A Constraint Redundancy Elimination Strategy to Improve Design Reuse in Parametric Modeling

Abstract: Strategies for design reuse play a fundamental role in the development of products that change and evolve over time. Design changes often involve modifications in the geometry of parts and assemblies, which are driven by changes in the digital representation of the product, i.e. the procedural and parametric CAD model. Consequently, constraint redundancies in the two-dimensional profiles that build the parametric model can significantly hinder alterability and reusability. This paper argues that constraint redundancy conditions are not solely a computational problem but a more complex issue that involves the interaction with the sketch throughout the modeling process and the modeling scheme used to convey a specific design intent. We analyze a representative group of 3D MCAD systems and the interaction with their corresponding geometric kernels and Geometric Constraint Solvers (GCS) to determine how constraint redundancy is managed at the profile level. Next, we report the results of a series of experiments to evaluate the influence of redundant constraints on model editing and reusability tasks and describe the development of a new software tool for identifying and parsing constraint redundancy conditions. We conclude that constraint redundancy in profiles significantly and unnecessarily compromises model conciseness, robustness, and the overall model quality, which negatively affects user productivity and downstream processes. The implications of constraint redundancy conditions for CAD training are emphasized. Our experiments also demonstrate the value of our parsing tool to assist users in maximizing model reuse (by removing redundancies) and the communication of design intent (by proposing an optimal set of non-redundant constraints), but further development is necessary to use it as a practical tool for engineering analysis.

Keywords: parametric CAD, design reuse, redundant constraints, design intent.

1. Introduction

Maximizing design reuse and enabling automation are fundamental goals in mechanical design [1] and a key aspect of Engineering Change Management (ECM). From a digital model standpoint, the inherent capabilities of parametric CAD technology provide a powerful mechanism to support and facilitate change, as modifications to a particular geometric element in a part can propagate downstream to other elements or other parts and assemblies [2]. However, without a proper modeling methodology, CAD models may not be sufficiently robust to effectively and efficiently react to design changes [3-5]. Specifically, decisions on the construction sequence and parameterization will determine and restrict what aspects of the geometry can be edited (and to what extent) after the model is built [6].

In a parametric modeling context, it has been argued that over- and under-constrained parametric 2D profiles result in low quality 3D models, which fail to convey design intent and are more likely to cause regeneration errors or behave unpredictably when altered [7, 8]. Design intent and how it is represented and managed are complex topics that have been the subject of numerous studies [9, 10]. In the context of this paper, we consider design intent as the construct or plan that allows designers to anticipate the way a geometric model must behave when it is modified [11].

According to the ISO 10303-108 standard [12], over-constrained profiles may contain redundant constraints, which are repetitive but consistent with other constraints without affecting the solution, or inconsistent constraints, which lead to an unsolvable system. Therefore, constraint redundancy is

a type of over-constraining condition that is not always perceived by the user or identified as incorrect or invalid by the CAD system. While redundant constraints do not remove any degrees of freedom in the profile, they add unnecessary complexity and overload the Geometric Constraint Solver (GCS) of the parametric modeler (which must deal with unnecessarily complex systems of equations). In most cases, redundant constraints do not result in errors. However, their effect on the alterability of the profile has yet to be studied.

In this paper, we examine the concept of constraint redundancy in 2D profiles of parametric 3D Mechanical CAD systems (MCAD) and hypothesize that constraint redundancy conditions negatively impact model reuse and communication of design intent. In the next section, the definition and scope of the term *redundant constraint* are established and a comparative study on how 3D CAD systems handle constraint redundancy conditions is presented. Next, we describe a series of experiments with a group of users to examine how the presence of redundant constraints affects reusability during the parametric sketching process. The paper also describes the development of a new software tool for identifying and parsing constraint redundancy conditions. The paper concludes with lessons learned and guidelines for the automatic detection of constraint redundancies in parametric profiles and the delivery of real-time feedback to users to ultimately maximize the design intent of CAD models.

2. Background

The use of fully defined profiles is generally considered a best practice in parametric modeling, as under- and over-defined profiles are unstable, unpredictable, and a source of potential errors [13-15]. Most MCAD systems provide mechanisms to check for under- or over-constrained profiles during the modeling process [16]. Similarly, CAD quality testers can help users to "clean" models by identifying, and sometimes repairing, data errors that could affect simplification, interoperability, and reusability. Nevertheless, it is easy to find examples that expose the limitations of MCAD Model Quality Testing Tools, such as the inappropriate uses of fix constraints [17], morphological errors derived from ineffective modeling practices [16] or the existence of redundant constraints.

A constraint is redundant if (1) it can be removed from a 2D profile without resulting in under-constrained geometry, and (2) it can be preserved without resulting in unsolvable geometry [18]. In algebraic terms, a redundant constraint can be expressed by an equation that is a linear combination of the equations that represent other constraints. A comparison between different methods for constraint redundancy identification was published by Paulraj and Sumathi [19], and although this problem has been addressed by other authors [20, 21], there is still a need for more effective strategies and practical implementations.

Some GCS's [22] can efficiently manage the most common cases of over-constrained profiles such as multiple coincident constraints between lines and points. Free CAD tools such as the constraint parameter solver in FreeCAD v0.18 [23] and the SHAPER module in SALOME [24] provide feedback to the user regarding the unconstrained degrees of freedom in a profile and can also detect certain redundancies. Complex situations, however, can significantly and inadvertently affect performance. Nevertheless, even solvers that can detect basic redundancies are not capable of identifying which specific constraints are best candidates for removal in a constraint redundancy scenario, based on the criteria of preventing model reuse and/or conveying design intent.

The goal of this paper is not to contribute to the mathematical and computational aspects of constraint solving. Instead, we focus on how redundant constraints, which do not necessarily cause a sketch to become invalid or unsolvable, may hinder subsequent modeling tasks, particularly when editing or reusing existing aspects of the 3D model. For example, if the model depicted in Fig. 1 left, is reoriented so that the slot runs vertically, the redundant constraints in the sloped hollow surface that connect the profile to the reference system will cause a regeneration error (Fig. 1, right).

Aldefeld defines redundancies as superfluous but compatible constraints [25]. According to Otey et al. [9], redundant constraints convey incorrect design intent, which makes editing tasks more difficult and increase the computational cost of processing the 3D model. For these reasons, some CAD software user manuals recommend caution when adding constraints to a model [22].

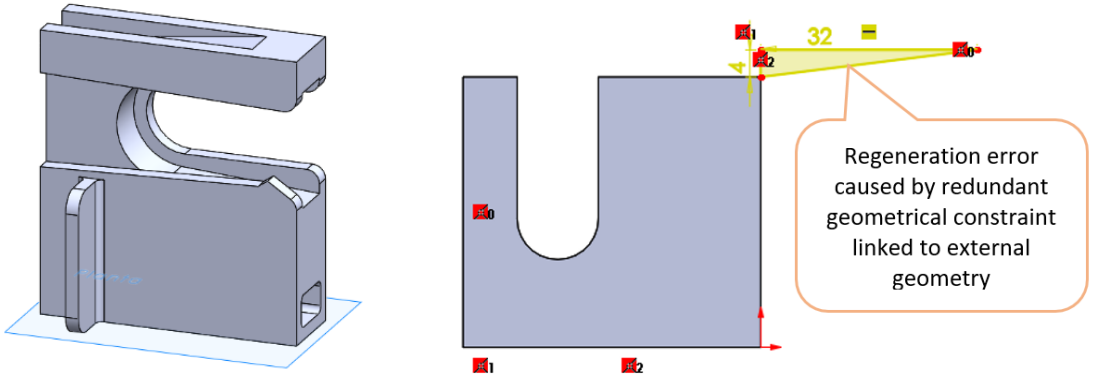


Fig. 1. Example of error caused by redundant constraints.

From an educational standpoint, some authors have recommended addressing the problem of constraint redundancy at the early stages of the learning process. Company et al. [12] demonstrated that modeling strategies can be improved by providing prompt feedback. Race [26] stated that, to be effective, feedback must be continuous in a formative sense. Similarly, some proposals to homogenize and improve CAD assessment have been published [27-30] and various guidelines have been suggested to build 3D CAD models with design change capabilities [31]. However, research has shown that even expert users have difficulties identifying certain types of geometric constraints and determining whether a profile is fully constrained when no assistance is provided [32, 33]. To the best of our knowledge, only a limited number of CAD tools provide some feedback to the user and implement mechanisms to manage the proliferation of redundant constraints. FreeCAD v.0.18, for example, allows the user to turn on or off the monitoring of redundant constraints in a profile [23]. However, current solvers are extremely limited in terms of analyzing the overall constraining scheme in a profile and identifying the specific constraints that are best candidates for removal.

According to the ISO 10303 standard, constraints are "relationships between two or more elements in a model, which should be maintained in any modifications made subsequent to a model transfer," while constraint solvers are "software systems for solving sets of equations that are mathematical representations of constraint relationships" [12]. A "logical" or geometric constraint enforces relationships between geometric entities such as tangency, collinearity, parallelism, perpendicularity, coincidence of points, symmetry, etc. Dimensional constraints may be used to specify the size of a particular entity or the relative location between different entities. Constraint solvers leverage the intrinsic differences between geometric and dimensional constraints to solve them separately and more efficiently [34, 35]. However, in our view, this distinction is insufficient when the goal is to maximize design intent and CAD model reusability.

Various classifications of modeling constraints have been proposed [36, 37]. In a recent study, authors Company et al. [38] proposed a new classification of geometric constraints which distinguishes between discrete and continuous constraints, acting at intrinsic or extrinsic levels. These criteria result in four main types of constraints: shape, size, location, and movement. In our view, any action aimed at marking constraints as redundant should consider their type, in order to preserve those constraints that better convey design intent and/or guarantee CAD reusability. Our argument can be illustrated with the following example: Is design intent conveyed more effectively by a perpendicularity

constraint, or by an angular dimensional constraint fixed at 90-degrees? We contend the answer does not depend solely on facilitating the most efficient operation to the GCS, but also on conveying the explicit intent of fixing the angle at a discrete value (90 degrees), or the alternative intent of enabling a controlled angle to continuously adapt to varying design requirements.

It is important to emphasize the type of redundant constraints that are the focus of this paper. Although industrial sketchers are designed to minimize the number of geometric constraints that over-constrain a profile, they do so only to prevent inconsistencies and unsolvable sketches which cannot be processed by the GCS. However, there are many scenarios where adding (redundant) geometric constraints does not cause a sketch to become unsolvable. For instance, many of the examples in our paper that include redundant geometric constraints are not detected by any GCS as unsolvable or even redundant, nor are these geometric constraints minimized or identified. In fact, the profiles are processed as valid, consistent, and solvable sketches, which is precisely the problem we are trying to describe in this paper. These scenarios, which can be easily replicated in any commercial parametric modeler, illustrate the idea that constraint redundancy is both a computational problem and an interaction problem connected to the modeling scheme used to convey a particular design intent.

Finally, a key concept in our research is defined by the term *granularity*, which describes the amount and complexity of the operations required to create a new geometric feature in a CAD model. The term was introduced to describe CAD exchange problems derived from the fact that distinct CAD formats may require different but equivalent sequences of one or more operations to describe the same geometric feature. The example described by Pratt and Kin [39, 40] to explain the concept of granularity is relevant to understand why different applications treat redundant constraints differently. According to the authors, "some CAD systems allow the creation of under constrained sketches or features, and permit later fine-tuning of the model in terms of lower-level operations. But other systems, with coarser granularity, only allow the creation of fully constrained constructs, possibly through the use of default options" [39, 40]. Therefore, different implementation strategies in commercial CAD systems may result in different levels of granularity, forcing users to adapt their modeling habits to the different types of behavior of the constrained sketches which, in turn, may depend on the computational limitations of the constraint solvers. This variability undoubtedly conditions the interaction with the CAD system, particularly in terms of CAD model reusability. We will use the term granularity in this context. A sketch with coarse granularity barely distinguishes between geometrical and dimensional constraints, whereas finer granularity sketches may differentiate between discrete and continuous constraints, acting at intrinsic or extrinsic levels, while also distinguishing between required and redundant constraints.

3. Constraint redundancy in commercial CAD applications

As a first step to analyze constraint redundancy, we conducted a comparative study to expose the differences and similarities between commercially available CAD applications. The behavior of a CAD application is governed by its geometric modeling kernel, which includes methods and algorithms to process geometry. Kernels are developed by a limited number of vendors, resulting in different CAD applications that share the same kernel. 3D ACIS Modeler (Spatial Technology, Dassault) [41], Parasolid (Siemens PLM) [42], C3D (ASCOM Group) [43] and Open CASCADE [44] are the most widely used geometric modeling kernels.

