Geometric Constraint Solving: the Witness Configuration Method

Dominique Michelucci, Sebti Foufou *,1

Laboratoire Le2i, UMR CNRS 5158, Université de Bourgogne
BP 47870, 21078 Dijon Cedex, France

Abstract

Geometric constraint solving is a key issue in CAD, CAM and PLM. The systems of geometric constraints are today studied and decomposed with graph-based methods, before their numerical resolution. However, graph-based methods can detect only the simplest (called structural) dependences between constraints; they can not detect subtle dependences due to theorems. To overcome these limitations, this paper proposes a new method: the system is studied (with linear algebra tools) at a witness configuration, which is intuitively similar to the unknown one, and easy to compute.

Key words: Geometric constraints, constraints dependences, decomposition and solving.

1 Introduction

Geometric Constraint Solving (GSC) is today a key issue for all geometric modellers. GCS has applications in CAD, CAM, PLM, computer vision, robotics, molecular chemistry etc. Users in CAD-CAM interactively provide an approximate sketch using some graphical interface, and specify constraints between geometric entities (point, line, circle, plane...), then the solver corrects the sketch in order to satisfy the specified constraints. Typical constraints are incidences, distances, angles, parallelisms, orthogonalities, tangencies, etc.

* Corresponding author.
Email addresses: `dmichel@u-bourgogne.fr` (Dominique Michelucci,), `sfoufou@u-bourgogne.fr` (Sebti Foufou).
[1] Sebti Foufou is currently a guest researcher at the Manufacturing Systems Integration Division, National Institute of Standards and Technology, Gaithersburg, MD 20899-8263. Email: `sfoufou@cme.nist.gov`

Solving means find a solution for a (typically algebraic) system of equations $F(X) = 0$, or more precisely $F(U,X) = 0$, where U is a vector of parameters such as distances, angle cosines, and non geometric values (costs, weights). The values of parameters are known just before solving. They are assumed generic.

Solvers use numerical resolution for big irreducible constraints systems, and formulas for small problems, *e.g.*, computing the length of the third side of a triangle knowing one angle and the two other side lengths. Industrial solvers in CAD-CAM rarely use computer algebra methods (Grobner bases, resultants, Wu-Ritt triangulation methods), because of their exponential complexity.

Before the effective resolution, a preprocessing step of qualitative study is performed to partition the system into well-, under-, and over-constrained parts. If the system is correct, the under-constrained and over-constrained parts are empty; the well-constrained part is further decomposed into irreducible well-constrained parts, which are separately solved and then assembled. These decompositions are essential to solve big systems of hundreds of equations or unknowns. They are graph-based and unfortunately they can detect only the simplest dependences (contradiction or redundancy), called structural dependences, in systems of constraints. Graph-based methods can not detect more subtle dependences, due to geometric theorems. Actually, these methods are not mathematically sound, and are used far beyond their scope of validity: a combinatorial characterization of well-constrainedness is unknown, and seems to be out of reach (Section 10.5).

To overcome these intrinsic limitations of graph-based methods, this paper proposes a strategy that mainly relies on a new method, called the Witness Configuration Method (WCM). The strategy consists in distinguishing a projective part, and an Euclidean part in the system of constraints, and in solving the projective part: for CAD-CAM problems, the latter is very under-constrained and trivially solved in the typical situations. The solution is a Witness Configuration (WC). Triviality is defined in Section 10.1.

We recall that projective geometry is concerned with incidences: collinearities and coplanarities, which are invariant through homographies (inversible linear transforms). Euclidean geometry is concerned with distances and angles, which are invariant through rigid motions and symmetries. Section 8 gives first insights on projective geometry.

The sketch provided by the user is often itself a good WC. The WC is very similar to the target (unknown) configuration; for instance collinear and coplanar points in the target are also in the WC; parallel and orthogonal lines or planes in the target are also in the WC, at least in a projective sense; the target and the witness configurations differ in their lengths and angles. We study this WC using a classical method well known in the rigidity theory called the Numerical Probabilistic Method (NPM) [1–3]. The NPM computes the structure of the Jacobian using only linear algebra; it is presented in Section 6.

The Witness Configuration Method (WCM) detects a number of dependences which are missed by graph-base methods. Moreover, it can also decompose systems into well-constrained subsystems (Section 6.3). It is also possible to interrogate the WC: if it has some property, then this property holds generically and a theorem has been (probabilistically) proved [4–6].

Section 2 discusses previous works on the use of probabilistic computations to discover geometric relationships and shows common points between the WCM and some of these earlier works. Section 3 defines problems and configurations. Section 4 presents the principle of the WCM, Section 5 shows the limitations of graph-based methods, Section 6 presents the NPM, its extension, how it can decompose systems (Section 6.3) and gives its remaining limitations (Section 6.4), overcome by the WCM. Section 7 explains the two possible definitions of a WC: reduction to incidence constraints (detailed in Section 8), and considering parameters as unknowns (detailed in Section 9). Section 10 explains the resolution of systems of incidences: the trivial case (Section 10.1), the almost-trivial case (Section 10.2), the universality of this problem (Section 10.5). Section 11 concludes. Appendix A presents some geometric theorems for completeness.

2 The probabilistic test

The probabilistic test used in the WCM is far from being new, but as far as we know, it is the first time it is used to study and decompose systems of geometric constraints in CAD-CAM. Our work shows that this probabilistic paradigm is not restricted to the field of computer algebra it originates from, but that it is relevant for CAD-CAM applications. The probabilistic paradigm has already been used in CAD-CAM, but up to now it is restricted to stochastic methods such as evolutionary methods for shape optimization, or Monte Carlo techniques for physical simulations of radiosity, heat transfer, etc. in all these cases the computed functions are continuous.

The probabilistic test used in the witness configuration method was introduced by Martin in 1971 [7] and independently by Schwartz in 1980 [8] in the field of computer algebra. This test decides in a probabilistic way the equality or the difference between two numbers or more generally between two algebraic expressions. It is still used today in the computer algebra field [9]. This idea is used in geometric computations, for instance by Benouamer *et al.* [10,11], and independently by Requicha *et al.* [6] (Fig. 8 illustrates the probabilistic proof of Pappus theorem), to settle the robustness issue: the probabilistic test detects singular geometric situations (*e.g.*, collinearities of 3 points, coplanarities of 4 points) typically due to geometric theorems (Pappus, Desargues, Pascal), which can not be detected by the floating-point arithmetic because of its inaccuracy.

The probabilistic test is used in computer geometry for probabilistically proving geometric theorems: by Hong [12], by Rege *et al.* [13,14], by Bouhineau for Cabri Géomètre [15], by Kortenkamp and Richter-Gebert for Cinderella [5], by Tulone, Yap and Li [16] –and likely by others. For the simplest theorems (where computations in the rationals or finite fields are sufficient), all the probabilistic tests mentioned above are equivalent. This idea is also used in the rigidity theory to test the rigidity of frameworks in polynomial time (since at least 1975 [1,17,3]) and more generally to compute their free infinitesimal motions. It is this last method, called the numerical probabilistic method, which directly inspires the witness method: we compute free motions of a witness.

We admit that Graver *et al.* [1] work on combinatorial rigidity is the starting point of the WCM. We just slightly extended the original NPM in two ways: (i) the original NPM can compute free motions but can not deal with collinearites/coplanarities; the WCM can; and (ii) the original NPM is not used to decompose; the WCM is.

Actually, the witness configuration method combines three main ideas (not all so widespread in the CAD-CAM field): (i) the probabilistic paradigm from computer algebra; (ii) the study of the free motions of a bar framework from a generic realization with linear algebra tools from rigidity theory [1,17] and the theory of mechanisms; and (iii) graph-based decompositions from the field of geometric constraints solving.

3 Some problems and configurations

3.1 Notations.

$u \vee v$ is the line joining points u and v, U is the vector of parameters, X is the unknown variable, X^* is the searched value of X, U^w is the parameter for the WC, X^w is the value of X for the WC, h is the line or plane at infinity. The line at infinity is the line that Desargues added to the Euclidean plane: each point on the line at infinity corresponds to a direction of lines in the Euclidean plane. All parallel lines of the Euclidean plane meet on the corresponding point on the line at infinity. The plane at infinity is defined similarly. Homographies are defined in Section 8.

3.2 Problems

This paper will use the following set of problems, or systems of constraints as examples.

- *3D Molecule:* given some inter-atomic distances, find the configuration of a molecule. These systems of only distance constraints are represented by a graph, one vertex per atom (*i.e.*, point); each specified distance is represented by an edge. The distances are assumed to be generic: a slight modification of the specified distances causes a slight modification of the solution; in other words, the

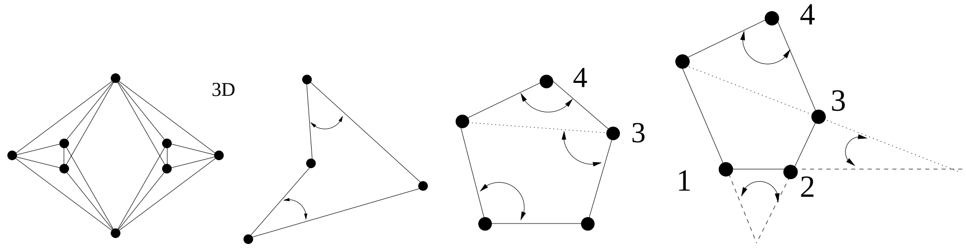

Fig. 1. In 3D, the double banana, and three Ortuzar's configurations (courtesy of Auxkin Ortuzar, Dassault Systèmes).

solution is a continuous implicit function of the specified distances. Several papers focus on this problem [3,18–23].

- *2D Molecule:* idem the previous one but in 2D.
- *Point-line incidence - PLI:* in 2D, a set of points and lines is specified with only a set of incidences. The two degenerate solutions (all points are equal, or dually all lines are equal) are forbidden.

3.3 Configurations

The theorems corresponding to these configurations are presented in Appendix A.

- *3 angles:* a triangle is specified by three angles.
- *Double Banana:* in 3D, 8 points are specified by distance constraints, represented by edges in the leftmost part of Fig. 1.
- *Ortuzar 1, 2, 3:* these 3D configurations are illustrated in Fig. 1; no 4 (or more) points are constrained to be coplanar; angles between (coplanar or not) lines are drawn with a curve arrow. In the third configuration, lines $(1,5)$ and $(2,3)$ are not coplanar, and idem for lines $(1,2)$ and $(3,5)$.
- *Icosahedron:* a 3D molecule problem with 12 vertices, where the graph is similar to the regular platonic icosahedron, but the distances on the edges are different. No convexity constraint at all. There are 30 point-point distances, thus the Bézout number is $2^{30} \approx 10^9$. This problem is difficult.
- *Dodecahedron:* in 3D, a dodecahedron, with the same topology as the regular platonic dodecahedron, is given by the length of its 30 edges, and by the coplanarity of its 12 pentagonal faces. No convexity constraint at all. Because of the coplanarity constraints, it is not a pure 3D molecule problem. The Bézout number is $2^{30} \approx 10^9$. This problem is as much difficult as the previous one maybe even more.
- *Desargues 2D hypothesis - D2H:* in 2D, the 3 lines (o,a,A), (o,b,B), (o,c,C) concur in o, or are parallel and o is a direction (*i.e.,* a point at infinity). Triangles *abc* and *ABC* are said to be perspective. Moreover $\gamma = ab \cap AB$, $\alpha = bc \cap BC$,

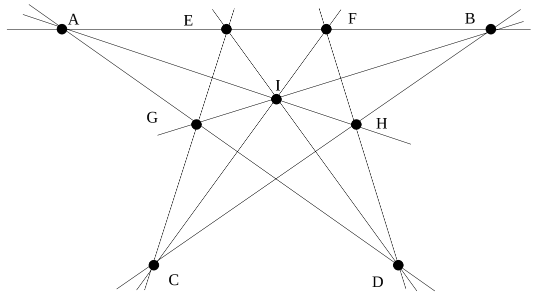

Fig. 2. The pentagonal star is a regular pentagon, modulo an homography; it has no rational realization; some cross ratios involve $\sqrt{5}$. This system of point-line incidences is not trivial to solve.

