Improved Computational Tools for Concept Development based on Sketches and Advanced CAD Technologies

Nickolas Sapidis[1], Sofia Kyratzi[1] and Philip Azariadis[1,2]

[1]University of the Aegean, {sapidis, skiratzi, azar}@aegean.gr
[2]ELKEDE – Technology & Design Centre SA

ABSTRACT

Sketching continues to be a vital part of methods and technologies related to Concept Development, including those analyzed in this paper: "concept design with constraints", "solid from sketch", "sketching on a point cloud" and "geometric modeling of design constraints". Existing methods are analyzed and critical subproblems are identified. New solution methods are proposed for the subjects "solid from sketch" and "sketching on a point cloud".

Keywords: Graph-theoretic analysis of a sketch, 3D digital curves, solid modeling of free spaces.

1. INTRODUCTION

Sketching continues to be one of the most important modeling activities in Concept Design & New Product Development, since it is a powerful tool for these early stages of the Product Life-Cycle. Indeed, recent research establishes that a sketch is much more than a vague description of 2D geometry. A sketch often includes information beyond the geometric description of the product, like a functional decomposition of the product, diagrammatic analyses of nongeometric concepts, and elements related to the designer's work method. A sketch is not just an "output device" but also an "input device" for the designer's mental processes. This paper examines four subjects, related to Concept Development, all of which involve sketching. More specifically, Section 2 presents a critical view on standard "concept design with constraints" and identifies shortcomings in this methodology. Section 3 focuses on the "solid from sketch" problem and proposes a new method that is not using standard heuristics, which, according to recent research, are highly unreliable. Section 4 explores a new subject, sketching on a point cloud, in the context of "point-based CAD & Concept Development". Finally, Section 5 deals with complex design-constraints and elaborates on a new solution methodology that combines Solid Modeling with Sketching and Geometric-Constraint Solving.

2. TWO-DIMENSIONAL CONCEPTUAL DESIGN: A METHODOLOGY AND CAD SYSTEM BASED ON SKETCHES AND CONSTRAINTS

We wish to comment on the standard "design with constraints" (DwC) methodology underlying many recent research-contributions in the area of "conceptual geometric design". DwC is also the foundation of the "concept-design component" of many current high-end mechanical CAD systems. We precede our critique with a short description of DwC, based on the recent publication [8]. (The first author of this publication holds full responsibility for the present summary of DwC extracted from [8].) According to [8], a constraint-based CAD-system for concept design includes two independent modules:

1. The *sketcher module* used by the designer in defining the mechanical part's parametric geometry (in 2D), and all geometric constraints that the final design must satisfy.
2. The *constraint solver*, which resolves the intended geometry and dimensions the mechanical part.

This CAD system is based on the following methodology:

2.1 "Design with Constraints" Methodology

The design process is divided into three phases [8, Sect. 1]: (a) sketching, (b) constraint definition and (c) constraint solving. In (a) and (b) the system only assists the user in faster specifying his/her intention. Part (c) is performed by the CAD system. The user provides a sketch, including linear and curved segments, associated to a *fixed number* of design variables (also called "generalized coordinates"); to finalize the design, one must specify values for all these variables.

The user complements the given sketch with a set of geometric constraints, expressed as mathematical equations, aiming at a full description of the given design-problem.

Most often, the system of geometric constraints has no unique solution, and some processing of them is required. Obviously, when the number of constraints is too small, the user is prompted to define additional constraints. The opposite situation, where some of the given constraints are redundant, is much harder to deal with. Indeed, [8] emphasizes that specification of a set of independent constraints "is essential to keeping the geometrical shape (of the given sketch)". [8] proposes a method for that based on analyzing the Jacobian matrix of the system of equations: the system identifies constraints with a "low pivot" and prompts the user to eliminate enough conditions until the number of constraints is equal to the number of unknown variables.

2.2 A Critique of the "Design with Constraints" Methodology

These are some obvious shortcomings of DwC:

1. According to DwC, the designer first creates a sketch of the part and, on the basis of that, he/she states the corresponding geometric constraints. In other words, the designer first specifies, in a completely ad hoc manner, a major part of the *"solution to the design problem"* (: parametric form of the part) and then uses that to state the *"design problem"* (: constraints) to be solved!

2. Even in [8], the geometric forms considered are very simple: combinations of lines, circles and arcs! The same is true for the constraints treated: dimensional constraints and the standard "shape constraints" dealing with perpendicularity, horizontality, symmetry, etc. A DwC methodology to be useful to a modern designer should include both advanced geometry (spline curves, set operations, etc) and also advanced constraints like G1, C1, C2-continuity, "regional constraints" and "interference constraints" [9].

3. DwC is based on the assumption that the design is fully described by a prescribed number of variables arbitrarily specified by the user during sketching. This assumption is complete unjustifiable.

4. When the derived system of mathematical equations is not solvable, DwC resolves the situation by adding or deleting constraints, i.e., by modifying the *"design problem"* to be solved. At the same time, the initial, ad hoc sketch is not modified. It is obvious that, for any solution strategy to be reasonable, exactly the opposite should be done.

5. DwC assumes that all geometric constraints may be expressed as a set of mathematical equations. Although, in a strict mathematical sense, this may be true, definitely it is not practical, e.g., when one deals with complex constraints like "regional constraints" and "interference constraints" [9].

