Solving Geometric Constraints by Homotopy

Hervé Lamure, Dominique Michelucci

Solving Geometric Constraints by Homotopy

Hervé Lamure & Dominique Michelucci

Abstract-Numerous methods have been proposed in order to solve geometric constraints, all of them having their own advantages and drawbacks. In this article, we propose an enhancement of the classical numerical methods, which are, up to now the only ones that apply to the general case.

Key words-Geometric constraints, numerical resolution, homotopy, continuation methods, constaints-based modeling

1 Introduction

Homotopy (also called the continuation method) has long served as useful theoretical tools in modern mathematics. Because of their versatility and robustness, numerical continuation methods have now been finding ever wider use in scientific applications. In Computer Graphics, Dobkin, Levy, Thurston, and Wilks [8] trace curves in R^n with a method inspired by a special kind of homotopy, i.e., piecewise linear approximations. In CAD, this method is used in robotics and kinematics, in particular by Morgan, Wampler, and Sommese [16], [27] who find all complex solutions of polynomial systems arising in these areas. In geometric modeling with free form surfaces, a special case of homotopy ('marching method') is also often used to follow intersection curves between surfaces in $\mathbb{R}^3$ [20]. But, as far as we know, it has not been used in the area of geometric modeling, except in a restricted way (see Section 2.1.3).

Here is a brief introduction to homotopy (see Appendix A for more details about continuation methods). Suppose you have to solve $G(X) = 0$, a system of n equations with n unknowns, Suppose, also, that you know the solution S of another system $F(X) = 0$, and that F is, in a certain sense, close to G, you can define an interpolation between F and G: $H(t, X) = tG(X) + 1 - t)F(X)$ called *homotopy*. $H(t, X) = 0$ defines a curve in $\mathbb{R}^{n+1}$, parameterized by t, that you can follow from $(0, S)$ to a solution of H such that $t = 1$.

The need of an initial solution is a constant for numerical methods, but homotopy has some great advantages over the classical Newton-Raphson method. Section 2 also presents a comparison with other methods used to solve the geometric constraints: decomposition methods and symbolic computation.

Section 3 presents the implementation that we have done of a homotopy in a constraint-based modeler. Numerous extensions of the basic method are also presented here. Some of them have already been done, while the others are only in preparation. Section 4 summarizes homotopy in constraint-based geometric modelers.

2 Homotopy Against Other Methods

In CAD, geometric modeling by constraints enables users to describe geometric objects such as points, lines, circles, conics, Bézier curves, etc. in 2D and planes, quadrics, tori, Bézier patches, etc. in 3D, by geometric constraints, i.e., distances or angles between elements, incidence, or tangency relations.... This modeling yields large systems of equations, typically algebraic ones. The problem is then to solve such constraints systems.

Since the seminal work of Sutherland [23], a lot of research has been done on the topic of solving constraints. We roughly classify resolution methods for constraint systems in three (nonexclusive) categories: numerical algorithms, symbolic computations, and decomposition methods.

2.1 Numerical Methods

Numerical methods are essential to solve big constraint systems met in the 'real world' of CAD. Numerical methods are used, amongst others, by Borning [3], Nelson [18], and Light, Lin, and Gossard [15]. The most popular numerical method is Newton-Raphson's iteration and its variants; it needs an initial guess of the solution, typically given by the user thanks to some interactive tools. In an interactive application, the need for an initial guess is not really a drawback. On the contrary, it allows us to select in a natural and interactive way the desired solution among an exponential number of possibilities. By Bezout's theorem, a polynomial system of total degree d, in n unknowns and equations, has $O(d^n)$ solutions. Moreover, Newton-Raphson's method is fast and works in polynomial time. Last, its use is compatible with decomposition methods (see [1] for instance) that speed up resolution.

If Newton-Raphson's method often works fine, sometimes—too often!—it does not converge; or it converges to an unwanted solution after a 'chaotic' behavior. In such a case, the user does not know what to do, apart from slightly changing his initial guess, until Newton-Raphson's method works—if it does! This section intends to show that the homotopic method is much more satisfactory in these situations. Its behavior is very easy to predict, intuitive, and self-explanatory.

2.1.1 Homotopy Convergence

Fig. 1 shows a typical failure of Newton-Raphson's iteration, and Fig. 2 shows the behavior of homotopy on the same example; the 10 circles must be tangent to each other, and must be tangent to the triangle. These images are extracted from those interactively displayed during the resolution process.

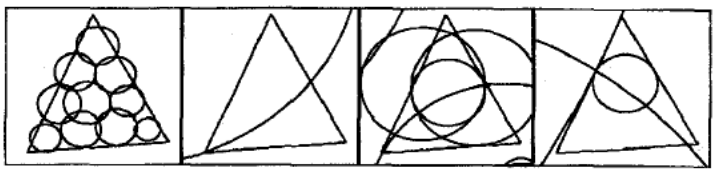

Fig. 1. Failure of Newton-Raphson's method.