$\beta = ac \cap AC$. Some of the points α, β, γ, possibly all, can be at infinity. See Fig. A.1

- *D2H1:* idem D2H; moreover the angle $\alpha\beta, \beta\gamma$ is specified. D2H1 is dependent.
- *D2H2:* idem D2H; moreover the distance of γ to line $\alpha\beta$ is specified. D2H2 is dependent.
- *D2:* idem D2H; moreover points α, β, γ must be collinear. Actually, by Desargues theorem, each time D2H holds, then α, β, γ are collinear. Thus D2 is redundant.
- *Desargues 3D hypothesis - D3H:* idem D2H but all points lie in 3D.
- *D3H1:* idem D3H; moreover the angle $\alpha\beta, \beta\gamma$ is specified. D3H is dependent.
- *D3H2:* idem D3H; moreover the distance of γ to line $\alpha\beta$ is specified. D3H2 is dependent.
- *D3:* idem D3H; moreover α, β, γ must be collinear. Actually, by Desargues theorem, each time D3H holds, then α, β, γ are collinear. D3 is redundant.
- *Pappus hypothesis - PH:* in 2D, points p_1, p_2, p_3 are collinear, as well as q_1, q_2, q_3. Moreover $i_1 = p_2q_3 \cap p_3q_2$, $i_2 = p_1q_3 \cap p_3q_1$, $i_3 = p_1q_2 \cap p_2q_1$.
- *PH1:* idem PH; moreover the angle i_1i_2, i_2i_3 is specified. PH1 is dependent.
- *PH2:* idem PH; moreover the distance i_1 to line i_2i_3 is specified. PH2 is dependent.
- *P:* idem PH; moreover points i_1, i_2, i_3 must be collinear. Actually, by Pappus theorem, each time PH holds, then i_1, i_2, i_3 are collinear. P is redundant.
- *Pentagonal star:* a PLI instance, which has no rational realization (see Fig. 2). Actually this problem ($p = 9$ points, $l = 9$ lines, $i = 28$ incidences) is well-constrained modulo homography: in 2D, such systems have $i = 2(p+l) - 8$ incidence constraints, and contains no over-constrained part. There are two distinct pentagonal stars (modulo homography).

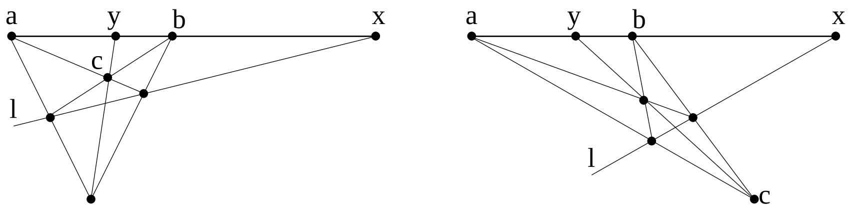

Fig. 3. Given 3 aligned points a,b,x, whatever l and c, y is unchanged: $c' = ((ac \cap l) \vee b) \cap ((bc \cap l) \vee a)$; $y = ab \cap cc'$. y is the harmonic conjugate of x relatively to a,b. No current graph-based method can detect that.

- *Harmonic Conjugate - HC:* a PLI instance, illustrated in Fig. 3. If x is at infinity (so that l is parallel to ab), then y is the middle of a and b.

4 Principle of the WCM

Subtle dependences due to geometric theorems are not detected (and probably not detectable) by graph-based methods. Examples of configurations containing such a subtle dependence are: D2H1, D2H2, D2, D3H1, D3H2, D3, PH1, PH2, P, HC. All these configurations contain the hypothesis of a theorem, and its conclusion or the negation of its conclusion. To bypass the limitations of the graph-based methods, our new strategy first extends the NPM, which initially applies only to the molecule problem (in any dimension), to deal with other configurations, and then develop the WCM as a modification of the extended NPM capable of solving the system of geometric constraints after decomposing it into a projective part and an Euclidean part.

The extended NPM detects the dependences in configurations: 3 angles, Ortuzar 1, 2 and 3, and the double banana (the classical NPM already treats correctly the double banana, since it is a molecule problem). Moreover the extended NPM is also able to decompose a system into rigid subsystems; the NPM has the same decomposition power with molecule problems, but it seems that nobody realizes or uses that up to now. However, the extended NPM is still misled by geometric theorems: if the constraints contain the hypothesis of some geometric theorem, and its conclusion (or the negation of its conclusion), the extended NPM does not detect the dependence. The WCM is a modification of the extended NPM, which is no more misled by theorems and detects the resulting dependences (if there are some of course), for instance in configurations: D2H1, D2H2, D2, D3H1, D3H2, D3, PH1, PH2, P and others omitted for conciseness. In D2H1, D2H2, D3H1, D3H2, PH1, PH2, the WCM detects that a metric constraint (distance and angle) is dependent with the incidence properties of the configuration: these properties are either incidence constraints explicitly specified, or implied by –known or unknown [2] – geometric theorems (Desargues in 2D, in 3D, Pappus, etc.). For P (for D2, for D3), the WCM

[2] Of course, it is very interesting that this method can detect theorems which are unknown from computer scientists, users, or even geometers!

detects that the constraint in P $-$ PH (in D2-D2H, in D3-D3H) is a consequence of PH (of D2H, of D3H). Of course, the WCM still detects the dependences which are already detected by the NPM or the extended NPM.

This detection is probabilistic, in the following (good) sense: if a theorem holds, it is detected, *i.e.*, it can not be missed; it may happen, less than one time over 10^9, that the WCM wrongly detects a theorem, *i.e.*, the system is accidentally dependent at the studied WC. This case of ”wrong positive” is as likely as guessing the root of an algebraic equation when proposing a random number. To reassure anxious, paranoiacs, and unlucky users the test can be repeated with several witness configurations. Moreover, the WCM is also able to decompose a system into its rigid subparts.

Here is a (non exhaustive) list of theorems which confuse the graph-based methods, but not the WCM: Pappus, Desargues, Pascal, hexamy, Brianchon, Beltrami (or Galucci). For conciseness, some 3D theorems or configurations, which are correctly handled by the WCM, are not detailed in the paper, but they are presented in Appendix A.

All geometric theorems of projective geometry arguably result from Pappus (or equivalently Pascal, or hexamy, or Brianchon) theorem: the main axiom of projective geometry is a restatement of Pappus, and the other axioms are actually definitions of points and lines [24]. It suggests that the WCM is robust against a lot of theorems of projective geometry, if not all. Finally, Cayley [24] claimed that projective geometry is all geometry (Cayley called ”descriptive geometry” what is today named ”projective geometry”).

The core idea of the WCM is to split the constraints system $F(U,X) = 0$ in two parts: projective part, and Euclidean part. Intuitively, the projective part contains all point-line, line-plane and point-plane incidences of the constraints system. Moreover, it is assumed that a solution to the projective part of the system is known, or easily computed, as it is the case for all the examples of this paper, except one: the pentagonal star; this solution is the WC (illustrated in Fig. 4, 5). It has the following properties: collinear (coplanar) points in X^*, the unknown and searched solution of $F(U,X) = 0$, are collinear (coplanar) in the WC X^w. Parallel (orthogonal) lines or planes in X^* are parallel (orthogonal) *in a projective sense* in the WC. Actually, X^* and X^w differ only in their lengths and generic angles (right angles occurring in orthogonality constraints are not generic).

Most of the time, the sketch (which users interactively enter to describe their system of geometric constraints) is by itself a WC, otherwise a WC X^w has to be computed.

A first method to compute the WC is to consider the parameters U as unknowns in the system $F(U,X) = 0$, and solve the resulting system: $F(Y,X) = 0$. Usually, for CAD-CAM applications, this system is very under-constrained and can be trivially solved (Section 10.1), even when starting from a difficult and challenging constraints system.

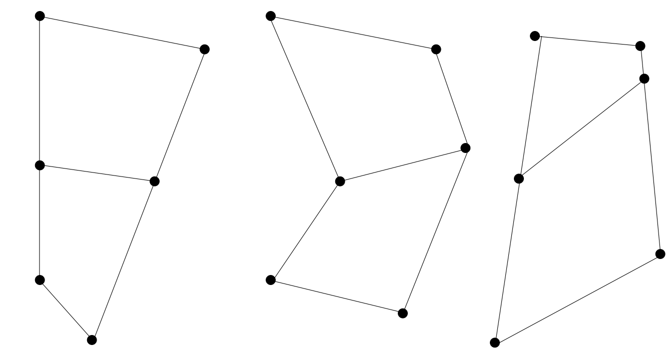

Fig. 4. Left: the unknown solution configuration. Middle: a random configuration, used by the extended NPM. Right: a witness configuration.

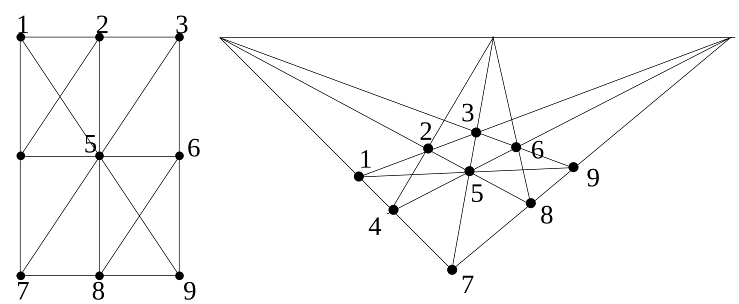

Fig. 5. Left: the unknown solution configuration, with parallel lines. Right: a witness configuration.

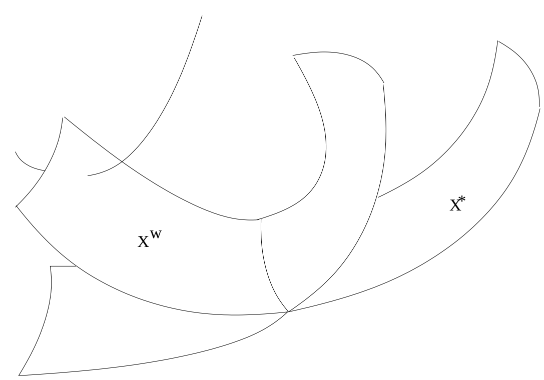

Fig. 6. The solution set (Y,X) of $F(Y,X)=0$ can be heterogeneous in dimension. We only assume that the WC X^w and the target one X^* have a similar Jacobian structure (here both have a tangent plane).

Another possible method to compute the WC is to formulate a projective system and find one solution. Projective constraints are constraints of incidences between flats (points, lines, planes). Indeed, parallelism and orthogonality constraints reduce to incidences constraints, as well as middleship (point I is the middle of points A and B).

There are two cases: either a WC is found, or it is not.

The WC is known. Suppose first that a WC X^w is available, for instance (V^w, X^w) is solution of $F(Y,X)=0$. For simplicity, assume that, as it is typically the case, the Jacobian at the target and the witness configurations has a similar structure; Fig. 6 gives an illustration of this assumption: *e.g.,* if X^* lies on a surface, then so does X^w otherwise it means that the solution set of the considered system is not a manifold, *i.e.,* it is heterogeneous in dimensions, for instance it contains curves and surfaces like Whitney's umbrella or Hermitte's umbrella, and the witness and the target configurations lie on parts with different dimension.

The existence of a WC first proves that the projective part of the system is consistent. The WC is then used to detect dependences in the complete system, *i.e.,* dependences between the constraints in the Euclidean part, as well as dependences between constraints in the projective and in the Euclidean parts. The idea is to use a variant of the NPM: the NPM studies the structure of the Jacobian of $F(Y,X) = 0$ at a random configuration; the WCM studies the structure of the Jacobian at the WC (U^w, X^w).