Constraint-based modeling functionality is provided by Geometric Constraint Solvers (GCS). These are software components that support the creation and modification of parametric geometric models, which are governed by constraints. GCS software is responsible for solving the configuration of geometric objects from a given set of constraints or provide a partial solution and information about the over-constrained parts of the geometric model when the configuration is not possible [45]. In the case of over-constrained but consistent models, the GCS can also find a valid solution [46].

Some of the most common GCS's include Dimensional Constraint Manager (DCM) (Siemens PLM) [46], C3D Solver (ASCON Group) [43] and LGS (Bricsys) [47]. Significant differences exist between GCS's in terms of managing over-constrained conditions and redundant constraints. In the case of DCM [22], models with redundant geometric constraints but without excessive dimensional constraints can be solved. By default, the C3D Solver [43] excludes any extra constraints from the calculations. They are only included when no conflicts exist. The solver LGS 2D [45] informs the CAD system about the conflict between geometric constraints, which then decides whether to mark the profile as over-constrained, based on the defined context option.

Many commercial CAD systems share both kernel and GCS, the diversity of these software components offers a number of alternatives to a CAD developer. As a result, different behaviors can be identified regarding the management of redundant or incompatible constraints in 2D profiles. To analyze this behavior, a representative set of profiles (shown in Fig. 2) was tested in various 3D CAD systems: Inventor Professional (Autodesk), Solid Edge (Siemens), Solidworks (Dassault Systemes), FreeCAD, Creo (PTC) and Onshape (PTC). A summary of the constraints used in the profiles is shown in Table 1 (adapted from [48]).

Table 1. Graphical representation of the constraints used in our study.

	Coincident	Forces two points to be coincident, or one point to lie on a line, a circle, or an arc.
	Collinear	Forces lines and linear polyline segments to lie on the same straight line.
	Concentric	Forces the center points of arcs, circles, ellipses, or elliptical arcs to coincide.
	Equal	Forces a set of entities to have the same dimensions (e.g. length, diameters, etc)
	Horizontal	Forces lines and linear polyline segments or pairs of points on entities to be parallel to the x-axis of the current coordinate system.
	Parallel	Forces two lines or linear polyline segments to be parallel to each other.
	Perpendicular	Fixes two lines or linear polyline segments at 90 degrees to one another.
	Symmetric	Forces two entities or constraint points on entities to lie symmetrically about a given axis.
	Tangent	Forces two entities (e.g. two arcs, or a line and an arc) to be tangent to each other.
	Vertical	Forces lines and linear polyline segments or pairs of points on entities to be parallel to the y-axis of the current coordinate system.
Numeration: Constraints that involve two or more entities (as in a collinear constraint) are placed next to all the entities involved and with the same number to disambiguate.		

The initial profiles with dimensional constraints are shown in Fig. 2, column 1. A valid set of geometric constraints that result in fully constrained profiles (without redundancies) are shown in Fig. 2, column 2. Columns 3 and 4 (Fig. 2) illustrate cases in which one or more redundant constraints are added to the profiles, but do not necessarily result in invalid or inconsistent sketches. For example, in Fig. $2a_3$, Fig. $2b_3$ and Fig. $2c_3$, the new perpendicular constraint (highlighted) is not necessary, as the perpendicularity condition is enforced indirectly by the existing horizontal and vertical constraints that converge at the same vertex. Therefore, this (redundant) constraint turns a fully constrained profile into an over-constrained one (which may or may not be recognized as such by the GCS). In the case of Fig. $2d_2$, a fully constrained sketch is obtained when the vertical lines are related by a symmetry constraint (global symmetry). However, when a new vertical constraint (which is redundant) is added in Fig. $2d_3$, the sketch becomes over-constrained. Column 4 represents cases in which more than one redundant constraint (highlighted with circles) are added to the profiles, i.e. the

highlighted constraints make the sketches over-constrained. In all cases, the resulting sketches (with the redundant constraints) may not necessarily be recognized as over-defined by the GCS. In fact, commercial CAD systems process these profiles as valid, consistent sketches. Column 5 (Fig. 2) shows the incompatible constraints added to the previous fully constrained profiles (column 2) as part of a second test. The incompatible constraints are indicated with a circle.

It is important to note that the definition of redundant constraints provided in section 2 (a constraint is redundant if (1) it can be removed from a 2D profile without resulting in under-constrained geometry, and (2) it can be maintained in the sketch without resulting in unsolvable geometry) is helpful to determine whether or not redundant constraints have been applied to the sketch, but not to determine precisely which constraints are redundant. In fact, many solutions may be possible, as we will discuss in section 6. Deciding which constraints should be marked as redundant in a profile with constraint redundancy conditions is not trivial and it is outside the scope of this paper. Nevertheless, as we discussed in Section 2, we suggest that constraints must be classified based on finer granularity and according to the amount and type of design intent they convey.

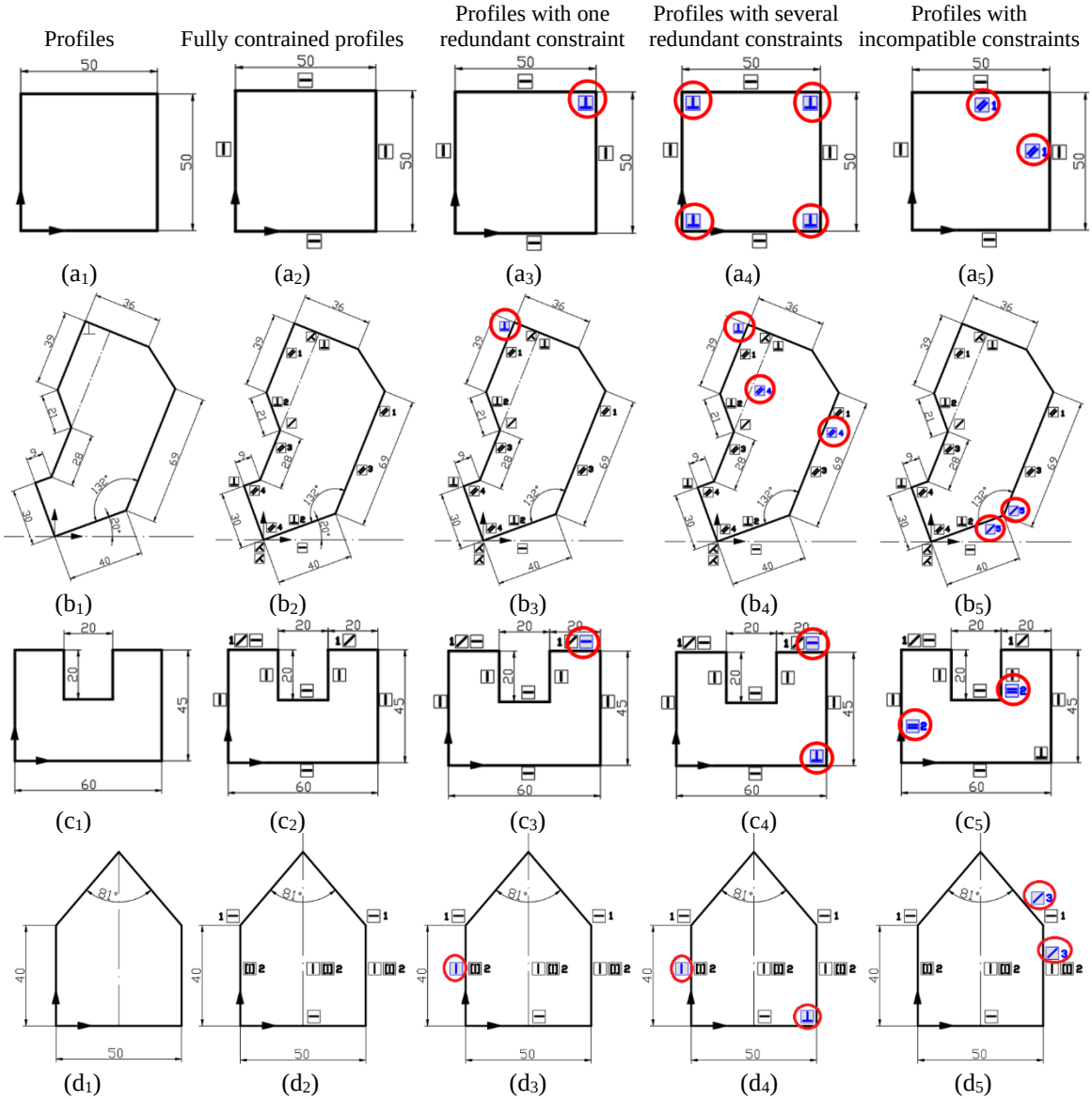


Fig. 2. Profiles used to assess the behavior of CAD systems in terms of redundant and incompatible constraints.

No common pattern was found on how different CAD systems manage redundant and incompatible constraints. Different types of feedback were observed.

First, some systems such as SolidWorks, Solid Edge, and Onshape detected redundancies, but no feedback or warning was displayed about the redundancy conditions. The user was allowed to continue with the modeling task as long as no incompatible constraints are present in the profile. Alternatively, SolidWorks warns the user about the unsolvable or conflicting item (highlighting it in different colors), and offers two options to solve the problem: to "diagnose" the sketch error, or to "delete" any of the current constraints in the profile. Onshape warns that the "Sketch could not be solved," but no mechanism is offered to solve the problem. Incompatible constraints are not allowed by Solid Edge. When the user attempts to add one, the program automatically removes the new constraint that conflicts with the existing ones. These solvers distinguish between redundant and incompatible constraints and can often find a geometric solution in a profile with redundant constraints. Nevertheless, they do not provide any feedback, warning, or alternative to the redundancy.

Other systems such as FreeCAD warn the user about the redundancy condition. In this case, the degrees of freedom that should be eliminated to produce a fully constrained profile are displayed while the profile is being edited. FreeCAD allows users to turn on and off the option "remove redundant constraints automatically." When turned on, the system does not allow the creation of either redundant or incompatible constraints and automatically removes the last constraint entered, even when the profile remains under-constrained. When the detection option is turned off, redundant and incompatible constraints can be added, but the system warns about the over-constrained condition and suggests their removal. In this case, the user can continue editing the profile and adding new constraints.

Autodesk Inventor and Salome-Meca Shaper also display the amount of degrees of freedom that remain under-constrained in a profile. However, unlike FreeCAD, these systems do not allow the user to continue to work when redundant constraints are present, i.e. the solvers do not distinguish between redundant and incompatible constraints. When redundant or incompatible constraints are detected, the system asks the user to remove the last constraint that was added or undo the last operation. In both cases, the program stops until the user solves the problem. Autodesk Inventor displays the same message as in the first test, even though the constraints were incompatible instead of redundant.

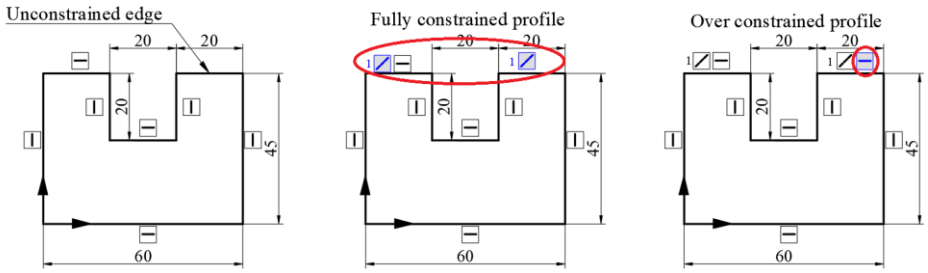
All systems showed different warnings when an incompatible constraint was introduced. More specifically, when an incompatible constraint is introduced in a 2D sketch in SolidWorks, the application warns the user about the unsolvable or conflicting item (highlighting it in different colors), and offers two alternatives: to "diagnose" to the sketch error, or to "delete" any of the current constraints in the profile to avoid the error. Onshape warned that the "Sketch could not be solved," but no mechanism was offered to solve the problem. The message displayed by Autodesk Inventor was the same as the one from the first test, even though the constraints were incompatible instead of redundant. Incompatible constraints are not allowed by Solid Edge. When the user attempts to enter one, the program automatically removes the new constraint that conflicts with the existing ones. FreeCAD allows users to turn on or off the automatic removal of redundant constraints. When turned on, the system does not allow the creation of incompatible constraints, even when the profile remains under-constrained. When turned off, incompatible constraint can be added, but the system warns about the over-constrained condition and suggests their removal. In this case, the user can continue editing the profile and adding new constraints. The behavior of the different CAD systems is summarized in Table 2.

