Geometric Algorithms Based on Computer Algebra

Franz Winkler *
RISC-Linz, Johannes Kepler Universität Linz
Altenbergerstrasse 69, A-4040 Linz, Austria
Franz.Winkler@risc.uni-linz.ac.at

Keywords: computer algebra, symbolic computation, computer aided geometric design, robotics, rational parametrization of curves

Abstract

Many problems in robotics and geometric design are really problems about the construction, analysis, and representation of curves and surfaces. If these curves and surfaces are algebraic, i.e. describable by polynomial equations, then they can be treated by standard algorithms and methods in computer algebra. This means that we can get reliable symbolic information about these problems.

1 Introduction

Many problems in robotics and geometric design are really problems about the construction, analysis, and representation of curves and surfaces. If these curves and surfaces are algebraic, i.e. describable by polynomial equations, then they can be treated by standard algorithms and methods in computer algebra [11]. This means that we can get reliable symbolic information about these problems.

Computer algebra is concerned with the design, analysis, implementation, and application of algebraic algorithms. Computer algebra is the symbolic version of scientific computation, i.e. its destinguishing features are

- a great variety of algebraic structures, such as various number domains, multivariate polynomials, finitely represented groups, rational and transcendental expressions among others,
- exact, i.e. non-approximative, computation on these algebraic structures,
- generating symbolic expressions and formulas as output.

In particular, computer algebra contains a great wealth of algorithms and methods for dealing with multivariate polynomials and corresponding systems of polynomial equations. Typically, when we solve a system of algebraic equations like

$$\begin{aligned} x^4 + 2x^2y^2 + 3x^2y + y^4 - y^3 &= 0, \\ x^2 + y^2 - 1 &= 0 \end{aligned}$$

by computer algebra methods, we are interested in an exact representation $(\sqrt{3}/2, -1/2)$ of the solution instead of an approximative one such as $(0.86602\ldots, -0.5)$.

In computer algebra we have a variety of concepts, methods, and algorithms for solving systems of such algebraic equations. Some of them concern the computation of greatest common divisors of polynomials, factorization of polynomials, resultants of systems of polynomial equations, characteristic sets for algebraic equations, and Gröbner bases for systems of algebraic equations, just to name the most important ones. So, for instance, we can triangularize the above system of algebraic equations by a Gröbner basis computation, yielding the equivalent system, in which the variables are introduced one at a time:

$$\begin{aligned} y - 4x^4 + 5x^2 - 1 &= 0, \\ 16x^6 - 24x^4 + 9x^2 &= 0. \end{aligned}$$

Various program systems, such as Maple, Mathematica, Reduce, just to name a few, are available for carrying out these exact symbolic algorithms. The program system CASA (see [4]), developed by the author's research group, is capable of performing all the computations in this paper. We will demonstrate in some selected examples, how methods and algorithms from computer algebra can be successfully applied to problems arising in the fields of robotics and geometric design.

2 Parametrization of Curves and Surfaces

Algebraic curves and surfaces are decribed by polynomial equations, i.e. the points on the corresponding curve or surface are the solutions of these algebraic equations. For instance, the algebraic curve $\mathcal{C}_1$ depicted in Fig. 1 is defined as the roots of the equation

$$(x^2 + 4y + y^2)^2 - 16(x^2 + y^2) = 0.$$

Such defining equations are convenient for checking whether a given point actually lies on the curve. But for

*Supported by the European project CHRX-CT94-0439 and by the Austrian FWF under project SFB F013/F1304.