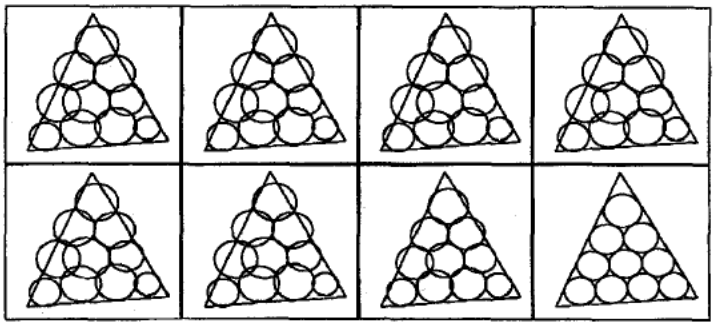

Fig. 2. Homotopy success.

2.1.2 Homotopy is Predictible

Another argument uses equation $z^3 - 1 = 0$ in $\mathbb{C}$. Of course, we do not generally need to work in $\mathbb{C}^n$ for CAD, and this equation is just a short cut for the system with two unknowns: $x, y \in \mathbb{R}^2$, and two equations: $x^3 - 3xy^2 = y^3 - 3yx^2 = 0$. Fig. 3 shows attraction basins for Newton-Raphson's method and for homotopy. Basins for Newton-Raphson's method are fractals [21]. This explains why it is so difficult for users to predict which solution this method will converge to. On the contrary, homotopy converges to the closest[1] solution. Homotopy basins have smooth frontiers: they are semi-algebraic sets[2] when the system to be solved is algebraic. This point is detailed further in -E. Generally, homotopy converges much more often than Newton-Raphson's iteration to the solution intuitively closest to the initial guess, though, of course, this claim cannot be proven in a rigorous way, since 'intuitively close' has no mathematical definition.

It is well known that attraction basins of the Newton-Raphson's resolution are fractals [21] (see Fig. 3 for instance). This explains the 'chaotic' behavior of this method. Semi-algebraic sets are less beautiful than fractals, but much smoother.

2.1.3 A Restricted Homotopic Method

Sometimes, people in constraints-based modeling use the following restricted homotopy [24]. Suppose a first solution X_0 to a constraint system c_0 is known, but the value of some parameter $p \in \mathbb{R}$ (a length, an angle, a radius...) has to be changed, say from p_0 to p_1. It is convenient to see the constraint system c as a function of p, say: $c = C(p)$ and so $c_0 = C(p_0)$, and we want to solve $c_1 = C(p_1)$. When directly solving $C(p_1)$ by Newton-Raphson's iteration with X_0 as an initial guess, there is a risk of divergence. So a natural idea is to first solve, say, $C(0.9\ p_0 + 0.1\ p_1)$ with starting guess X_0 to get $X_{0.1}$ by using Newton-Raphson's method or Secant method, then to solve $C(0.8\ p_0 + 0.2\ p_1)$ with the starting guess $X_{0.1}$ to get $X_{0.2}$, and so on, until solution X_1 of $C(p_1) = 0$ is found. Of course, this method can be generalized to any number of steps (not only 10), and for changing any number of parameters. It is a kind of naive climbing homotopy.

This method has a serious limitation; it only works when the followed path has no turning points. Since turning points are found even with very simple examples, the climbing homotopy is relevant only in the field of complex numbers (and with some kind of perturbation to avoid bifurcation points).

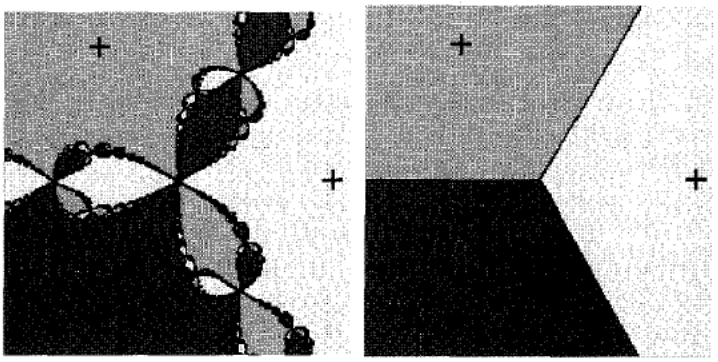

Fig. 3. $z^3 - 1 = 0$, $z \in \mathbb{C}$: Attraction basins with Newton-Raphson's method and homotopy. The +'s represent the three roots of unity.

2.1.4 Homotopy Limitations

Of course, homotopy has some limitations, but it appears that they have not a great importance in the interactive context. Roughly speaking, homotopy is on average 10 to 20 times slower than Newton-Raphson's method, which it calls about 10 to 20 times.

2.2 Homotopy and Symbolic Computation

Symbolic computation is used, for instance, by Ericson and Yap [9], and by Kondo [12]. This method typically uses Gröbner bases, some kind of resultants, or other elimination techniques. It is exact but very slow for our purpose of an interactive modeler. Moreover, according to Lazard [13], it is definitely impossible to compute the Gröbner bases of an irreducible polynomial system of degree 2 in 10 unknowns and variables. In CAD we often have to handle hundreds of equations with higher degree. This kind of method seems to be more suited for automatic demonstrations of geometric theorems, like for example in [7], [28].

