Numerical Algebraic Geometry and Algebraic Kinematics

Charles W. Wampler* Andrew J. Sommese†

January 14, 2011

Abstract

In this article, the basic constructs of algebraic kinematics (links, joints, and mechanism spaces) are introduced. This provides a common schema for many kinds of problems that are of interest in kinematic studies. Once the problems are cast in this algebraic framework, they can be attacked by tools from algebraic geometry. In particular, we review the techniques of numerical algebraic geometry, which are primarily based on homotopy methods. We include a review of the main developments of recent years and outline some of the frontiers where further research is occurring. While numerical algebraic geometry applies broadly to any system of polynomial equations, algebraic kinematics provides a body of interesting examples for testing algorithms and for inspiring new avenues of work.

Contents

1 Introduction **4**

2 Notation **5**

I Fundamentals of Algebraic Kinematics **6**

3 Some Motivating Examples **6**

3.1 Serial-Link Robots 6

3.1.1 Planar 3R robot 6

3.1.2 Spatial 6R robot 11

3.2 Four-Bar Linkages 13

3.3 Platform Robots 18

*General Motors Research and Development, Mail Code 480-106-359, 30500 Mound Road, Warren, MI 48090-9055, U.S.A. *Email:* Charles.W.Wampler@gm.com *URL:* www.nd.edu/˜cwample1. This material is based upon work supported by the National Science Foundation under Grant DMS-0712910 and by General Motors Research and Development.

†Department of Mathematics, University of Notre Dame, Notre Dame, IN 46556-4618, U.S.A. *Email:* sommese@nd.edu *URL:* www.nd.edu/˜sommese. This material is based upon work supported by the National Science Foundation under Grant DMS-0712910 and the Duncan Chair of the University of Notre Dame.

3.3.1 3-RPR Planar Robot 18
3.3.2 6-SPS "Stewart-Gough" Platform 20

4 Algebraic Kinematics **23**
4.1 Rigid-Body Motion Spaces 23
4.2 Algebraic Joints 24
4.3 Mechanism Types, Families, and Spaces 27
4.4 Kinematic Problems in a Nutshell 28
4.4.1 Analysis 30
4.4.2 Synthesis 33
4.4.3 Exceptional Mechanisms 35
4.5 Fitting into the Nutshell 35
4.5.1 Planar 3R Robots 35
4.5.2 Spatial 6R robots 37
4.5.3 Four-Bar Linkages 38
4.5.4 Planar 3-RPR Platforms 38
4.5.5 Spatial 6-SPS Platforms 38

5 History of Numerical Algebraic Geometry and Kinematics **38**

II Numerical Algebraic Geometry **41**

6 Finding Isolated Roots **46**
6.1 Homotopy 46
6.2 Path Tracking 48
6.3 Multiplicities and Deflation 49
6.4 Endgames 51
6.5 Adaptive Precision 53

7 Computing Positive-Dimensional Sets **54**
7.1 Randomization 56
7.2 Slicing 57
7.3 The Monodromy Membership Test 58
7.4 Deflation Revisited 59
7.5 Numerical Irreducible Decomposition 60

8 Software **63**

III Advanced Topics **64**

9 Nonreduced Components **64**
9.1 Macaulay Matrix . . . 64
9.2 Local Dimension . . . 65

10 Optimal Solving **67**
10.1 Coefficient Parameter Homotopies . . . 68
10.2 Equation-by-Equation Methods . . . 70

IV Frontiers **73**

11 Real Sets **73**

12 Exceptional Sets **75**

V Conclusions **77**

VI Appendices **78**

A Study coordinates **78**

1 Introduction

While systems of polynomial equations arise in fields as diverse as mathematical finance, astrophysics, quantum mechanics, and biology, perhaps no other application field is as intensively focused on such equations as rigid-body kinematics. This article reviews recent progress in *numerical algebraic geometry*, a set of methods for finding the solutions of systems of polynomial equations that relies primarily on homotopy methods, also known as polynomial continuation. Topics from kinematics are used to illustrate the applicability of the methods. High among these topics are questions concerning robot motion.

Kinematics can be defined as the study of the geometry of motion, which is the scaffolding on which dynamics builds differential equations relating forces, inertia, and acceleration. While not all problems of interest in kinematics are algebraic, a large majority are. For instance, problems concerning mappings between robot joints and their hand locations fall into the algebraic domain. As our interest here is in numerical algebraic geometry, we naturally restrict our attention to the subset of kinematics questions that can be modeled as polynomial systems, a subfield that we call *algebraic kinematics*. We give a high-level description of kinematics that encompasses problems arising in robotics and in traditional mechanism design and analysis. In fact, there is no clear line between these, as robots are simply mechanisms with higher dimensional input and output spaces as compared to traditional mechanisms. We discuss which types of mechanisms fall within the domain of algebraic kinematics.

While algebraic geometry and kinematics are venerable topics, numerical algebraic geometry is a modern invention. The term was coined in 1996 [89], building on methods of numerical continuation developed in the late 1980s and early 1990s. We will review the basics of numerical algebraic geometry only briefly; the reader may consult [99] for details. After summarizing the developments up to 2005, the article will concentrate on the progress of the last five years.

Before delving into numerical algebraic geometry, we will discuss several motivational examples from kinematics. These should make clear that kinematics can benefit by drawing on algebraic geometry, while a short historical section recalls that developments in algebraic geometry in general, and numerical algebraic geometry in particular, have benefitted from the motivation provided by kinematics. Numerical algebraic geometry applied to algebraic kinematics could be called "numerical algebraic kinematics," which term therefore captures much of the work described in this paper. With a foundational understanding of kinematics in place, we then move on to discuss numerical algebraic geometry in earnest, occasionally returning to kinematics for examples.

Our presentation of numerical algebraic geometry is divided into several parts. The first part reviews the basics of solving a system of polynomial equations. This begins with a review of the basic techniques for finding all isolated solutions and then extends these techniques to find the numerical irreducible decomposition. This describes the entire solution set of a system, both isolated points and positive dimensional components, factoring the components into their irreducible pieces. The second part describes more advanced methods for dealing effectively with sets that appear with higher multiplicity and techniques that reduce computational work compared to the basic methods. Finally, we turn our attention to the frontiers of the area, where current research is developing methods for treating the real points in a complex algebraic set and for finding exceptional examples in a parameterized family of problems.

Like any computational pursuit, the successful application of numerical algebraic requires a robust implementation in software. In § 8, we review some of the packages that are available, including the Bertini package [13] in which much of our own efforts have been invested.

2 Notation

This paper uses the following notations.

- $i = \sqrt{-1}$, the imaginary unit.
- a^* for $a \in \mathbb{C}$ is the complex conjugate of a.
- $\overline{A}$ for set $A \subset \mathbb{C}^N$ is the closure of A in the complex topology.
- For polynomial system $F = \{f_1, \ldots, f_n\}$, $F : \mathbb{C}^N \to \mathbb{C}^n$, and $y \in \mathbb{C}^n$,

  $$F^{-1}(y) = \{x \in \mathbb{C}^N \mid F(x) = y\}.$$

  We also write

  $$\mathcal{V}(F) = \mathcal{V}(f_1, \ldots, f_n) := F^{-1}(0).$$

  (Our use of $\mathcal{V}()$ ignores all multiplicity information.)
- "DOF" means "degree(s)-of-freedom."
- We use the following abbreviations for types of joints between two links in a mechanism:
  - R = 1DOF rotational joint (simple hinge),
  - P = 1DOF prismatic joint (a slider joint allowing linear motion), and
  - S = 3DOF spherical joint (a ball-and-socket).

  These are discussed in more detail in § 4.2.
- $\mathbb{C}^*$, pronounced "Cee-star," is $\mathbb{C} \setminus 0$.
- $\mathbb{P}^n$ is n-dimensional projective space, the set of lines through the origin of $\mathbb{C}^{n+1}$.
- Points in $\mathbb{P}^n$ may be written as homogeneous coordinates $[x_0, \ldots, x_n]$ with $(x_0, \ldots, x_n) \neq (0, \ldots, 0)$. The coordinates are interpreted as ratios: $[x_0, \ldots, x_n] = [\lambda x_0, \ldots, \lambda x_n]$ for any $\lambda \in \mathbb{C}^*$.
- A quaternion u is written in terms of the elements $\mathbf{1}, \mathbf{i}, \mathbf{j}, \mathbf{k}$ as

  $$u = u_0\mathbf{1} + u_1\mathbf{i} + u_2\mathbf{j} + u_3\mathbf{k}, \quad u_0, u_1, u_2, u_3 \in \mathbb{C}.$$

  We call u_0 the *real part* of u and $u_1\mathbf{i} + u_2\mathbf{j} + u_3\mathbf{k}$ the *vector part*. We may also write $u = (u_0, \mathbf{u})$, where $\mathbf{u}$ is the vector part of u.

- If u and v are quaternions, then $u * v$ is their quaternion product. This may be written in terms of the vector dot-product operator "$\cdot$" and vector cross-product operator "$\times$" as

$$(u_0, \mathbf{u}) * (v_0, \mathbf{v}) = (u_0 v_0 - \mathbf{u} \cdot \mathbf{v}, u_0 \mathbf{v} + v_0 \mathbf{u} + \mathbf{u} \times \mathbf{v}).$$

  Note that $*$ does not commute: in general, $u * v \neq v * u$.

- When $u = (u_0, \mathbf{u})$ is a quaternion, $u' = (u_0, -\mathbf{u})$ is its quaternion conjugate, and $u * u'$ is its squared magnitude.

Part I

Fundamentals of Algebraic Kinematics

3 Some Motivating Examples

Before attempting to organize kinematic problems into a common framework, let us begin by examining some examples.