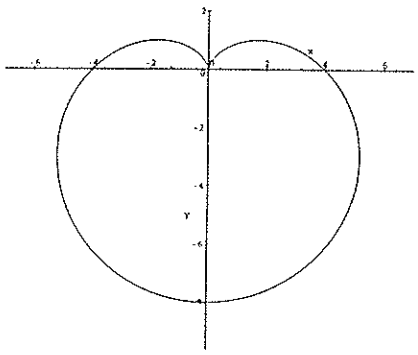

Figure 1: Algebraic curve $\mathcal{C}_1$.

other applications, such as creating points, computing intersections, determining offsets, etc., it is much more convenient to work with parametrizations of curves and surfaces, in particular rational parametrizations. A rational parametrization of an algebraic curve is nothing else than a pair of rational functions (i.e. polynomial numerator and denominator) $(x(t), y(t))$, such that whenever we substitute for the parameter t, we get a point on the curve, and every point on the curve can be produced in this way (with finitely many exceptions).

It is well known, that not every algebraic curve can be rationally parametrized. In fact, this is possible only for irreducible curves of genus 0, i.e. these curves have to have enough singular points. Nevertheless, rational curves appear in numerous applications (e.g. rational Bézier curves), and often it is important to be able to test whether an algebraic curve is rational, and if so, compute a rational parametrization.

Let us consider $\mathcal{C}_1$ for an example of the parametrization process descibed in [8] and [9]. Some of the singularities of $\mathcal{C}_1$ are "at infinity", so they are visible only in the projective plane. In fact, $\mathcal{C}_1$ has 3 double points, namely

$$O = (0:0:1), \quad P_{1,2} = (1:\pm i:0).$$

We consider the system of all quadratic curves through these 3 points (in analogy to the system of lines in the previous example). The system has 2 free parameters, and we reduce the number of parameters by requiring also that every curve in the system should pass through the point $Q = (0:-8:1)$ on $\mathcal{C}_1$. Now the defining equation of the system of quadratic curves is

$$h(x, y, t) = tx^2 + ty^2 + x + 8ty$$

for an undetermined coefficient t. Intersecting $\mathcal{C}_1$ with an arbitrary conic of this system, we get (by Bézout's Theorem) 2-fold intersections at $O, P_{1,2}$, a 1-fold intersection at Q, and exactly one more intersection point depending on t. This point has coordinates depending rationally on the parameter t, leading to the parametrization

$$x(t) = \frac{-1024t^3}{256t^4 + 32t^2 + 1}, \quad y(t) = \frac{-2948t^4 + 128t^2}{256t^4 + 32t^2 + 1}$$

of the curve $\mathcal{C}_1$.

In general, this process can introduce algebraic numbers in the coefficients of the parametrization, which could in fact be non-real algebraic numbers. This raises the question of whether a "real" algebraic curve (i.e. a curve having infinitely many points in the real plane) having any rational parametrization (with possibly complex coefficients), can in fact be parametrized with coefficients in the field of real numbers. Fortunately, the answer to this question is "yes" (see [9] and [10]).

The rational parametrization of algebraic surfaces poses additional problems. But also for surfaces there are parametrization algorithms available, see [7].

3 Geometric Design: Offset Curves

In computer aided geometric design we frequently encounter the problem of having to offset a curve or surface. This means to determine, for a given generating curve $\mathcal{C}$ and a given distance d, a parallel or offset curve $\mathcal{D}$ at distance d, having the following property: for every point P on $\mathcal{C}$ there is a point Q on the offset curve $\mathcal{D}$ such that the distance between P and Q is exactly d, and the line $\overline{PQ}$ is perpendicular to $\mathcal{C}$ at P and vice versa.

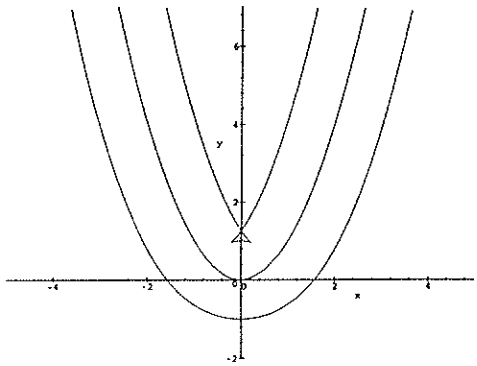

Figure 2: Offset to the parabola.