Moreover, the WC makes possible the decomposition of the complete system into rigid subsystems (Section 6.3). It can also be interrogated; for instance, a WC of the Pappus hypothesis PH (of the Desargues hypothesis D2H in 2D, or of the Desargues hypothesis D3H in 3D), will satisfy the Pappus conclusion (the Desargues conclusion in 2D, in 3D): this is a (probabilistic) proof of the theorem; it is the principle of Jurzak's prover [4]. If the WC has not the conjectured incidence property, then the conjecture is wrong – with no doubt at all: thus it is useless to repeat the test with another WC.

If the strategy proves that the system is rigid (well-constrained modulo isometry), it can happen that the solver fails to find a solution for the values U of the parameters. If the solver is complete (*e.g.,* it uses interval analysis), the existence of the WC proves that the problem is due to a bad value U of parameters: for instance, U violates a triangular inequality. In passing, triangular inequalities are not the only inequalities which must be fulfilled, in order for the geometric system to have a solution in real space; for instance, using Cayley Menger relations, Philippe Serré exhibits in his PhD thesis a tetrahedral inequality, *i.e.,* a realizability condition involving the lengths of the 6 edges of a tetrahedron [25]; actually, expliciting all inequalities $C_i(U) \geq 0$ for any configuration parameterized by a vector U is an open problem.

No WC is known. This case occurs only with the pentagonal star; the absence of a WC prevents to detect, *e.g.,* the collinearity of I, $DG \cap CH$ and $DH \cap CG$ (it would be detected in a WC). Actually, in this system, a WC corresponds not to a continuum of solutions, but only to a finite number (2) of solutions (modulo homography).

For the moment, the absence of a WC only means that the system is neither trivially solvable nor almost trivial and dependent (which is already a relevant information), but it does not mean that the projective part is not solvable: for the moment we choose to focus our work on the use of the WC, and we postpone the numerical resolution of non trivial projective systems. The latters seem rare in CAD-CAM:

$$\begin{vmatrix} a & b & 0 \\ u & v & 0 \\ r & s & t \end{vmatrix} = avt - but$$

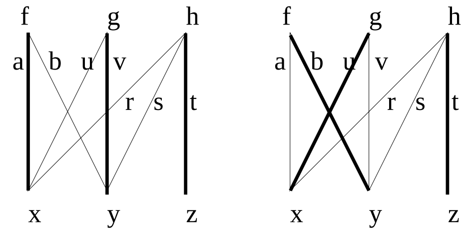

Fig. 7. Each term *avt* and *but* of the determinant corresponds to a perfect matching in the bipartite graph. Thus when there is no perfect matching, the determinant is identically zero. If the nullity of the Jacobian, *i.e.*, the dependence between rows, is due to the numerical value of coefficients, and not to the structure (0 or not) of the matrix, graph-based methods do not detect the dependence. The NMP does when the dependence occurs generically: the WCM does every time.

we find only examples which are not natural for CAD-CAM such as the pentagonal star. Anyway, all projective systems reduce to algebraic systems; actually they even reduce to a molecule problem (Section 10.3) – non generic however. Thus classical methods of GCS should be used to solve and decompose them; the resolution will provide a WC –or the proof that the complete system has no solution since its projective part has none. Then the previous WC machinery applies, but with a modification, which is worth mentioning: if the projective system is difficult, it is probably non linear, and the found WC has probably non rational coordinates (as the pentagonal star); it implies that the numerical study of the Jacobian can not use exact arithmetics in $\mathbb{Q}$ or in a finite field (*i.e.*, a Galois field) which –very conveniently– eliminates all inaccuracy difficulties. Thus computations must be performed with floating-point arithmetic (computer algebra being usually not practicable), which may cause numerical inaccuracy difficulties when computing the rank of a set of vectors; some SVD method and the ε heuristic must be used to limit inaccuracy.

5 The limitations of graph-based methods

Today, the qualitative study step is graph-based. For simplicity, we summarize the theory for the case of systems of algebraic equations, *i.e.*, for the case of a well-constrained system where the number of unknowns equals the number of equations and no over-constrained subsystem is contained. An over-constrained subsystem is a subset of equations that constrains a smaller (in cardinality) subset of unknowns; *e.g.*, in the system $f(x,y,z) = g(z) = h(z) = 0$, the subset g, h over-constrains z. Roughly, one of the numerous variants relies on the bipartite graph equations-unknowns where equations and unknowns are represented by vertices; an equation-vertex is linked to an unknown-vertex iff the unknown occurs in the equation. Matching theory provides polynomial time algorithms to decompose the graph. A matching is a subset of the edges, at most one edge per vertex. A vertex is covered or saturated by a matching iff one edge in the matching is incident to this vertex. A matching is maximum iff its number of edges is maximal in the set of all possible matchings. A matching is perfect iff it covers all vertices. A perfect matching is maximum, of course. A system is structurally well-constrained iff the corresponding graph has a perfect matching. More precisely, the rank of the Jacobian is smaller or equal to the cardinality of the maximum matching; it is equal to, in the generic case. This easily follows from the fact that the determinant of a square matrix is a sum of products, and each of these products corresponds to a perfect matching in the bipartite graph associated to the linear system. From a perfect matching, it is possible to decompose the (structurally well-constrained) graph into its irreducible well-constrained parts.

Considering systems of geometric constraints, instead of systems of algebraic equations, introduces several difficulties or complications.

A difficulty is that geometric constraints are usually supposed to be independent of all coordinates system; thus geometric constraints can not determine the location and orientation of a geometric configuration; this location and orientation are determined by 3 parameters in 2D (a base point, and an angle for orientation), and 6 in 3D (a base point, and 3 Euler angles for orientations). Thus, a well-constrained (modulo location and orientation, *i.e.,* modulo isometry) geometric system has 3 unknowns more than equations in 2D, 6 unknowns more than equations in 3D, and $d(d+1)/2$ unknowns more than equations in d dimensions. For instance in 2D, a triangle is represented by 6 unknowns $((x_i, y_i), i = 1, 2, 3)$ and is well-constrained by 3 constraints. Symptomatically, there is today no commonly accepted definition for the decomposition of rigid systems into rigid subsystems, and current definitions are often polluted with algorithmic considerations; it contrasts with systems of equations, where there is a *unique* decomposition of systems into the under-, over- and well-constrained parts, and a *unique* decomposition of the well-constrained part into irreducible well-constrained parts.

A complication is that unknown-vertices attached to the same geometric entity (point, line, plane) are typically merged; a weight (called DoF, Degree of Freedom) is attached to each resulting vertex; similarly, equation-vertices representing the same geometric constraint are merged, and a weight (called DoF or DoR, Degree of Restriction) is attached to each resulting constraint-vertex. Finally, the algorithms are often presented using network flows, rather than maximum matching; however the two parlances are equivalent.

An essential problem is due to the intrinsic limitations of graph-based methods. Since they consider only a graph, they can detect only the simplest errors, called structural errors (Fig. 7); they can not detect more subtle dependences (redundancies or contradictions) between equations, *i.e.,* when an equation is a consequence of the others (the polynomial lies in the radical of the others) or contradicts the others (it is possible to deduce 0=1 from the equations). Such subtle dependences often occur in geometric constraints. The simplest example is the 3 angles configuration: a triangle is well constrained by three lengths, but not by three angles (though, in spherical geometry, *i.e.,* on the unit sphere, a triangle is indeed well-constrained by three angles).

Graph-based methods are mathematically sound only for a very restricted class of geometric systems: in 2D, when all unknowns are points, and all constraints are distance constraints between points; moreover the distances must be generic, which forbids collinear (and cocyclic, ”coconic”, ”cocubic”, etc.) points. Laman’s theorem states that a system of m distance constraints between $n \geq 2$ points is well-constrained modulo isometry (other words are: isostatic, rigid) iff $m = 2n-3$ and there is no over-constrained subsystem, which means that every subset of $n' \geq 2$ points is constrained by at most $m' \leq 2n' - 3$ distance constraints [1].

The same point-point distances problem in 3D is sometimes called the molecule problem, because of its applications in molecular chemistry: given some distances between its atoms, determine the configuration of the molecule. This problem also occurs in the rigidity of 2D or 3D frameworks for buildings, and the rigidity theory originated from this application.

The natural extension of Laman’s theorem in 3D (replacing $2n-3$ by $3n-6$) is wrong; the double banana (Fig. 1, leftmost) is the most famous counter example; it respects the previous counts, but it is not rigid: the two halves can rotate independently around their common axis (an under-constrainedness), and there is no reason for the height of the two halves to be equal (this over-constrainedness cancels out the previous under-constrainedness, and permits this configuration to pass the rigidity test).

Computer scientists are aware of the lack of combinatorial characterization of rigidity in 3D and of the limitations of graph-based methods. Nevertheless, they extend (abusively) Laman’s theorem, and they apply graph-based methods, to more general geometric constraints in 2D (not only point to point distance constraints) where Laman’s theorem is no more valid, as in the triangle example, and in 3D. Though this approach sometimes fails, it often works and makes possible the resolution of constraints systems of industrial size.

Recently, GCS becomes victim of its success: the size of constraints systems is still increasing, as well as the probability for a non structural dependence between constraints, *i.e.*, a dependence which is not detected -or even non detectable- by graph-based methods used during the qualitative study step. This phenomenon is somehow aggravated by the fact that today not only engineers but also designers or artists use more and more geometric constraint solvers, *e.g.*, for designing constrained surfaces and blends. It seems also that the risk of subtle dependences is bigger in 3D, for various reasons: the lack of human intuition in 3D, the greater difficulty of 3D problems (consistent with the existence of Laman’s theorem in 2D, and its absence in 3D), the limitations of human-computer graphical interfaces.

To overcome the limitations of combinatorial approaches, a first possibility is to add some ad hoc tests, in order to detect at least the most frequent and simple counter examples, such as 3 angles, and perhaps Ortuzar 1, 2, 3. Though useful, this approach can give no definitive guarantee; moreover it yields to complicated and non easily reproducible methods.

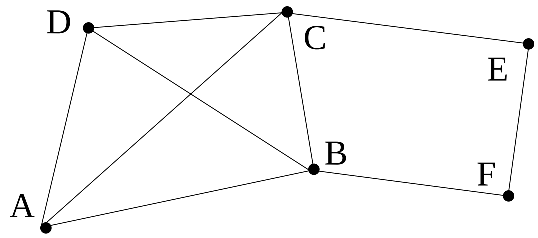

Fig. 8. A 2 molecule system, studied with the NPM.

A second research track is to search a *mathematically sound* combinatorial characterizations of well-constrained geometric systems, and indeed several combinatorial characterizations for rigidity have been conjectured. However computer scientists need to consider problems more general than the molecule problem, *e.g.*, constraints involving angles, orthogonality, parallelism ... or even more basically incidence constraints: point-line incidences in 2D, or point-line-plane incidences in 3D. These last constraints may seem trivial, but it turns out that every geometric constraints *in any dimension*, actually every algebraic systems, reduces (rather cryptically) to a system of point-line incidence constraints *in 2D*; in this sense, 2D point-line incidence constraints are universal: see Section 10.5. Though baffling, this universality property is just a restatement of an ancient theorem due to von Staudt (at the times of Hilbert). This universality property suggests that the combinatorial characterization of well-constrainedness (modulo homography) for systems of 2D point-line incidence constraints (and thus, by the universality theorem, of all kind of geometric constraints) is terribly difficult, to say the least.

6 The Numerical Probabilistic Method

6.1 The principle of the NPM

Equations $F(U,X) = 0$ are independent if the Jacobian has full rank at the searched root. Unfortunately, the latter is unknown, but it is still possible to study the Jacobian structure at a generic (random) configuration X_0. To speed up, and to avoid inaccuracy difficulties, it is possible to compute a base of the Jacobian in some finite field $\mathbb{Z}/P\mathbb{Z}$ with $P \approx 10^9$ for example.

