, 1998.
- [24] Kramer G. (1990) *Solving Geometric Constraint Systems*. AAAI-90 Proceedings, pp 708–714.
- [25] Light, R.; Gossard, D.: Modification of geometric models through variational geometry, *Computer-Aided Design*, 14(4), 1982, 209–214.
- [26] Lin, V. C.; Gossard, D. C.; Light R.A.: Variational geometry in computer-Aided Design, *Computer Graphics*, 15(3), 1981, 171–177.
- [27] Mantyla, M.; Shah, J.: *Parametric and Feature Based CAD/CAM: Concepts, Techniques and Applications*, John Wiley & Sons, 1995.
- [28] Menary, G.; Robinson, T.: Novel approaches for teaching and assessing CAD, *International Conference on Engineering Education, ICEE*, 2011.
- [29] Otey, J.; Company, P.; Contero, M.; Camba, J. D.: Revisiting the Design Intent Concept in the Context of Mechanical CAD Education, *Computer-Aided Design and Applications*, 15(1), 2018, 47–60. <https://doi.org/10.1080/16864360.2017.1353733>.
- [30] Roller, D.; Schonek, F.; Verroust, A.: Dimension-driven Geometry in CAD: A Survey. In: Straßer W., Seidel HP. (eds) *Theory and Practice of Geometric Modeling*, Springer, Berlin, Heidelberg, 1989, [https://doi.org/10.1007/978-3-642-61542-9\\_32](https://doi.org/10.1007/978-3-642-61542-9_32).
- [31] Rynne, A.; Gaughran, W.: Cognitive Modeling Strategies for Optimum Design Intent in Parametric Modeling, *Computers in Education Journal*, 18(3), 2007, 55–68.
- [32] Shah, J.; Mantyla M.: *Parametric and Feature based CAD/CAM: Concepts, Techniques and Applications*, John Wiley, New York, ISBN#0471-00214-3, 1995.
- [33] Shih, C. H.; Anderson, B.: A design/constraint model to capture design intent, *SMA '97 Proceedings of the fourth ACM symposium on Solid modeling and applications*, 1997, 255–264.
- [34] Sutherland, I.E.: Sketch pad a man-machine graphical communication system. *Proceeding DAC '64. Proceedings of the SHARE design automation workshop*, 1964, 6.329 - 6.346.