6. Despite its long history, the DwC methodology is still limited to 2D and to simple geometry. Early techniques for 3D DwC were focusing on specific applications. (A promising effort towards 3D DwC has been recently presented by M. Sitharam et al; see [4] and references therein.)

Clearly, the most profound shortcomings of DwC are 1 & 4: The sketch, despite the fact that it is part of the "solution", precedes, and actually domineers, statement of the "design problem" to be solved. Also, when an iterative solution-process is employed, this does not touch the sketch but rather modifies the "design problem" so that it fits to the ad hoc geometric sketch provided by the user. Surprisingly, in the field of "mechanism (: device with moving parts) design", despite the fact that the corresponding design problems are far more complex, standard methodologies treat sketches in a much more reasonable manner [11]: CAD tools analyze the given sketch, repair it when flaws are identified and transform it into an abstract representation capturing desired behavior/function "while abstracting away its specific implementation" [11, Sect. 1]. Then, sophisticated methods are used that map this "abstract behavior representation" to multiple new implementations (precise models). The end result is a wide range of design alternatives and, as a by-product, a repaired version of the original sketch.

3. TWO-DIMENSIONAL CONCEPTUAL DESIGN: AUTOMATIC CONSTRUCTION OF A POLYHEDRON FROM A SINGLE LINE SKETCH

Here, the researchers' objective is development of a robust Sketch-to-Polyhedron (StP) algorithm that, from a given line-sketch, constructs a complete description of the implied polyhedron. The related literature has always been focusing on two distinct versions of this problem, involving a complete wireframe [7] or only visible lines ("natural sketch") [15], respectively. This paper deals with the latter version as this is most often involved in industrial applications. Also, the scope of this research is limited to the simplest case of a *trihedral polyhedron*, where exactly three faces meet at each vertex.

The given sketch must depict the object in "general position", i.e., no face or edge is parallel to the view direction, and it is drawn from the "most informative viewpoint", i.e., there is nothing at the rear of the object which could not be inferred from the visible part. The lines, junctions and regions of the sketch correspond to edges, vertices and faces of the implied polyhedron. The projection is assumed to be orthographic, therefore the 2D coordinates of each junction are the (x, y) coordinates of the corresponding vertex.

3.1 Line Labeling: Is It Useful?

One of the standard tools employed by StP algorithms has been "Line Labeling (LL)"; see [12], [14], [15] and references therein. LL assigns to each line of a sketch one of the labels {+, -,→,←}, meaning

"+" ("-") : The line is convex (concave), i.e., the two faces meeting along the corresponding edge form an angle > (<) 180°.

"→" : The line is occluding, i.e., of the two regions sharing this line only the one on the right side of the arrow (i.e., oriented line) defines a face including the corresponding edge; the other region describes a partially visible face.

The best-established LL methodology is based on using a *Junction Catalogue* (Fig. 1(a)) listing all possible cases of junctions found in a sketch. (Note that, in the present context of trihedral polyhedra, a T-junction always corresponds to two collinear "occluding lines" and one "occluded line".)

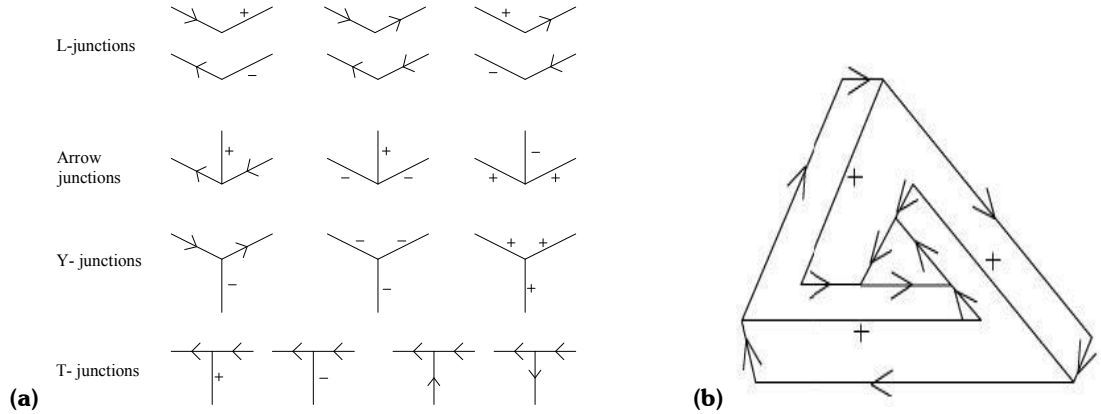


Fig. 1. (a) Junction catalogue for trihedral polyhedra. (b) Impossible object with a consistent labeling.

The LL algorithm labels lines so that the resulting configuration at all junctions belongs to the Junction Catalogue and no line receives different labels from its two endpoints [12] [15]. The algorithm first associates to each junction ***s*** a set **J(s)** of candidate labels, initially containing all possible configurations. Then, whenever an element of **J(s)** turns out to be inconsistent with labels at neighboring junctions, it is removed from **J(s)** [12]. A sketch is called *labelable* iff it can be associated to at least one valid labeling. This is a necessary condition for a sketch to correspond to a polyhedron.