Table 2. Behavior of commercial 3D CAD systems regarding constraint redundancy.

CAD system		1st Test (one redundant constraint)		2nd Test (multiple redundant constraints)		3rd Test (incompatible constraints)		
		Warning	*Feedback*	*Warning*	*Feedback*	*Warning*	*Feedback*	*Execution stopped*
SolidWorks		No	No	No	No	Yes	Yes	No
Onshape		No	No	No	No	Yes	No	Yes
Inventor Professional		Yes	Redundant constraints not allowed	Yes	Redundant constraints not allowed	Yes, but incorrect warning	Same as previous tests	Yes
Creo		Yes	Redundant constraints not allowed	Yes	Redundant constraints not allowed	Yes, but incorrect warning	Same as previous tests	Yes
Solid Edge		No	No	No	No	Yes	No	Yes
FreeCAD	Remove redundant constraints automatically (ON)	Yes	Yes	Yes	Yes	Yes	Remove constraints	No
	Remove redundant constraints automatically (OFF)	Yes	No	No	No	Yes	Remove constraints	No
Salome-Meca Shaper		Yes	Redundant constraints not allowed	Yes	Redundant constraints not allowed	Yes	Remove constraints	Yes

We report two additional observations on the CAD systems that do not allow redundant constraints:

1. The order in which constraints are applied may influence the detection of constraint redundancy in the CAD system. For example, if the profile in Fig. 3a$_1$, is constrained by first adding the collinear constraint pointed out in Fig. 3a$_2$ then the profile is fully constrained and when adding a horizontal constraint (Fig. 3a$_3$), the profile will be considered as an over-constrained by some systems. However, if the same profile is first constrained by introducing the horizontal constraint as depicted in Fig. 3a$_4$, then the edge remains unconstrained until the collinearity constraint of Fig. 3a$_5$ is included, in this case the systems will incorrectly consider the profile not redundant.
2. Certain constraints are added implicitly and remain hidden for users. For instance, an implicit parallelism constraint is applied in a profile when the distance between two parallel lines is dimensioned. The scenario is illustrated in Fig. 3b. If the profile is constrained as shown in Fig. 3b$_1$, the profile becomes fully-constrained when the dimensional constraints are applied to the lines. However, if the dimensional constraints are applied to the vertices (Fig. 3b$_2$), additional constraints (e.g. one vertical and one horizontal) are necessary to fully constrain the profile.



(a$_1$) (a$_2$) (a$_3$)

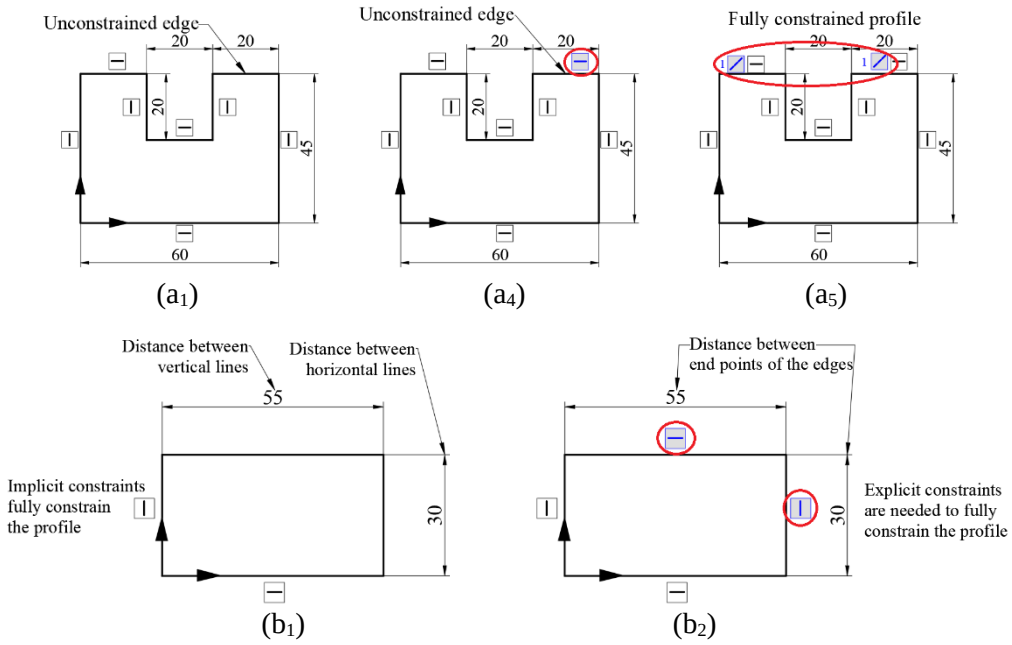


Fig. 3. Failure to detect redundant constraints (a_1-a_5) and implicit constraints linked to dimensions (b_1-b_2).

We conclude that GCSs provide valid solutions in the case of over-constrained sketches, although most applications manage the information that is shared with the user differently when a constraint causes an over-constrained sketch, or when the constraint is incompatible with current constraints. None of the applications, however, provided the user with finer granularity information about the types of constraints and the implications for design intent and the reusability of the CAD model. In terms of constraint redundancy, most applications do not provide feedback to the user when redundant constraints are detected by the GCS. Some systems delete certain redundant constraints automatically, whereas others do not allow users to add them to the sketch. When users add incompatible constraints, all the applications trigger warnings, which in some cases are inaccurate and not helpful to easily identify and correct the mistake. More importantly, none of the applications provides any assistance by suggesting alternative and more appropriate constraints to maximize quality, particularly in terms of design intent and reusability.

We speculate sketchers behave as black boxes to the users because of historical reasons. Historic GCSs were less powerful and robust than modern ones. Therefore, allowing complex interactions often resulted in unsolvable sketches and even application crashes. Modern sketches have inherited most of the opaque and automatic procedures that have been implemented to prevent failures, as they simplify implementation while still being acceptable for basic use. However, there are multiple indicators of CAD interoperability and reusability issues, which clearly show that this strategy is no longer appropriate for modern engineering paradigms such as the Model-Based Enterprise. In these scenarios, reusable high-quality CAD models are paramount, and interactive sketches that can parse the quality of the constraints during the sketching process and mitigate or even eliminate undesirable practices such as constraint redundancy, can be helpful.

4. Experimental determination of the influence of redundant constraints on editing and reusing profiles

When a design engineer edits an existing profile, several factors may influence the time and likelihood of success. We hypothesize that the quantity of redundant constraints is one of those factors. This

hypothesis may seem obvious if we consider the terms *over-constrained* and *redundant* as synonymous. Although many authors agree that over-constraining is bad practice that affects the validity of the models, it is important to distinguish between these two conditions. Geometry-wise, redundant constraints do not cause geometrical inconsistencies. However, from a CAD user perspective, they seem to be perceived differently. Over-constraining is considered a bad strategy, whereas redundant constraining is usually perceived as a neutral or even beneficial action that helps to improve robustness without compromising how design intent and reusability are conveyed. To validate the different hypotheses that redundant constraining practices compromise reusability, we conducted an experiment with junior Mechanical Engineering students. Developing proper constraining skills is not an easy task. Our studies and experience show that novices tend to introduce redundant constraints even in simple sketches, even after being properly trained in parametric modeling and using basic help tools in the 3D CAD system. These practices do not provide any benefits. Rather, they have proven to be highly inefficient. Other studies [32] suggest that many users, even experts, often fail to use constraints in a robust and efficient manner that avoids redundant constraints. Other authors have recommended addressing the problem of constraint redundancy at the early stages of the learning process.

Participants were enrolled in one of the four available laboratory sections and received the same level and amount of training using SolidWorks® prior to the experiment. Training included the creation of 2D profiles, the use of constraints as well as the differences between over-, under-, and fully constrained profiles [49]. Students are considered to have sufficient knowledge and experience to perform the tasks of the experiment successfully. In our study, participants were asked to edit a profile (*profile A*, as shown in Fig. 4) created and constrained by the research team based on specific criteria.

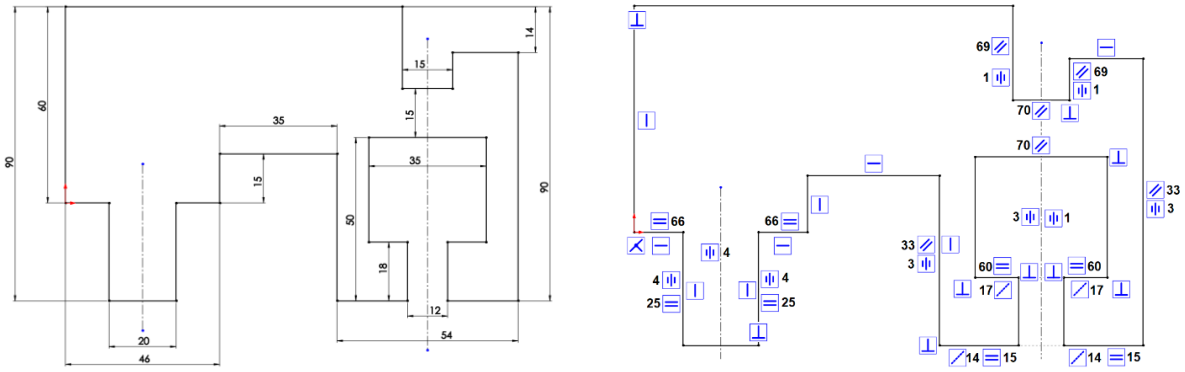


Fig. 4. Profile with dimensional constraints (left) and fully constrained profile used as Profile A (right).

Two additional versions (profiles B and C) were created with five and ten redundant constraints respectively, as shown in Fig. 5. The redundant constraints in profiles B and C were selected based on their potential interactions with the entities affected by the modifications. The most frequent types of redundant constraints were added to the profile, including parallelism, horizontality / verticality, and perpendicularity.

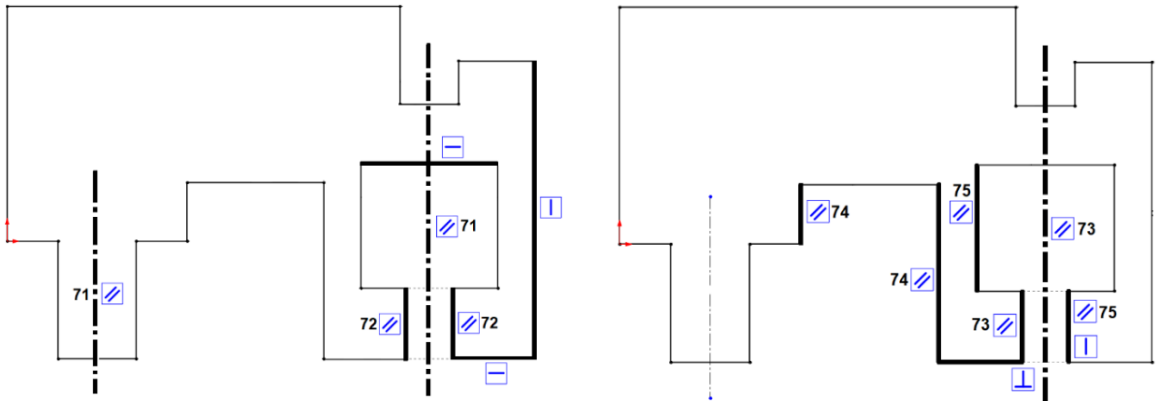
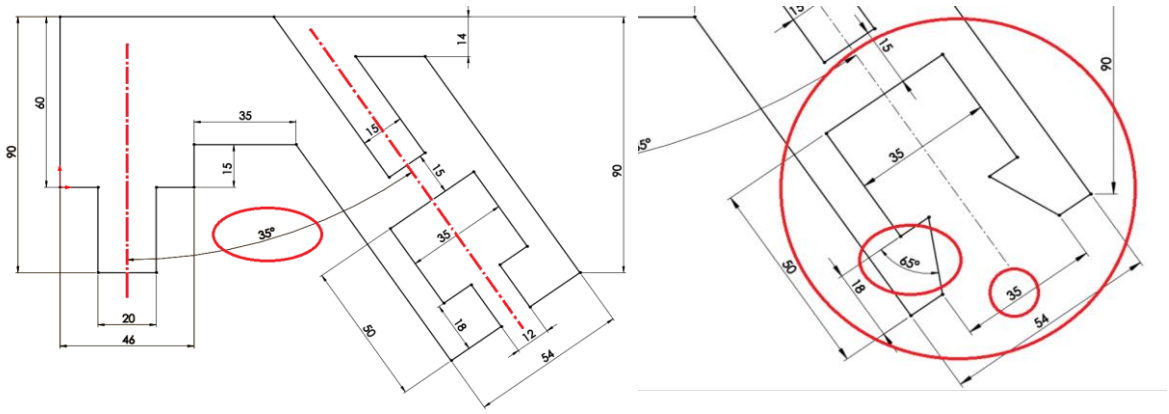


Fig. 5. Redundant constraints in profiles B (left) and C (right).