Such offset curves arise for instance, when we want to cut out an object whose boundary is a curve $\mathcal{C}$, and we want to use a circular cutter of diameter d. Then the center of the cutter has to traverse exactly the offset curve to $\mathcal{C}$ at distance d. Other applications in computer aided geometric design are in robot path planning, geometric optics, and the formulation of geometric procedures such as growing and shrinking, blending, filleting, etc. For symbolic algorithms on offset curves we refer to [3], [1].

Let us demonstrate some of the problems and techniques for the parabola (compare Fig. 2). The parabola is given parametrically by $x(t) = t, y(t) = t^2$. In general, if the generating curve $\mathcal{C}$ is given parametrically by $(f_1(t), f_2(t))$, then the offset curve at distance d is the solution of the system

$$\begin{aligned} (x - f_1(t))^2 + (y - f_2(t))^2 - d^2 &= 0, && \text{distance } d \\ \tfrac{\partial f_1}{\partial t}(t)(x - f_1(t)) + \tfrac{\partial f_2}{\partial t}(t)(y - f_2(t)) &= 0. && \text{perpendicular} \end{aligned}$$

For the parabola this means that we have to eliminate the variable t from the equations

$$x^2 - 2tx + t^2 + y^2 - 2t^2y + t^4 - 1 = 0,$$
$$x - t + 2ty - 2t^3 = 0.$$

Computing the resultant of these two polynomials w.r.t. the variable t (any computer algebra system can do this for us), we get the defining equation of the offset curve at distance 1 to the parabola:

$$\begin{aligned} 16y^4 - 32x^2y^3 - 40y^3 + 16x^4y^2 + 9y^2 - 40x^4y \\ +6x^2y + 40y + 16x^6 - 47x^4 + 28x^2 - 25 \quad = \quad 0. \end{aligned}$$

This equation describes the two-sided offset curve, and in fact this two-sided offset to the parabola is an irreducible curve, i.e. we cannot decompose it into a left- and right-sided offset. This is, of course, not always so. For instance the offset to a circle clearly decomposes into 2 circles.

Since the parabola is a rational curve, it is natural to ask whether the offset to the parabola is also rational. Given a rational curve $\mathcal{C}$ defined by $x = f_1(t), y = f_2(t)$, we get a parametric representation $\mathcal{P}(t)$ of the offset curve at distance d by setting

$$\mathcal{P}(t) = (x(t), y(t)) \pm \frac{d}{m(t)}\mathcal{N}(t),$$

where $\mathcal{N}(t)$ is the normal vector $(-f_2'(t), f_1'(t))$ and $m(t)$ is the norm of the normal vector, i.e. $m(t) = \sqrt{f_1'^2(t) + f_2'^2(t)}$. (Observe that this definition does not work for singular points, so we have to consider the Zariski closure of $\mathcal{P}(t)$.) If the norm $m(t)$ is not rational, then the parametrization $\mathcal{P}(t)$ will not be a rational parametrization of the offset. This is indeed the case for the parabola. But still the offset to the parabola is a rational curve, and we can get a rational parametrization either by replacing the parameter t in $\mathcal{P}(t)$ by $(s^2-16)/16s$, or by directly applying the parametrization algorithm decribed above. In any case, we get a rational parametrization of the offset as

$$\begin{aligned} x(t) &= \frac{(t^2-16t+16)(t^2-16)}{16(t^3+16t)}, \\ y(t) &= \frac{t^6-16t^4+2048t^3-256t^2+4096}{256(t^4+16t^2)}. \end{aligned}$$

We have a complete overview of the situations that can occur w.r.t. the rationality of offset curves:

generator curve	offset curve
rational	2 rational components 1 rational component 1 non-rational compent
non-rational	1 rational and 1 non-rat. comp. 2 non-rational components 1 non-rational component

Of course, offsets are also very important for surfaces. However, the results for surfaces are by no means so complete as for curves. Special cases, such as canal surfaces, can be handled successfully (see [6]).