The main disadvantage of LL is that there is no efficient method to examine whether a labelable sketch corresponds to a polyhedron. E.g., Sugihara [12] details a necessary and sufficient condition for that, yet this approach is numerically unstable. Regarding current research on LL, this is best described in the PhD Thesis [14] and subsequent publications by Varley et al; see references in [15]. [14] introduces a deterministic and also a probabilistic LL method aiming at "good labelings in interactive time". Although these methods are better than earlier techniques, they have many disadvantages detailed in Sect. 4.5 of [14]. The primary disadvantage is that one can easily devise counterexamples for them, i.e., simple sketches for which incorrect labels are produced; see Sect. 4.5 in [14] and Fig. 2. After presenting, in a series of papers (see references in [15]) improved versions of the methods in [14], Varley et al. reconsider the usefulness of LL [15] and conclude that this is limited only to distinguishing occluding from non-occluding T-junctions [ibid. Sect. 2.5]. They propose that emphasis must be changed away from LL and describe a method to solve the above problem without using LL.

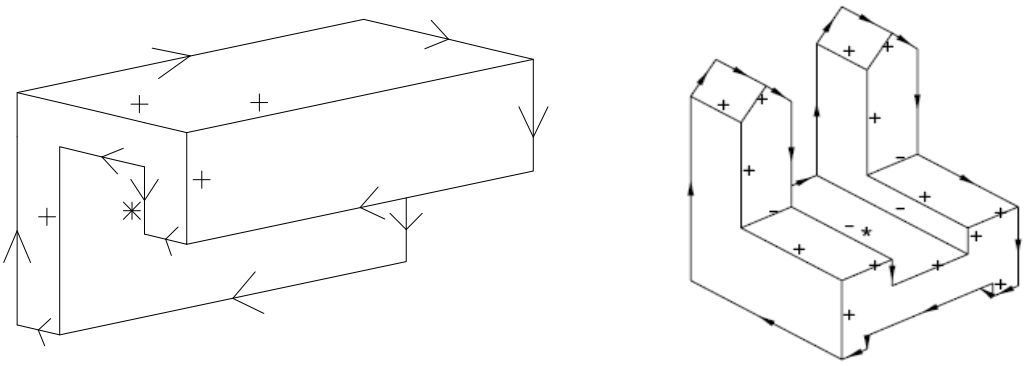


Fig. 2. From [14]: For these two sketches, the Line Labeling methods in [14] produce incorrect labelings for the edges marked with *.

3.2 Line Labeling for the "Sketch to Polyhedron" Process: Proposals for a New Research-Strategy

Although we agree with [15] that, during the last 20 years, LL has not succeeded in significantly contributing to the solution of the StP problem, we are not convinced that this failure relates to the fundamental ideas underlying LL. We suspect that this failure may be related to the following two "principles" characterizing most of the past research-efforts in the field "LL for StP":

(a) LL for Sketch-to-Polyhedron should be fast: use heuristics to achieve that.

(b) LL produces, for a given sketch, too many valid labelings: use heuristics to significantly reduce this number.

Our first strategic proposal may be stated as follows: Since not much has been achieved, on the basis of (a) and (b), LL research should consider eliminating these two principles. Research on "LL for Sketch-to-Polyhedron" should free itself from the burden of "interactive performance" and focus on producing a mathematically and algorithmically complete solution to LL. After producing a solution which is proven both theoretically and experimentally robust, then the research community should start warring about accelerating that solution or devising faster ones.

Our second strategic proposal relates to user involvement in a "LL preprocessor" and, more generally, in the "input analysis" stage of the whole StP process. We feel that user involvement in this initial stage is unavoidable as, in many cases, the geometric information implied by a sketch is extremely vague. We propose to limit the requirement of "automatic processing" only to the main part of StP (: sketch analysis and polyhedron modeling), and tolerate a system-user dialogue during the "input analysis" stage, aiming at clarifying parts of the given sketch. Clearly, this dialogue should be free of any particular StP terminology and it should use only standard CAD- and geometric-modeling-language. A typical example is presented in Fig. 3. This sketch admits six different LL interpretations, making it a non-trivial case for the StP process. However, cases (b), (d) and (f) are easily eliminated by asking the designer the very plain question *"does the face F2 touch the face F3 at the edge E2?"*. A similar query, about the edge *E3*, allows the system to eliminate also case (e). The two remaining cases, (a) and (c), must be processed by the StP algorithm, as resolving them would require asking the user "is E2 convex or concave?", which indeed involves him/her in the core of the LL procedure.

3.3 A "Sketch to Polyhedron" Method based on Graph Theory and Solid Modeling

The obvious alternative to Line Labeling is: Use "Graph Theory" to analyze the given sketch, and "Solid Modeling" to construct the corresponding polyhedron. Indeed, these two fields offer an abundance of results and algorithmic tools that have found, up to the present time, limited application in StP. The authors have been developing a method, based exactly on the above strategy, to construct the topological description of the plainest polyhedron corresponding to a given sketch. (This method will be detailed elsewhere due to space limitations.) The algorithm includes two major components: The first one applies graph-theoretic and solid-modeling results to the given sketch in order to estimate the minimum number of invisible vertices/edges/faces that should be added to the "visible topology" to produce the corresponding polyhedron. The second component constructs a complete topological-description of that polyhedron using two kinds of tools: (a) Tools that add invisible vertices/edges/faces so that each L- and each T-junction is associated to a trihedral vertex; see examples in Figs. 4(a, b) and 5(a, b). (b) Tools that simplify the evolving "invisible topology". The latter tools focus on preserving the local and global topological-validity of the polyhedron, and, at the

same time, identify as many as possible invisible edges that may be replaced by a single vertex (see example in Figs. 4(b)-(e)), and/or invisible faces that may be replaced by a single vertex or edge (see examples in Figs. 4 & 5).