The NPM is very well known in the rigidity theory [26,1,3]. It deals with the molecule problem: for a given dimension, for a given graph, assuming edges carry generic lengths, is the graph rigid, or not? For the 2D case, Laman's theorem gives a characterization of rigidity, and several polynomial time algorithms were given [27,28,3]. For the 3D case and beyond, a combinatorial characterization is still unknown. Anyway, in any dimension, the NPM applies, and it is polynomial time. Its probabilistic nature is not a tabu for engineers.

We illustrate the principle of the NPM with the 2D molecule problem in Fig. 8.

Remark this system is generic, *i.e.*, no three points are collinear for instance. The NPM computes a base of the gradient vectors of distance equations, for random coordinates of the points $A,\dots F$. It can detect that the gradient vectors for equations: AB, BC, CD, DA, AC, BD are dependent (for instance the vector for BD lies in the vectorial space generated by the gradient vectors AB, BC, CD, DA, AC). It also detects that the complete system is not rigid: a rigid system has a kernel with rank 3 in 2D (6 in 3D); actually a possible base for the kernel in the rigid case is a translation in x, one in y, and an instantaneous rotation around the origin (more details in [2]). In this example the kernel has rank 4, and this is due to the under-constrainedness of the part $BCEF$. The Jacobian is called the rigidity matrix in the rigidity theory.

Computation of a base, and a cobase (the base of the kernel) need only standard linear algebra. We use only Gauss' elimination, and not the Gram-Schmit orthogonalization procedure. We perform computations in a finite field to avoid inaccuracy problems when computing ranks; in finite field, isotropic vectors can occur: they are non zero vector with a null norm, *e.g.*, $(1,1,1)$ modulo 3, and thus the Gram-Schmit procedure does not apply. This is not a problem, since indeed Gauss method is sufficient.

When the NPM is used on a configuration involving floating point numbers, for instance a WC obtained after solving a non linear system (though rare, it can happen), exact arithmetic and computations in a finite field can no more be used; instead some SVD (singular value decomposition) and ε threshold heuristic must be used to compute the rank of a set of vectors.

6.2 The extended NPM

We extend the NPM to other geometric constraints. We try several translations of constraints into equations, and several representations (cartesian or barycentric coordinates). Here is one which works. Points are represented by cartesian coordinates. Unknowns may be non geometric (*e.g.,* densities, weights, forces, torques) or geometric. Values of geometric unknowns are either dependent on the cartesian frame (*e.g.,* cartesian coordinates of points), or independent (*e.g.,* distances, scalar products, signed areas or volumes) [29]. Vectorial equations $t_1 \times \overrightarrow{P_iP_j} + t_2 \times \overrightarrow{P_kP_l} + \ldots = \overrightarrow{0}$ specify collinearities, coplanarities, incidences and intersections. For instance, to specify I is the middle of AB use equation $\overrightarrow{AI} - \overrightarrow{IB} = 0$; to specify $J = AB \cap CD$, introduce two new scalar unknowns α, β and use equations $\overrightarrow{AJ} = \alpha\overrightarrow{AB}$ and $\overrightarrow{CJ} = \beta\overrightarrow{CD}$.

Scalar equations (*i.e.,* non vectorial equations) are used to specify distances and angles between vectors. The constraint $\text{distance}(A,B) = d$ is translated into the equation: $d^2 - \overrightarrow{AB}.\overrightarrow{AB} = 0$. The constraint $\text{angle}(\overrightarrow{AB},\overrightarrow{CD}) = \theta$ is translated into equations:

$$\begin{cases} \overrightarrow{AB}.\overrightarrow{CD} - l_{AB}l_{CD}k_\theta = & 0 \\ l_{AB}^2 - \overrightarrow{AB}.\overrightarrow{AB} = & 0 \\ l_{CD}^2 - \overrightarrow{CD}.\overrightarrow{CD} = & 0 \end{cases}$$

where k_θ represents the $\cos\theta$; it is either a parameter or an unknown.

Scalar products are translated trivially, *e.g.,* in 2D: $\overrightarrow{AB}.\overrightarrow{CD}$ is rewritten: $(x_A - x_B)(x_D - x_C) + (y_A - y_B)(y_D - y_C)$. Finally, vectorial equations are straightforwardly translated into d scalar equations in dimension d. This translation of constraints into equations is systematic and automatic. The user specifies what are parameters, *i.e.,* have known values; all coordinates are unknown. The language used to specify constraints guarantees that all constraints are indeed independent of the cartesian frame, *e.g.,* the user can not specify point coordinates. With this formulation, the NPM indeed detects the bad constrainedness in all pathologic configurations of Fig. 1 (the classic NPM already detected the problem with the double banana). Other representations should be tried: Grassmann Cayley coordinates for instance.

6.3 Decomposition with the NPM

The NPM (and the extended NPM, as well as the WCM) can also detect rigid subsystems in polynomial time. Actually, our method detects *maximal rigid subsystems*. Maximal is intended for inclusion. Searching maximal rigid subsystems may seem strange, since usually people are looking for the smallest rigid subsystems – actually the smallest *non trivial* rigid subsystems, to avoid isolated points and lonely edges which are not relevant.

Two points P and Q are "relatively fixed" by a system $F(X) = 0$ iff the gradient vectors of the equations resulting from the constraint: $\overrightarrow{PQ}.\overrightarrow{PQ} - l_{PQ}^2 = 0$ (either l_{PQ} is a parameter, or the value of the length in the witness) lie in the vectorial space spanned by the Jacobian F' of the system; in other words if their distance is fixed by the system. Note that the two points P and Q can be far from each other in the bipartite graph of constraints, and still be fixed relatively to each other by the system. As usual, all computations are done at some random configuration (at the witness configuration, for the WCM).

In 2D, two relatively fixed points are an anchor. In 3D, three non collinear points and pairwise fixed are an anchor. A system contains at most a polynomial number of anchors: $O(n^2)$ in 2D, and $O(n^3)$ in 3D for a system with n points. For a given system, a given anchor lies in exactly one maximal rigid subsystem; conversely, non trivial maximal rigid subsystems contain several anchors. Thus every system contains at most a *polynomial number* of maximal rigid subsystems (one maximal rigid subsystem per anchor), while it can contain an *exponential number of (non maximal) rigid subsystems*. Maximal rigid subsystems guarantee a polynomial complexity in a very straightforward way. Note that two distinct maximal rigid subsystems always meet in less than one anchor.

The following greedy method finds in polynomial time the maximal rigid subsystem containing a given anchor for a given system: start with the anchor and add, in *any* order, all points in the system which are relatively fixed with the points in the anchor. If the system is rigid, the maximal rigid subsystem containing the anchor is the system itself.

To find *all* maximal rigid subsystems in a given system, just compute with the previous greedy method the maximal rigid subsystem of every anchor of the system. Again, if the system is rigid, it is the maximal rigid subsystem containing all anchors, and it is found several times by the previous method. Thus this method is not optimal but it is polynomial time. To optimize, only anchors not completely inside a previously found maximal rigid part need to be computed with the greedy method.

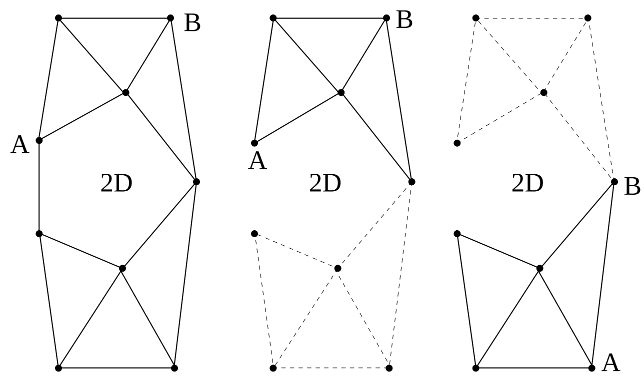

Fig. 9. Decomposition with the NPM. Maximal rigid parts with anchor A, B are represented with solid lines. Middle and right: removing a constraint creates (in this example) 2 maximal rigid parts.

To decide if a rigid system S contains smaller rigid subsystems, and to detect them, just find for every constraint c in S all maximal rigid subsystems of $S - c$, with the previous method. See Fig. 9 for an example. The method is polynomial time, whether the system is reducible or not; its optimization partly depends on the global strategy used for the decomposition, for instance whether a detected rigid subsystem is recursively decomposed in an eager way, or all systems $S - c$ are considered, before the most promising decomposition is selected according to some criteria (for instance the C-tree criterion by Gao *et al.* [30]).

These methods for computing all maximal rigid subsystems in a rigid and in a flexible system are polynomial time, and more reliable than pure graph-based methods. They provide relevant tools to implement and test strategies for decomposing systems of geometric constraints. These decomposition strategies can be inspired by current graph-based methods, but also by matroid theory.

The extended NPM and all graph-based methods of course are still misled by geometric theorems: if constraints contain the hypothesis and the conclusion (or its negation) of a theorem, the extended NPM can not detect the dependence between constraints. Basically, geometric theorems do not reduce the rank of the Jacobian everywhere, whereas the extended NPM computes the rank only at a generic (random) configuration.

To illustrate this facts let us consider a simple example: In 3D, if we specify that lines AB and CD meets at point I and that A,B,C,D are coplanar, the second constraint is redundant. But, the extended NPM does not detect this redundancy. Of course this is a very trivial theorem. More sophisticated ones are also not detected.

If the extended NPM is used not at a random configuration, but at a WC, it detects the dependence between equations, *i.e.*, the rank deficiency. Thus a natural idea is to find a WC.

7 Constraints of the WC

This section presents how to formulate equations to find a WC (when the interactive sketch does not provide one). First, we assume that all parameters in U have generic values, thus every orthogonality constraint is expressed as the vanishing of a scalar product $\overrightarrow{AB}.\overrightarrow{CD} = 0$; it is *not* expressed by the system for the (generic) angle constraint:

$$\begin{cases} \overrightarrow{AB}.\overrightarrow{AB} - l_{AB}^2 = & 0 \\ \overrightarrow{CD}.\overrightarrow{CD} - l_{CD}^2 = & 0 \\ \overrightarrow{AB}.\overrightarrow{CD} - k_\theta l_{AB} l_{CD} = & 0 \end{cases}$$

with a value 0 for the parameter k_θ; in other words, $k_\theta = 0$ is not generic. Similarly, $k_\theta = 1$ is not generic; thus parallelism constraints can not use the previous system, which is used for the generic angle constraint only.

The first possibility, the easier to explain, is to consider the parameters (distances, angles cosine, etc.) as unknowns. The resulting system of equations is likely very under-constrained because of the addition of unknowns. This operation leaves equations unchanged, thus parallelisms and orthogonalities are preserved in solutions (each solution giving a WC). This system is usually completely trivial to solve, even when starting from a very difficult system, such as the icosahedron (just choose at random 12 points in 3D, and deduce the distances to get a WC) or the dodecahedron (choose 20 planes at random, deduce the vertices and then the distances to get a WC). Section 9 gives some possible tracks to solve this kind of under-constrained systems, when they are not trivial.

The second possibility is to explicit all projective constraints, *e.g.*, the incidence constraints induced by parallelisms and orthogonalities. Indeed Section 8 shows that Euclidean constraints of parallelism, orthogonality and middleship reduce to incidence constraints between flats: points, lines, planes. A solution to this incidence constraints provides a WC. This system is most of the time under-constrained, because of the deletion of metric constraints (the pentagonal star is the only counter example).

8 Reducing to incidence constraints

8.1 A first insight on incidence constraints

Incidence constraints, also called: projective constraints, do not permit to fix angles nor dimensions. Thus these constraints and the related domain of projective geometry are not well known in the CAD-CAM community. This section intends to give some insight on incidence constraints and projective geometry.