3.1 Serial-Link Robots

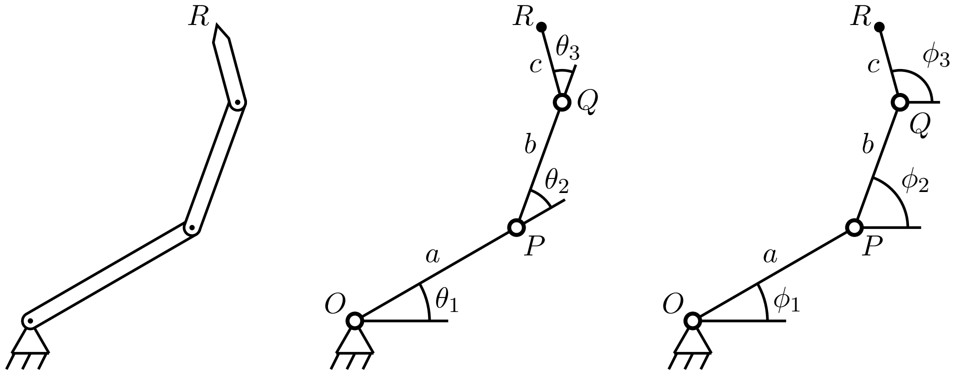
To motivate the discussion, let us see how one of the simplest problems from robot kinematics leads to a system of polynomial equations. It will also serve to introduce some basic terminology for robot kinematics. The generalization of this problem from the planar to the spatial case is one of the landmark problems solved in the 1980s.

3.1.1 Planar 3R robot

Consider the planar robot arm of Figure 1(a) consisting of three moving links and a base connected in series by rotational joints. (Hash marks indicate that the base is anchored immovably to the ground.) Kinematicians use the shorthand notation "R" for rotational joints and hence this robot is known as a planar 3R robot. The rotation of each joint is driven by a motor, and a sensor on each joint measures the relative rotation angle between successive links. Given the values of the joint angles, the *forward kinematics problem* is to compute the position of the tip, point R, and the orientation of the last link. Often, we refer to the last link of the robot as its "hand." The converse of the forward kinematics problem is the *inverse kinematics problem*, which is to determine the joint angles that will place the hand in a desired position and orientation.

The forward kinematics problem is easy. We assume that the position of point O is known: without loss of generality we may take its coordinates to be $(0, 0)$. The relative rotation angles, $(\theta_1, \theta_2, \theta_3)$ in Figure 1(b), are given, and we know the lengths a, b, c of the three links. We wish to compute the coordinates of the tip, $R = (R_x, R_y)$, and the orientation of the last link, absolute rotation angle ϕ_3 in Figure 1(c). To compute these, one may first convert relative angles to absolute ones as

$$(\phi_1, \phi_2, \phi_3) = (\theta_1, \theta_1 + \theta_2, \theta_1 + \theta_2 + \theta_3). \tag{1}$$



(a) (b) (c)

Figure 1: Three-link planar robot arm and its kinematic skeleton.

Then, the tip is at

$$\begin{bmatrix} R_x \\ R_y \end{bmatrix} = \begin{bmatrix} a\cos\phi_1 + b\cos\phi_2 + c\cos\phi_3 \\ a\sin\phi_1 + b\sin\phi_2 + c\sin\phi_3 \end{bmatrix}. \tag{2}$$

In this case, the forward kinematics problem has a unique answer.

The inverse kinematic problem is slightly more interesting. It is clear that if we can find the absolute rotation angles, we can invert the relationship Eq. 1 to find the relative angles. Moreover, we can easily find the position of point $Q = (Q_x, Q_y)$ as

$$Q = (Q_x, Q_y) = (R_x - c\cos\phi_3, R_y - c\sin\phi_3).$$

However, to find the remaining angles, ϕ_1, ϕ_2, we need to solve the trigonometric equations

$$\begin{bmatrix} Q_x \\ Q_y \end{bmatrix} = \begin{bmatrix} a\cos\phi_1 + b\cos\phi_2 \\ a\sin\phi_1 + b\sin\phi_2 \end{bmatrix}. \tag{3}$$

Of course, an alternative would be to use basic trigonometry to solve for θ_2 as the external angle of triangle OPQ having side lengths a, b, and $\|Q\|$, which can be done with the law of cosines. But, it is more instructive for our purposes to consider converting Eq. 3 into a system of polynomials.

One way to convert the trigonometric equations to polynomials is to introduce for $i = 1, 2$ variables $x_i = \cos\phi_i$, $y_i = \sin\phi_i$, $i = 1, 2$, and the trigonometric identities $\cos^2\phi_i + \sin^2\phi_i = 1$. With variable substitutions, the four trigonometric equations become the polynomial equations

$$\begin{bmatrix} ax_1 + bx_2 - Q_x \\ ay_1 + by_1 - Q_y \\ x_1^2 + y_1^2 - 1 \\ x_2^2 + y_2^2 - 1 \end{bmatrix} = 0. \tag{4}$$

This system of two linears and two quadratics has total degree $1 \cdot 1 \cdot 2 \cdot 2 = 4$, which is an upper bound on the number of isolated solutions. However, for general a, b, Q_x, Q_y it has only two isolated solutions.

An alternative method often used by kinematicians for converting trigonometric equations to polynomials is to use the tangent half-angle relations for rationally parameterizing a circle. This involves the substitutions $\sin\phi = 2t/(1+t^2)$ and $\cos\phi = (1-t^2)/(1+t^2)$, where $t = \tan\phi/2$. (Denominators must be cleared after the substitutions.) This approach avoids doubling the number of variables but the degrees of the resulting equations will be higher. Either way, the conversion to polynomials is a straightforward procedure.

The pathway of first writing trigonometric equations and then converting them to polynomials is not really necessary. We have presented it here to illustrate the fact that for any robot or mechanism built by connecting rigid links with rotational joints there is a straightforward method of arriving at polynomial equations that describe the basic kinematic relations.

A different way of arriving at polynomial equations for this simple example is to first concentrate on finding the location of the intermediate joint P. Clearly P is distance a from O and distance b from Q, both of which are known points. Thus, P can lie at either of the intersection points of two circles: one of radius a centered at O and one of radius b centered at Q. Letting $P = (P_x, P_y)$, we immediately have the polynomial equations

$$\begin{bmatrix} P_x^2 + P_y^2 - a^2 \\ (P_x - Q_x)^2 + (P_y - Q_y)^2 - b^2 \end{bmatrix} = 0. \tag{5}$$

Here, only (P_x, P_y) is unknown, and we have two quadratic polynomial equations. Subtracting the first of these from the second, one obtains the system

$$\begin{bmatrix} P_x^2 + P_y^2 - a^2 \\ 2P_xQ_x + Q_x^2 + 2P_yQ_y + Q_y^2 - b^2 + a^2 \end{bmatrix} = 0, \tag{6}$$

which consists of one quadratic and one linear polynomial in the unknowns (P_x, P_y). Hence, we see that the $3R$ inverse kinematics problems has at most two isolated solutions. In fact, by solving a generic example, one finds that there are exactly two solutions. Of course, this is what one would expect for the intersection of two circles.

The situation can be summarized as follows. The robot has a *joint space* consisting of all possible joint angles $\theta = (\theta_1, \theta_2, \theta_3)$, which in the absence of joint limits is the three-torus, T^3. The *parameter space* for the family of 3R planar robots is $\mathbb{R}^3$, consisting of link-length triplets (a, b, c). The *workspace* of the robot is the set of all possible positions and orientations of a body in the plane. This is known as special Euclidean two-space $SE(2) = \mathbb{R}^2 \times SO(2)$. Since $SO(2)$ is isomorphic to a circle, one can also write $SE(2) = \mathbb{R}^2 \times T^1$. Above, we have represented points in $SE(2)$ by the coordinates (R_x, R_y, ϕ_3). Equations 1 and 2 give the *forward kinematics map* $K_{3R}(\theta; q) : T^3 \times \mathbb{R}^3 \to SE(2)$. For any particular such robot, say one given by parameters $q^* \in \mathbb{R}^3$, we have a forward kinematics map that is the restriction of K_{3R} to that set of parameters. Let's denote that map as $K_{3R,q^*}(\theta) := K_{3R}(\theta; q^*)$, so $K_{3R,q^*} : T^3 \to SE(2)$.

The forward kinematics map $K_{3R,q}$ is generically two-to-one, that is, for each set of joint angles $\theta \in T^3$, the robot with general link parameters q has a unique hand location $H = K_{3R,q}(\theta)$ while for the same robot there are two possible sets of joint angles to reach any general hand location, i.e., $K_{3R,q}^{-1}(H)$ is two isolated points for general $H \in SE(2)$. To be more precise, we should distinguish between real and complex solutions. Let $T_{\mathbb{R}}^3$ be the set of real joint angles while $T_{\mathbb{C}}^3$ is the set of complex ones. Similarly, we write $SE(2,\mathbb{R})$ and $SE(2,\mathbb{C})$ for the real and complex workspaces. The *reachable workspace* of the robot having link parameters q is the set

$$W_q = \{H \in SE(2,\mathbb{R}) \mid \exists\theta \in T^3_{\mathbb{R}}, K_{3R}(\theta; q) = H)\}. \tag{7}$$

One can confirm using the triangle inequality that the reachable workspace is limited to locations in which point Q lies in the annulus

$$\{Q \in \mathbb{R}^2 \mid |a - b| \leq \|Q\| \leq |a + b|\}.$$

In the interior of the reachable workspace, the inverse kinematics problem has two distinct real solutions, outside it has none, and on the boundary it has a single double point solution. An exception is when $a = b$. In that case, the interior boundary of the annulus for Q shrinks to a point, the origin O. When Q is at the origin, the inverse kinematics problem has a positive-dimensional solution with $\theta_2 = \pm\pi$ and $\theta_1 = T^1$. We will refer to cases where the inverse kinematics problem has a higher than ordinary dimension as *exceptional sets*.