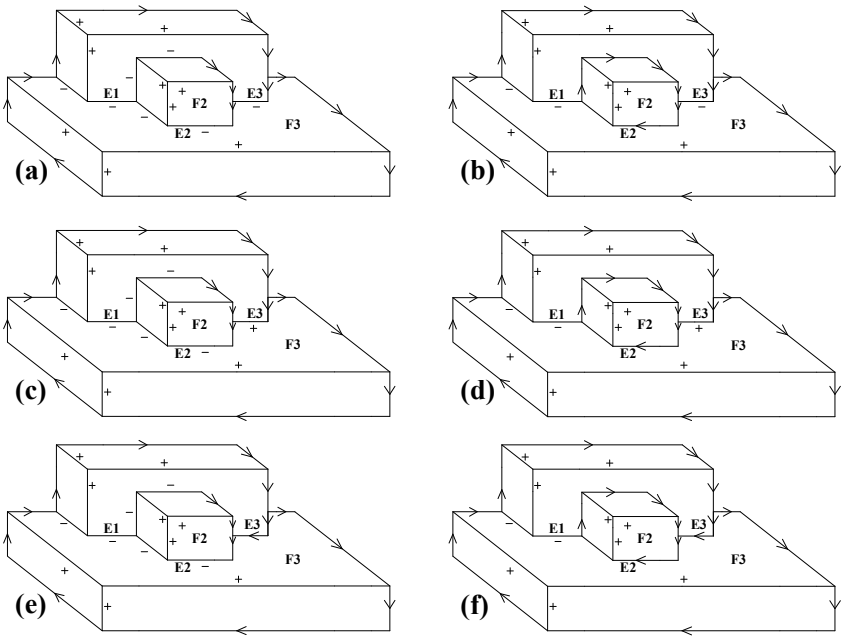


Fig. 3. A sketch admitting six different labelings.

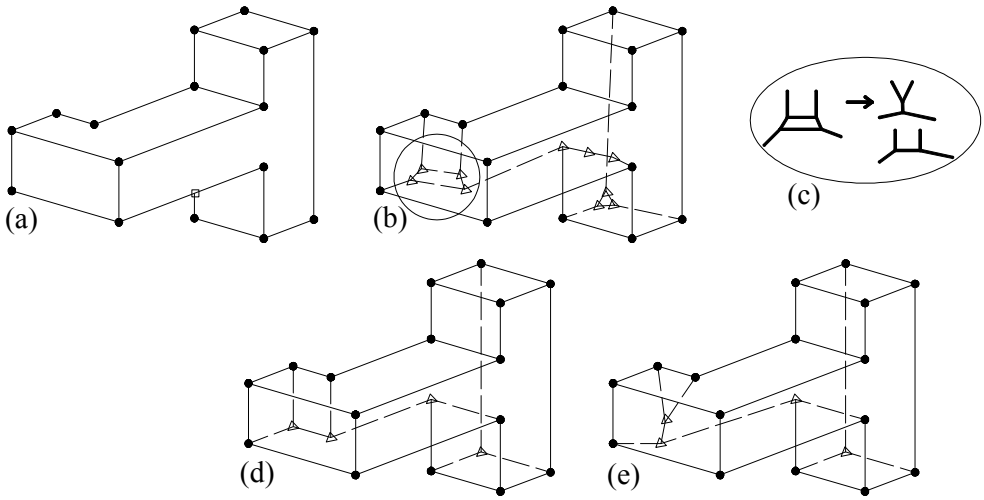


Fig. 4. Constructing the topological description of the plainest polyhedron corresponding to the sketch (a): depending on what version of the "face-simplification tool" (c) one employs, the final outcome is the polyhedron (d) or (e).

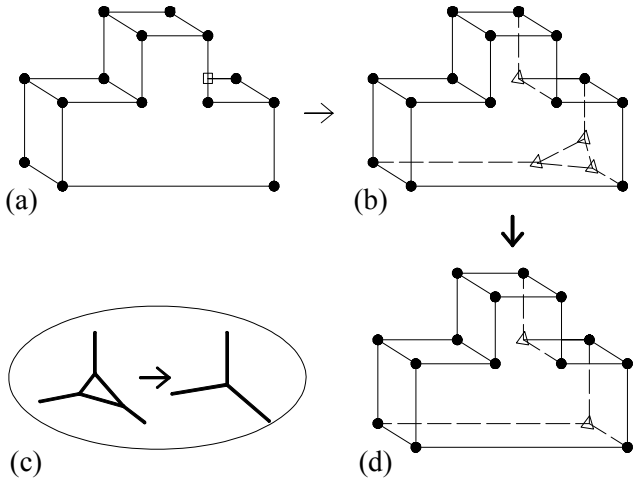


Fig. 5. Constructing the topological description of the plainest polyhedron corresponding to the sketch (a): The polyhedron (b) is the outcome of "Component 1"; see Sect. 3.3. The polyhedron (d) is the outcome of "Component 2" where the "face-simplification tool" (c) has been employed.