In 2D, the simplest projective constraints are incidence constraints between points and lines; they can specify neither lengths nor angles, nor even the order of three aligned points. In 3D, the simplest projective constraints are incidence constraints between points, lines, and planes. Points, lines and planes are called flats.

More complex incidence constraints involve conic curves (quadric surfaces) and algebraic curves (surfaces) with higher degree. Considering only incidence constraints between flats (as mainly this paper does) is not a loss of generality: remarkably, it is possible (and not so difficult) to constrain a point to lie on any algebraic curve (surface) using only incidence constraints between flats. For instance, in 2D, after Pascal theorem, the conic defined by 5 given points (no three of them collinear) $p_1, \dots p_5$ is the locus of the points p_6 such that $p_1p_2 \cap p_4p_5$, $p_2p_3 \cap p_5p_6$, $p_3p_4 \cap p_6p_1$ are collinear; variants of this theorem permit to define a conic from 5 any geometric elements (point or tangent line) under mild assumption.

Incidence constraints are preserved through homography; assuming points are represented with homogeneous coordinates (x,y,h) in 2D and (x,y,z,h) in 3D, an homography is any inversible linear mapping, *i.e.*, a 3×3 (a 4×4 in 3D) inversible matrix acting on these homogeneous coordinates.

A 2D homography is defined by 8 coefficients: the matrix has 9 entries but all proportional matrices represent the same homography. Projective constraints can only define a configuration up to homography, *i.e.*, in 2D a well-constrained system modulo homography has 8 remaining degrees of freedom (it is well known that a well-constrained system modulo isometry, also called a rigid or an isostatic system, has 3 remaining degrees of freedom). An homography is uniquely defined by its action on 4 generic (*i.e.,* no 3 aligned) points; in other words, all generic quadrilateral (4 points, no 3 collinear) are congruent modulo homography, *i.e.,* any generic quadrilateral is mapped to a square by some homography.

A 3D homography is defined by 15 coefficients: in 3D a well constrained system modulo homography has 15 remaining degrees of freedom (a rigid 3D configuration has 6 remaining degrees of freedom).

Homographies obviously preserve incidences, and degrees of algebraic curves (surfaces). However, they do not preserve parallelism, orthogonality, distances, angles, distance ratios; but they preserve cross ratios, which are ratios of distance ratios of collinear points.

To reduce Euclidean constraints to incidence constraints, a line at infinity in 2D, or a plane at infinity in 3D, is needed; call it h; from a projective point of view, h is an arbitrary line or plane.

8.2 Parallelism reduces to incidence

Though homographies do not preserve parallelism, something more subtle is kept. Consider the 2D case, parallel lines do not meet in the Euclidean plane. Desargues proposed to complete the Euclidean plane with a special line called line at infinity, or line of directions, where all parallel lines of the Euclidean plane meet. Each point on the line at infinity corresponds to one direction of lines in the Euclidean plane. The Euclidean plane completed this way is the projective plane. An homography acts in the projective plane; two parallel lines $l \parallel m$ (*i.e.,* two lines which meet on the line at infinity h) usually stop to be parallel, after an homography: they no more meet on h; h usually stops being at infinity, after the homography; however the transformed parallel lines $l' = H(h)$ and $m' = H(m)$ always meet on the transformed line at infinity $h' = H(h)$. This kind of property extends in 3D, for parallel lines or planes: the Euclidean 3D space is completed with a plane at infinity, or plane of directions, where all parallel lines meet; moreover all parallel planes meet on a line in the plane at infinity.

Parallelism constraints in the Euclidean plane or space can thus be translated into incidence constraints only, in the projective plane or space; for instance, in 2D, the Euclidean constraints: $l_1 \parallel l'_1$ and $l_2 \parallel l'_2$ are translated into projective incidence constraints: lines h, l_1, l'_1 concur, lines h, l_2, l'_2 concur; h is the line at infinity of the Euclidean plane. Fig. 10 shows a more complex example: given a square $abcd$, trace the contiguous square $a'bcd'$; the following construction applies both in the Euclidean plane (where parallelism is relevant) and in the projective plane: $i = ad \cap bc$, $j = ab \cap cd$, $k = ij \cap ac$, $d' = bk \cap cd$, $a' = id' \cap ab$; i, j, k are points at infinity in the Euclidean plane. This construction uses no Euclidean notion of parallelism but only projective point-line incidences.

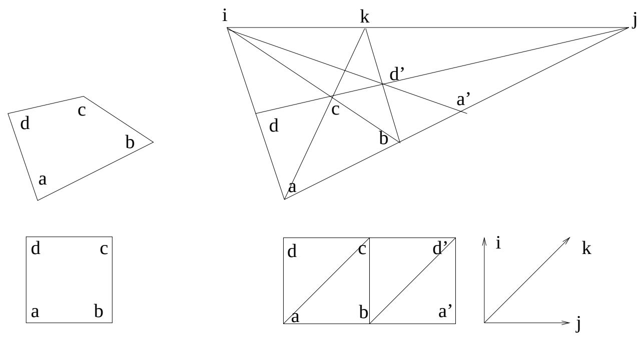

Fig. 10. Projective construction of the mirror ”square” $dbcd'$ of the ”square” $abcd$ in the Euclidean plane (top) and in the projective plane (bottom). To rely on intuition, it can be convenient to interpret the projective plane as the image of an Euclidean horizontal plane by a perspective. The line ijk is the line at infinity (the horizon) of the Euclidean plane.

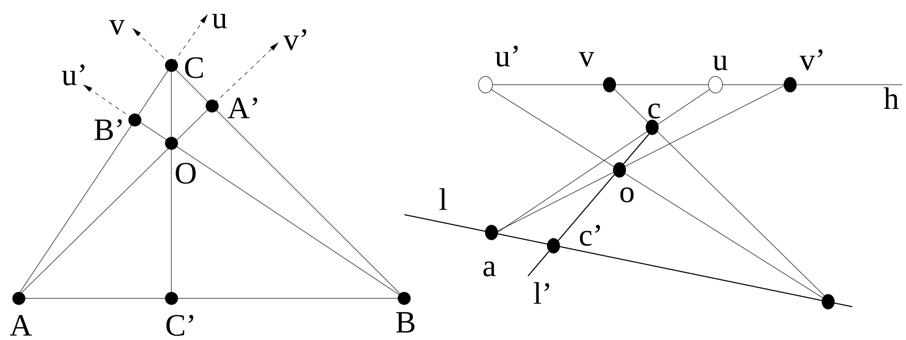

Fig. 11. Pouzergues’s construction.

8.3 Orthogonality reduces to incidence

As with parallelism, homographies do not preserve orthogonality, *e.g.*, in 2D, the three altitudes of a triangle concur in a point called orthocentre (the extension to 3D is wrong: the 4 altitudes of a tetrahedron do not concur, generically). After an homography, the orthogonality between the three sides of the triangle and their altitudes is lost, but the concurence is preserved. This property permits to define a ”projective orthogonality”, *i.e.*, an orthogonality relationship which is preserved by homographies. This projective orthogonality is first defined in Euclidean words for clarity, then translated in the projective parlance (Fig. 11).

Assume two pairs of orthogonal directions $u \perp u'$ and $v \perp v'$ are known (Fig. 11); then the line passing through a given point c and orthogonal to a given line l can be constructed as follows: draw a triangle abc, where a is the intersection point between l and the line parallel to u through c; b is the intersection point between l and the line parallel to v through c; thus $ab = l$. The altitude through b is parallel to u'; the altitude through a is parallel to v'; these two altitudes meet at the orthocentre o of abc; the third altitude through c is co; thus co is the searched line, through c and orthogonal to l.

This construction can be translated in the projective parlance. u, u', v, v' are now 4 directive points aligned on h, the line at infinity (or vanishing line). In this new parlance, two lines are parallel iff they meet on the line $h=(u,u',v,v')$. The construction from c and l of the line "orthogonal" to l through c straightforwardly follows: $a=cu\cap l$, $b=cv\cap l$, $o=av'\cap bu'$; co is "orthogonal" to l and goes through c. This projective definition of orthogonality suggests a projective definition of a "cercle" with given diameter ab as the set of points p such that ap is "orthogonal" to pb (Fig. 12). Thus, introducing 4 collinear directive points $u\perp u', v\perp v'$ on the line at infinity, all orthogonality constraints in the Euclidean plane can be translated into projective point-line incidence constraints only.

These ideas extend in 3D for orthogonal lines or planes: [31] defined a projective theory of orthogonality in 3D and solved 3D orthogonality constraints in a drawing (*i.e.*, in an image of a 3D scene) without using a 3D model of the scene. For completeness, we summarize the main facts of this theory: if u,v,w are the points at infinity of three 3D lines non pairwise orthogonal, and u',v',w' are the line at infinity of corresponding planes orthogonal to u,v,w, then the lines $(u,v'\cap w')$, $(v,u'\cap w')$, $(w,u'\cap v')$ concur: the two triangles are perspective. If u,v,w are pairwise orthogonal, then $w'=(uv), v'=(uw), u'=(vw)$, *i.e.*, the triangle uvw is auto-polar. Actually, in the plane at infinity, points at infinity and lines at infinity of orthogonal lines and planes are poles and polars.

Given u,v,w,u',v',w' in the plane at infinity, all orthogonality constraints reduce to projective point-line incidence constraints in the plane at infinity, and the geometric constructions use only the ruler, *i.e.*, they are solvable with linear algebra: let l' be the line at infinity of some plane π, and the problem is to build l, the point at infinity of the lines orthogonal to π. Then triangles (u,v,l) and $(v'\cap l', u'\cap l', u'\cap v')$ are perspective, thus $l\in(u'\cap v')\vee\omega$ with $\omega=((l'\cap v')\vee u)\cap((l'\cap u')\vee v)$ is the "center of the perspective" (again, $a\vee b$ is the line ab); similarly triangles (u,w,l) and $(w'\cap l', u'\cap l', u'\cap w')$ are perspective, which gives another line passing through l; thus l is known; a third and useless line through l is obtained, considering triangles (w,v,l) and $(v'\cap l', w'\cap l', w'\cap v')$ are perspective. A dual construction gives l' from l.

To summarize, Euclidean constraints of parallelism and orthogonality reduce to projective incidence constraints; it is not so obvious a priori. Moreover, the basic problems (*e.g.*, find the line in 2D (plane in 3D) orthogonal to a given line) are linearly solvable, *i.e.*, geometrically speaking they only need the ruler.

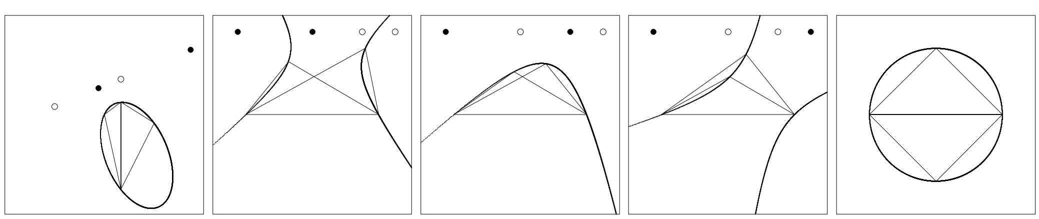

Fig. 12. Projective cercles. Small black disks are two directive points in involution. Idem for small white disks. The order of directed points along the vanishing line determines the kind of the conic.

8.4 Middleship reduces to incidence

The Euclidean constraint y is the middle of ab reduces to incidence constraints: $x = ab \cap h$ is the point at infinity of ab, and y is the harmonic conjugate of x relatively to a, b (Fig. 3).