4 Robotics: Inverse Kinematics

We consider robots with two types of joints, namely prismatic and revolute joints. The kinematics of such robots can be described by multivariate polynomial equations, after having represented angles α by their sines and cosines and having added the equation $\sin^2(\alpha) + \cos^2(\alpha) = 1$ to the set of polynomial equations. This description can be found, for instance, in [5].

There are basically two kinds of kinematics problems related to this situation: forward kinematics and inverse kinematics. The former determines the position of the end-effector for given lengths of prismatic joints and angles of revolute joints, whereas the latter determines possible lengths and angles from a predetermined goal position of the end-effector. Whereas a forward kinematics problem always has exactly one solution, an inverse kinematics problem could have no, exactly one, or several (possibly infinitely many) solutions.

As an example, let us consider a robot consisting of two bars of fixed lengths joined by a revolute joit. The first bar is fixed to the origin of the coordinate system in (x, y, z)-space, always stays perpendicular to the (x, y)-plane, but can rotate relative to the (x, y)-plane describing an angle δ_1. The second bar may assume an angle δ_2 relative to the first one. The end-effector is located at the other end of the second bar. So this robot has two "degrees of freedom", i.e. we can – within suitable bounds – set two of the coordinates of the end-effector, and determine the third coordinate and the angles producing this position. See Fig. 3.

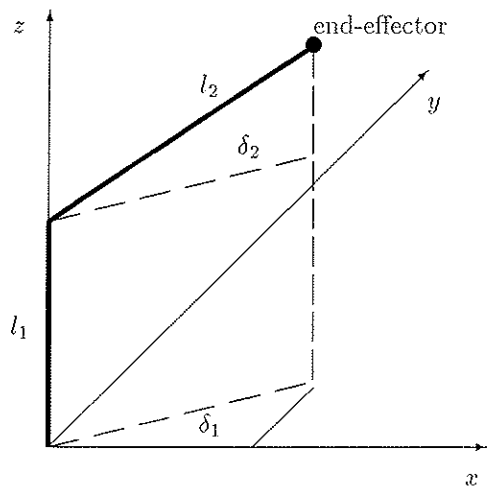

Figure 3: Robot arm with 2 degrees of freedom.

A variant of this robot is studied in [2]. The inverse kinematics problem for such a robot consists in fixing a point (actually only 2 coordinates of the desired point can be fixed, since the robot has only 2 degrees of freedom) in space, and subsequently determining the possible adjustments of the angles for achieving this position.

For transforming the original problem into a system of algebraic equations, we introduce the following variables and sines and cosines of angles:

l_1, l_2	lengths of the two robot arms
px, py, pz	x-, y-, and z-coordinates of the position of the end-effector
δ_1, δ_2	angles describing the rotations of the revolute joints
s_1, s_2, c_1, c_2	sines and cosines of δ_1, δ_2, respectively

Now the inverse kinematics problem for given values of the geometrical variables (i.e. l_1, l_2) and position variables (i.e. px, pz) is solvable with joint variables (i.e. s_1, c_1, s_2, c_2) and third coordinate py if and only if these variables satisfy the following system of algebraic equations:

$$\begin{aligned} l_2 \cdot c_1 \cdot c_2 - px &= 0, \\ l_2 \cdot s_1 \cdot c_2 - py &= 0, \\ l_2 \cdot s_2 + l_1 - pz &= 0, \\ c_1^2 + s_1^2 - 1 &= 0, \\ c_2^2 + s_2^2 - 1 &= 0. \end{aligned}$$