4. INDUSTRIAL PRODUCT DESIGN WITH POINT CLOUDS: A NEW FRAMEWORK FOR CAD

Many industrial products are designed onto certain base surfaces by following rules derived either from long-term study or from the day-to-day experience. Typical examples are the industries of apparel and footwear, which develop new products by designing them onto appropriate base surfaces (avatars and lasts, respectively). Usually, through a variety of sketches, a shoe- or a garment-stylist iteratively develops his/her ideas onto a base surface encapsulating (at the early stages of the product life-cycle) the basic product concept. Supporting such a design process in a CAD environment is not a trivial task. Existing CAD systems adopt standard geometric models based on continuous, analytic curves and surfaces (e.g., NURBS). Thus, when the input is a point cloud, a Reverse Engineering (RE) method is used to "translate" the given point-model into a curve/surface model.

There is an important problem that an industrial designer must face when working with point clouds: the reconstruction of an appropriate continuous base surface from the given point-cloud. Although, existing "specialized" CAD systems offer a variety of surfacing toolkits, it is widely accepted that - despite impressive progress during the last 10 years - RE technology is still far from offering a robust solution to the above problem. The designer, consequently, has to devote a large portion of his/her time in order to cope with the demands of this "routine task", instead of focusing exclusively on the development of his/her product concept. This has led some CAD researchers and CAD companies to consider using, as primary free-form model, a discrete model based on polygons/triangles (continuous model) or just points (discontinuous or "digital" model). The latter approach is the subject of this section: sketching digital curves onto a 3D point-cloud (also called as "digital surface").

Point-based approaches are gaining a significant attention in many fields of Computer Graphics, like "surface modeling", "rendering", etc. Regarding CAD/CAM, few researchers have already proposed point-based solutions for a variety of problems: a point-based approach to NC machining is presented in [5], while Cripps [3] presents algorithms for a comprehensive point-based CAD/CAM system. Regarding digital curves, [6] constructs a grid of curves onto a dense polygon mesh in order to fit a surface to that grid. The proposed method ignores "curve smoothness" and thus easily produces unacceptable results [2]. The latter work proposes improved techniques for constructing curves onto a dense polygon mesh, yet these cannot be applied to point clouds.

4.1 Interactive 3D Sketching onto a Digital Surface

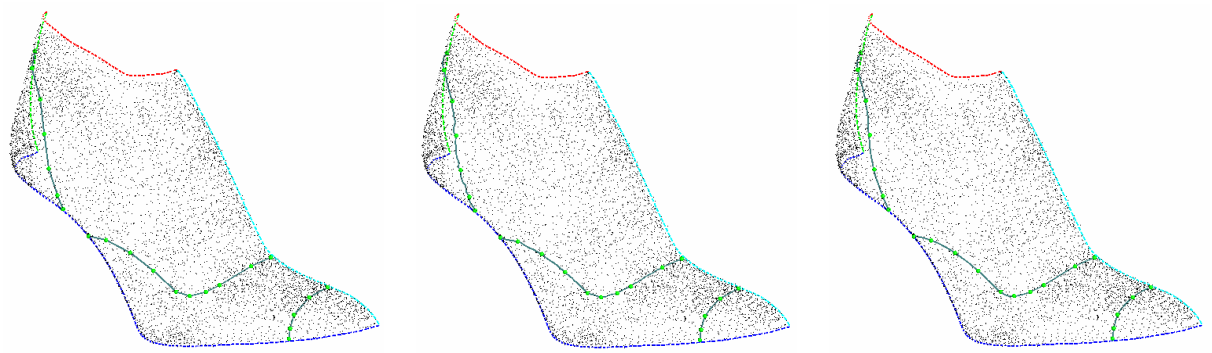
Using digital surfaces in product design necessitates the existence of efficient methods for drawing/editing digital curves (i.e., piecewise linear curves) onto a point cloud. These methods must be sufficiently robust to be used by a designer or stylist with minimal (or no) knowledge of geometric modeling. For this purpose, [1] proposes the following 3-step methodology (see also Figs. 6 and 7).

4.1.1 Step A: Sketching a "Design Polyline"

The most essential part of the proposed design-paradigm is the definition of "design polylines" that comprise the product concept. This is achieved using a directed-projection tool which allows the user to project, according to a preferred viewing direction, sequences of vertices onto the point cloud. Projected vertices are connected to each other with line segments which define a "design polyline". E.g., Figs. 6(a) and 7(a) present several design polylines defined through corresponding sets of projected vertices depicted with small green spheres.