2.3 Homotopy and Decomposition Methods

Decomposition methods reduce constraint systems into simpler ones; solutions of irreducible subsystems are then merged. In 2D, typical irreducible systems are triangles, of

1. In this example, attraction basins for homotopy are the cells of the Voronoi's diagram of the solution points, but this is not true in general.
2. A semi-algebraic set is the projection of an algebraic set. An algebraic set is the solution set of a polynomial system.

which the relative location of vertices is defined by three constraints (e.g., three distances, or one distance and two angles, ...) or systems soluble by 'ruler and compass' like Appolonius's problem; here an explicit formula does the job.

The decomposition is performed either by the inference engine of some expert system like in Buthion's work [6] or Verroust, Schonek, and Roller's work [25] to cite a few, either by the matching mechanism provided by some programming language such as Prolog, as for example in Brüderlin [5], or by searches in graphs, like in Owen [19], Bouma, Fudos, Hoffmann, Cai, and Paige [4], Ait-Aoudia, Jegou, and Michelucci [1], and many others.

Decomposition methods are fast but, since the programmer cannot predict all possible cases, complex irreducible subsystems have to be solved by symbolic or numerical computations. As we will see in Section 3.1.5 it is possible to use decomposition to accelerate the resolution by homotopy.

3 A 2D Constraint-Based Modeler

3.1 Description

3.1.1 General Points

We have implemented in Lelisp an experimental 2D constraint-based modeler to compare behavior of resolution by Newton-Raphson's method and by homotopy. The user creates points, edges, circles in an interactive way, as with, say, MacDraw or XFig. Moreover, he can specify constraints. Predefined constraints are: distance between two points or between a point and a line, angle between two edges, tangency between a line and a circle or between two circles, incidence between a point and a line or a circle. He can also specify (with an algebraic formula) any algebraic equation. He can declare coordinates and radius circles to be 'movable,' or not; the modeler can only modify movable parameters. They are unknowns, and their numerical value (interactively provided by the user) are used as the initial guess by resolution methods. The user may call resolution by Newton-Raphson's method, or by homotopy. Our resolution methods need constraint systems to be well- or under-constrained. So the user can incrementally add constraints and solve the system. Homotopy with an under-constrained system is explained in Section 3.1.2. For the moment, the modeler considers over constrained systems as mistakes from the user, and gives him a warning.

When a Newton-Raphson's resolution fails or converges to an unwanted solution, the user recovers the previous state, and uses homotopy. Generally it works much better. However, sometimes, homotopy may also converge to an unwanted solution. But homotopy is really self-explanatory. During the resolution process, intermediate states are displayed (see Fig. 2), so the user can easily see what is wrong in his initial guess. For instance, when he sees some circle going to the wrong side of some line, he interrupts the resolution (or waits for its end), restores the previous state, moves the circle or the line, and then calls for the resolution again; it works.

In our very first experiments, homotopy nearly always worked at the first try. Then, failures became more and more usual. The reason is that, seeing homotopy work so well, we became more and more lazy and we gave initial guesses further and further away from the solution....

We have not met problems with bifurcation points, except when we do it on purpose, to test the software. Typically, these situations occur when there are several symmetrical solutions for a constraint system, and when the initial guess has the same symmetry. For instance, if a movable point P has two symmetrical solutions P_1 and P_2 relative to some line L. If the initial guess for P belongs to L, then the homotopy method cannot 'choose' between the two symmetrical paths, one leading to P_1 and the other to P_2. The initial guess is itself a bifurcation point. A slight perturbation of the initial guess is sufficient to break the symmetry and to remove the bifurcation point.

3.1.2 Under-Constrained Homotopy

For a 2D constraint-based modeler to be truly user-friendly and interactive, we also wanted to solve under-constrained systems, and not only well-constrained ones. But, if there are n unknowns (without t) and u missing equations, homotopy $H(t, X) = 0$ does not describe anymore a curve in $\mathbb{R}^{n+1}$, but a 'hypersurface' with dimension $1 + u$.

At the starting point $M_0 = (0, S)$, we project the 'upward' vector $V = (t = 1, x_1 = x_2 \ldots = x_n = 0)$ on the tangent space of this hypersurface to get T_0 (see Fig. 4); the predicted point is then $M_0 + \epsilon\, T_0 / |T_0|$ and a correction step gives M_1. At M_k, we project the previous tangent vector T_{k-1} on the tangent space of $H(M_k) = 0$ to get the tangent vector T_k and the predicted point is $M_k + \epsilon\, T_k / |T_k|$, etc.

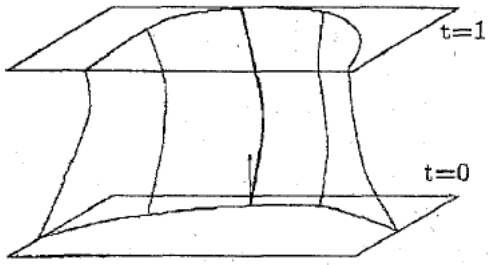

Fig. 4 An example of under-constrained system.