9 Solving under-constrained systems

Finding one solution to an under-constrained system $F(Y,X) = 0$ (where unknowns Y replace known parameters U) provides a WC. Typically under-constrained systems have a continuum of solutions (though there are exceptions: *e.g.*, $(x+y+z)^2 + 1 = 0$ has no solution in $\mathbb{R}$, but it has a continuum in $\mathbb{C}$; $x+y+z = x+y+z+1 = 0$ has no solution at all). Any point lying on this continuum gives a WC. In our context (searching a WC), the system is typically trivial to solve (*i.e.*, the icosahedron or dodecahedron problems); note however that it is possible to build under-constrained problems which are arbitrarily difficult: just take an arbitrarily difficult well-constrained problem and replace one of its unknowns by the sum of 2 new auxiliary unknowns (of course this problem is artificial). We do not use this approach (solving an under-constrained system) to get a WC; we suggest three possible tracks.

First track: several methods were recently proposed to deal with under-constrained systems [32–35,30,36].

Second track: a well known subdivision method, relying on interval computations [37–40], displays 2D algebraic curves $f(x,y) = 0$ and 3D algebraic surfaces $s(x,y,z) = 0$, 3D algebraic curves: $s(x,y,z) = t(x,y,z) = 0$. Actually this subdivision method covers with small boxes the solution set of under-constrained systems; of course it can be stopped as soon as it finds a first point, which gives a WC. To limit a combinatorial explosion in high dimension space, several strategies have been proposed to halve the current box (an interval vector) according to the more relevant unknown [40].

Third track: find which key unknowns to fix in order to minimize the algebraic degree of the system. Label each edge of the bipartite graph equation-unknown with the $\log_2$ of the corresponding degree, and compute a maximum matching with minimal cost. The unknown-vertices which are uncovered give the key unknowns, *i.e.*, the unknowns to fix. If the maximum matching has cost 0, the system is linear, and the problem is very similar to trivial systems of incidences. For instance, with $e : x^2 - y = 1$, the graph is (e,x) with cost 1, (e,y) with cost 0, and the best matching is the edge (e,y) with total cost 0: x is the unknown to fix and the system is linear. Once key unknowns are found, the system is structurally well-constrained, and decomposed into irreducible well-constrained subsystems, and solved with some numerical or interval method (as in the first track). A possible flaw is that, if the key unknowns are fixed at random, possibly with bad values, the system may have no solution. One can also imagine to subdivide the space of key unknowns, in the spirit of the previous track, and in the wake of the locus method by Gao, Hoffmann and Yang [41]. Another potential weakness is that this track assumes that constraints are independent and relies on graph-based methods to compute a WC: it is ironical since the WCM is just designed to overcome the limitations of graph-based methods.

10 Solving systems of incidences

Solving systems of incidence constraints, more precisely finding one solution, is a possible way to obtain a WC. Most of the time, the sketch (which the user interactively provides to specify the system of constraints) is a solution. This section describes trivial systems of incidences, almost trivial systems, how these systems reduce to algebraic systems and more specifically to a molecule problem, and finally shows their universal feature.

10.1 Trivial systems

This section defines trivial systems of point-line incidences in 2D. A point is removable iff it is constrained to lie on 2 lines or less: indeed, it can be removed from the set of constraints, the remaining constraints are solved, then the removed point is added (where the 2 lines meet). Dually for a line. Removing a point or a line may make removable another point or line. If all points and lines can be removed, the system is trivial. All configurations in Section 3 are trivial, except D, P and the pentagonal star; they are almost trivial (see Section 10.2). This may be easily extended to 3D.

The icosaedron problem is difficult but it is an instance of the 3D molecule problem, so finding a WC is trivial: the previous method fixes 12 vertices at random in 3D and deduces the distances (as in Section 9, third track). The dodecahedron problem is as or more difficult, and it is not a molecule problem. However, finding a WC is also trivial: the previous method reduces to choose 12 planes at random in 3D, and then computes the 20 intersection points and finally compute the distances (as in Section 9, third track).

10.2 Almost trivial systems

A system is almost trivial if removing an incidence constraint makes it trivial. The P (for Pappus), D (for Desargues) and the pentagonal star configurations are almost trivial. One may remark that for P and D, removing any one of the incidences makes them trivial.

The strategy is obvious: if S is almost trivial, remove an incidence $I \in S$ to get a trivial system $S-I$; find a WC x^w for $S-I$; if x^w satisfies the removed incidence I, then I is (very probably) a consequence of $S-I$: it is the case for the Pappus and Desargues theorems (again, to gain more confidence in the probabilistic proof, the test can be repeated with another WC of $S-I$). In this case, we have proved a redundancy, and found a WC otherwise I is not a consequence of $S-I$, but a constraint independent of $S-I$ (and repeating the test with another WC of $S-I$ is absurd and useless); this case occurs with the pentagonal star; the WCM proves that the constraints of the star are independent, and indeed they are. Unfortunately, no WC is found in this case for the moment (since we do not even try to solve the system). The presence of a WC would make possible to detect, say, the collinearity of I, $DG \cap CH$ and $DH \cap CG$.

Two remarks conclude this section: We admit that the pentagonal star seems unlikely and not natural for CAD-CAM. More natural examples in this contexte may be given. The notion of almost trivial seems generalizable. A system S is k almost trivial iff removing k incidences gives a trivial system S'; compute a WC of S', and check if each of the k removed constraints are satisfied or not. If they are all satisfied, then we have proved k theorems and found a WC! However, there is a flaw in this approach: if a removed incidence I is a consequence of H, with $S' \subset H \subseteq S-I$, the WCM will not detect it.

10.3 Point-line incidences is a 3D molecule problem

2D point-line incidence problems obviously reduce to algebraic systems. Represent each point P_i with homogenized coordinates (x_i, y_i, z_i). Represent each line L_j with a vector (a_j, b_j, c_j). Represent each incidence $P_i \in L_j$ by an equation: $P_i.L_j = a_j x_i + b_j y_i + c_j z_i = 0$. Possibly, normalize vectors of points: $P_i.P_i = 1 \Rightarrow x_i^2 + y_i^2 + z_i^2 - 1 = 0$; idem for lines. Each solution to the algebraic system gives a (possibly degenerate: all points are equal, all lines are equal) solution to the PLI system.

To reduce the PLI problem: $P_i \in L_j$ to the 3D molecule problem, we can suppose that vectors P_i and L_j have norm 1, so P_i and L_j are points on the unit sphere; each incidence $P_i \in L_j$ gives a distance ($\sqrt{2}$) between P_i and L_j. Add the center of the unit sphere to the set of points of the molecule problem, and distances (all equal to 1) between this center and every point P_i and L_j. Every solution to the molecule problem gives a solution (possibly degenerate) to the PLI problem. It is possible to avoid degenerate solutions by imposing a distance between one P_i and a non incident L_j, or imposing 4 points (no 3 collinear) to lie on some square on the sphere.

The molecule problem is the variant EDM_3 of the Euclidean Distance Matrix completion (EDM) [20]. The EDM problem is to determine whether a partially filled square symmetric matrix A can be completed to a distance matrix, *i.e.*, whether there are points P_i in $\mathbb{R}^k$ for some k such that $A_{ij} = \overrightarrow{P_iP_j}.\overrightarrow{P_iP_j}$. Specifying a value for

k gives the EDM_k problem. The EDM and EDM_k problems are closely connected to the Positive (semi) Definite Matrix Completion problem (PMC) which consists in determining whether a partially filled square matrix A can be completed to a positive (semi) definite matrix, *i.e.*, a Gram matrix . EDM_1 is NP-complete, and EDM_k is NP-hard for $k \geq 2$ [20].

10.4 Solving point-plane incidence problems

In 3D, for problems involving incidences between points, lines and planes, it suffices to consider points and planes: indeed, to express points are collinear, it suffices to say they lie on two auxiliary planes. Then point-plane incidences straightforwardly reduce to a 4D molecule problem.

After the universality theorem, systems of point-plane incidences reduce (though cryptically) to systems of point-line incidences.

10.5 Universality of the PLI problem

Though the incidence systems considered in this paper are all trivial or almost trivial, these systems can be arbitrarily difficult. Actually all algebraic systems (*e.g.*, geometric systems in 2D or 3D) can be reduced to (or rather encrypted into) a PLI problem with a size of the same magnitude.

Theorem 1 (Universality theorem) *All algebraic systems of equations with coefficients in $\mathbb{Z}$ or $\mathbb{Q}$ and unknowns in a field K ($\mathbb{R}$, $\mathbb{C}$, $\mathbb{Z}/p\mathbb{Z}$) reduce to a system of point-line incidence constraints in the projective plane on $\mathbb{K}$.*

The universality theorem is striking, but its proof is trivial, and needs ingredients known since more than one century. As far as we know, it has not been published so far.

Proof: Numbers are represented by points along a particular line, where three distinct points are arbitrarily chosen, and called $0, 1$ and ∞. A geometric construction in Fig. 13 gives the point representing $a+b$, from the points representing a and b. The construction is first illustrated in the affine plane, using parallelism, then in the projective plane. The proof of the correctness of this construction, *e.g.*, that the resulting point $a+b$ is independent of the used auxiliary points or lines, that the point $a+b$ equals the point $b+a$, that the addition so defined is indeed associative, etc. uses the Desargues property [42].

Similarly, another construction in Fig. 14 gives the point representing $a \times b$, from the points representing a and b. The construction is first illustrated in the affine plane, using parallelism, then in the projective plane. The proof of the correctness

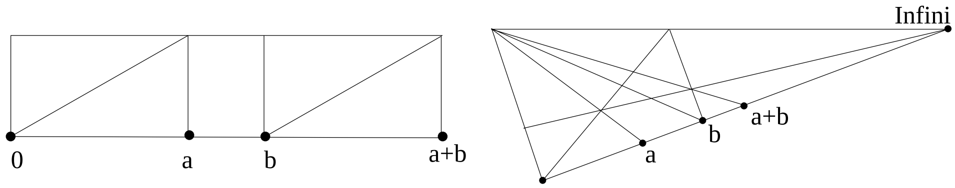

Fig. 13. the operation $a+b$. Affine and projective variants.

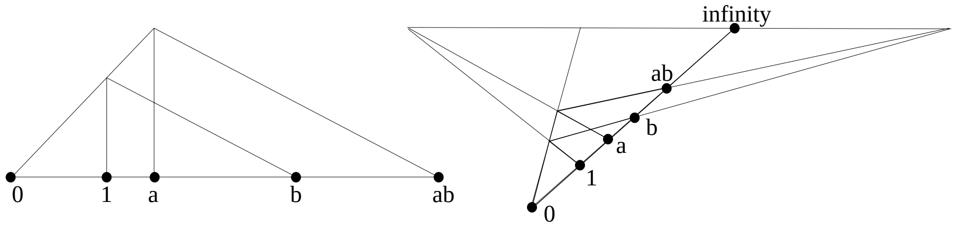

Fig. 14. The operation $a \times b$. Affine and projective variants.

of this construction uses the Desargues property again; the proof that the defined multiplication is commutative uses the Pappus property [42].

The geometric construction for one addition, and for one multiplication, uses a constant number of points, lines, and incidences. The point representing every integer coefficient occurring in the algebraic system is needed. Using iterated squaring, an integer n can be constructed with $O(\log_2 |n|)$ incidences from the points $0, 1$ and ∞. Thus the geometric representation of an integer has the same size as the usual binary one.

Every unknown of the algebraic system is also represented by a point on the line $0, 1, \infty$. Iterated squaring is also used to construct the point x^k for given $k \in \mathbb{N}$. The point representing a monomial $x_1^{d_1} x_2^{d_2} \ldots$ is deduced by geometric multiplication of the points representing $x_1^{d_1}$, $x_2^{d_2}$, etc. The point representing a polynomial is the geometric sum of the points representing its monomials. Each equation $E_i = 0$ is represented in the PLI system by merging the point representing E_i and the point 0, *i.e.*, by replacing every reference to the point E_i by a reference to the point 0 in all point-line incidences. Fig. 15 shows a solution of the PLI system of the equation $x^2 - 2 = 0$.