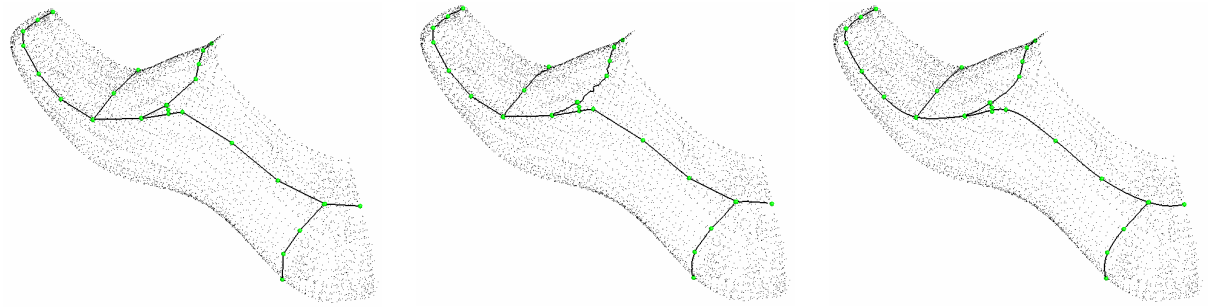
4.1.2 Step B: Projecting a Design Polyline onto a Point Cloud

Digital surfaces are allowed to be doubly-curved, which implies that line segment defined by two successive projected vertices may be off the point cloud. This step of the method aims at minimizing the distance between a design polyline and the corresponding digital surface. This is achieved as follows: (i) A finite number of *nodes* is selected on each design polyline, according to a discretization technique. Each node is coupled with an appropriate projection direction which is computed by interpolating the initial direction vectors at the projected vertices. (ii) This set of nodes is projected onto the point cloud along the computed directions, thus minimizing the distance between the design polyline and the digital surface.



(a) (b) (c)

Fig. 6. The three basic steps of the proposed digital-design methodology for the design of a ladies' shoe.



(a) (b) (c)

Fig. 7. The three basic steps of the proposed digital-design methodology for the design of a men's shoe.

This step of the method is illustrated in Figs. 6(b) and 7(b), where the initial design polylines are projected onto the point cloud and produce the corresponding "projected polylines". Unfortunately, this step often produces projected polylines with "wrinkles", which are mainly due to the sampling density of the point-cloud.

4.1.3 Step C: Smooth Projection of a Design Polyline onto a Point Cloud

The fact that industrial applications often involve dense "thick clouds" or clouds with a variable sampling density implies that Step B must also employ "smoothing functionals" in order for the computed projected polylines not to include "wrinkles". In previous attempts [1] we tackled this problem by introducing a "minimal chord-length" criterion into the calculation of the projected polylines. In this paper, we improve the initial smoothing technique by introducing a second-order functional which resembles the digital-curve discrete-curvature. Specifically, we incorporate into the

projection process two smoothing functionals minimizing the total chord-length and the total discrete curvature of the projected polylines, respectively. The "smooth projection problem" is formulated as

$$E(\mathbf{x}) = \lambda D(\mathbf{x}) + (1-\lambda)\left[L(\mathbf{x}) + C(\mathbf{x})\right],\ \lambda \in [0,1] \quad (1)$$

where $D(\mathbf{x})$, $L(\mathbf{x})$ and $C(\mathbf{x})$ represent the distance metric, length functional and second-differences functional, respectively. Vector $\mathbf{x}$ includes the unknown positions of the projected nodes, which are calculated by minimizing (1). According to our experiments this new method succeeds in eliminating wrinkles and produces very smooth digital curves.

4.1.4 Designing Smooth Digital Curves onto a Point Cloud

A higher-order digital curve on a point cloud is constructed by first using an interpolation method to calculate a polynomial spline passing through the vertices of the corresponding projected polyline. Then, similarly to Step B, a set of nodes and associated projection-directions are specified on the polynomial spline. Finally, employing the "smooth projection method" of Section 4.1.3, one projects each set of nodes onto the point cloud and constructs "higher-order digital curves". Figs. 6(c) and 7(c) present examples of such digital curves.

4.1.5 Editing Digital Curves

Editing toolkits are vital components of any CAD system to support the user in efficiently modifying his/her sketches. In fact, the proposed digital-design methodology may easily incorporate several editing tools for geometric modeling with digital curves. The proposed tool set includes all standard editing tools, which are split into two groups: (a) Vertex modificators (move, insert, delete, merge), and (b) digital-curve modificators (split, merge, trim). Our current research focuses on enhancing this tool set by developing robust solutions to new "geometry processing" problems related specifically to digital curves.

5. DESIGN SPECIFICATIONS AND SPACE REQUIREMENTS IN 3D LAYOUT PROBLEMS: FROM SKETCHES TO INFORMATIONALLY-COMPLETE DESCRIPTIONS

Engineers designing a complex system (plant, industrial facility, or a modern highly complex consumer-product) deal with identification, management and satisfaction of a large number of design requirements/restrictions. These are expressed as "design constraints", divided into "layout constraints" and "functionality constraints". The former focus on the system's viability while the latter aim at ensuring that the system also accomplishes all its tasks. In this paper, we focus on "geometric constraints", i.e., constraints from both families referring to all aspects of the geometric description and position of *layout elements* (: components, free spaces, and structural parts).

5.1 Standard layout-design methodology using "geometric constraints"

Standard approaches (see, e.g., [9] and references therein) limit geometric constraints to "dimensional constraints" and "topological constraints". The first group includes distance and angular dimensions determining the size, location and orientation of components and of required free spaces. The second group usually includes

Inclusion constraints	:	they restrict the area where a component or free space may lie.
Non-overlapping constraints	:	they specify layout elements that must not overlap.
Adjacency constraints	:	they specify objects that must be adjacent.