The profiles were randomly assigned to participants, along with detailed instructions about the required modifications. Participants were given 35 minutes to complete the tasks described in Fig. 6, which consisted of (a) adding a 35-degree angle between the symmetry axes, and (b) modifying the shape and number of dimensions of the slot located at the lower right corner of the profile. Participants were asked to manipulate the constraints without removing any lines from the profile. The automatic constraints option in SolidWorks was turned off to avoid the addition of unintended constraints.



(a) (b)

Fig. 6. Requested changes: a new angle between the symmetry axes (a) and a new shape for the slot (b).

A total of 44 files were submitted by the participants in our study: 15 files for profile A, 14 for profile B, and 15 for profile C. Three files were rejected (two were submitted out of time and one was deemed invalid as the automatic constraints option in SolidWorks had not been turned off). The descriptive statistics of the time spent on each profile (in seconds) to perform the modification is shown in Table 3 and illustrated in Fig. 7.

Table 3. Descriptive statistics of the time spent on each profile (in seconds).

Profile	N	Mean	Std. Dev.	Lower limit*	Upper limit*	Minimum	Maximum
A	15	1089.6	407.8	863.8	1315.4	513	1783
B	14	2689.7	426.4	2443.5	2935.9	2055	3363
C	15	2801.3	609.2	2463.9	3138.6	1629	3871

* 95% confidence interval for the mean

To determine whether editing time was compromised by the amount of redundant constraints in a profile, a single factor ANOVA (time) between groups (profiles A, B and C with different amount of redundant constraints) was performed. The null hypothesis (H_0) is defined as "there is no difference on the mean time between the profile groups." Our analysis revealed a statistically significant difference among the three groups $F(2, 44) = 56.52$; $p = <.001$, which suggests that the existence of redundant constraints increases the time required to modify a profile.

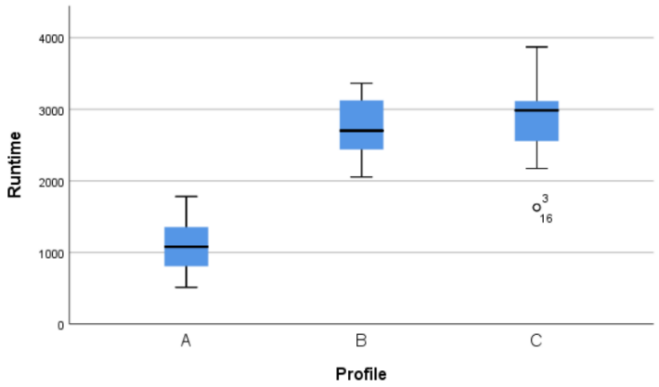


Fig. 7. Time to perform the modifications by profile

A second analysis was performed to study the influence of redundant constraints on the success rate of the requested modifications. The null hypothesis H_0 is defined as "there is no difference on the success rate between profiles." We consider a task successful if (1) all modifications were applied correctly and the final shape of the profile coincides with the one shown in Fig. 6b, and (2) the profile is not under-constrained. To validate this hypothesis, a contingency table was applied to compare the success ratio among the groups of profiles A, B, and C. The observed (Count) and expected (Expected) results for each group are shown in Table 4.

Table 4. Success rate and Profiles, Cross tabulation.

		Profile A	Profile B	Profile C	Total
Success	Count	10	5	3	18
	Expected	6.14	5.73	6.14	18
Non-success	Count	5	9	12	26
	Expected	8.86	8.27	8.86	26
	Total	15	14	15	44

The null hypothesis was tested using a Chi-Square Test of Independence. This test can be used to determine independence between groups of study when the studied variable is qualitative. In our case, we used a dichotomous variable (success/no success) to classify the results from the participants (Table 4). Our analysis suggests that there are statistically significant differences (X^2 (2, N=44) =6.99, p=0.03<.05). We conclude that the existence of redundant constraints hinders the reusability of the profile. In other words, a lack of redundant constraints can be considered an appropriate quality metric of the profile.

5. Pilot tool to detect constraint redundancy

As a proof-of-concept to demonstrate the feasibility of automating the detection of redundant constraints, a greedy algorithm was implemented as a custom software tool using Visual Basic .NET that is compatible with current versions of the SolidWorks Application Programming Interface (API) to calculate the number of redundant constrains in a profile. Contrary to current CAD modeling tools,

the proposed algorithm searches for global redundant constraints, which affect other constraints in the profile, and applies heuristic criteria to establish a priority for processing the constraints.

A constraint is marked as redundant if it can be removed from a 2D profile without resulting in under-constrained or unsolvable geometry. The functionality of the tool is described as follows: first, the tool determines the constraining status of the profile. If it is fully constrained, then an algorithm identifies the redundant constraints (if they exist). If the profile is under-constrained, a complementary algorithm fully constrains the profile before applying the first algorithm.

The algorithm to identify redundant constraints is based on sequentially activating and deactivating the profile constraints and verifying the profile status. The first step is to retrieve all the profile constraints. Next, for each constraint, the algorithm determines the maximum number of geometric constraints that can be suppressed while keeping the profile fully constrained. To accomplish this, the first constraint is deactivated, and the profile status is checked. If the profile remains fully constrained, the constraint is stored as redundant. Otherwise, the constraint is reactivated, and the next constraint is deactivated. The process is repeated until all the constraints have been processed, as shown in Fig. 8. After all constraints have been traversed, a resulting set of redundant constraints (if not empty) is obtained. However, the set may not necessarily be unique. To find alternative sets of redundant constraints, the system repeats the process multiple times starting with a different constraint in every iteration.

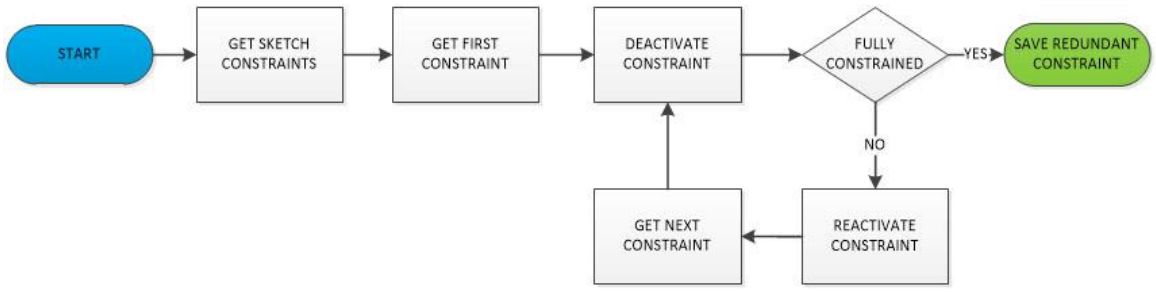


Fig. 8. Flowchart of a single iteration of the algorithm to identify redundant constraints

The algorithm to fully define a profile that is under-defined is based on the iterative addition of fix constraints to determine the point at which the profile becomes fully constrained. If it does not, other actions such as suppressing dimensions are performed. The algorithm traverses all vertices in a profile, adding fix relations in the process when possible (i.e. fixing the coordinates of the vertices to their current values). In each iteration, the status of the profile is checked. If fully constrained, the algorithm ends the task and calls the module that identifies redundant constraints (described previously). However, if it is not under-constrained but over-constrained, the dimensional constraints in the profile are processed to determine whether any of them can be removed and make the profile fully defined, as shown in Fig. 9. When this occurs, the algorithm ends, and the module responsible for identifying redundant constraints is called. A key function in the algorithm is provided by the SolidWorks API, which checks the status of the sketch. The possible results are the profile states (fully, over- and under-constrained) plus three additional error states, i.e. *noSolution*, *unknownConstraint*, and *invalidSolution*. If a particular modification implemented by the algorithm causes the sketch to move to an error state, then the last action must be undone, as the sketch becomes unusable (see Fig. 9).

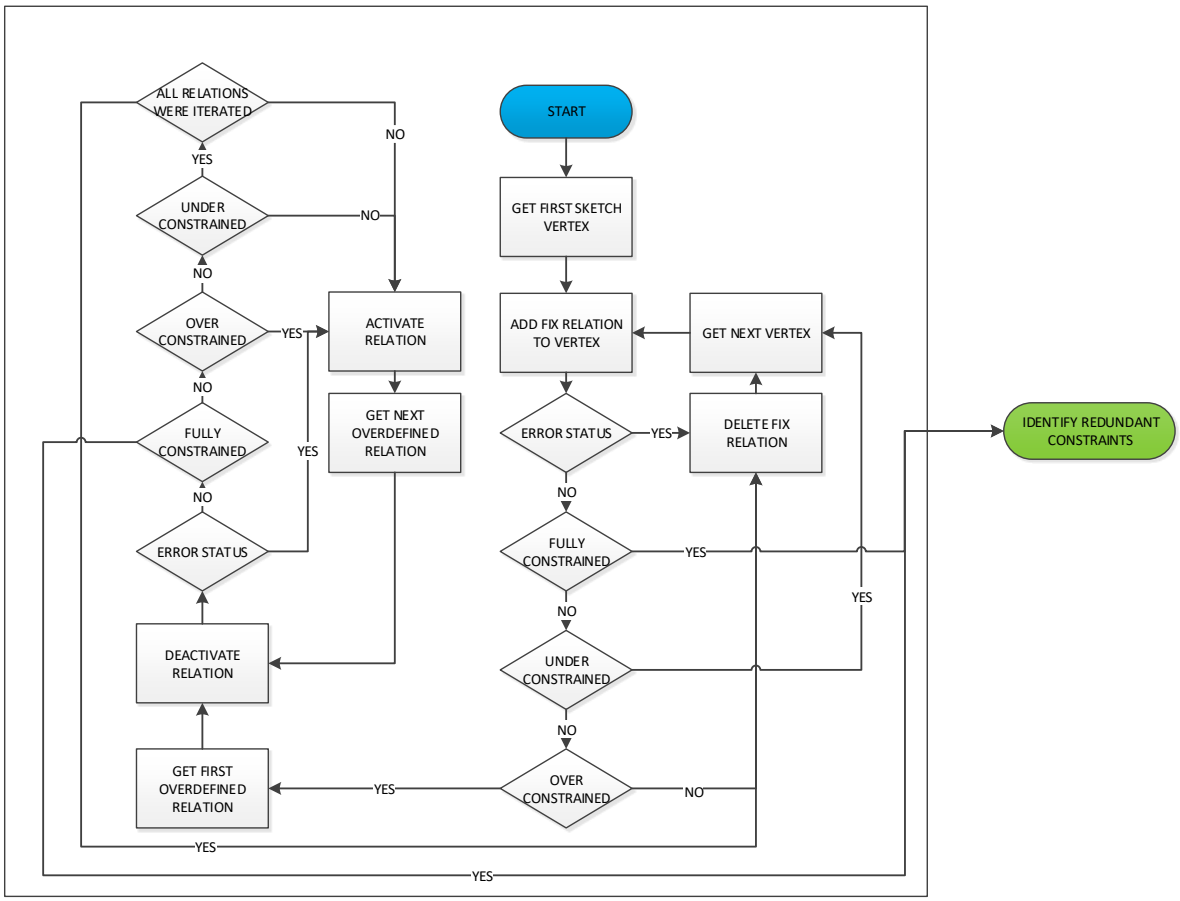
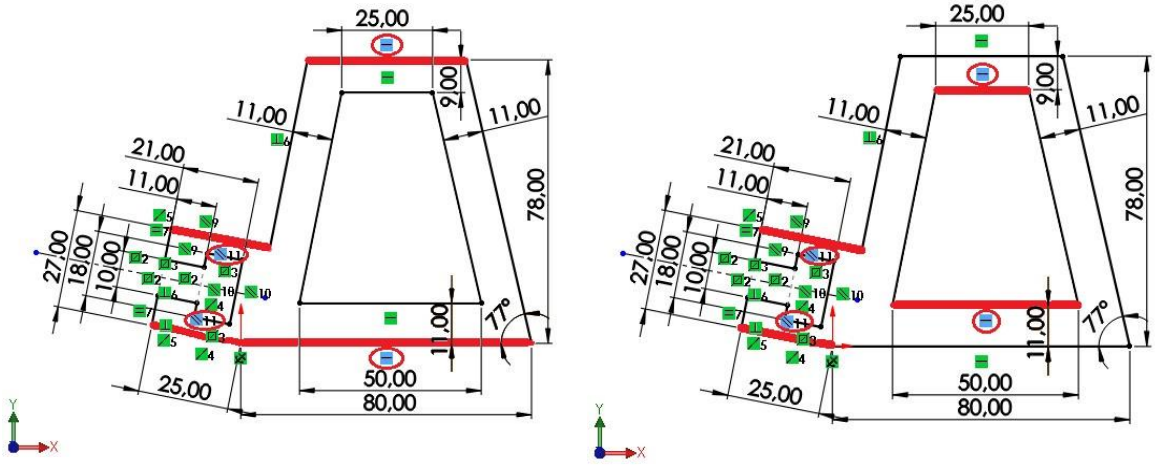


Fig. 9. Flowchart of the algorithm to fully define a profile that is under-constrained.