The forward kinematics equations extend naturally to the complexes, where the situation is somewhat simpler due to the completeness of the complex number field. Over the complexes, for generic a, b, the inverse kinematics problem has two isolated solutions everywhere except at singularities where it has a double root. The forward and inverse kinematics maps still make sense, but the reachable workspace is no longer relevant. Since a real robot can only produce real joint angles, the maps over the reals and the corresponding reachable workspace are ultimately of paramount interest, but to get to these, it can be convenient to first treat the problem over the complexes. The roots of the inverse kinematics problem over the complexes are real whenever the hand location is within the reachable workspace.

The problem of analyzing the reachable workspace for the robot with parameters q comes down to finding the singularity surface

$$S_q = \{\theta \in T^3_{\mathbb{R}} \mid \operatorname{rank} d_\theta K_{3R}(\theta; q) < 3\}, \tag{8}$$

where $d_\theta K_{3R}$ denotes the 3×3 Jacobian matrix of partial derivatives of K_{3R} with respect to the joint angles. The boundaries of the reachable workspace are contained in the image of the singularities, $K_{3R,q}(S_q)$. Again, it is often simplest to first treat this problem over the complexes and then to extract the real points in the complex solution set.

We note that the forward and inverse kinematics problems are defined by the analyst according to the intended usage of the device; there may be more than one sensible definition for a given mechanism. These are all related to the same essential kinematics of the device, usually related by simple projection operations. As a case in point, consider the variant of the forward and inverse kinematics problems, and the associated reachable workspace problem, occurring when the robot is applied not to locating the hand in position and orientation but rather simply positioning its tip without regard to the orientation with which this position is attained.

In the position-only case, we have a new forward kinematics map $K'_{3R} : T^3 \times \mathbb{R}^3 \to \mathbb{R}^2$ which may be written in terms the projection $\pi : SE(3) \to \mathbb{R}^2$, $\pi : (R_x, R_y, \phi_3) \mapsto (R_x, R_y)$, as $K'_{3R} = \pi \circ K_{3R}$. Holding the parameters of the robot fixed, the forward kinematics problem for K'_{3R} is still unique, but the inverse kinematics problem (finding joint angles to reach a given position) now has a solution curve. The positional 3R planar robot is said to be *kinematically* *redundant*, and motion along the solution curve for a fixed position of the endpoint is called the robot's *self-motion*.

The reachable workspace for the positional 3R robot is either a disc or an annulus in the plane, centered on the origin. The singularity condition, similar to Eq. 8,

$$S'_q = \{\theta \in T^3_{\mathbb{R}} \mid \text{rank } d_\theta K'_{3R}(\theta; q) < 2\}, \tag{9}$$

gives an image $K'_{3R}(S'_q) = \pi(K_{3R}(S'_q))$ that is the union of four circles having radii $|a \pm b \pm c|$. These correspond to configurations where points O, P, Q, and R are collinear: as shown in Figure 2(a), this means the linkage is fully outstretched or folded into one of three jackknife configurations. Rotation of the first joint does not affect the singularity, so the projection of each singularity curve gives a circle in the workspace.

The outer circle is a workspace boundary, while the inner circle may or may not be a boundary depending on the particular link lengths. If any point in the interior of the smallest circle can be reached, then the entire inner disk can be reached. In particular, the origin can be reached if and only if the link lengths satisfy the triangle inequality:

$$\max(a, b, c) \le a + b + c - \max(a, b, c).$$

For the particular case we are illustrating, this is satisfied, as is shown in Figure 2(b). The two intermediate circles are never workspace boundaries, but they mark where changes occur in the topology of the solution curve for the inverse kinematics problem.

This illustrates the general principle that potential boundaries of real reachable workspaces are algebraic sets given by equality conditions, and the determination of which patches between these is in or out of the workspace can be tested by solving the inverse kinematics problem at a test point inside the patch.

In short, we see that the 3R planar robot has three associated kinematics problems, as follows. For all these problems, the robot design parameters $q = (a, b, c)$ are taken as given.

1. The *forward kinematics problem* for a given robot q is to find the hand location from knowledge of the joint angles. This has the unique solution $(R_x, R_y, \phi_3) := K_{3R}(\theta; q)$.

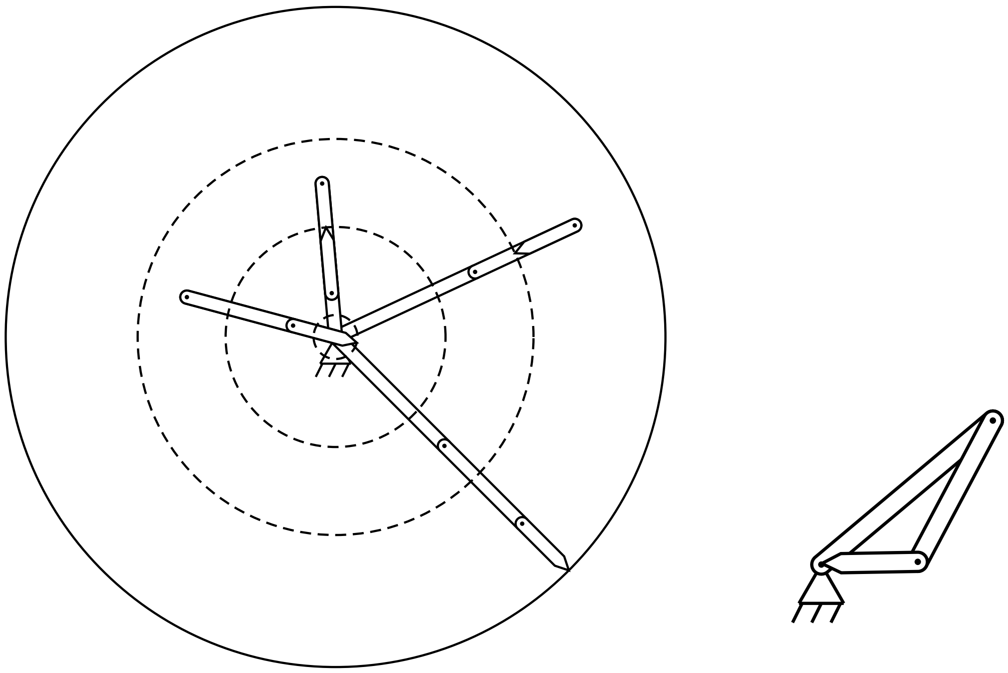
2. The *inverse kinematics problem* for a given robot q is to find the joint angle sets that put the hand in a desired location, $H \in SE(2)$. The answer, denoted $K^{-1}_{3R,q}(H)$, is the solution for θ of the equations $H - K_{3R}(\theta; q) = 0$. For generic cases, the solution set is two isolated points.

3. The *reachable workspace problem* is to find the boundaries of the real working volume of a given robot q. This can be accomplished by computing the singularity surface S_q of Eq. 8.

An additional problem of interest is to identify exceptional cases, especially cases where the dimensionality of the solutions to the inverse kinematic problem are larger than for the general case. More will be discussed about this in § 12. For this problem, we let the robot parameters $q = (a, b, c)$ vary and seek to solve the following.

4. The exceptional set problem for forward kinematics function K_{3R} is to find

$$\mathcal{D}_1 = \{(\theta, q, H) \in T^3 \times \mathbb{R}^3 \times SE(2) \mid \dim\{\theta \in T^3 \mid K_{3R}(\theta; q) = H\} \ge 1\}, \tag{10}$$



(a) (b)

Figure 2: Singularities and positional workspace for 3R planar robot.

where $\dim X$ means the dimension of set X. As will be described later, it is preferable in general to use *local dimension* instead of dimension, but it makes no difference in the 3R case. We described one exceptional set in the discussion of the inverse kinematic problem above. The exceptional set problem is significantly more difficult than the other three; the difficulty is only hinted at by the fact that all the symbols $(\theta_1, \theta_2, \theta_3, a, b, c, R_x, R_y, \phi_3) \in T^3 \times \mathbb{R}^3 \times SE(2)$ are variables in the problem.

All four problems exist as well for the positional 3R robot and its forward kinematics map $K'_{3R}(\theta; q)$. The generic inverse kinematics solutions become curves, and the exceptional set problem would find robots in configurations where the inverse solution has dimension greater than or equal to two.

In the problem statements of items 1–4 above, we have reverted to writing functions of angles, implying a trigonometric formulation. However, it is clear from the earlier discussion that all of these problems are easily restated in terms of systems of polynomial equations.