5.2 A critique of the standard "geometric constraint" methodology

In general, the domain of application, for standard methods, is only 2-D problems with very simple geometric objects ("orthogonal polygons") [9], yet the corresponding solutions are still highly heuristic and rely heavily on user interaction. In addition to that, these methods often involve ad hoc sketches and geometric descriptions that are never altered during the design process (compare that to point 4 of Section 2.2). More specifically, since topological constraints focus on relative position of layout elements, only dimensional constraints deal with the geometric description of individual layout elements. Surely, these dimensional constraints refer to some geometric framework (e.g., a sketch or a detailed 2D profile) created by the user. Despite the fact that this geometric framework is a vital part of the "design solution", standard methods, in general, do not modify it. It is reasonable to expect that such modifications may lead to drastic improvement of the final outcome as this geometric framework significantly affects statement (and thus treatment) of all topological and dimensional constraints.

Let us consider a typical example: "free-space for maintenance" around/near a piece of equipment. Usually, the geometric description employed for these spaces is an orthogonal parallelepiped including in its interior also this specific component [9]. The dimensions of the paralleliped are based on the designer's judgment or on recommendations given by the equipment's manufacturer. It is obvious that using such oversimplified geometric descriptions for layout elements is not appropriate for locating optimal solutions. Our research is based on exactly the opposite strategy: use exact descriptions for free-space requirements and robust solid-modeling methods to locate optimal layouts. More specifically, this research (see [10], [13], and references therein), pursued in collaboration with industrial experts, produced a detailed analysis of free-space requirements for an extensive collection of ship components and derived the required geometric-modeling tools. This work establishes that geometric modeling of such requirements relates to complex solid-modeling operations (solid-model simplification, modeling "illegal solids", simultaneous handling of fully- and partially-evaluated solids, 3D solid sweep, extending assembly-modeling to include also free spaces, interactive editing of nonmanifold solids, etc) and sometimes gives rise to new geometric-modeling problems like the problem "solid between a solid and a surface" treated in [13] (see also Fig. 9). [10] & [13] propose that layout constraints should be modeled/interrogated using exactly the same solid-modeling technology used for defining electromechanical components and assemblies. Indeed, [10] & [13] describe a sophisticated "Free-Space Modeler", combining three-dimensional geometric modeling with sketching, to allow a user to efficiently describe and analyze layout constraints in 2D or 3D; see examples in Fig. 8 and 9.

6. CONCLUSIONS

This paper has reviewed four geometric-modeling subjects of vital importance to Concept Design and New-Product Development. Two of these subjects are very old, classic CAD-problems: "Design with Constraints" and "Sketch to Polyhedron". The other two are brand new: "Point-based CAD" and "Solid Modeling of Design Constraints". Current methods, in all these fields, have been briefly described and evaluated regarding robustness and relevance to practical design-applications. Regarding new results, a new method has been introduced for constructing the topological description of the plainest polyhedron corresponding to a given sketch. Also, an improved technique has been presented for sketching digital-curves on point clouds.

7. ACKNOWLEDGEMENTS

This work was supported by the Ministry of Development of Greece through the Research Projects "E-MERIT: An Integrated E-Collaborative Environment for Product & Process Modeling Using 3D Models and Avatars" and "Information Management and Exchange in Network-Centric Product Design: Modeling Component Knowledge in Design Repositories". This research was also funded by the European Union and the Greek Ministry of National Education and Religious Affairs through the "Heraclitus Research Grant: Preliminary Industrial Design: Geometric/Information Models and Methods for Interactive Product-Design".

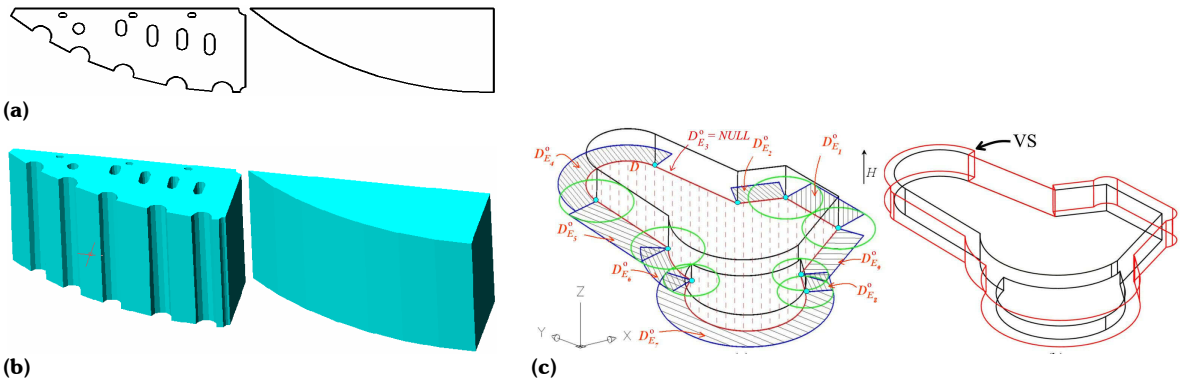
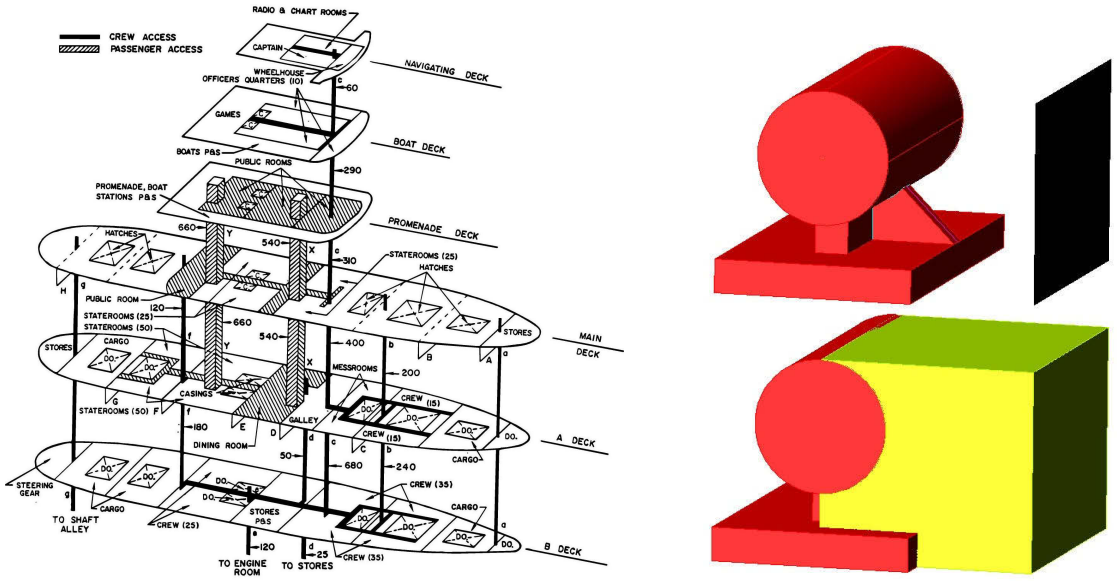