After preliminary tests that demonstrated its viability, our custom software tool was repackaged as a SolidWorks plugin that can be fully integrated within the CAD environment. The plugin allows users to optimize their profiles on demand during the modeling process by identifying and locating the specific constraints that are redundant. Furthermore, the plugin is able to suggest different solutions to fully constrain a profile without any constraint redundancy conditions, as shown in Fig. 10, where the solutions to an example with redundant constraints can be viewed from the plugin's interface.



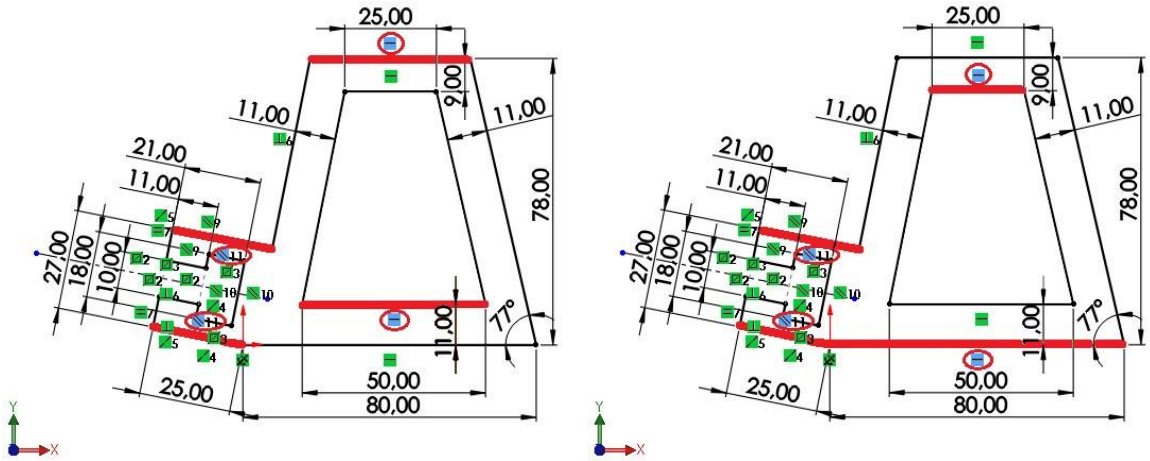


Fig. 10. Four different sets of redundant constraints for a given profile identified by our software tool.

5.1. Improved parser

The exhaustive generate and test search approach described in the previous section was implemented in our "ATA" application (see Fig. 11). ATA is a standalone tool that requires a fully functional SolidWorks installation.

We further improved ATA to validate the feasibility of a heuristic approach to identify and remove redundant constraints. Using the classification defined in [38], we produced a set of rules for determining the sequence and priority for processing the constraints and inform an algorithm that can identify and remove constraints that contribute to over-constrain the profile but do not necessarily convey design intent. Intrinsic constraints relate geometric components of a figure to each other. For example, a parallel constraint relates two lines of a sketch to each other by making them parallel. Extrinsic constraints relate geometric elements of a figure to the scene they belong [49]. For example, two horizontal constraints applied to two lines indirectly make the lines parallel, but at the cost of aligning them with the external reference system. Company et al. [38] stated that not all constraints are universally perceived as able to convey a particular kind of intent. In fact, some are usually questionable. However, the authors also showed that some constraints are perceived as consistent: positional constraints represented by dimensions are perceived visibly as such, whereas perpendicularity constraints are consistently perceived as describing shape. Building on this idea, our algorithmic approach for under-constrained profiles can be described as follows:

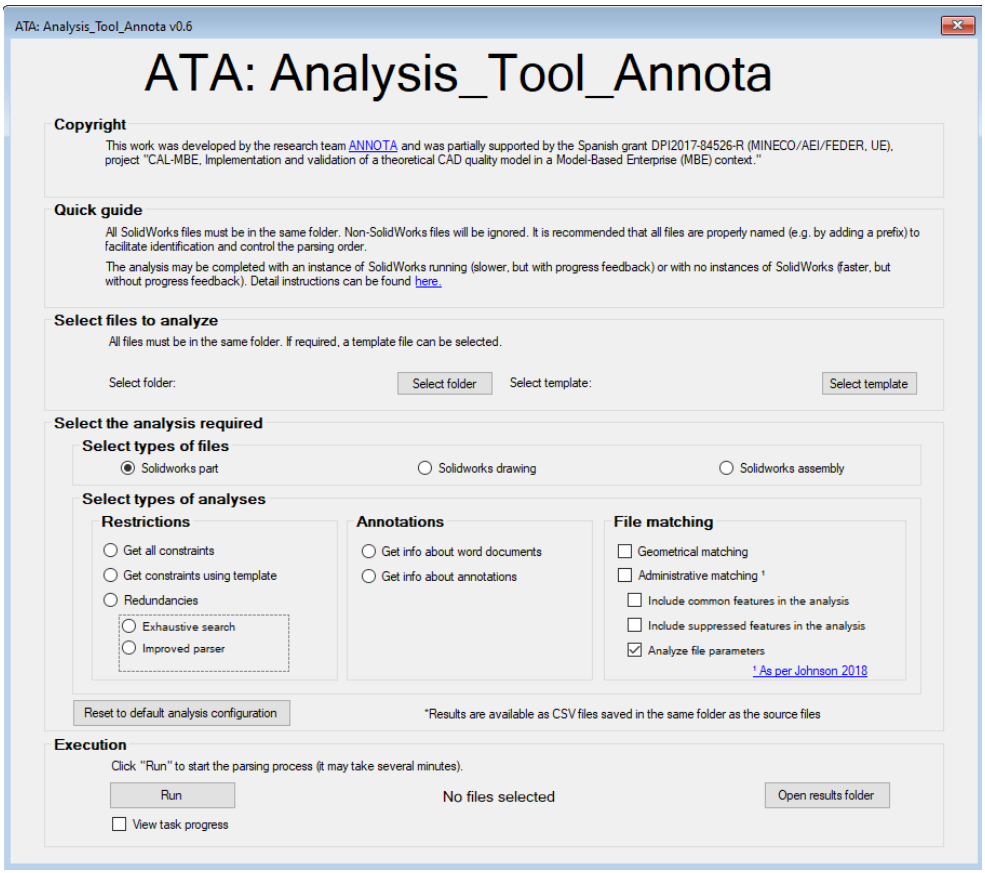


Fig. 11. User interface of the experimental software tool to parse CAD models.

1. Determine if the profile is intrinsically or extrinsically inconsistent (or both).

    1.1 Sequentially modify the coordinates of each vertex in the profile.

    1.2 If no vertices can be moved without causing errors, the profile is considered fully constrained.

    1.3 If one or more vertices can be moved independently from the rest, the profile is marked as intrinsically under-constrained.

    1.4 If moving any vertex causes other vertices to move, then the figure is rigid but extrinsically under-constrained.

2. If the profile is intrinsically inconsistent:

    2.1 Find all unfixed geometric elements.

    2.2 Order and group unfixed geometric elements hierarchically (first, symmetrical elements; second, elements belonging to the outer contour, etc.).

    2.3 Fix any unfixed elements according to the hierarchical order. Fix them to other elements of the same group by prioritizing intrinsic constraints (such as perpendicularity, instead of horizontality/verticality).

3. If the profile is extrinsically inconsistent:

3.1 Find any possible symmetry axes. Currently, our algorithm cannot automatically detect symmetry axes if they are not available in the profile. However, it can search for auxiliary lines linked by symmetry conditions added by the designer.

3.2 If more than one symmetry condition is found, then order them by placing the most important symmetry axis first (the one related to most geometric entities)

3.3 Fix the symmetry axes from most to least important, until the profile is fully constrained.

3.4 After all symmetry axes are fixed, but while the profile is not yet fully constrained, sort all vertices in ascending order by their distances to the origin, then fix the vertices in the list until the profile becomes fully constrained.

4. While the profile remains under-constrained:

4.1 Find and fix all elements whose size is not fixed by adding suitable dimensions.

4.2 Find and fix all elements whose location is not fixed by adding suitable dimensions.

We modified our exhaustive search algorithm according to the rules described above. We used the improved parser to conduct the following experiments:

(Experiment #1).

#1.1 Select a set of profiles representative of both over and under-constrained situations.

#1.2 Use the algorithm to improve them.

#1.3 In parallel, ask a group of design engineers to improve them.

#1.4 Compare performances.

(Experiment #2).

#2.1. We asked a group of expert design engineers to redesign the profiles. These profiles came from (a) the initial profiles selected in #1.1, (b) profiles supervised by other experts (previously obtained from #1.3), (c) or by the ATA algorithm (solutions from #1.2).

The experts are university professors from Mechanical Engineering and Engineering Graphics departments with extensive experience in parametric solid modeling. Most experts also have significant professional experience as design engineers in various firms. They were carefully selected based on their qualifications and experience, and all agreed to participate in the study.

#2.2 Compare performances.

The main goal of these experiments is to prove two hypotheses. First, the solutions provided by the ATA algorithm are at least as good as the solutions provided by the expert designers, in terms of producing improved profiles with no redundant constraints (minimum constrained DOF). Second, both the improved profiles provided by the experts and the ATA algorithm ease subsequent redesigns.

The results of our two experiments are detailed below and summarized in Tables 6 to 8.

Experiment #1

The profiles shown in Fig. 12 were used in experiment #1 and created in SolidWorks®. The goal of this experiment is to compare the results of the ATA algorithm and the expert designers when fully constraining these profiles. The group of experts is comprised of 15 engineering professors from the areas of Computer Graphics, Engineering Projects, and Mechanical Engineering. All experts have extensive experience working and teaching engineering design using SolidWorks®.