In other words, we try to follow the path that has an 'upward' tangent T_0 at the point $(0, S)$ and that is as straight as it can be while remaining in the homotopy surface. Up to errors (due to the fact that our ϵ is not infinitesimal), we thus follow a *geodesic curve* of the surface: in all points of such a curve, the principal normal to the curve coincides with the normal to the surface. Due to the accumulation of errors, the followed path progressively diverges from the 'true' geodesic curve; despite this limitation, the homotopy defined in this fashion has an intuitive behavior. For instance, suppose an unknown point P must belong to a given line L, but the initial guess P_0 for P does not. Then the user sees the point getting closer to the line by following the shortest path.

3.1.3 Solution at Infinity

Solutions at infinity (for instance $xy = 1$, $y = 0$ has solution $x = \infty$) pose a problem with the homotopy method, since paths to these solutions have infinite length.... We have not met this problem during our experiments; anyway, it is easily (and classically) solved by homogenization. Suppose

for instance unknowns are the (x, y) coordinates of some *point*. Just use homogenized coordinates (X, Y, H) for this point, i.e., $xH = X$, $yH = Y$, translate constraints $g_i(x, y) = 0$ into $G_i(X, Y, H) = H^{\deg(g_i)} g_i(X/H, Y/H) = 0$, add a new constraint like, say: $X^2 + Y^2 + H^2 = 1$, and solve. All solutions of the homogenized system are finite; in the previous example, the homogenized system is $XY = H^2$, $Y = 0$, $X^2 + Y^2 + H^2 = 1$, the solution is $X = \pm 1$, $Y = H = 0$. See [2], [17].

3.1.4 Inequalities

Inequalities are a convenient way to select a solution. Among others, for instance, there are two circles tangent to a given circle and passing through two given points; an inequality can be used to select the wanted circle. Our application assumes that the choice is implicitly performed by the user through his initial guess. Our constraints only lead to equations and not to inequalities.

However, providing explicit inequalities is perhaps useful for some applications. It is possible, at least theoretically, to translate inequalities into equations: $f(X) \geq 0 \Leftrightarrow f(X) - a^2 = 0$ and $f(X) > 0 \Leftrightarrow b^2 f(X) - 1 = 0$ where a and b are auxiliary real unknowns. In this case we obtain an under-constrained system that we already know how to solve by homotopy.

Though further studies are needed, our present tests show that, for imprecision reasons inherent to numerical methods, it is not possible to distinguish the cases $f(X) > 0$ and $f(X) \geq 0$. The best results are obtained with the inequality $f(X) \geq 0 \Leftrightarrow f(X) - a^2 = 0$ where we use

$$a_0 = \begin{cases} 0 & \text{when } f(X_0) \leq 0 \\ \sqrt{f(X_0)} & \text{when } f(X_0) > 0 \end{cases}$$

as the initial numerical value of the auxiliary unknown a since here the user has no way to interactively provide it.

3.1.5 Diagnosis and Decomposition

Our modeler provides some tools for diagnosis of constraint systems: recognition of under-, well-, or over-constrained unknowns by computing the Dulmage-Mendelsohn's decomposition of the bipartite graph associated to constraint systems [1], or diagnosis of rigidity like in [10] for instance. These tools are essential to help users 'debug' complex constraint systems. Moreover, they allow us to decompose huge constraint systems into smaller ones (by using the Konig-Hall's decomposition [1]), and so to speed up resolution. The homotopy method (like the Newton-Raphson's method, incidentally) is compatible with such decomposition methods.

3.2 Open Problems

3.2.1 Control

When a constrained system has no solution, it is easily seen. The homotopy enters in a loop (supposing a homogenization is used to handle solution at infinity, and so to avoid infinite paths). Though feasible [22], the automatic detection of loops is not implemented just now. The user has to interrupt the endless homotopy process (his work is not lost!), or wait for the automatic stop of the process after a fixed number of steps.

3.2.2 Speed

Our homotopy method is not as fast as it could be (though the constraints resolution always remains faster than their interactive specification), especially with more than 40 or 50 unknowns. When implementing, our first goal was to verify the relevance of the homotopy method, not speed... So the slowness of our implementation is not relevant. More relevant is the number of correction-prediction steps needed on average. Most often, about 20 steps are enough; 60 or 70 iterations may be needed, when the followed path is very close to another homotopy curve, i.e., at 'quasi-bifurcations points' (see Fig. 5). Thus, in practice and in average, homotopy will be 20 times slower than the secant method, used in each step.

3.2.3 Imprecision

Looking closely to attraction basins in Fig. 3, one can see some little errors on the frontiers. Ideally, frontiers would be straight lines (in this example). This problem is mainly due to imprecision. In some areas, this kind of problem is a severe drawback of homotopic methods, because the confusion between distinct (and very close) paths may lead to logical or topological inconsistencies; however, for our applications, and in an interactive use, this is not a serious problem.