The size of the resulting PLI system (the number of incidences) is proportional to the bit size of the algebraic system. Remark that all numbers (coefficients, degrees) occurring in the algebraic system are represented by points, lines and incidences only, *i.e.*, by a *pure bipartite graph*, with size proportional to the bit size of the algebraic system. No edge weight is needed, contrarily to graphs of the molecule problem in the rigidity theory (so, the genericity assumption is neither needed nor possible). □

Consequences of the universality theorem: solving algebraic systems is NP-hard, thus PLI as well; the combinatorial (*e.g.*, graph-theoretical) characterization of

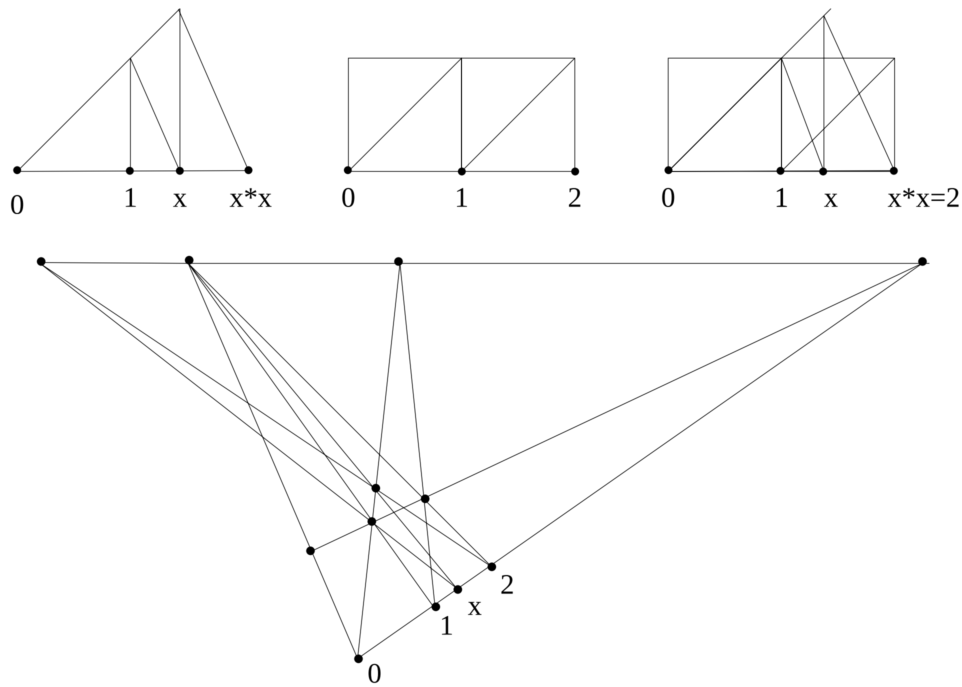

Fig. 15. Above: affine representation of the equation $x^2 - 2 = 0$; left: construction of the point x^2; middle: construction of the point 2; right: superposition of the two constructions. Below: projective representation of the equation $x^2 - 2 = 0$.

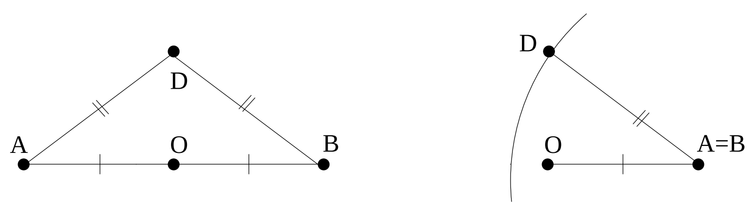

Fig. 16. This system, due to C. Jermann, is built in order to have two kinds of solutions – thus two kinds of witness configurations.

well-constrained PLI systems, of PLI systems realizable in $\mathbb{R}$, of PLI systems realizable in $\mathbb{C}$ seem difficult.

11 Conclusion

This paper proposed a new strategy for studying and decomposing systems of geometric constraints. A witness configuration is computed (very easily most of the time), and then studied and decomposed with classical linear algebra tools. This paper has shown that probabilistic proofs, which are used in computer algebra, are also practicable and relevant for GCS in CAD-CAM-PLM. We conclude with future works:

- There are basically two ways to obtain a witness configuration: i) consider parameters as unknowns and then solve the resulting under-constrained system;

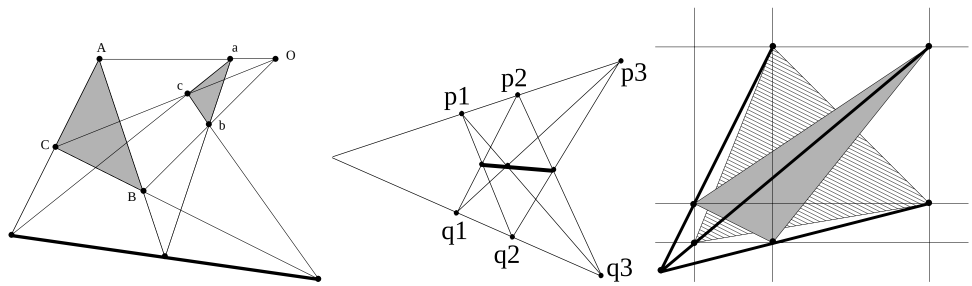

Fig. A.1. Left to right: Desargues, Pappus, Pappus' dual.

and ii) explicit all projective constraints. This second option is discussed in section 8, while the first option (solving under-constrained systems) is not yet tried and will be subject to further investigations.

- We do not use classical resolution methods used in GCS to find a witness configuration when the system of incidences is not trivial. It would result in interleaving the qualitative study and the numeric resolution, which are today performed sequentially. It would significantly increase the power of solvers.
- We do not treat (nor detect) configurations such as Jermann's one (Fig. 16), with several, qualitatively distinct, solutions – and thus with several witness configurations.

A Some theorems

W.A. Martin [7] and J.T. Schwartz [8] introduced probabilistic proofs (to prove algebraic identities). A counter example is sufficient to prove a conjecture is wrong; conversely, the principle of probabilistic proofs is that a generic example is sufficient to prove a conjecture: if a generic example, satisfying the hypothesis of a conjecture, and randomly chosen in an infinite set, satisfies also the conclusion of the conjecture, then the conjecture is true with probability 1, *i.e.,* with a null probability of error. In practice, the example is chosen in a finite set (the representation of examples has finite size) with size Ω, so the probability of error is no more 0 but $\varepsilon \approx 1/\Omega$. Note that the genericity assumption is also needed for counter examples: indeed some geometric theorems do not hold for degenerate situations (*e.g.,* a degenerate triangle). Probabilistic proofs can be made deterministic (no more probabilistic), but it has an exponential cost. Using this principle, the WCM detects all the following theorems (for the 3 quadrics theorem, the WC was built by hand, so that the intersection points have rational coordinates).

Theorem 2 (Desargues theorem) *If two triangles abc and ABC are perspective (i.e., aA, bB, cC concur), then homologous sides cut in 3 aligned points, i.e., $ab \cap AB, bc \cap BC, ca \cap CA$ are collinear. The converse is true as well.*

Theorem 3 (Pappus theorem) *If p_1, p_2, p_3 are three distinct aligned points of P, and if q_1, q_2, q_3 are three distinct aligned points, then the three intersection points*

$p_1q_2 \cap p_2q_1$, $p_1q_3 \cap p_3q_1$ and $p_2q_3 \cap p_3q_2$ are aligned as well.

Theorem 4 (Harmonic conjugate) *Let a, b, x be 3 distinct aligned points. Let l be any line through x, c any point outside l and abx. Then the point y defined by the construction: $p_1 = ca \cap l$, $p_2 = cb \cap l$, $p_3 = p_1b \cap ap_2$, $y = cp_3 \cap ab$, depends neither on l nor on c. See Fig. 3.*

The point y is called the harmonic conjugate of x relatively to a, b. The harmonic conjugate of y is x.

Theorem 5 (Pascal theorem) *If $p_1, p_2, p_3, q_1, q_2, q_3$ lie on a common conic in P, then the three intersection points $p_1q_2 \cap p_2q_1$, $p_1q_3 \cap p_3q_1$ and $p_2q_3 \cap p_3q_2$ are aligned.*

Coming from the Pascal theorem, we have:

Theorem 6 (Pouzergues hexamy theorem) *In P, a hexamy is a hexagon (possibly concave and self intersecting) such that its three opposite sides cut in three aligned points. Then every permutation of a hexamy is a hexamy too.*

The duality principle, due to Poncelet, Plucker and Gergonne, exchanges the role of points and lines, and permits to find and prove dual theorems. *e.g.,* by applying duality to Pappus's theorem, one can find:

Theorem 7 (Pappus dual) *If p_1, p_2, p_3 are three distinct concurrent lines, and q_1, q_2, q_3 are three other concurrent lines, then $(p_1 \cap q_2, p_2 \cap q_1)$, $(p_2 \cap q_3, p_3 \cap q_2)$, $(p_3 \cap q_1, p_1 \cap q_3)$ are concurrent lines as well.*

or, in other words, if two triangles are perspective in two ways, then they are perspective in three ways. The dual of Desargues theorem is its converse. The dual of Pascal theorem is Brianchon theorem. Applying duality to the hexamy theorem, one finds that:

Theorem 8 (Hexamy dual) *A symhex is an hexagon such that the three lines through the three pairs of opposite vertices are concurrent. Then every permutation of a symhex is a symhex.*

Theorem 9 (Chasles) *In the projective plane, if Q_1 and Q_2 are two cubic curves without a common component, which cut in 9 distinct points, then all cubic passing through 8 of these points also passes through the 9th point.*

Actually Pappus theorem is a consequence of Chasles' theorem.

Proof: let Q_1 be the cubic curve composed of the 3 lines p_1q_2, p_2q_3, p_3q_1. Let Q_2 be the cubic curve composed of the 3 lines p_2q_1, p_3q_2, p_1q_3. Let C be the conic composed of the 2 lines $p_1p_2p_3$ and $q_1q_2q_3$. Let K be the cubic $C \cup i_3i_1$ where $i_3 = p_1q_2 \cap p_2q_1$, $i_1 = p_2q_3 \cap p_3q_2$. We note $i_2 = p_3q_1 \cap p_1q_3$. The 2 cubic curves Q_1 and Q_2 cut in the 9 points p_k, q_k, i_k, $k \in 1, 2, 3$. The cubic K passes through all these points too, except i_2. By Chasles theorem, since it passes by 8 of the 9 points, it also passes by the 9th point i_2. Thus $K = p_1p_2p_3 \cup q_1q_2q_3 \cup i_3i_1$ passes through i_2. But i_2 does not lie on $p_1p_2p_3 \cup q_1q_2q_3$, thus it lies on i_3i_1, and i_1,i_2,i_3 are aligned. $\square$

Proof of Pascal theorem: This proof is similar to the previous one, we only need to replace the conic C by the conic passing through the 6 points $p_k, q_k, k \in 1,2,3$. $\square$

Chasles' theorem will be proved below, with a combinatorial argument which can be extended to the 3D case, for instance to prove the following theorems:

Theorem 10 (Three quadrics) *In 3D projective space, let Q_1,Q_2,Q_3 be three quadrics, which cuts in 8 distinct points. Then all quadrics passing through 7 of these 8 points also goes through the 8th point.*

Theorem 11 (Beltrami or Galucci) *16 points theorem: In 3D projective space, be given 3 black lines, non pairwise coplanar, and 3 white lines, non pairwise coplanar, such that each black line cuts all white lines; black and white lines cut in 9 points. Then every line cutting the 3 black lines cut every line cutting the 3 white lines.*