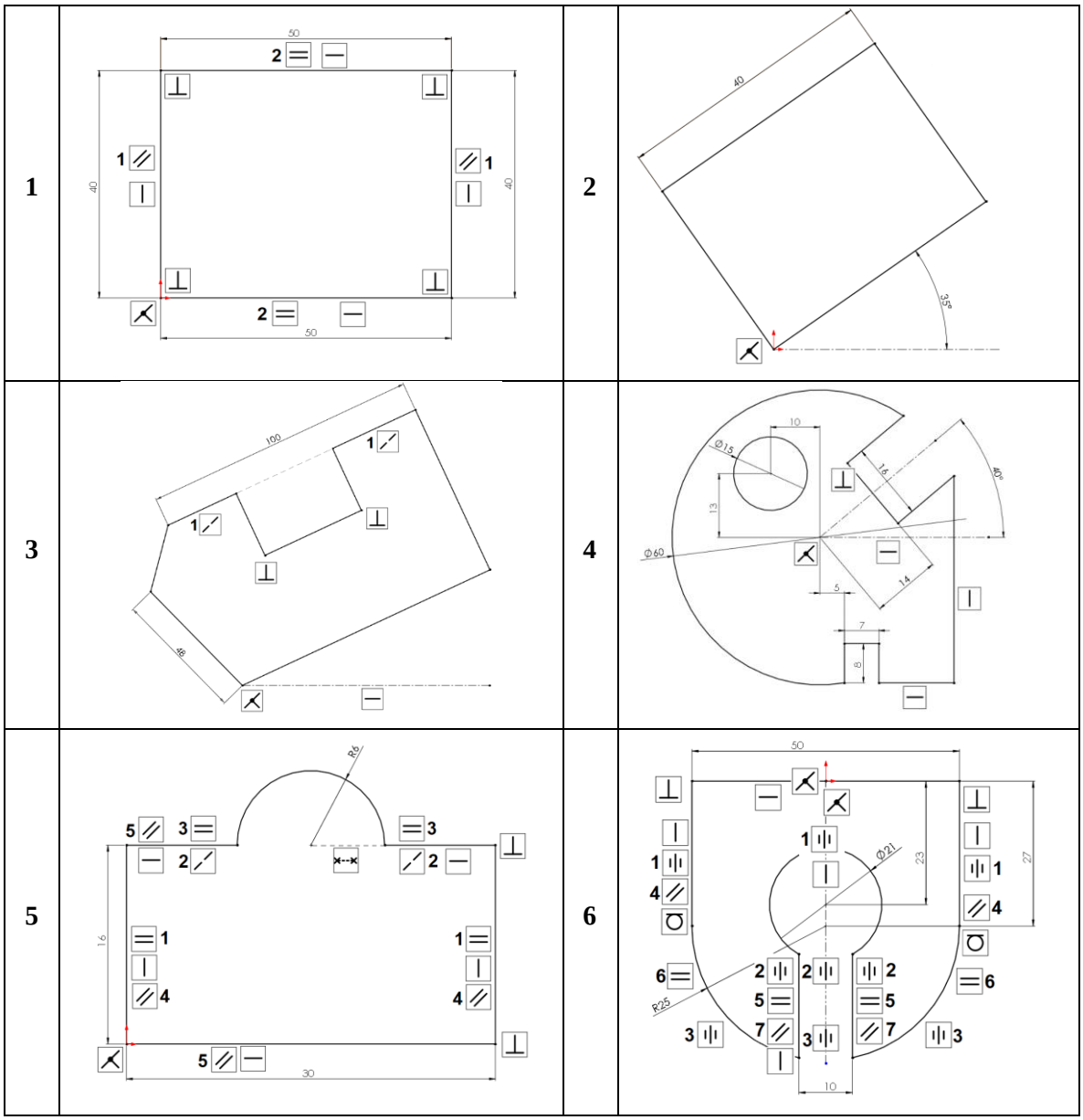


Fig. 12. Profiles used in experiment #1.

Our results are summarized in Table 6. The state of the initial profiles is shown in column 2. Specifically, profiles 1, 5, and 6 are over-constrained, and profiles 2, 3, and 4 are under-constrained. The number of CDOF in the initial profiles before any improvements is shown in column 3. The number of constrained degrees of freedom (CDOF) in each solution are shown in columns 4 (experts) and 6 (ATA algorithm) based on the use of both geometric and dimensional constraints. It should be noted that significantly different solutions were submitted by the experts (for instance, ranging from 8 to 14 CDOF for Profile 1, which initially included 16 CDOF). The frequencies for each alternative are tabulated in the *%freq Expert* column. The column "ATA qty sol" defines the number of solutions proposed by the ATA algorithm in each case.

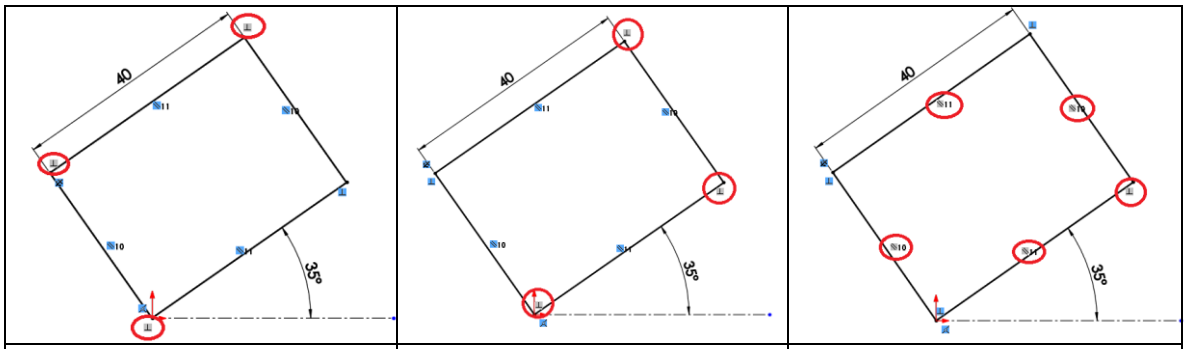
Table 6. Results (experiment #1).

Nº profile	Initial state	Initial CDOF	CDOF Expert solutions	%freq Expert	CDOF ATA solutions	%freq ATA	ATA qty sol
Profile 1	Over-constrained	16	8	46.7%	8	100%	10
			9	20%			
			10	6.7%			
			12	6.7%			
			14	20%			
Profile 2	Under-constrained	4	9	86.7%	9	100%	5
			10	13.3%			
Profile 3	Under-constrained	8	18	86.7%	18	100%	14
			19	13.3%			
Profile 4	Under-constrained	16	24	84.6%	24	100%	8
			27	15.4%			
Profile 5	Over-constrained	18	12	50%	12	45.5%	11
			13	35.7%	13	54.5%	
			14	14.3%			
Profile 6	Over-constrained	31	21	6.7%	21	5.9%	17
			22	20%	22	94.1%	
			23	40%			
			24	6.7%			
			25	13.3%			
			27	6.7%			
			30	6.7%			

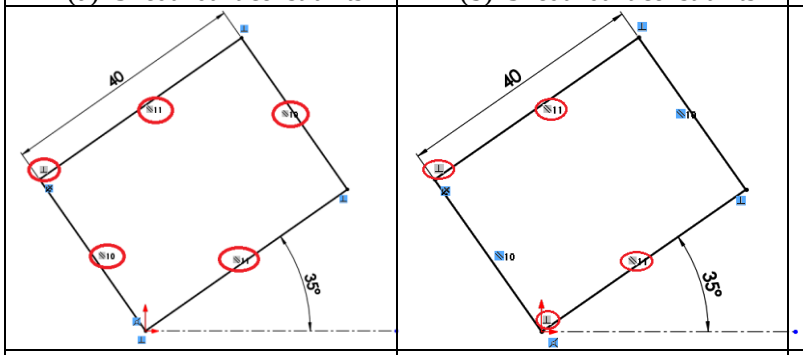
The results show that the most frequent CDOF in the solutions proposed by the experts generally match those of the algorithm. In Profile 5, the solutions are similar (solutions with 12 and 13 CDOF) for both groups. There is not a clear solution, but the two alternatives are the same in both cases. In profile 6, despite the variation among the solutions provided by experts, the most frequent solutions revolve around 22 and 23 CDOF, which correspond to the solution provided by our ATA algorithm. Therefore, the analysis of the data validates the hypothesis that the proposed algorithm performs at least as well as expert designers when fully constraining a profile while avoiding redundant constraints.

Additional observations from the results reveal that when the initial profile is over-constrained, the dispersion in the solutions provided by the experts seems to be greater than in the case of under-constrained profiles. This result proves that redundant constraints are difficult to find even for expert engineers. Conversely, there is more consistency in the case of under-constrained profiles since experts generally stop adding constraints to the profile when the CAD application changes the profile state to fully constrained. Two experts submitted profile 4 as under-constrained, thus the sample of expert solutions was reduced to 13 solutions. It should also be noted that in all the examples, the ATA algorithm found more optimized solutions than the experts.

The different sets of redundant constraints identified by the ATA algorithm are illustrated in Figs. 13 and 14. The solutions for profile 2, which was originally under-constrained, are shown in Fig. 13. We assume that the profile was initially constrained by applying the heuristic criteria and then the sets of redundant constraints (which should be removed to produce a fully defined profile) were identified. The sets of redundant constraints obtained for profile 5, which was originally over-constrained, are shown in Fig. 14.

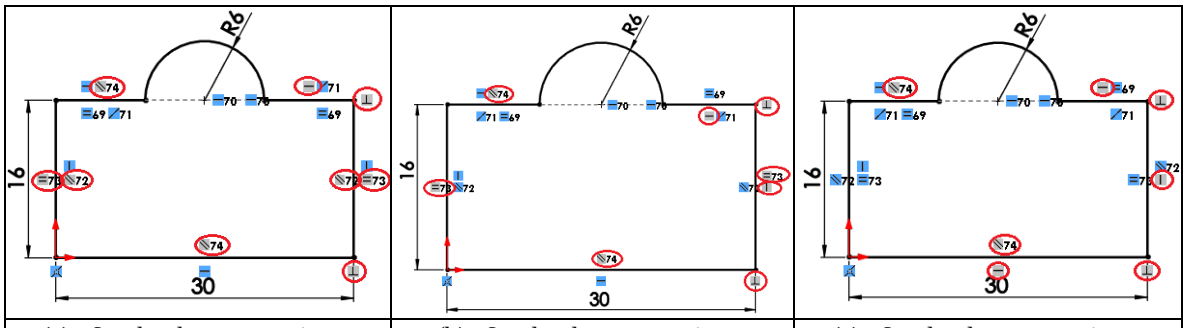


(a) 3 redundant constraints (b) 3 redundant constraints (c) 3 redundant constraints

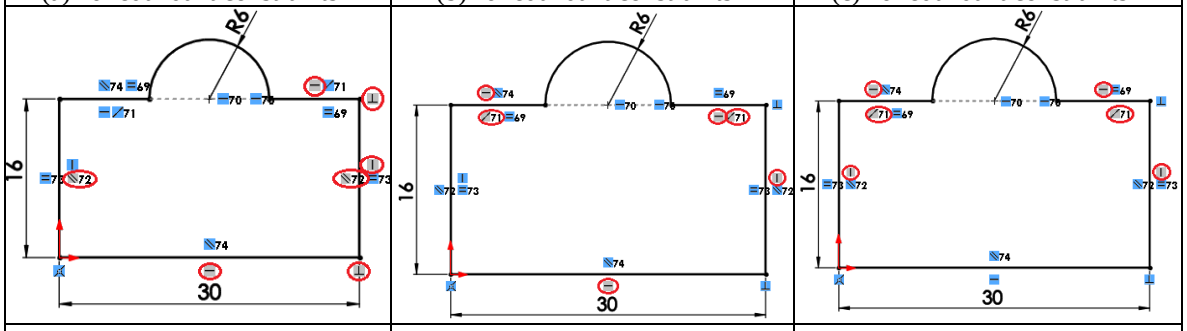


(d) 3 redundant constraints (e) 3 redundant constraints

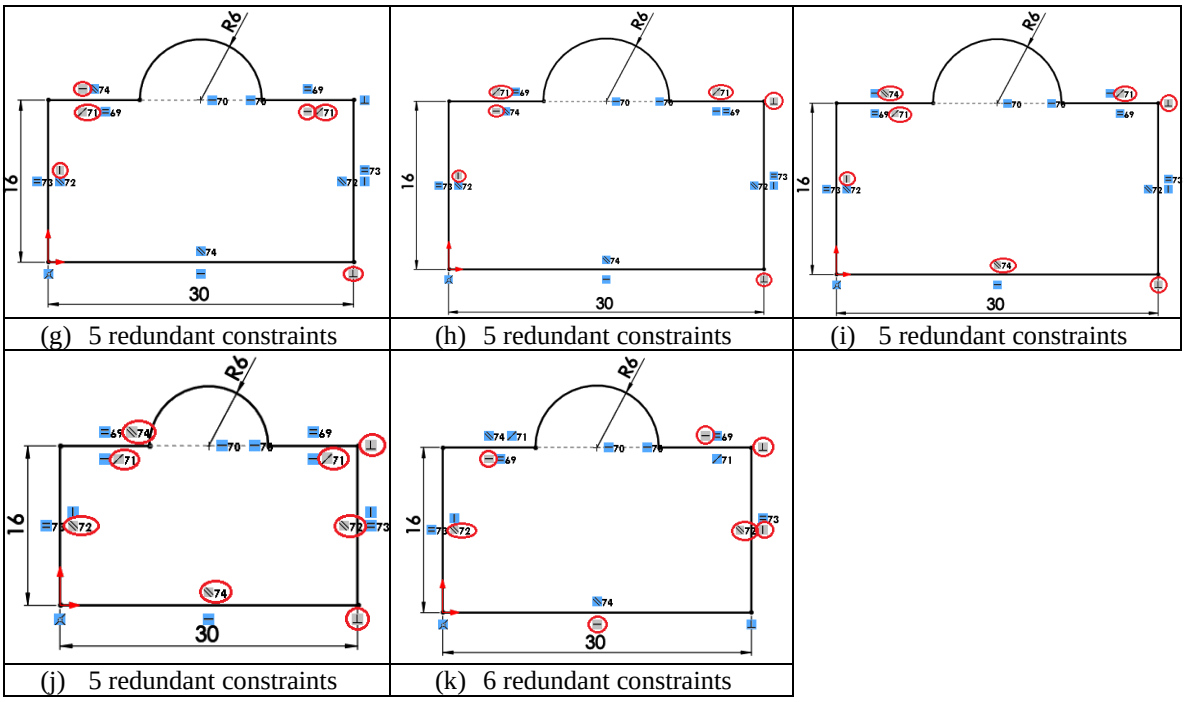
Fig. 13. Solutions provided by the ATA algorithm for sketch 2 (original sketch was under-constrained)



(a) 6 redundant constraints (b) 6 redundant constraints (c) 6 redundant constraints



(d) 6 redundant constraints (e) 5 redundant constraints (f) 5 redundant constraints



(g) 5 redundant constraints | (h) 5 redundant constraints | (i) 5 redundant constraints

(j) 5 redundant constraints | (k) 6 redundant constraints

Fig. 14. Solutions provided by the ATA algorithm for sketch 5 (original sketch was over-constrained)

Experiment #2

The goal of this experiment was to validate the hypothesis that there are no differences in terms of reusability between 2D profiles enhanced by the ATA algorithm and those by experts. The same group of experts from the previous study participated in this experiment.