For solving this system of algebraic equations, we compute a Gröbner basis (w.r.t. a lexicographical term ordering) for this system. This, in effect, changes the original system into a new equivalent one, which is "triangular", i.e. the variables are introduced one after the other. So we can apply a recursive process for solving the new system. Technically, we compute a Gröbner basis for this system in the polynomial ring

$$\mathbf{Q}(l_1, l_2, px, pz)[c_1, c_2, s_1, s_2, py],$$

getting

$$\begin{aligned} (l_2^2 - l_1^2 + 2l_1pz - pz^2 - px^2) \cdot c_1 - px \cdot s_1 \cdot py &= 0, \\ (l_2^3 - l_2l_1^2 + 2l_2l_1pz - l_2pz^2 - l_2px^2) \cdot c_2 & \\ +(-l_2^2 + l_1^2 - 2l_1pz + pz^2) \cdot s_1 \cdot py &= 0, \\ (l_2^2 - l_1^2 + 2l_1pz - pz^2) \cdot s_1^2 - l_2^2 + l_1^2 - 2l_1pz & \\ +pz^2 + px^2 &= 0, \\ l_2s_2 + l_1 - pz &= 0, \\ -l_2^2 + l_1^2 - 2l_1pz + pz^2 + px^2 + py^2 &= 0. \end{aligned}$$

From this Gröbner basis for the equations of the robot — which is typically computed "off-line" — we can easily determine particular solutions of the inverse kinematics problem: from the last equation we get py, from the 4th equation we get s_2, and so on. Let us assume that the lengths of the two arms are $l_1 = 30, l_2 = 45$. Since our robot has only 2 degrees of freedom, we can predetermine for instance the x- and the z-coordinates of the position of the end-effector. Let us set

$px = \frac{45 \cdot \sqrt{6}}{4} = 27.5567\ldots$	x-coordinate of end-effector
$pz = \frac{45 \cdot \sqrt{2}}{2} + 30 = 61.8198\ldots$	z-coordinate of end-effector

Then there are several solutions, one of them being

$$py = \frac{45\sqrt{2}}{4}, s_2 = \frac{\sqrt{2}}{2}, s_1 = \frac{1}{2}, c_2 = \frac{\sqrt{2}}{2}, c_1 = \frac{\sqrt{3}}{2},$$

i.e. the angles have to be set to

$$\delta_1 = 30°, \delta_2 = 45°.$$

References

[1] E. Arrondo, J. Sendra, J.R. Sendra, "Genus Formula for Generalized Offset Curves", to appear in *J. Pure and Applied Algebra*

[2] B. Buchberger, "Applications of Gröbner Bases in Non-Linear Computational Geometry", in J.R.Rice (ed.), *Mathematical Aspects of Scientific Software*, No.14 in The IMA Volumes in Mathematics and its Applications, 59–87, Springer-Verlag, 1988.

[3] R.T. Farouki, C.A. Neff, "Algebraic Properties of Plane Offset Curves", *Computer Aided Geometric Design* 7, 101–127, 1990.

[4] M. Mňuk, F. Winkler, "CASA — A System for Computer Aided Constructive Algebraic Geometry", in J.Calmet and C.Limongelle (eds.), *Design and Implementation of Symbolic Computation Systems*, LNCS 1128, 297–307, Springer-Verlag, 1996.

[5] R.P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*, The MIT Press, 1981.

[6] M. Peternell, H. Pottmann, "Computing Rational Parametrizations of Canal Surfaces", *J. Symbolic Computation* 23/2&3, 255–266, 1997.

[7] J. Schicho, "Rational Parametrization of Surfaces", to appear in *J. Symbolic Computation.*

[8] J.R. Sendra, F. Winkler, "Symbolic Parametrization of Curves", *J. Symbolic Computation* 12/6, 607–631, 1991.

[9] J.R. Sendra, F. Winkler, "Parametrization of Algebraic Curves over Optimal Field Extensions", *J. Symbolic Computation*, 23/2&3, 191–207, 1997.

[10] J.R. Sendra, F. Winkler, "Real Parametrization of Algebraic Curves", in J.Calmet (ed.), *Proc. 4th Internat. Conf. on Artif. Intell. and Symbolic Computation*, LNAI, Springer-Verlag, 1998.

[11] F. Winkler, *Polynomial Algorithms in Computer Algebra*, Springer-Verlag, 1996.