3.1.2 Spatial 6R robot

The spatial equivalent of the 3R planar robot is a 6R chain, illustrated in Figure 3. In the drawing, the six spool-like shapes indicate the rotational joints, which have axes labeled $\mathbf{z}_1, \ldots, \mathbf{z}_6$. Vectors $\mathbf{x}_1, \ldots, \mathbf{x}_6$ point along the common normals between successive joint axes. Frame $\mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0$ is the world reference frame, and a final end-effector frame $\mathbf{x}_7, \mathbf{y}_7, \mathbf{z}_7$ marks where the hand of the robot arm would be mounted. The purpose of the robot is to turn its joints so as to locate the end-

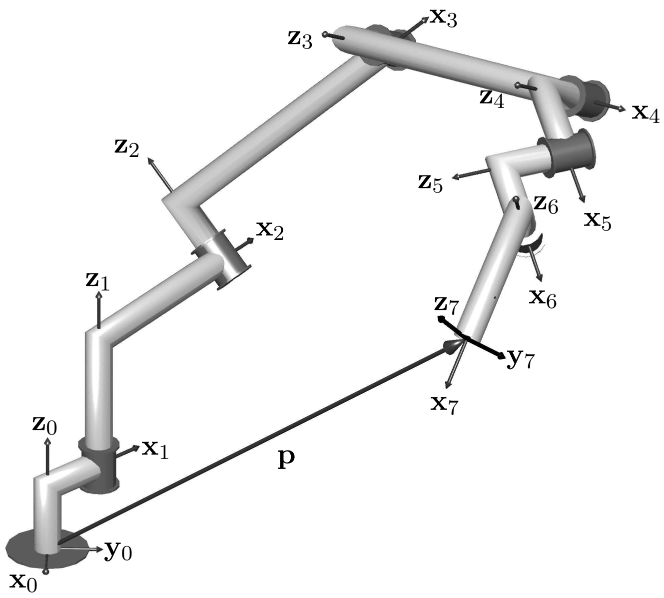


Figure 3: Schematic six-revolute serial-link robot

effector frame in useful spatial locations with respect to the world frame. For example, these destinations may be where the hand will pickup or drop off objects.

The set of all possible positions and orientations of a rigid body in three-space is $SE(3) = \mathbb{R}^3 \times SO(3)$, a six-dimensional space. Accordingly, for a serial-link revolute-joint spatial robot to have a locally one-to-one mapping between joint space and the workspace $SE(3)$, it must have six joints. With fewer joints, the robot's workspace would be a lower dimensional subspace of $SE(3)$. With more, the robot is *kinematically redundant* and has a positive dimensional set of joint angles to reach any point in the interior of its workspace. With exactly six joints, the robot generically has a finite number of isolated points in joint space that produce the same hand location in $SE(3)$.

For a general 6R robot and a general hand location, the 6R inverse kinematics problem has 16 isolated solutions over the complexes, while the number of real solutions may change with the hand location. The singularity surfaces divide the reachable workspace into regions within which the number of real isolated solutions is constant. This includes, of course, the boundaries between the reachable workspace and its exterior, where no solutions exist. Robots with special geometries, which are very common in practice, may have a lower number of isolated solutions, and for some hand locations and special robot geometries, positive-dimensional solution sets may arise. Thus, we see that the four types of problems discussed for the 3R robot extend naturally to the 6R robot, although one may expect the answers to be somewhat more complicated and harder to compute.

The most common way to formulate the kinematics of a serial link robot is to use 4×4 transformation matrices of the form:

$$A = \begin{bmatrix} C & d \\ 0_3 & 1 \end{bmatrix}, \quad C \in SO(3), \quad d \in \mathbb{R}^{3\times 1}, \quad 0_3 = [0\ 0\ 0]. \tag{11}$$

The set of all such matrices forms a representation of $SE(3)$. Pre-multiplication by A is equivalent to a rotation C followed by a translation d. When put in this form, a rotation of angle θ around the $\mathbf{z}$ axis is written

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{12}$$

We may use the 4×4 representation to build up the chain of transformations from one end of the robot to the other. The displacement from the origin of link 1, marked by $\mathbf{x}_1, \mathbf{z}_1$, to the origin of link 2, marked by $\mathbf{x}_2, \mathbf{z}_2$, is $R_z(\theta_1)A_1$ where A_1 is a constant transform determined by the shape of link 1. In this manner, the final location of the hand with respect to the world reference system, which we will denote as $H \in SE(3)$, can be written in terms of seven link transforms $A_0, \dots, A_6 \in SE(3)$ and six joint angles as

$$H = K_{6R}(\theta; q) := A_0 \prod_{j=1}^{6} R_z(\theta_j) A_j. \tag{13}$$

Here, the set of parameters q is the link transforms, so $q \in SE(3)^7$, while the joint space is the six-torus: $\theta \in T^6$. As in the planar case, to convert from this trigonometric formulation to an algebraic one, we simply adopt unit circle representations of the joint angles.

It is clear the four basic problem types discussed for the 3R robot (forward, inverse, workspace analysis, and exceptional sets) carry over to the 6R case with K_{6R} playing the role of K_{3R}.

3.2 Four-Bar Linkages

The reachable workspaces of 3R and 6R robots have the same dimension as the ambient workspace, $SE(2)$ and $SE(3)$, respectively. This is not true for many mechanisms used in practice. In fact, it is precisely the ability to restrict motion to a desired subspace that makes many mechanisms useful.

Consider, for example, the four-bar linkage, illustrated in Figure 4. As is customary, we draw the three moving links, two of which are attached to the ground at points A_1 and A_2. The ground link counts as the fourth bar; a convention that was not yet adopted in some of the older papers (e.g., [23, 26, 83]). The middle link, triangle B_1B_2P, is called the *coupler triangle*, as it couples the motion of the two links to ground, A_1B_1 and A_2B_2. Four bars can be used in various ways, such as

- *function generation*, i.e., to produce a desired functional relationship between angles Θ_1 and Θ_2;
- *path generation*, i.e., to produce a useful path of the *coupler point* P, as shown in Figure 4(b); or
- *body guidance*, in which the coordinated motion in $SE(2)$ of the coupler link is of interest.

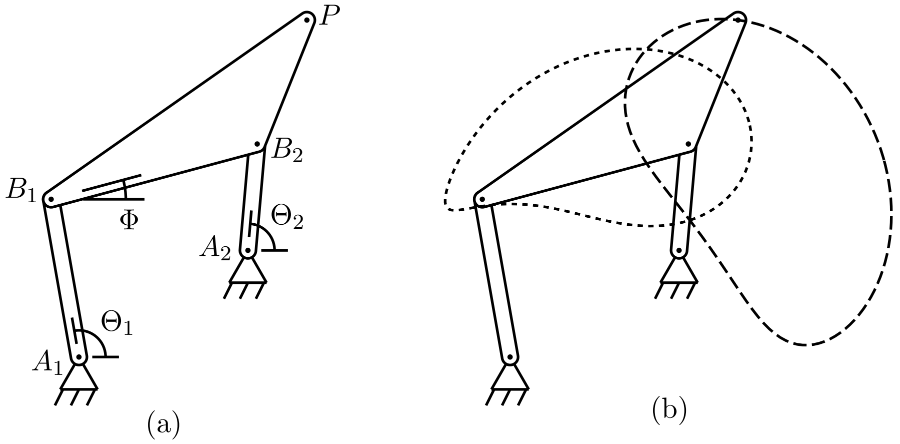


Figure 4: Four-bar linkage.

The path of the coupler point P is called the *coupler curve.* In the example shown in Figure 4(b) the coupler curve has two branches; the mechanism must be disassembled and reassembled again to move from one branch to the other. (Over the complexes, the curve is one irreducible set. Whether the real coupler curve has one or two branches depends on inequality relations between the link sizes.)

To specify a configuration of a four-bar, one may list the locations of the three moving links with respect to the ground link. That is, a configuration is a triple from $SE(3)^3$. We may take these as (A_1, Θ_1), (A_2, Θ_2) and (P, Φ). For generic link lengths, the mechanism has a 1-DOF motion, a curve in $SE(3)^3$. The whole family of all possible four-bars is parameterized by the shapes of the links and the placement of the ground pivots. These add up to 9 independent parameters, which we may enumerate as follows:

- positions of ground pivots A_1 and A_2 in the plane (4 parameters);
- lengths $|A_1B_1|$, $|A_2B_2|$, $|B_1P|$, and $|B_2P|$ (4 parameters); and
- angle $\angle B_1PB_2$ (1 parameter).

Clearly, the motion of a four-bar is equivalent to the self-motion of a kinematically redundant positional 3R robot whose endpoint is at A_2. The workspace regions drawn in Figure 2 now become the regions within which we may place A_2 without changing topology of the motion of the four-bar. The boundaries delineate where one or more of the angles Θ_1, Θ_2, and Φ change between being able to make full 360° turns or just partial turns.

As in the case of the 3R robot, we may use several different ways to write down the kinematic relations. One particularly convenient formulation for planar mechanisms, especially those built with rotational joints, is called *isotropic coordinates.* The formulation takes advantage of the fact that a vector $\mathbf{v} = a\mathbf{x} + b\mathbf{y}$ in the real Cartesian plane can be modeled as a complex number $v = a + bi$. Then, vector addition, $\mathbf{u} + \mathbf{v}$, is simple addition of complex numbers, $u + v$, and rotation of vector v by angle Θ is $e^{i\Theta}v$.