3.2.4 Nonalgebraic Equations

Homotopy is only used with algebraic constraints for the moment, though it may work with transcendental ones.

4 CONCLUSION

We are convinced homotopy will soon become very popular in constraint-based geometric modelers. It is not very difficult to implement, and its behavior is much more intuitive, predictable, and self-explanatory than those of the Newton-Raphson's method. Finally our implementation shows that it is compatible with interactivity, and with decomposition methods.

APPENDIX A

A recent survey on homotopy theory is [2].

A.1 Homotopy Definition

Let $G(X) = 0$ be a system of n independant (say) polynomial equations, in n unknowns, that is $X = (x_1, x_2 \ldots, x_n)$ and $G = (g_1, g_2 \ldots, g_n)$. Suppose now a solution $S = (s_1, s_2 \ldots, s_n)$ of another system $F(X) = 0$ is known, with $F = (f_1, f_2 \ldots, f_n)$, and that F is, in a certain meaning, 'close' to G. In our application, S is nothing else but the vector of values (vertices coordinates and circles radius) defining the initial guess interactively provided by the user, and system $F(X)$ is defined by $F(X) = G(X) - G(S)$; by construction, S is a solution of $F(X) = 0$. F and G are then embedded in a *homotopy*:

$$H(t, X) = tG(X) + (1 - t)F(X)$$

such that $H(0, X) = F(X)$ and $H(1, X) = G(X)$. System H is a linear interpolation between F and G; some homotopies use nonlinear interpolation, but they are beyond the scope of this paper, see [2].

System H has $n + 1$ unknowns and n equations. If $P = (t_p, X_p)$ is such that $H(P) = 0$, and if P is a regular point of $H = 0$ (i.e., the jacobian $H'(P)$ has maximal rank) then, from the implicit function theorem, $H^{-1}(0)$ is locally parameterizable by t at P. In more geometric words, $H(t, X) = 0$ defines a curve (the *homotopy curve*) in $n + 1$ dimensional space, passing through P and parameterized by t. Such a point P is known: it is $P = (0, S)$.

The main idea of resolution by homotopy is to follow the homotopy curve (also called *homotopy path*), starting from $t = 0$, $X = S$. If the homotopy path passes through a point (t, X) with $t = 1$, then a solution to $H(1, X) = 0$, and thus to $G(X) = 0$, has been found. Methods for following homotopy curves are summarized below. Fig. 5 shows a sampling of 12^2 homotopy 3D paths corresponding to our example: $z^3 - 1 = 0, z \in \mathbb{C}$ of Fig. 3. The checkerboard is in the plane $t = 0$.

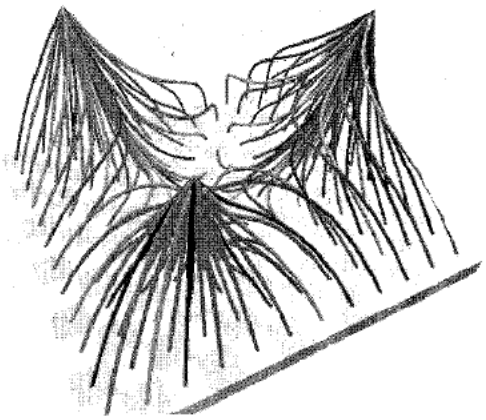

Fig. 5. $z^3 - 1 = 0$, $z \in \mathbb{C}$: Homotopy curves in 3D.

A.2 Topologic Considerations

If a homotopy curve only contains regular points, its topology is either that of a circle (i.e., the curve is a loop), or that of a line (i.e., it comes from infinity, and goes back to infinity). Of course, in each case, it may cross, or not, hyperplane with equation $t = 1$. See Fig. 6.

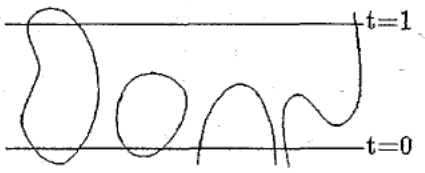

Fig. 6. Some examples of homotopy curves.

A homotopy curve may contain *singular points*, i.e., points where the jacobian has a nonmaximal rank. In the homotopy context, such points are called *bifurcation points*: two (or more) homotopy curves collide. The simplest and more frequent bifurcation points are *quadratic bifurcation points*, i.e., two curves in $\mathbb{R} \times \mathbb{C}^n$ meet in a point $Q = (t_0, X_0) \in \mathbb{R} \times \mathbb{C}^n$ where Q is a regular solution of $H(Q) = 0$ and $det(H'_X(Q)) = 0$. If ϕ is a vector tangent to one of the two curves at a quadratic bifurcation point, then $i\phi$ is the tangent vector to the second curve, where $i^2 = -1$ as usual. Moreover, the two tangent vectors have a vanishing t component. See [14] for a full characterization of quadratic bifurcation points.