Expert participants were split into three groups of five individuals and asked to perform a series of modifications (Fig. 15) to three different groups of profiles selected from experiment #1: (1) the first group of experts performed the modifications directly to the original profiles (under and over-constrained profiles of Fig. 12), (2) the second group performed the redesigns on a selection of the profiles improved by the expert's results (selected from #1.3), (3) the last group of experts performed the redesigns on a selection of the profiles improved by the ATA algorithm from #1.2.

The profiles for both the ATA and expert groups were selected from the solutions with higher %freq (Table 6). In the case of profile 6, the expert solution that was most similar to the ATA profile was selected. Each group of profiles was randomly distributed to the experts as SolidWorks files. Participants were asked to modify the constraints without removing any lines from the profile. A total of 14 redesigns were submitted: 5 redesigns from the original profiles, 4 from experts, and 5 from the ATA profiles. Our analysis focused on the following aspects:

- Success rate. The redesign was considered successful when all the modifications to the profile were conveyed correctly and the profile was fully constrained.
- Average editing time (in seconds) that participants within the same group spent performing the redesign modifications. To this end, a SolidWorks plugin was implemented to collect the editing time.
- Amount of CDOF applied in the solutions.

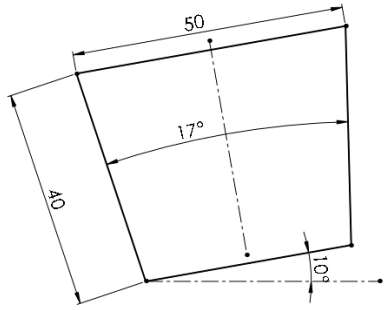
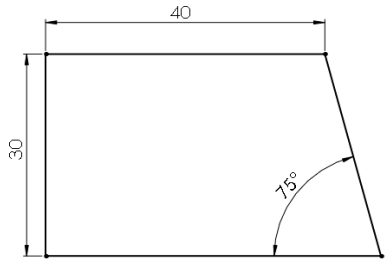
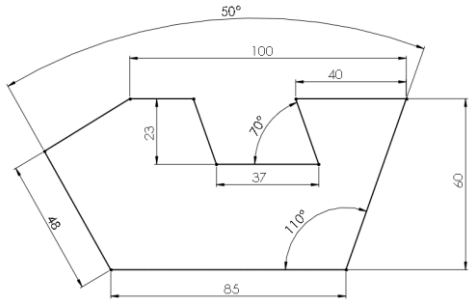
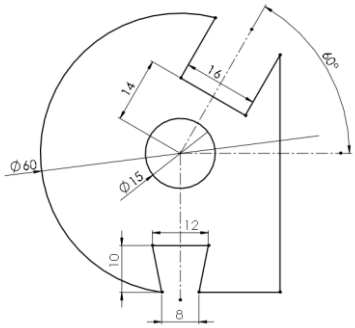
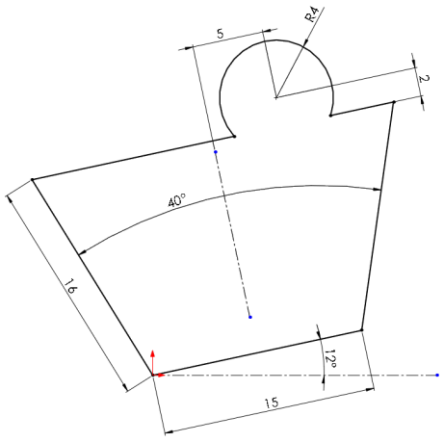
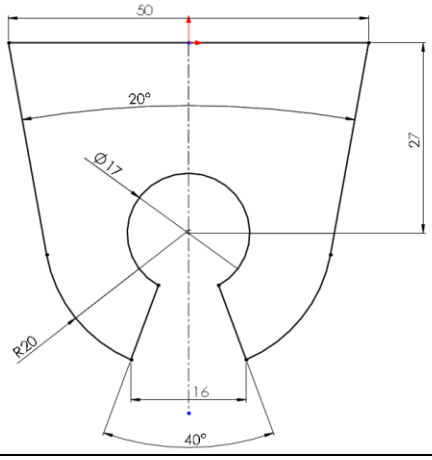
Profile	Redesign	Profile	Redesign
1		**2**	
3		**4**	
5		**6**	

Fig. 15. Redesigns proposed in experiment #2.

The analysis of the redesigned profiles is summarized in Tables 7 and 8. The success ratios reflected in Table 7 are similar for both experts and ATA groups. However, the success ratio decreases in examples 3 and 6 of the group of original profiles. These results corroborate other studies that show that under- and over-defined profiles are sources of potential errors [13-15]. The main causes of failure were unconstrained vertices (Fig. 16, left) (only occurred once), incorrect geometry (Fig. 16, middle), and errors in some dimensions (Fig. 16, right). It is important to note that none of the solutions included fix constraints (although examples 2, 3, and 4 in the ATA group did include them in the initial profiles), which proves that fix constraints are not appropriate, as proven by [17].

Table 7. Summary of results (experiment #2).

Nº Profile	Success Rate			Editing Time (seconds)					
	Original	Experts	ATA	Mean Orig.	STD	Mean Exp.	STD	Mean ATA	STD
Profile 1	100%	100%	80%	294.8	141.8	244.3	182.4	176.5	104.3
Profile 2	100%	100%	100%	95.4	42.5	94.3	17.8	87.2	31.7
Profile 3	60%	75%	100%	252.7	82.7	337.0	131.7	269.0	76.2
Profile 4	100%	75%	80%	363.0	140.5	331.3	64.6	331.0	143.2
Profile 5	80%	75%	100%	422.5	167.9	282.0	72.2	302.4	94.4
Profile 6	40%	100%	80%	336.0	86.3	334.3	142	278.8	74.5

Regarding editing time, only the files that were successfully redesigned were included in Table 7. In general, the highest values of editing time are related to the cases in which the initial profile has a greater number of CDOF (e.g. profiles 1, 5, and 6 of the original profile group), which are also the profiles that contain more redundant constraints. However, our sample is not large enough to determine statistical significance.

The results shown in Table 8 confirm the conclusions from experiment #1. First, the highest frequencies of CDOF in the redesigns proposed by the experts agree with those by the ATA algorithm. Second, the more redundant constraints there are in the initial profiles, the greater the dispersion in the solutions. Once again, this fact is mostly shown in profiles 1, 5, and 6 where the initial profiles of the original group contain redundant constraints, whereas the initial profiles of experts and ATA groups are fully constrained profiles.

Table 8. Summary of CDOF of the profiles after redesign in experiment #2

	Original Profiles			Expert Profiles			ATA Profiles		
Nº profile	Initial CDOF	Redesign CDOF	% freq	Initial CDOF	Redesign CDOF	% freq	Initial CDOF	Redesign CDOF	% freq
Profile 1	16	11	20%	8	11	25%	8	11	25%
		13	20%		12	25%		13	25%
		14	40%		14	50%		14	50%
		15	20%						
Profile 2	4	8	80%	9	8	75%	9	8	80%
		9	20%		10	25%		9	20%
Profile 3	8	18	25%	18	17	25%	18	18	60%
		19	25%		18	75%		19	20%
		20	50%					20	20%
Profile 4	16	23	20%	24	23	50%	24	23	40%
		24	40%		25	25%		24	20%
		27	20%		26	25%		26	20%
		31	20%					28	20%
Profile 5	18	15	20%	12	15	25%	12	16	20%
		17	20%		17	50%		17	60%
		18	20%		20	25%		19	20%
		19	20%						
		22	20%						
Profile 6	31	22	40%	22	21	25%	22	22	60%
		23	20%		22	50%		23	20%
		24	40%		24	25%		25	20%

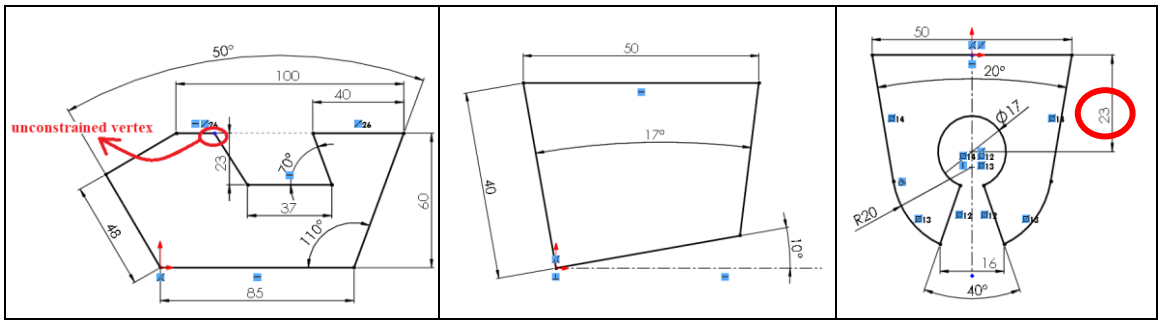


Fig. 16. Causes of unsuccessful redesigns in experiment #2. Unconstrained vertex (left), incorrect geometry (middle), and incorrect dimension (right).

Our pilot experiment has a low statistical power due to the reduced sample of experts who participated in the experiment. However, results are conclusive enough to support our initial hypothesis that the ATA algorithm performs at least as well as expert designers when optimizing profile constraints without redundant constraints. Our results also lead us to hypothesize that fewer redundant constraints imply more agreement between designers.

Our tool can also fully define an unconstrained profile. However, further improvements are needed to reduce the computational cost of the algorithm and more extensive tests are required to increase the statistical power of our analyses. As future work, we are interested in examining the influence of different types of constraints on profile quality. This analysis may define a methodology that can be implemented by our algorithm so the most appropriate constraints can be selected during the optimization process, eliminating redundant constraints, and maximizing design intent.

This approach demonstrates a practical application that not only increases the efficiency of our original tool for detecting redundant constraints, but also suggests alternative more effective constraining schemes in a given design scenario. We are currently expanding the functionalities of the system to classify solutions by quality, more specifically in terms of design intent and reusability, based on the different types of constraints. As a long-term goal, we plan to develop a quality checker to detect and measure parametric CAD quality. The checker will consist of a quality kernel wrapped by a layer of Application Programming Interfaces (APIs). The kernel will implement a set of smart heuristics which will describe, in a formalized language, the quality criteria that our research team has been developing during the past few years (including, but not limited to, constraint redundancy conditions). The layer of APIs will connect the kernel to the different commercial CAD systems and provide various quality assurance services required throughout the different stages of the product development process.

The prototype tool uses fix constraints to temporarily produce a fully-constrained sketch which is convenient for our analyses. Those temporary constraints should not be confused with permanent constraints in the final sketch, which should not have any fix constraints. This is in fact, an additional improvement we are contemplating for future versions of the tool. Ultimately, we should be able to replace these fix constraints by permanent constraints that are better suited to each particular sketch, thus providing a significant level of automation to the sketching and constraining processes.

6. Conclusions

In our view, redundant constraints in 2D profiles of parametric 3D CAD models unnecessarily compromise model conciseness and reusability, and hinder design intent communication. Arguably, constraint redundancy may also compromise the efficiency of geometric constraint solvers. Nevertheless, we consider this field to be sufficiently mature to provide valid solutions, and have focus our efforts on the implications of constraint redundancy conditions on design intent and CAD model reusability. In our analysis of CAD systems, we identified two strategies to manage constraint redundancy conditions: a strategy that allows redundant constraints and enables the user to design

more quickly at the expense of creating models that are less portable, and a strategy that is less permissible with incompatible and redundant constraints but results in more portable and robust models. Both strategies align with the extreme cases of *granularity* in CAD models, as defined in the field of CAD model exchange and interoperability. None of the systems we tested provides any assistance or feedback to users during the parametric sketching process. Therefore, we classify them as *coarse granularity* systems in terms of their ability to facilitate sketch interactions and enable effective modeling schemes for conveying a specific design intent and leveraging reusability.