Suppose that in Cartesian coordinates $P = (x, y)$, so that the location of the coupler link is represented by the triple (x, y, Φ), which we may view as an element of $SE(2)$. To make the Cartesian representation algebraic, we switch to the unit-circle coordinates $(x, y, c_\Phi, s_\Phi) \in \mathcal{V}(c_\Phi^2 + \mathbf{s}_\Phi^2 - 1)$, which is an isomorphic representation of $SE(3)$. In isotropic coordinates, the representation of $SE(3)$ is $(p, \bar{p}, \phi, \bar{\phi}) \in \mathcal{V}(\phi\bar{\phi} - 1)$, where

$$p = x + iy, \quad \bar{p} = x - iy, \quad \phi = e^{i\Phi}, \quad \bar{\phi} = e^{-i\Phi}. \tag{14}$$

When (x, y, Φ) is real, we have the complex conjugate relations

$$p^* = \bar{p}, \quad \text{and} \quad \phi^* = \bar{\phi}, \tag{15}$$

where "*" is the conjugation operator. The isotropic coordinates are merely a linear transformation of the unit-circle coordinates:

$$\begin{bmatrix} p \\ \bar{p} \end{bmatrix} = \begin{bmatrix} 1 & i \\ 1 & -i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} \phi \\ \bar{\phi} \end{bmatrix} = \begin{bmatrix} 1 & i \\ 1 & -i \end{bmatrix} \begin{bmatrix} c_\Phi \\ s_\Phi \end{bmatrix}.$$

To model the four-bar linkage using isotropic coordinates, we draw a vector diagram in the complex plane, shown in Figure 5. Point O marks the origin, from which vectors a_1 and a_2 mark the fixed ground pivots and vector p indicates the current position of the coupler point. As the coupler rotates by angle Φ, its sides rotate to ϕb_1 and ϕb_2, where $\phi = e^{i\Phi}$. The links to ground have lengths ℓ_1 and ℓ_2. Letting $\theta_j = e^{i\Theta_j}$, $j = 1, 2$, we may write two vector loop equations:

$$\ell_1\theta_1 = p + \phi b_1 - a_1, \tag{16}$$
$$\ell_2\theta_2 = p + \phi b_2 - a_2. \tag{17}$$

Since a_1, a_2, b_1, b_2 are complex vectors, their isotropic representations include the conjugate quantities $\bar{a}_1, \bar{a}_2, \bar{b}_1, \bar{b}_2$, with $\bar{a}_1 = a_1^*$, etc., for a real linkage. Accordingly, the vector loop equations imply that the following conjugate vector equations must also hold:

$$\ell_1\bar{\theta}_1 = \bar{p} + \bar{\phi}\bar{b}_1 - \bar{a}_1, \tag{18}$$
$$\ell_2\bar{\theta}_2 = \bar{p} + \bar{\phi}\bar{b}_2 - \bar{a}_2. \tag{19}$$

Finally, we have unit length conditions for the rotations:

$$\theta_1\bar{\theta}_1 = 1, \quad \theta_2\bar{\theta}_2 = 1, \quad \text{and} \quad \phi\bar{\phi} = 1. \tag{20}$$

Altogether, given a four-bar with the parameters $a_1, \bar{a}_1, a_2, \bar{a}_2, b_1, \bar{b}_1, b_2, \bar{b}_2, \ell_1, \ell_2$, we have 7 equations for the 8 variables $p, \bar{p}, \phi, \bar{\phi}, \theta_1, \bar{\theta}_1, \theta_2, \bar{\theta}_2$, so we expect in the generic case that the mechanism has a 1DOF motion.

While Eqs. 16–20 describe the complete motion of the four-bar, we might be only interested in the path it generates. In general, in such cases we might just invoke a projection operation,

$$K : (p, \bar{p}, \phi, \bar{\phi}, \theta_1, \bar{\theta}_1, \theta_2, \bar{\theta}_2) \mapsto (p, \bar{p}),$$

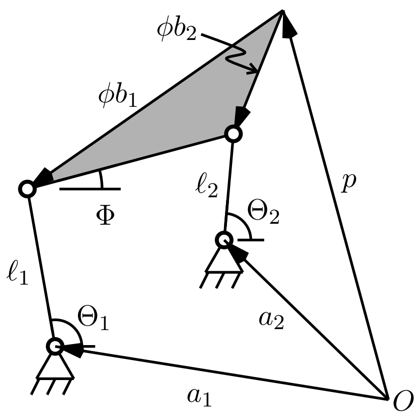


Figure 5: Vector diagram of a four-bar linkage.

and keep working with all the variables in the problem. However, in the case of the four-bar, we can easily perform the elimination operation implied by the projection. First, the rotations associated to Θ_1 and Θ_2 are eliminated as follows:

$$\ell_1^2 = (p + \phi b_1 - a_1)(\bar{p} + \bar{\phi}\bar{b}_1 - \bar{a}_1), \tag{21}$$

$$\ell_2^2 = (p + \phi b_2 - a_2)(\bar{p} + \bar{\phi}\bar{b}_2 - \bar{a}_2). \tag{22}$$

After expanding the right-hand sides of these and using $\phi\bar{\phi} = 1$, one finds that the equations are linear in $\phi, \bar{\phi}$. Consequently, one may solve eliminate these using Cramer's rule to obtain a single coupler-curve equation in just $p, \bar{p}$. Using $|\alpha\ \beta|$ to denote the determinant of a 2×2 matrix having columns α and β, we may write the coupler curve equations succinctly as

$$f_{cc}(p, \bar{p}; q) = |v\ \bar{u}||v\ u| + |u\ \bar{u}|^2 = 0, \tag{23}$$

where

$$v = \begin{bmatrix} (p - a_1)(\bar{p} - \bar{a}_1) + b_1\bar{b}_1 - \ell_1^2 \\ (p - a_2)(\bar{p} - \bar{a}_2) + b_2\bar{b}_2 - \ell_2^2 \end{bmatrix}, \quad u = \begin{bmatrix} b_1(\bar{p} - \bar{a}_1) \\ b_2(\bar{p} - \bar{a}_2) \end{bmatrix}, \quad \text{and} \quad \bar{u} = \begin{bmatrix} \bar{b}_1(p - a_1) \\ \bar{b}_2(p - a_2) \end{bmatrix}, \tag{24}$$

and where q is the set of linkage parameters:

$$q = (a_1, a_2, b_1, b_2, \bar{a}_1, \bar{a}_2, \bar{b}_1, \bar{b}_2, \ell_1, \ell_2).$$

Inspection of Eqs. 23, 24 shows that the coupler curve is degree six in $p, \bar{p}$, and in fact, the highest exponent on p or $\bar{p}$ is three. In other words, the coupler curve is a bi-cubic sextic. The number of monomials in a bi-cubic equation in two variables is 16, which is sparse compared to the possible 28 monomials for a general sextic. The bi-cubic property implies that a coupler curve can intersect a circle in at most 6 distinct points, one of the important properties of coupler curves.

The coupler curve equation answers the workspace analysis question for four-bar path generation. For example, the question of where the coupler curve crosses a given line comes down to solving a sextic in one variable, a process that can be repeated for a set of parallel lines to sweep out a picture of the whole curve in the plane. (This is not the simplest procedure that can be applied for this mechanism, but it is a workable one.)

A much tougher kind of question inverts the workspace *analysis* problem. These are *synthesis* problems of finding four-bars whose coupler curve meets certain specifications. For example, we might request that the coupler curve interpolate a set of points. The most extreme case is the nine-point synthesis problem of finding all four-bars whose coupler curve interpolates nine given points.[1] Essentially, this means solving for the parameters q that satisfy the system:

$$f_{cc}(p_j, \bar{p}_j; q) = 0, \qquad j = 0, \ldots, 8, \tag{25}$$

for nine given points $(p_j, \bar{p}_j) \in \mathbb{C}^2$, $j = 0, \ldots, 8$. This problem, first posed in 1923 by Alt [4], finally succumbed to solution by polynomial continuation in 1992 [113, 114], where it was demonstrated that 1442 coupler curves interpolate a generic set of nine points, each coupler curve generated by three cognate linkages (Robert's cognates [83]).

The system of Eq. 25 consists of nine polynomials of degree eight. Roth and Freudenstein [85] formulated the problem in a similar way, but ended up with eight polynomials of degree seven. Since this version has become one of our test problems, we give a quick re-derivation of it here. The key observation is that since we are given the nine points, we may choose the origin of coordinates to coincide with the first one: $(p_0, \bar{p}_0) = (0, 0)$. Moreover, the initial rotations of the links can be absorbed into the link parameters, so we may assume that $\Theta_1, \Theta_2, \Phi = 0$ at the initial precision point. Accordingly, at that point Eqs. 21,22 become

$$\ell_1^2 = (b_1 - a_1)(\bar{b}_1 - \bar{a}_1), \tag{26}$$
$$\ell_2^2 = (b_2 - a_2)(\bar{b}_2 - \bar{a}_2). \tag{27}$$

Substituting from these into the expression for v in Eq. 24, one may eliminate ℓ_1^2 and ℓ_2^2. Accordingly, the Roth-Freudenstein formulation of the nine-point problem is

$$f_{cc}(p_j, \bar{p}_j; q) = 0, \qquad j = 1, \ldots, 8, \tag{28}$$

with the precision points $(p_j, \bar{p}_j)$ given and with variables $a_1, a_2, b_1, b_2, \bar{a}_1, \bar{a}_2, \bar{b}_1, \bar{b}_2$. Although the equations appear to be degree eight, the highest degree terms all cancel, so the equations are actually degree seven. Hence, the total degree of the system is 7^8. The original Roth-Freudenstein formulation did not use isotropic coordinates, but their equations are essentially equivalent to the ones presented here.

Various other synthesis problems can be posed to solve engineering design problems. The nature of the synthesis problem depends on the use to which the linkage is being applied. For example, if the four-bar is being used for function generation, one may seek those mechanisms whose motion interpolates angle pairs $(\Theta_{1,j}, \Theta_{2,j}) \in \mathbb{R}^2$ (for up to five such pairs). For body guidance, one may specify triples $(p_j, \bar{p}_j, \Phi_j) \in SE(2)$ for up to five poses. Instead of just interpolating points, synthesis problems can also include derivative specifications (points plus

[1] Nine is the maximum possible number of generic points that can be interpolated exactly, because that is the number of independent parameters. Although we have 10 parameters in q, there is a one-dimensional equivalence class. This could be modded out by requiring that b_1 be real, i.e., $\bar{b}_1 = b_1$. We enumerated a count of nine independent parameters earlier in this section.

tangents). Kinematicians sometimes distinguish between these as "finitely separated" versus "infinitesimally separated" precision points. Every synthesis problem has a maximum number of precision points that can be exactly interpolated. If more points are specified, one may switch from *exact synthesis* to *approximate synthesis* in which one seeks mechanisms that minimize a sum of squares or other measure of error between the generated motion and the specified points.