Turning points are a special case of quadratic bifurcation points. They arise with real systems F and G; at a turning point, one of the two curves is *real*, and the other is *imaginary* or *complex*. Fig. 7 shows a typical example of turning points. Turning points arise even in very simple problems, so homotopy methods must take them into account. Moreover, after [14], they are the only bifurcation points that arise in real systems, in the generic case.

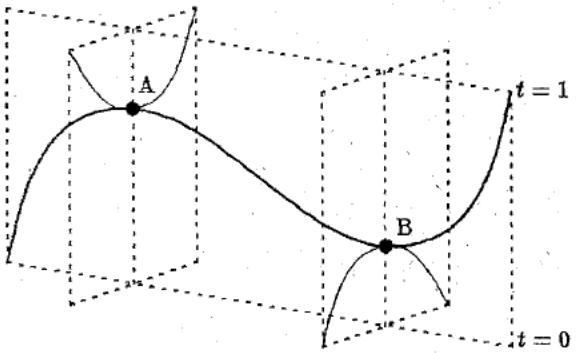

Fig. 7. A real homotopy curve, in thick line, with two turning points A and B: tangent vector has vanishing t component. In thin lines, corresponding complex branches (here, imaginary). Equations are (say):

$$F(X) = 3X^3 - X + \tfrac{1}{2}$$
$$G(X) = 3X^3 - X - \tfrac{1}{2}$$
$$H(t, X) = 3X^3 - X + \tfrac{1}{2} - t$$
$$A_t = \tfrac{13}{18}$$
$$A_X = -\tfrac{1}{3}$$
$$B_t = \tfrac{5}{16}$$
$$B_X = \tfrac{1}{3}.$$

A.3 Homotopy Method

In our case, a starting point for homotopy is known; we are only interested in the homotopic curve crossing this point. However, in a more general setting, people want to find all (maybe complex) solutions of a given system, and have no starting points. Thus, they typically proceed in two steps [2]. First they build a starting system of equations; it must be easily solved, and it must have at least as many solutions as the system to be solved. This number of solutions may be bounded, classically, by the product of degrees, after Bezout's theorem, or, more closely, with Newton's polytope and BKK bounds [26]. Secondly, paths are followed from starting points.

A.4 Methods for the Following Paths

A.4.1 Climbing Complex Homotopy

For constraint systems in CAD modeling, only real roots of system G are relevant, and the followed paths are curves in $\mathbb{R}^{n+1}$. In other areas, complex roots Z of system G are needed, and homotopy paths are curves in $\mathbb{R} \times \mathbb{C}^n$ (since, usually, t goes from 0 to 1, staying in $\mathbb{R}$). In this last case, it is possible to 'climb' along homotopy paths, starting with $(t = 0, Z = Z(0))$, and tracking $Z(t)$ as t monotonously increases from 0 to 1. By linearization of H at a known point (t, Z) on the curve, we get:

$$H(t + \Delta t, Z + \Delta Z) \approx H(t, Z) + H'_t \Delta t + H'_Z \Delta Z$$

We use the prediction: $\Delta Z = -(H'_Z)^{-1} H'_t \Delta t$ where Δt may possibly be scaled to control the step size $|(\Delta t, \Delta Z)|$. Then, leaving t constant (so $\Delta t = 0$), we correct several times Z by iterating: $\Delta Z := -(H'_Z)^{-1} H(t, Z)$, $Z := Z + \Delta Z$, until $|\Delta Z|$ is sufficiently close to 0.

This method faces a difficulty when a turning point, or a bifurcation point, is met. One possible solution (among others) is to use the perturbed homotopy:

$$H(t, Z) = t\gamma G(Z) + (1 - t) F(Z)$$

where $\gamma = \rho e^{i\theta}$ is a random complex number: it is proven that there is only a finite number of θ for which H has bifurcation points in $t \in [0, 1)$. So, with probability 1, perturbed homotopy removes turning and bifurcation points. Of course, this perturbation cannot remove possible singular solutions of $G = 0$, and corresponding bifurcation points when $H = G$, i.e., when $t = 1$.

A.4.2 Predictor-Corrector

To follow the homotopic path from a given point M_k, the so called *predictor-corrector* method first computes T_k, the tangent vector (or an approximation) to the curve in M_k, predicts that the point $P = M_k + \epsilon T_k / |T_k|$ (see Fig. 8) is close to the curve, and corrects P by some variant of the Newton-Raphson's iteration (for instance using the secant method, or using the Moore-Penrose's pseudo-inverse) or some gradient method to obtain M_{k+1}, the point on the curve closest to P, and so on restarting from M_{k+1}. Material about numerical linear algebra can be found in [2]. We do not detail this method further, very similar to marching methods used in CAD for following intersection curves between surfaces in 3D geometric modeling [20], [11].

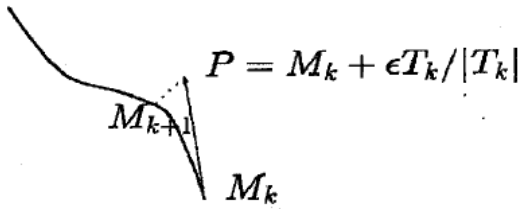

Fig. 8. The principle of the predictor-corrector method.