Our experiments confirmed the hypothesis that redundant constraints are a common tendency among CAD users and significantly affect the time and effort required to modify a parametric profile. Our results also showed that implicit geometric constraints and their effect on profiles are not evident to many users. In fact, most are not aware of the amount or the types of constraints that are added to a profile during sketching processes. Therefore, we conclude that a lack of redundant constraints can be considered an appropriate quality metric for parametric profiles.

Current CAD applications do not provide mechanisms to support decision making in constraint redundancy scenarios. Efficient strategies and decisions for deleting redundant constraints in a manner that ensures proper communication of design intent and the maximization of model reuse are left entirely to the user. Researchers have stated that an effective strategy to eliminate, or at least minimize, constraint redundancy conditions requires developing proper training and providing real-time feedback during the modeling session on the adequacy of the constraining strategy that is being used. To this end, an automated mechanism that can assist users during the creation of profiles could be of great value. To explore the viability of such a tool, we developed a plugin for a CAD system that can detect redundant constraints on demand and provide feedback to users during the parametric sketching process. Our system can suggest alternative solutions to a constraint schema that uses more appropriate and effective combinations of constraints, which can assist users in creating parametric sketches without constraint redundancy conditions and thus increasing quality. This pilot tool has been proven to be valuable, but further developments are required. As future work, we plan to simplify our algorithm to allow for on-line use by detecting redundant constraints to automatically check the profile every time a rebuild operation occurs. Finally, we are interested in further studying the types of constraints that may result in higher quality parametric profiles, and implementing these findings in the prototype tool described in this paper.

ACKNOWLEDGMENTS

This work was partially supported by UJI-A2017-15 (Universitat Jaume I), project "New metrics to improve the quality of 3D CAD models" and 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.”

REFERENCES

[1] Amadori, K., Tarkian, M., Ölvander, J., Krus, P. (2012). Flexible and robust CAD models for design automation, Advanced Engineering Informatics 26, 180–195. https://doi.org/10.1016/J.AEI.2012.01.004.

[2] Eltaief, A., Louhichi, B., Remy, S. (2018). Associations management and change propagation in the CAD assembly, Computers in Industry 98, 134-144. https://doi.org/10.1016/j.compind.2018.02.012.

[3] Bodein, Y., Rose, B., Caillaud, E. (2014). Explicit reference modeling methodology in parametric CAD system, Computers in Industry 65:1, 136–147. https://doi.org/10.1016/j.compind.2013.08.004.

[4] Camba, J.D., Contero, M., Company, P. (2016). Parametric CAD modeling: An analysis of strategies for design reusability, Computer-Aided Design 74, 18-31. https://doi.org/10.1016/j.cad.2016.01.003.

[5] Camba, J. D., Cosin, A., Contero, M. (2014, November). An evaluation of formal strategies to create stable and reusable parametric feature-based 3D models. In ASME 2014 International Mechanical Engineering Congress and Exposition. American Society of Mechanical Engineers Digital Collection. https://doi.org/10.1115/IMECE2014-37859

[6] Shah, J.J. (1998). Designing with parametric CAD: Classification and comparison of construction techniques, International Workshop on Geometric Modelling, 53-68. Boston, MA. https://doi.org/10.1007/978-0-387-35490-3_4

[7] Ault, H.K. (2004). Over-Constrained , Under-Constrained or Just Right? Goldilocks Evaluates DOF of Sketched Profiles, in: American Society for Engineering Education. 59th Annual Midyear Meeting Past, Present & Future?, Williamsburg, Virginia., pp. 127–131.

[8] Lukaszewicz, A., Skorulski, G., Szczebiot, R. (2018). Main aspects of training in field of Computer-Aided techniques (CAx) in mechanical engineering, in: engineering for rural development, Jelgava, 865–870. https://doi.org/10.22616/ERDev2018.17.N493.

[9] Otey, J., Company, P., Contero, M., Camba, J.D. (2018). Revisiting the design intent concept in the context of mechanical CAD education, Computer-Aided Design and Applications 15, 47–60. https://doi.org/10.1080/16864360.2017.1353733.

[10] Otey, J.M., Company, P., Contero, M., Camba, J.D. (2014). A Review of the Design Intent Concept in the Context of CAD Model Quality Metrics. 2014 ASEE Annual Conference & Exposition, Indianapolis, Indiana. 10.18260/1-2--19992.

[11] Alducin-Quintero, G., Rojo, A., Plata, F., Hernández, A., Contero, M. (2012). 3D model annotation as a tool for improving design intent communication: A case study on its impact in the engineering change process. In Proceedings of the ASME design engineering technical conference, 2012 (parts A and B, vol. 2, pp. 349–356). https://doi.org/10.1115/DETC2012-70872)

[12] ISO 10303-108: 2005. Industrial automation systems and integration—Product data representation and exchange: Integrated application resource: Parameterization and constraints for explicit geometric product models. International Organization for Standardization, Geneva, Switzerland.

[13] Company, P., Contero, M., Otey, J., Plumed, R. (2015). Approach for developing coordinated rubrics to convey quality criteria in MCAD training, Computer-Aided Design. 63, 101–117. https://doi.org/10.1016/j.cad.2014.10.001.

[14] Otto, H.E., Mandorli, F. (2015). A framework to support 3D explicit modeling education and practice, Computer-Aided Design and Applications 12(1), 104-117. https://doi.org/10.1080/16864360.2014.949581.

[15] Branoff, T., Devine, K.L., Brown, J. (2016). Evaluating a rubric for assessing constraint-based solid models, In American Society for Engineering Education Annual Conference & Exposition, New Orleans, LA.

[16] González-Lluch, C., Company, P., Contero, M., Camba, J.D., Plumed, R. (2017). A Survey on 3D CAD Model Quality Assurance and Testing Tools, Computer-Aided Design, 83, 64-79. https://doi.org/10.1016/j.cad.2016.10.003.

[17] González-Lluch, C., Company, P., Contero, M., Pérez-López, D., Camba, J.D. (2018). 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 29, 821–841. https://doi.org/10.1007/s10798-018-9458-z.

[18] Autodesk, Glossary for the Engineer's Handbook | Inventor 2018 | Autodesk Knowledge Network, (n.d.). https://knowledge.autodesk.com/support/inventor/learn-explore/caas/CloudHelp/cloudhelp/2018/ENU/Inventor-Help/files/GUID-3E0794C5-75B4-4425-8163-45FCCF8597B3-htm.html (accessed April 11, 2020).

[19] Paulraj, S., Sumathi, P. (2010). A Comparative Study of Redundant Constraints Identification Methods in Linear Programming Problems, Mathematical Problems in Engineering. 2010, 1–16. https://doi.org/10.1155/2010/723402.

[20] Sumathi, P., Yuvarekha, K., Nandharkumar, V., Karunagaran, K. (2015). Basics of Redundancy and Solutions in Linear Programming Problems, International Journal of Innovate Research in Science, Engineering and Technology, 4, 596-598.

[21] Estiningsih, Y., Farikhin, Tjahjana, R.H. (2019). Modified Stojkovic-Stanimirovic method to find redundant constrains in linear programming problems, Journal of Physics: conference Series 1321.

[22] Siemens Industry Software Limited, D-Cubed 2D DCM Manual Version 69.0, (2018) 1-360.

[23] FreeCAD online tutorials: https://wiki.freecadweb.org/Online_Help_Toc, last access March 2021.

[24] Salome (software). March 2021. In wikipedia. URL: https://en.wikipedia.org/wiki/Salome_(software)

[25] Aldefeld, B. (1988). Variation of geometries based on a geometric-reasoning method, Computer-Aided Design 20, 117–126. https://doi.org/10.1016/0010-4485(88)90019-X.

[26] Race, P. (2013). The lecturer's toolkit : a practical guide to assessment, learning and teaching, Routledge-Falmer, Glasgow, Great Britain, 2013.

[27] Kirstukas, S.J. (2016). Development and Evaluation of a Computer Program to Assess Student CAD Models, in: ASEE Annual Conference & Exposition, Paper ID15834, New Orleans, Lousiana, 26–29.

[28] Tshibalo, A.E. (2007). The Potential Impact of Computer-Aided Assessment Technology in Higher Education, South African Journal of Higher Education, South African Journal of Higher Education 21, 695.

[29] Ault, H.K., Fraser, A. (2013). A Comparison of Manual vs. Online Grading for Solid Models, in: 120th ASEE Annual Conference & Exposition, Atlanta, American Society for Engineering Education, 23–26.

[30] Hekman, K.A., Gordon, M.T. (2013). Automated Grading of First Year Student CAD Work, in: 120th ASEE Annual Conference & Exposition, Atlanta, GA, 23–26.

[31] Cheng, Z., Ma, Y. (2017). A functional feature modeling method, Advanced Engineering Informatics 33, 1–15. https://doi.org/10.1016/J.AEI.2017.04.003.

[32] González-Lluch, C., Plumed, R. (2019). Are we training our novices towards quality 2D profiles for 3D models?, in: Lecture Notes in Mechanical Engineering, Cartagena (Spain), 714–721. https://doi.org/10.1007/978-3-030-12346-8_69.

[33] Plumed, R., González-Lluch, C., Otey, J., Pérez-Belis, V. (2021). Training Engineers in the Use of Constraints to Create Quality 2D Profiles for 3D Models. Computer-aided design and applications 18(3), 612–623.

[34] Kondo, K. (1992). Algebraic method for manipulation of dimensional relationships in geometric models, Computer-Aided Design, 24, 141-147. https://doi.org/10.1016/0010-4485(92)90033-7.

[35] Ge, J.X., Chou, S.C., Gao, X.S. (1999). Geometric constraint satisfaction using optimization methods, Computer-Aided Design, 31, 867-879. https://doi.org/10.1016/S0010-4485(99)00074-3.

[36] Mantyla, M., Shah, J. (1995). Parametric and Feature Based CAD/CAM: Concepts, Techniques and Applications, John Wiley & Sons, 1995.

[37] Shih, C.H., Anderson, B. (1997). A design/constraint model to capture design intent, SMA '97 Proceedings of the fourth ACM symposium on Solid modeling and applications, 255-264.

[38] Company, P., Naya, F., Contero, M., Camba, J.D. (2020). On the role of geometric constraints to support design intent communication and model reusability**.** *Computer-Aided Design and Applications,* 17(1), 61-76. https://doi.org/10.14733/cadaps.2020.61-76.

[39] Pratt M.J., Kim J. (2006) Experience in the exchange of procedural shape models using ISO 10303 (STEP). SPM '06: Proceedings of the 2006 ACM symposium on Solid and physical modeling, pp. 229–238. https://doi.org/10.1145/1128888.1128920.

[40] Kim J., Pratt M.J., Iyer R.G., Sriram R.D. (2008) Standardized data exchange of CAD models with design intent. Computer-Aided Design, 40, 760–777. https://doi.org/10.1016/j.cad.2007.06.014.

[41] S.C. Dassault Systèmes, 3D ACIS Modeler | Spatial, (2019). https://www.spatial.com/products/3d-acis-modeling (accessed April 10, 2020).

[42] Tech Soft 3D, Parasolid Geometric Modeling | Tech Soft 3D, (2019). https://www.techsoft3d.com/products/parasolid/ (accessed April 10, 2020).

[43] ASCON, C3D Labs, (n.d.). https://c3dlabs.com/en/ (accessed April 10, 2020).

[44] Open CASCADE (2020) https://www.opencascade.com/ (accessed June 9, 2020)

[45] BRICSYS, LGS 2D Geometric Solver Overview, (2013) 1–28.

[46] SIEMENS, PLM−Product Lifecycle Management: Siemens PLM Software, (n.d.). https://www.plm.automation.siemens.com/global/en/ (accessed April 10 2020).

[47] Bricsys NV, Bricsys Component Technology, (n.d.). https://www.bricsys.com/en-intl/applications/developers/components/ (accessed April 10, 2019).

[48] Dassault systemes, SolidWorks web help, 2021. http://help.solidworks.com/HelpProducts.aspx (accessed January, 2021)

[49] Company, P., González-Lluch, C. (2018). CAD 3D con SolidWorks Tomo I: Diseño básico. Publicacions Universitat Jaume I. Ed. 2. http://cad3dconsolidworks.uji.es.