Fig. 8. (a) Given the technical drawing on the left, the designer sketches a "simplified profile" (a: right) so that the corresponding "extruded space" (b) is much simplified. (c) Free space around a component: the designer sketches a 2D "print" of this space, on a base plane, which is then constructed automatically by the Free-Space Modeler.



(a) **(b)**

Fig. 9. (a) A 3D sketch defining "access diagrams" in a ship. (b) Given an "opening", defined by a sketch, on a "wall" (b: top), the designer uses the Free-Space Modeler to construct the volume between that "opening" and a mechanical component.

8. REFERENCES

[1] Azariadis, P., and Sapidis, N., Drawing curves onto a cloud of points for point-based modeling, *Computer-Aided Design*, Vol. 37, No. 1, 2005, pp 109-122.

[2] Bonneau, G.-P., and Hahmann, S., Smooth polylines on polygon meshes, in G. Brunnett, B. Hamann, H. Mueller (eds.): *Geometric Modeling for Scientific Visualization*, Springer, 69-85, 2003.

[3] Cripps, R., Algorithms to support point-based cadcam, *Intern. Journal of Machine Tools and Manufacture*, Vol. 43, No. 4, 2003, pp 425-432.

[4] Hoffmann, C. M., Sitharam, M., and Yuan, B., Making constraint solvers more usable: overconstraint problem, *Computer-Aided Design*, Vol. 36, No. 4, 2004, pp 377-399.

[5] Jerard, R., Angleton, J., Drysdale, R., and Su, P., The use of surface points sets for generation, simulation, verification and automatic correction of NC machining programs, Proc. NSF Design and Manuf. Systems Conf., Tempe, AZ, 1990, pp 143-148.

[6] Krishnamurthy, V., Levoy, M., Fitting smooth surfaces to dense polygon meshes, SIGGRAPH'96, pp 312-324.

[7] Lipson, H., and Shpitalni, M., Optimization-based reconstruction of a 3D object from a single freehand line drawing, *Computer-Aided Design*, Vol. 28, No. 8, 1996, pp 651–663.

[8] Martinez, M. L. and Felez, J., A constraint solver to define correctly dimensioned and overdimensioned parts, *Computer-Aided Design*, in press.

[9] Medjdoub, B., Richens, P., and Barnard N., Generation of variational standard plant room solutions, *Automation in Construction*, Vol. 12, No. 2, 2003, pp 155-166.

[10] Sapidis, N., and Theodosiou, G., Informationally complete product models of complex arrangements for simulation-based engineering: modeling design constraints using virtual solids, *Engineering with Computers*, Vol. 16, Nos. 3&4, 2000, pp 147-161.

[11] Stahovich, T. F., Davis, R., and Shrobe, H., Generating multiple new designs from a sketch, *Artificial Intelligence,* Vol. 104, No. 1-2, 1998, pp 211-264.

[12] Sugihara, K., *Machine Interpretation of Line Drawings,* MIT Press, Cambridge, MA, 1986.

[13] Theodosiou G., and Sapidis N., Information models of layout constraints for product life-cycle management: a solid-modeling approach, *Computer-Aided Design*, Vol. 36, No. 6, 2004, pp 549-564.

[14] Varley, P. A. C., *Automatic Creation of Boundary-Representation Models from Single Line Drawings*, Ph.D. Thesis, Cardiff University, 2002.

[15] Varley, P.A.C., Martin, R.R., and Suzuki, H., Frontal geometry from sketches of engineering objects: is line labeling necessary?, *Computer Aided Design,* in press.