A practical difficulty (and a difficult theoretic problem of numerical analysis) is the choice of a good ϵ. If it is too big, the correction-step may fail; if it is too small, path following is slowed down (moreover, some numerical problems due to imprecision sometimes appear). Research has been done for safely and automatically choosing ϵ [2], [29].

However, we have found the following heuristic good enough for our limited needs and easy to implement. At each correction-prediction step, we update the pseudo-inverse of H' and choose $\epsilon = 0.05$; if the correction step does not converge fast enough (i.e., in at most four iterations), we divide ϵ by 2 and try again. Let d the distance between the starting point M_k and the predicted point P_0; let P_1, P_2, P_3, and P_4 the successive corrections of the point P_0. As soon as distance $|P_0P_i|$ is greater than d, the system is said to diverge. It convergences when $|P_iP_{i-1}|$ is less than 0.02 d, and the angle between tangent vectors at M_k and at P_i is less than 10 degrees.

Our experimental constraint-based 2D modeler uses the predictor-corrector method, mainly because we already need a Newton-Raphson's iteration to compare the behaviors of the latter and of the homotopic method. However, this method requires the computation and (some kind of) inversion of the jacobian[3] H'. The next method only requires to evaluate H at some points.

A.4.3 Piecewise Linear Approximation

Another method for following homotopy paths is known as *piecewise linear approximation*. In Computer Graphics, a variant of this method has been used for tracing curves in $\mathbb{R}^n$ by (at least) Dobkin, Levy, Thurston, and Wilks [8].

In $\mathbb{R}^2$, this method is very simple (see Fig. 9). To trace the homotopy curve $H(t, x) = 0$, $\mathbb{R}^2$ is triangulated by, say, equilateral triangles with side ϵ. Assume a first triangle ABC traversed by H is known. H enters by the edge AB and leaves by AC because $H(A) > 0$, $H(B) < 0$, $H(C) < 0$, or the contrary. So the next triangle cut by H is ACB', where $B' = A - B + C$ is the point symmetrical to B relatively to the line AC. In ABC, H is approximated by the unique linear map $L(t, x)$ from $\mathbb{R}^2$ to $\mathbb{R}$ such that $L(A) = H(A)$, $L(B) = H(B)$ and $L(C) = H(C)$, and the curve in ABC is approximated by the edge $\{L(x, y) = 0\} \cap ABC$.

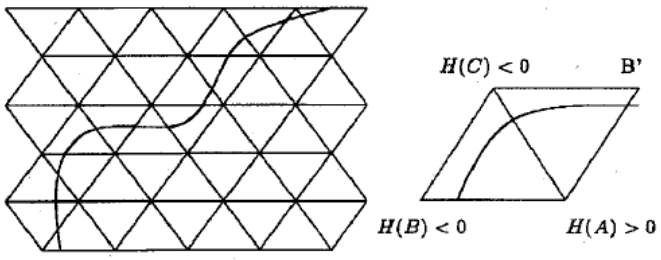

Fig. 9. An example of piecewise linear approximation.

In $\mathbb{R}^{n+1}$, the space is triangulated by 'hyper tetrahedrons,' i.e., simplices. Assume we know a starting simplex T traversed by the homotopy path; suppose also that values of the n functions defining the homotopy H are available at the vertices of T. They define a unique linear map L from $\mathbb{R}^{n+1}$ to $\mathbb{R}^n$; in T, approximate H by L and the homotopy curve by the edge $\{L(t, x_1, ..., x_n) = 0\} \cap T$. Deduce the hyperfacet of T by which this edge leaves T, and follow it in the neighboring simplex.

A.4.4 Attraction Basins

An attraction basin for homotopy is a maximal connected points set $S \in \mathbb{R}^n$ leading to the same solution $(t = 1, X_1)$, or to no solution, when the homotopic method is applied with the starting point: $(t = 0, S)$. Two neighboring basins are separated by points leading to bifurcation points. The latter are solutions of the algebraic system: $H(t, X) = det(H'_X(t, X)) = 0$ in $n + 1$ equations and unknowns, and constitute by definition an algebraic set (assuming G to be algebraic). The projection on the hyperplane having equation: $t = 0$ then gives a semi-algebraic set. The reader can easily verify that, with the example: $z^3 - 1 = 0$, $z = x + iy$, frontiers between basins are (part of) lines: $y = 0$, $y = \pm x\sqrt{3}$.

3. The jacobian H' is symbolically computed for each constraint system.

REFERENCES

[1] S. Ait-Aoudia, R. Jegou, and D. Michelucci, "Reduction of Constraint Systems," *Compugraphics,* pp. 83-92, Alvor, Portugal, 1993. Also available at http:/ /www.emse.fr/-micheluc/.

[2] E.L. Allgower and K. Georg, "Continuation and Path Following," *Acta Numerica,* pp.'l-64,1993.

[3] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige, "A geometric Constraint Solver," Technical Report CSD-TR-93-054, Dept. of Computer Science, Purdue Univ., Aug. 1993.