Theorem 12 (Cox) *Let $p_1,\dots p_n$ be $n=4$ coplanar points. Let P_{ij} be $n(n-1)/2 = 6$ planes, where $p_i \in P_{ij}, p_j \in P_{ij}$, with $i=1,\dots n$. Define n points $s_{ijk} = P_{ij} \cap P_{ik} \cap P_{jk}$. Then these n points s_{ijk} are coplanar too. Actually, the theorem holds for any integer $n \geq 4$. The dual theorem also holds.*

Theorem 13 (Mobius's tetrahedrons) *In 3D projective space, let abcd and ABCD be two tetrahedrons such that $a \in BCD$, $b \in ACD$, $c \in ABD$, $d \in ABC$, and $A \in bcd$, $B \in acd, C \in abd$. Then $D \in abc$. In other words, each vertex of one tetrahedron lies on a face of the other, and vice versa (abcd and ABCD are Mobius's tetrahedrons). It gives 8 incidences; any 7 of the 8 incidences force the 8th one.*

These theorems all share the same combinatorial flavor. Actually, their proof will use the same combinatorial argument as this seemingly trivial theorem:

Theorem 14 *In 3D space, if 3 distinct points lie on 2 distinct planes, then every plane (or line) passing through two of these points also passes through the third.*

Proof of Chasles theorem: points in the projective plane are represented by 3-vectors $v=(x,y,z)$. A cubic lifting maps the 3-vector $v=(x,y,z)$ to the 10-vector

$$\phi_3(x,y,z) = (x^3,y^3,z^3,xyz,x^2y,x^2z,y^2z,xy^2,xz^2,yz^2)$$

Let Q be a cubic curve; then $\phi_3(Q)$ is an hyperplane in a 10 dimensional vectorial space, thus $\phi_3(Q)$ has vectorial rank 9 (thus 9 points are needed to define a cubic curve). By hypothesis, Q_1 and Q_2 are 2 cubic curves in the projective plane; they have no common component, and they intersect in 9 distinct points; Bézout theorem bounds the number of roots of polynomial systems. Q_1 has vectorial rank 9. Idem for Q_2. The intersection of 2 hyperplanes with rank 9 has rank 8. Thus the 9 vectors of the intersection points are not linearly independent. Every vectorial space, such as an hyperplane, through 8 of them also contains the remaining 9th. In other words, every cubic curve through 8 of the 9 intersection points goes through the remaining 9th. $\square$

Consequence: a cubic is defined by 9 points, under mild assumptions (the independence of the 9 points). If these 9 points come from the intersection of two cubics, they are not independent.

Proof of the 3 quadrics theorem:
the quadratic lifting maps the 4-vector $v = (x, y, z, w)$, representing a point in 3D space, to the 10-vector $\phi_2(v) = (x^2, y^2, z^2, w^2, xy, xz, xw, yz, yw, zw)$. Let C be a quadric surface in the projective space (a 4 dimensional vectorial space). Then $\phi_2(C)$ is an hyperplane in a 10 dimensional vectorial space, thus $\phi_2(C)$ has vectorial rank 9; thus 9 points are needed to define a quadric surface. If the 9 points are in generic position, there is only one quadric. Let Q_1, Q_2, Q_3 be 3 quadric surfaces in projective space, which intersects in 8 distinct points (Bézout's theorem bounds the number of intersection points). $\phi_2(Q_1)$ is an hyperplane in a 10 dimensional vectorial space, thus $\phi_2(Q_1)$ has rank 9. Idem for $\phi_2(Q_2)$ and $\phi_2(Q_3)$. $\phi_2(Q_1) \cap \phi_2(Q_2)$ has rank 8, $\phi_2(Q_1) \cap \phi_2(Q_2) \cap \phi_2(Q_3)$ has rank 7. Thus the 8 lifted vectors representing the 8 intersection points are not independent. Every vectorial space (such as an hyperplane) through 7 of them also contains the remaining 8th. In other words, every quadric surface through 7 of the 8 intersection points goes through the remaining 8th. $\square$

This proof is similar to the proof of Chasles theorem. The proof of the other theorems: Beltrami, Cox, Mobius ([43], pages 138-140) use similar combinatorial arguments. Automatizing this kind of proofs seems interesting but it is an open question.

References

[1] J. Graver, B. Servatius, H. Servatius, Combinatorial Rigidity, Vol. 2 of Graduate Studies in Mathematics, American Mathematical Society, Providence, RI, USA, 1993.

[2] H. Lamure, D. Michelucci, Qualitative study of geometric constraints, in: Geometric Constraint Solving and Applications, Springer-Verlag, 1998.

[3] B. A. Hendrickson, Conditions for unique graph realizations, SIAM J. Computing 21 (1) (1992) 65–84.

[4] S. Foufou, D. Michelucci, J.-P. Jurzak, Numerical decomposition of geometric constraints, in: ACM Symp. on Solid and Physical Modelling, MIT, Boston, USA, ACM Press, 2005, pp. 143–151.

[5] J. Richter-Gebert, U. Kortenkamp, The Interactive Geometry Software Cinderella, Springer, 1999.

[6] A. Agrawal, A. Requicha, A paradigm for the robust design of algorithms for geometric modeling, Computer Graphics Forum (EUROGRAPHICS'94) 13 (3) (1994) C–33–C–44.

[7] W. Martin, Determining the equivalence of algebraic expressions by hash coding, J. ACM 18 (4) (1971) 549–558.

[8] J. Schwartz, Fast probabilistic algorithms for verification of polynomial identities, J. ACM 4 (27) (1980) 701–717.

[9] M. Monagan, G. Gonnet, Signature functions for algebraic numbers, in: ISSAC '94: Proceedings of the international symposium on Symbolic and algebraic computation, ACM Press, 1994, pp. 291–296.

[10] M. Benouamer, P. Jaillon, D. Michelucci, J.-M. Moreau, A lazy arithmetic library, in: Proc. of the IEEE 11th Symp. on Computer Arithmetic, Windsor, Ontario, 1993, pp. 242–269.

[11] M. Benouamer, P. Jaillon, D. Michelucci, J. Moreau, Hashing lazy numbers, Computing 53 (3–4) (1994) 205–217.

[12] J. Hong, Proving by example and gap theorem, in: I. C. S. Press (Ed.), 27th symposium on Foundations of computer science, Toronto, Ontario, 1986, pp. 107–116.

[13] I. Emiris, A. Rege, Monomial bases and polynomial system solving (extended abstract), in: ISSAC '94: Proceedings of the international symposium on Symbolic and algebraic computation, 1994, pp. 114–122.

[14] A. Rege, A complete and practical algorithm for geometric theorem proving, in: SCG '95: Proceedings of the eleventh annual symposium on computational geometry, ACM Press, 1995, pp. 277–286.

[15] D. Bouhineau, Constructions automatiques de figures géométriques et programmation logique avec contraintes, Ph.D. thesis, Université Joseph Fourier, Grenoble (1997).

[16] D. Tulone, C.-K. Yap, C. Li, Randomized zero testing of radical expressions and elementary geometry theorem proving, in: ADG '00: Revised Papers from the Third International Workshop on Automated Deduction in Geometry, Springer-Verlag, London, UK, 2001, pp. 58–82.

[17] H. Gluck, Geometric Topology, Springer-Verlag, Berlin, 1975, Ch. Almost all simply connected closed surfaces are rigid, pp. 225–239.

[18] T. F. Havel, Some examples of the use of distances as coordinates for euclidean geometry, Journal of Symbolic Computation 11 (1991) 579–593.

[19] L. M. Blumenthal, Theory and applications of distance geometry, Clarendon Press, Oxford, 1953, (for the second edition: Chelsea, New York, 1970).

[20] M. Laurent, Matrix completion problems, The Encyclopedia of Optimization 3 (Interior - M) (2001) 221–229.

[21] B. A. Hendrickson, The molecule problem: determining conformation from pairwise distances, Ph.D. thesis, Cornell University, Ithaca, NY, USA (1991).

[22] B. A. Hendrickson, The molecule problem: Exploiting structure in global optimization, SIAM Journal on Optimization 5 (1995) 835–857.

[23] G. M. Crippen, T. F. Havel, Distance Geometry and Molecular Conformation, Research Studies Press, Taunton, U.K., ISBN 0-86380-073-4, 1988.

[24] H. Coxeter, Projective geometry, Springer-Verlag, Heidelberg, 1987.

[25] P. Serré, Cohérence de la spécification d'un objet de l'espace euclidien à n dimensions, Ph.D. thesis, Ecole Centrale de Paris (2000).

[26] A. Recski, Matroid theory and its applications, Springer Verlag, 1986.

[27] L. Lovasz, Y. Yemini, On generic rigidity in the plane, SIAM J. Alg. Discr. Meth. 3 (1) (1982) 91–98.

[28] L. Lovasz, M. Plummer, Matching Theory, North-Holland, 1986.

[29] D. Michelucci, S. Foufou, Using Cayley-Menger determinants for geometric constraint solving, in: ACM Symposium on Solid Modeling, ACM Press, Genova, Italy, 2004, pp. 285–290.

[30] X.-S. Gao, G. Zhang, Geometric constraint solving via c-tree decomposition, in: ACM Solid Modelling, ACM Press, New York, 2003, pp. 45–55.

[31] P. Macé, D. Michelucci, L.-P. Untersteller, Solving space orthogonality constraints in a 2d drawing, in: Fifth International Workshop on Automated Deduction in Geometry, University of Florida in Gainesville, Florida, USA, 2004.

[32] R. Joan-Arinyo, A. Soto-Riera, S. Vila-Marta, J. Vilaplana-Pasto, Revisiting decomposition analysis of geometric constraints graphs, in: Proc. ACM SM02, ACM Press, 2002, pp. 105–115.

[33] R. Joan-Arinyo, A. Soto-Riera, S. Vila-Marta, J. Vilaplana-Pasto, Transforming an under-constrained geometric constraint problem into a well-constrained one, in: Proc. ACM SM03, ACM Press, 2003, pp. 33–44.

[34] R. Joan-Arinyo, A. Soto-Riera, S. Vila-Marta, Tools to deal with under-constrained geometric constraints graphs, in: Workshop on geometric Constraints Solving, Beijing, China, 2003.

[35] X.-S. Gao, G. Zhang, Geometric constraint solving based on connectivity of graphs, Tech. Rep. MMRC, AMSS, Academia Sinica, Beijing, 22, MM Research Preprints (2003).

[36] M. Wilczkowiak, G. Trombettoni, C. Jermann, P. Sturm, E. Boyer, Scene modelling based on constraint system decomposition techniques, in: Proc. Int. Conf. on Computer Vision, 2003, pp. 1004–1010.

[37] G. Taubin, An Accurate Algorithm for Rasterizing Algebraic Curves, in: Second Symp. on Solid Modeling and Applications, ACM/IEEE, 1993, pp. 221–230.

[38] L. de Figueiredo, J. Stolfi, Adaptive Enumeration of Implicit Surfaces with Affine Arithmetic, in: Proc. Eurographics Workshop on Implicit Surfaces, INRIA, 1995, pp. 161–170.

[39] D. Michelucci, S. Foufou, L. Lamarque, D. Ménégaux, Bernstein based arithmetic featuring de Casteljau, in: Canadian Conference on Computational Geometry, University of Windsor, Canada, 2005, pp. 212–215.

[40] COPRIN, ALIAS-C++: A C++ algorithms library of interval analysis for equation systems, version 2.3, Tech. rep., INRIA Sophia-Antipolis (2004).

[41] X.-S. Gao, C. M. Hoffmann, W.-Q. Yang, Solving spatial basic geometric constraint configurations with locus intersection, in: SMA '02: Proceedings of the seventh ACM symposium on Solid modeling and applications, ACM Press, New York, NY, USA, 2002, pp. 95–104.

[42] J. E. Bonin, Introduction to matroid theory, Tech. rep., The George Washington University, http://home.gwu.edu/˜jbonin/survey.ps (2002).

[43] H. Coxeter, The beauty of geometry. 12 essays, Dover publications, 1999.