While this subsection has focused on the four-bar planar linkage, it should be appreciated that similar questions will arise for other arrangements of links and joints, and like the 6R spatial generalization of the 3R planar robot, some of these move out of the plane into three-space.

3.3 Platform Robots

Suppose you are given the task of picking up a long bar and moving it accurately through space. While it may be possible to do so by holding the bar with one hand, for a more accurate motion, one naturally grasps it with two hands spread a comfortable distance apart. This reduces the moments that must be supported in one's joints. On the down side, a two-hand grasp limits the range of motion of the bar relative to one's body due to the finite length of one's arms.

Similar considerations have lead engineers to develop robotic mechanisms where multiple chains of links operate in parallel to move a common end-effector. Generally, this enhances the robot's ability to support large moments, but this comes at the cost of a smaller workspace than is attained by a serial-link robot of similar size. While many useful arrangements have been developed, we will concentrate on two "platform robots": one planar and one spatial. In each case, a moving end-effector link is supported from a stationary base link by several "legs" having telescoping leg lengths. As $SE(2)$ and $SE(3)$ are 3- and 6-dimensional, respectively, it turns out that the number of legs that make the most useful mechanisms is also 3 and 6, respectively. We begin with the planar 3-RPR platform robot and then address the related 6-SPS spatial platform robot. (These notations will be explained shortly.)

3.3.1 3-RPR Planar Robot

Recall that R indicates a rotational joint and P indicates a prismatic joint. An RPR chain is a set of four links in series where links 1 and 2 are connected by an R joint, 2 and 3 by a P joint, and 3 and 4 by another R joint. A 3-RPR planar robot has three such chains wherein the first and last links are common to all three chains. This arrangement is illustrated in Figure 7(a).

Before working further with the 3-RPR robot, we first consider what happens to a four-bar linkage if we add a fifth link between its coupler point and a new ground pivot, A_3, as shown in Figure 6(a). As it has five links, this new linkage is known as the *pentad* linkage. Due to the new constraint imposed on the motion of P by the link A_3P, the pentad becomes an immobile structure. The possible assembly configurations can be found by intersecting the four-bar coupler curve (branches C_1 and C_2) that we first met in Figure 4(b) with the circle C_3 centered on A_3 with radius $|\overline{AP}|$. As shown in Figure 6(b), this particular example has six real points of intersection, one of which gives the illustrated configuration. We already noted in the previous section that due to the bi-cubic nature of the coupler curve, there are generically six such intersection points over the complexes. In this example all of them are real.

The pentad could be called a 3-RR mechanism, as the three links (legs) between ground and

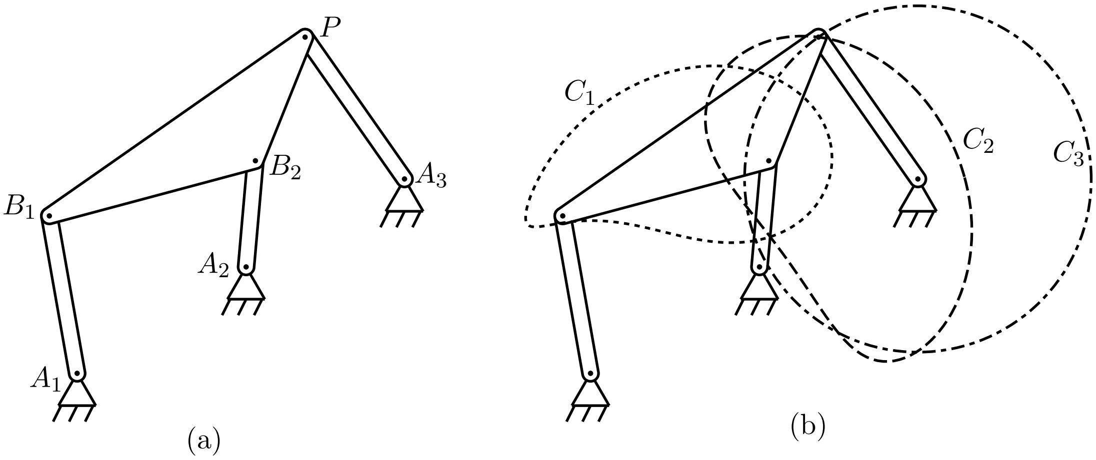


Figure 6: Pentad (3-RR) linkage.

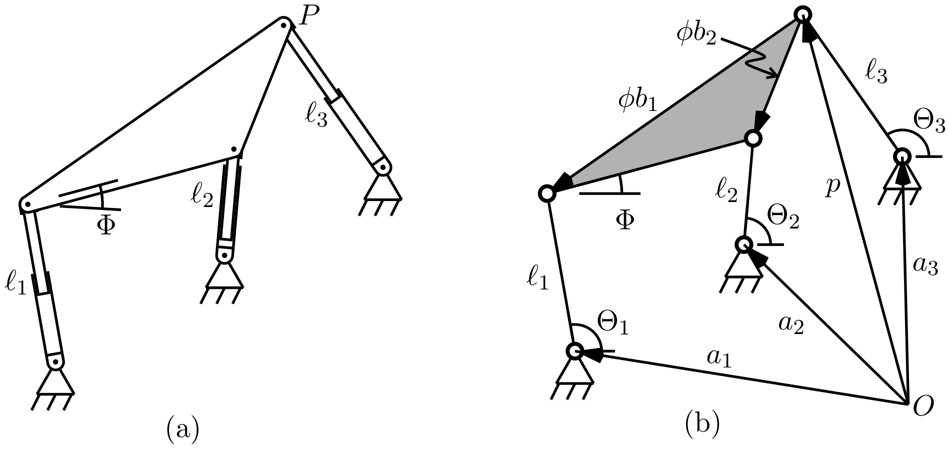


Figure 7: 3-RPR planar platform robot.

the coupler triangle have R joints at each end. If we add "prismatic" slider joints on each leg, it becomes the 3-RPR platform robot shown in Figure 7(a). The prismatic joints allow the leg lengths ℓ_1, ℓ_2, ℓ_3 to extend and retract, driven by screws or by hydraulic cylinders. Thus, the input to the robot are the three leg lengths $(\ell_1, \ell_2, \ell_3) \in \mathbb{R}^3$, and the output is the location of the coupler triangle, a point in $SE(2)$.

We have already done most of the work for analyzing the 3-RPR robot when we formulated equations for the four-bar linkage. The vector diagram in Figure 7(b) is identical to Figure 5 except for the addition of vector a_3 to the new ground pivot and the new link length ℓ_3. Using the same conventions for isotropic coordinates as before, we simply add the new equations

$$\ell_3\theta_3 = p - a_3, \quad \ell_3\bar{\theta}_3 = \bar{p} - \bar{a}_3, \quad \text{and} \quad \theta_3\bar{\theta}_3 = 1. \tag{29}$$

If we wish to eliminate Θ_3, we replace these with the circle equation

$$\ell_3^2 = (p - a_3)(\bar{p} - \bar{a}_3). \tag{30}$$

The inverse kinematics problem is to find the leg lengths (ℓ_1, ℓ_2, ℓ_3) given a desired location of the coupler triangle $(p, \bar{p}, \phi, \bar{\phi}) \in SE(2)$. Since the leg lengths must be positive, they can be evaluated uniquely from Eqs. 21,22,30.

In the opposite direction, the forward kinematics problem comes down to computing the simultaneous solution of Eqs. 23 and 30. This is the same as solving the pentad problem illustrated in Figure 6(b), which has up to six distinct solutions. Notice that once we have found $(p, \bar{p})$ from this intersection, we may recover $(\phi, \bar{\phi})$ from Eqs. 21,22, where they appear linearly. Using the notation of Eq. 24 for 2×2 determinants, this solution reads

$$\phi = |\bar{u}\ \ v|/|u\ \ \bar{u}|, \qquad \bar{\phi} = |v\ \ u|/|u\ \ \bar{u}|. \tag{31}$$

3.3.2 6-SPS "Stewart-Gough" Platform

A natural generalization of the 3-RPR planar platform robot is the 6-SPS spatial platform robot, where the rotational R joints are replaced by spherical S joints and the number of legs is increased to six. This arrangement, illustrated in Figure 8, is usually referred to as a "Stewart-Gough[2] platform." The device has a stationary base A and a moving platform B connected by six legs of variable length. The six legs acting in parallel can support large forces and moments, which makes the platform robot ideal as the motion base for aircraft simulators, although it has many other uses as well.

Most practical versions of the device resemble the arrangement shown in Figure 9, where the six joints on the base coincide in pairs, forming a triangle, and those on the moving platform are arranged similarly. The edges of these triangles along with the six connecting legs form the twelve edges of an octahedron. In more general arrangements, such as the one shown in Figure 8, the joints have no special geometric configuration.

It is easy to see that if the legs of the platform are allowed to extend and retract freely without limit, the moving platform can be placed in any pose, say $(\mathbf{p}, C) \in SE(3)$, where $\mathbf{p} \in \mathbb{R}^3$ is the position of a reference point on the platform and $C \in SO(3)$ is a 3×3 orthogonal matrix representing the orientation of the platform. Let $\mathbf{a}_j \in \mathbb{R}^3$, $j = 1, \ldots, 6$ be vectors in reference frame A to the centers $A_1, \ldots, A_6$ of the spherical joints in the base, and let $\mathbf{b}_j \in \mathbb{R}^3$, $1, \ldots, 6$ be the corresponding vectors in reference frame B for the joint centers $B_1, \ldots, B_6$ in the moving platform. Figure 8 labels the joints and joint vectors for leg 1. The length of the jth leg, denoted L_j is just the distance between its joint centers:

$$\begin{aligned} L_j^2 &= |\overline{A_j B_j}|^2 = ||\mathbf{p} + C\mathbf{b}_j - \mathbf{a}_j||_2^2 \\ &= (\mathbf{p} + C\mathbf{b}_j - \mathbf{a}_j)^T(\mathbf{p} + C\mathbf{b}_j - \mathbf{a}_j), \qquad j = 1, \ldots, 6. \end{aligned} \tag{32}$$

This spatial analog to Eqs. 21,22 solves the inverse kinematic problem for the 6-SPS robot. There is only one inverse solution having all positive leg lengths.