[4] Borning, "The Programming Language Aspects of Thinglab, a Constraint Oriented Simulation Laboratory," ACM *Trans, Programming Languages and Systems,* vol. *3,* pp. 353-387, Oct. 1981.

[5] B. Bruderlin, "Constructing Three-Dimensional Geometric Objects Defined by Constraints," *Interactive 3D Graphics,* pp. 111-129, Oct. 1986.

[6] M. Buthion, "Un Programme Qui Résout Formellement Des Problèmes de Constructions Géométriques," *RAIRO Infomzatique,* vol. 3, no. 4, pp. 353-387, Oct. 1979.

[7] S.C. Chou, W.F. Schelter, and J.G. Yang, "Characteristic Sets and Grobner Bases in Geometry Theorem Proving," *Workshop Computer Aided Geometric Reasoning,* pp. 29-56, INRIA, France, June 1987.

[8] D.P. Dobkin, S.V.F. Levy, W.P. Thurston, and A. Wills, "Contour Tracing by Piecewise Linear Approximations," *ACM Trans. Graphics,* vol. 9, no. 4, pp. 389423, Oct. 1990.

[9] L.W: Ericson and C.K. Yap, "The Design of Linetool a Geometric Editor," Symp. *Computational Geometry,* pp. 83-92, 1988.

[10] B. Hendrickson, "Conditions for Unique Realizations," *SIAM 1.Computing,* vol. 21, no. 1, pp. 65-84, Feb. 1992.

[11] C.M. Hoffmann, "A Dimensionality Paradigm for surface interrogations," *Computer Aided Geometric Design,* vol. *7,* pp. 517-532, 1990.

[12] K. Kondo; "Algebraic method for manipulation of dimensional relationships in geometric models," *Computer Aided Design,* vol. 24, no. 3, pp. 141-147, Mar. 1992.

[13] D. Lazard, "Systems of Algebraic Equations: Algorithms and Complexity," Technical Report LITP 92.20, LITP, Université Paris VI, VII, CNRS, Mar. 1992.

[14] T.Y Li and Xiaoshen Wang, "Solving Real Polynomial Systems with Real Homotopies," *Math. of Computation,* vol. 60, no. 202, pp.

[15] R. Light, V. Lin, and D.C. Gossard, "Variational Geometry in CAD," *Computer Graphics,* vol. 15, no. 3, pp. 171-177, Aug. 1981.

[16] A.P. Morgan, *Solving Polynomial Systems Using Continuation for Scientific and Engineering Problems.* Englewood Cliffs, N.J.: Prentice-Hall, 1983.

[17] A.P. Morgan, "A Transformation to Avoid Solutions at Infinity for Polynomial Systems," *Applied Math. and Computation,* vol. 18.

[18] G. Nelson, "Juno, a Constraint-Based Graphic System," *Roc.* ACM *Siggraph Conf.,* 1985.

[19] J.C. Owen, "Algebraic Solution for Geometry from Dimensional Constraints," *Proc. Symp. on Solid Modeling Foundations and CAD/CAM Applications,* pp. 397407,1991.

[20] N.M. Patrikalakis, "Surface-to-Surface Intersections," *IEEE* Com*puter Graphics and Applications,* vol. 13, no. 1, pp. 89-95, Jan. 1992.

[21] H.O. Peitgen and P.H. Richter, *The Beauty of Fractals, Images of Complex Dynamical Systems.* Springer-Verlag, 1986.

[22] P. Schramm, "Intersection Problems of Parametric Surfaces in cagd," *Computing,* vol. 53, pp. 355-364'1994.

[23] I.E. Sutherland, "Sketchpad, a man machine graphical communication system," *AFIPS, Spring Joint Computer Conf.,* pp. 329-346, Detroit, Mich., May 1963.

[24] Verroust, "Etudes de problèmes liés à la définition, la visualisation et l'animation d'objets complexes en informatique graphique," PhD thesis, Univ. de Paris Sud, Centre d'Orsay, 1990.

[25] Verroust, F. Schonek, and D. Roller, "Rule Oriented Method for Parametrized Computer Aided Design," *Computer Aided Design* ,vol. 24, no. 3, pp. 531-540, Oct. 1992.

[26] J. Verschelde, P. Verlinden, and **K.** Cools, "Homotopies Exploiting Newton Polytopes for Solving Sparse Polynomial Systems," SIAM *J. Numerical Analysis,* 1993.

[27] C.W. Wampler, A.P. Morgan, and A.J. Sommese, "Numerical Continuation Methods for Solving Polynomial Systems Arising in Kinematics.,"ASME J. *on Design,* vol. 112, pp. 59-68,1990.

[28] F. Wmkler, "Algorithms in Polynomial Ideal Theory and Geome**try**", *Second Austrian-Hungarzan lnformatzcs Conf* , Retzhof Austria, Sept. 1988.

[29] J.C. Yakoubsohn, "An Universal Constant for the Convergence to the Newton Method and Application to the Classical Hornotopy Method," *Numerical Algorithms,* 1995.