Another way to view Eq. 32 is that when leg length L_j is locked, it places one constraint on the moving platform: joint center B_j must lie on a sphere of radius L_j centered on joint center

[2]For historical reasons, the arrangement first became widely known as a Stewart platform, although D. Stewart's invention actually had a different design. As it appears that E. Gough was the true first inventor of the device, current usage [19] now credits both contributors.

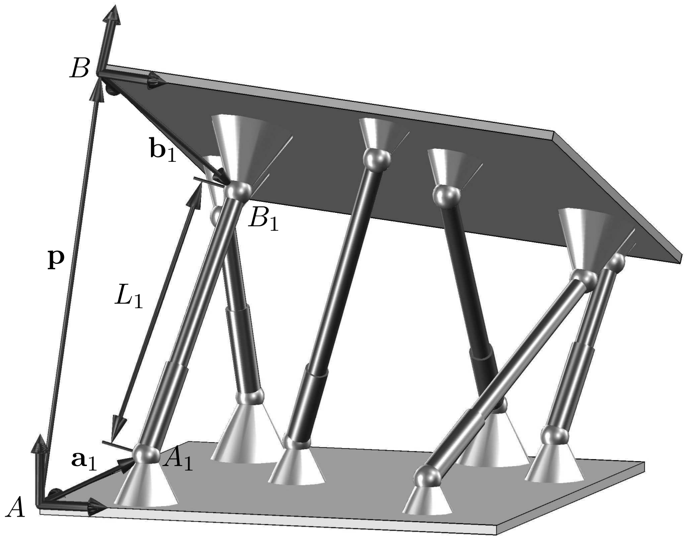


Figure 8: General Stewart-Gough platform robot

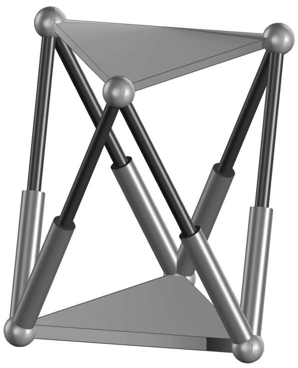


Figure 9: Octahedral Stewart-Gough platform robot

A_j. Since $SE(3)$ is six-dimensional, it takes six such constraints to immobilize plate B, turning the robot into a 6-SS structure. The forward kinematics problem is to find the location of B given the six leg lengths, that is, to solve the system of Eqs. 32 for $(\mathbf{p}, C) \in SE(3)$ given the leg lengths $(L_1, \ldots, L_6)$ and the parameters of the platform $(\mathbf{a}_j, \mathbf{b}_j)$, $j = 1, \ldots, 6$.

Notice that for the platform robots, the inverse kinematics problem is easy, having a unique answer, while the forward kinematics problem is more challenging, having multiple solutions. This is the opposite of the situation for serial-link robots. It can happen that both directions are hard; for example, consider replacing the simple SPS-type legs of the Stewart-Gough platform with ones of type 5R.

It is somewhat inconvenient to solve the forward kinematics problem as posed in Eq. 32, because we must additionally impose the conditions for C to be in $SO(3)$, namely $C^T C = I$ and $\det C = 1$. In the planar 3-RR case, isotropic coordinates simplified the treatment of displacements in $SE(2)$. Although isotropic coordinates do not generalize directly to $SE(3)$, there is an alternative applicable to the spatial case: Study coordinates. Just as isotropic coordinates $(p, \bar{p}, \phi, \bar{\phi}) \in \mathbb{C}^4$ are restricted to the bilinear quadric $\phi\bar{\phi} = 1$, Study coordinates $[e, g] \in \mathbb{P}^7$ are restricted to the bilinear quadric $S_6^2 = \mathcal{V}(Q)$, where

$$Q(e, g) = e^T g = e_0 g_0 + e_1 g_1 + e_2 g_2 + e_3 g_3 = 0. \tag{33}$$

The isomorphic mapping between Study coordinates $[e, g] \in S_6^2 \subset \mathbb{P}^7$ and $(\mathbf{p}, C) \in \mathbb{C}^3 \times SO(3)$ is discussed in Appendix A. In Study coordinates, translation $\mathbf{p}$ is rewritten in terms of $[e, g]$ via quaternion multiplication and conjugation as $\mathbf{p} = g * e'/(e * e')$ while rotation of vector $\mathbf{v}$ by rotation matrix C is rewritten as $C\mathbf{v} = e * \mathbf{v} * e'/(e * e')$.

In Study coordinates, the leg length equations become:

$$L_i^2 = ||(g * e' + e * \mathbf{b}_j * e')/(e * e') - \mathbf{a}_j||_2^2, \qquad j = 1, \ldots, 6. \tag{34}$$

Using the relations $||\mathbf{v}||^2 = \mathbf{v} * \mathbf{v}'$, and $(a * b)' = b' * a'$, one may expand and simplify Eq. 34 to rewrite it as

$$\begin{aligned} 0 &= g * g' + (\mathbf{b}_j * \mathbf{b}_j' + \mathbf{a}_j * \mathbf{a}_j' - L_j^2) e * e' + (g * \mathbf{b}_j' * e' + e * \mathbf{b}_j * g') \\ &\qquad - (g * e' * \mathbf{a}_j' + \mathbf{a}_j * e * g') - (e * \mathbf{b}_j * e' * \mathbf{a}_j' + \mathbf{a}_j * e * \mathbf{b}_j' * e') \\ &= g^T g + g^T B_j e + e^T A_j e, \qquad j = 1, \ldots, 6. \end{aligned} \tag{35}$$

In the last expression, g and e are treated as 4×1 matrices and the 4×4 matrices A_j (symmetric) and B_j (antisymmetric) contain entries that are quadratic in a_j, b_j, L_j. Expressions for A_j and B_j can be found in [112]. In the forward kinematics problem, A_j and B_j are considered known, so these equations along with the Study quadric equation, Eq. 33, are a system of seven quadrics on $\mathbb{P}^7$. The forward kinematics problem is to solve these for $[e, g] \in \mathbb{P}^7$ given the leg lengths $(L_1, \ldots, L_6)$.

The forward kinematics problem for general Stewart-Gough platforms received extensive academic attention leading up to its effective solution in the early 1990s. It can be shown that a generic instance of the problem has 40 isolated solutions in complex $\mathbb{P}^7$. The first demonstration of this was numerical, based on polynomial continuation [81, 82], while around the same time others published demonstrations based on Gröbner bases over a finite field [59, 78] and a proof by abstract algebraic geometry [84]. The formulation of the problem in Study coordinates was arrived at independently by Wampler [112], who gave a simple proof of the root count of 40, and Husty [50], who showed how to reduce the system to a single degree 40 equation in one variable. Using current tools in numerical algebraic geometry, the solution of seven quadrics is a simple exercise, requiring from a few seconds to a minute or so of computational time, depending on the computer applied to the task, to track 2^7 homotopy paths. As we shall discuss further below, once the first general example has been solved to find just 40 solutions, all subsequent examples can be solved by tracking only 40 homotopy paths.

4 Algebraic Kinematics

Now that we have discussed a variety of problems from kinematics, let us examine how we may fit them all into a common framework. This framework will help show why so many problems in kinematics are algebraic at their core.

4.1 Rigid-Body Motion Spaces

In the examples, we have already dealt in some detail with $SE(3)$, the six-dimensional set of rigid-body transformations in three-space. The most useful representations of $SE(3)$ are as follows.

- $(\mathbf{p}, C) \in \mathbb{R}^3 \times SO(3)$, where $SO(3) = \{C \in \mathbb{R}^{3\times 3} \mid C^T C = I, \det C = 1\}$. This acts on a vector $\mathbf{v} \in \mathbb{R}^3$ to transform it to $\mathbf{u} = C\mathbf{v} + \mathbf{p}$.
- The 4×4 homogeneous transform version of this where $(\mathbf{p}, C)$ are placed in a matrix so that the transform operation becomes

$$\begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = \begin{bmatrix} C & \mathbf{p} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix}.$$

- Study coordinates $[e, g] \in S_6^2 \subset \mathbb{P}^7$, where $S_6^2 = \mathcal{V}(e^T g)$ and the transform operation is

$$\mathbf{u} = (e * \mathbf{v} * e' + g * e')/(e * e').$$

In all three cases, the representations live on an algebraic set and the transform operation is also algebraic.

There are several subgroups of $SE(3)$ that are of interest. Most prominent is $SE(2)$, the set of planar rigid-body transformations, with representations as follows.

- $(\mathbf{p}, C) \in \mathbb{R}^2 \times SO(2)$, where $SO(2) = \{C \in \mathbb{R}^{2\times 2} \mid C^T C = I, \det C = 1\}$. The transform rule looks identical to the spatial case: $\mathbf{u} = C\mathbf{v} + \mathbf{p}$.
- The unit-circle form $\{(x, y, s, c) \in \mathbb{R}^4 \mid c^2 + s^2 = 1\}$. This is the same as the former with $\mathbf{p} = x\mathbf{i} + y\mathbf{j}$ and

$$C = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}.$$

- The tangent half-angle form $(x, y, t) \in \mathbb{R}^3$ in which rotations become

$$C = \frac{1}{1+t^2}\begin{bmatrix} 1-t^2 & -2t \\ 2t & 1-t^2 \end{bmatrix}$$

- Isotropic coordinates $\{(p, \bar{p}, \theta, \bar{\theta}) \in \mathbb{C}^4 \mid \theta\bar{\theta} = 1\}$. Real transforms must satisfy $p^* = \bar{p}$ and $\theta^* = \bar{\theta}$. The action of transform $(p, \bar{p}, \theta, \bar{\theta})$ on a vector given by isotropic coordinates $(v, \bar{v})$ is the vector $(u, \bar{u})$ given by
$$(u, \bar{u}) = (p + \theta v, \bar{p} + \bar{\theta}\bar{v}).$$

Again, each of these representations lives on an algebraic set and has an algebraic transform operation. Clearly, $SE(2)$ is a three-dimensional space.

Another subspace of interest is the set of *spherical transforms*, that is, just $SO(3)$, another three-dimensional space. This is $SE(3)$ with the translational portion set identically to zero. The terminology "spherical" derives from the fact that this is the set of motions allowed by a spherical ball set in a spherical socket of the same diameter.

At root, all of these are algebraic because an essential property of a rigid body transform is the preservation of distance between any two points. Squared distances are, of course, algebraic. The second property is that the transform must preserve handedness, as we do not wish to transmute a rigid body into its mirror image. It is this latter consideration that restricts the rotational portion to $SO(3)$, which is the component of the set of all orthogonal matrices, $O(3)$, that contains the identity matrix.

4.2 Algebraic Joints

A mechanism is a collection of rigid bodies connected by joints. Without the joints, each body could move with six degrees of freedom anywhere in $SE(3)$. Typically, we declare one body to be "ground" and measure the locations of all the other bodies relative to it, so a collection of n bodies lives in $SE(3)^{n-1}$. Joints are surfaces of contact between bodies that constrain the motion of the mechanism to a subset of $SE(3)^{n-1}$. *Algebraic joints* are those which constrain a mechanism to *algebraic subsets* of $SE(3)^{n-1}$.

The most important joints for building mechanisms are the *lower-order pairs*. These are pairs of identical surfaces that can stay in full contact while still allowing relative motion. In other words, they are formed by a surface that is invariant under certain continuous sets of displacements. The lower-order pairs form six possible joint types, having the following standard symbols.

R "Revolute." A surface of revolution is invariant under rotation about its axis of symmetry. A general R pair allows a 1DOF rotational motion equivalent to $SO(2)$. An example is a door hinge.

P "Prismatic." A surface swept out by translating a planar curve along a line out of its plane. A general P pair allows a 1DOF translational motion. An example is a square peg in a matching square hole.

H "Helical," also known as a "screw joint." A surface swept out by a curve that simultaneously rotates and translates about a fixed axis, with the two rates in fixed proportion. This yields a 1DOF motion for a general H pair. The ratio of the linear rate to the rotational rate is called the "pitch" of the screw. An example is a nut on a screw.

C "Cylindrical." A round peg in a matching round hole can both rotate around the axis of symmetry and translate along it independently, generating a 2DOF motion.

E "Plane," from the German "ebener" since P is already taken. Plane-to-plane contact allows a 3DOF motion, namely the set $SE(2)$.

S "Spherical." A spherical ball in a matching spherical socket allows a 3DOF motion, the set $SO(3)$.

The importance of the lower-order pairs derives from the fact that surface-to-surface contact spreads forces of contact over a larger area, reducing stresses that might wear out the machinery.

Fortunately—from the viewpoint of an algebraic geometer—five of these six joint types are algebraic. The exception is the H joint. The relative motion of a helical joint of pitch ρ with axis through the origin along the $\mathbf{z}$ direction is given by 4×4 transforms of the form

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & \rho\theta \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{36}$$

The mixture of θ with $\cos\theta$ and $\sin\theta$ makes the motion truly trigonometric, hence non-algebraic. An alternative line of reasoning is to observe that a helix and a plane containing its symmetry axis intersect in an infinite number of isolated points. Any algebraic curve in $\mathbb{R}^3$ intersects a plane in at most a finite number of isolated points.

Even more fortuitously, helical joints are rarely used as a direct motion constraint in a manner that impacts kinematic analysis. Instead, screws are usually used to transmit power along a prismatic joint. Consequently, the geometric motion of a great many mechanisms is governed by algebraic joints.

Not all joints in mechanism work are lower-order pairs. Line contact between two surfaces can be adequate to spread stresses, and it is typical of many rolling contacts (e.g., wheel on plane). With sufficient lubrication, even sliding line contacts can endure, such as happens in some cam contacts.

To demonstrate that a joint type is algebraic, one may write down the constraint conditions it imposes between the transforms for the two bodies in contact. Suppose $(\mathbf{p}_1, C_1), (\mathbf{p}_2, C_2) \in SE(3)$ are the rigid-body transforms for bodies 1 and 2. To impose constraints equivalent to the lower-order joints, we need to specify geometric features that are associated with the joints. To this end, let $\mathbf{a}$ be a point of body 1, given in body 1 coordinates. The world coordinates of the point are then $\hat{\mathbf{a}} = \mathbf{p}_1 + C_1\mathbf{a}$. Similarly, let $\mathbf{b}$ be a point in body 2, so that its world coordinates are $\hat{\mathbf{b}} = \mathbf{p}_2 + C_2\mathbf{b}$. Let $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ be a dextral set of mutually orthogonal unit vectors in body 1, that is, the matrix formed with these as its column vectors is in $SO(3)$. Let $(\mathbf{v}_1, \mathbf{v}_2)$ be orthogonal unit vectors in body 2. The unit vectors transform into world coordinates as $\hat{\mathbf{u}}_i = C_1\mathbf{u}_i$, $i = 1, 2, 3$, and $\hat{\mathbf{v}}_i = C_2\mathbf{v}_i$, $i = 1, 2$. With these features, the constraints imposed by lower-order pairs R, P, C, E, and S can be modeled with equations that are algebraic in the transforms (p_1, C_1) and (p_2, C_2) and the features $\mathbf{a}, \mathbf{b}, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{v}_1, \mathbf{v}_2$ as follows.

R: $\hat{\mathbf{a}} = \hat{\mathbf{b}}$ and $\hat{\mathbf{u}}_1 = \hat{\mathbf{v}}_1$, which after substituting from the expressions above become

$$\mathbf{p}_1 + C_1\mathbf{a} = \mathbf{p}_2 + C_2\mathbf{b}, \quad \text{and} \quad C_1\mathbf{u}_1 = C_2\mathbf{v}_1.$$

For brevity, in the remaining cases we only give the constraints on the world coordinates, which the reader may easily expand to get equations in the transforms and features.

P: $\hat{\mathbf{u}}_2^T(\hat{\mathbf{b}} - \hat{\mathbf{a}}) = 0$, $\hat{\mathbf{u}}_3^T(\hat{\mathbf{b}} - \hat{\mathbf{a}}) = 0$,$\hat{\mathbf{u}}_1 = \hat{\mathbf{v}}_1$, $\hat{\mathbf{u}}_2 = \hat{\mathbf{v}}_2$.

C: $\hat{\mathbf{u}}_2^T(\hat{\mathbf{b}} - \hat{\mathbf{a}}) = 0$, $\hat{\mathbf{u}}_3^T(\hat{\mathbf{b}} - \hat{\mathbf{a}}) = 0$,$\hat{\mathbf{u}}_1 = \hat{\mathbf{v}}_1$

E: $\hat{\mathbf{u}}_1^T(\hat{\mathbf{b}} - \hat{\mathbf{a}}) = 0$, $C_1\mathbf{u}_1 = C_2\mathbf{v}_2$.

S: $\hat{\mathbf{a}} = \hat{\mathbf{b}}$.

Equating two points imposes three constraints, whereas equating two unit vectors imposes only two independent constraints. From this, we see that the R joint imposes 5 constraints, the C joint imposes 4, and the E and S joints each impose 3 constraints. For the P joint, we are equating a pair of orthogonal unit vectors to another such pair, which together imposes three constraints. With the other two scalar equations, this brings the total number of constraints imposed by a P joint to 5. These models are not minimal with respect to the number of parameters involved, so in practice we usually to write down equations in other ways.[3] Nevertheless, this suffices to show that the lower-order pairs R, P, C, E, and S are, indeed, algebraic.

Joints can be described either *extrinsically* in terms of the constraints they impose or *intrinsically* in terms of the freedoms they allow. The foregoing description is extrinsic. We have already seen how to model R joints intrinsically in the discussion of spatial 6R serial-link robots (see § 3.1.2). Note that this was written in terms of 4×4 transforms $R_z(\theta)$ from Eq. 12, which is equivalent to the transform for a helical joint in Eq. 36 with zero pitch, $\rho = 0$. The lower-order pairs can be modeled intrinsically using matrices of the form

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & a \\ \sin\theta & \cos\theta & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{37}$$

R: Use Eq. 37 with $a = b = c = 0$, leaving θ as the joint variable.

P: Use Eq. 37 with $\theta = a = b = 0$, leaving c as the joint variable.

C: Use Eq. 37 with $a = b = 0$, leaving θ and c both as joint variables.

E: Use Eq. 37 with $c = 0$, leaving θ, a, and b all as joint variables.

[3] The Denavit-Hartenberg formalism is a minimal parameterization. See [68] or any modern kinematics textbook for a definition.

	R	P	C	E	S
constraints	5	5	4	3	3
freedoms	1	1	2	3	3

Table 1: Algebraic lower-order pairs: constraints and freedoms

S: Use
$$\begin{bmatrix} C & 0 \\ 0 & 1 \end{bmatrix}$$
with $C \in SO(3)$ as the joint freedom.

In an intrinsic formulation, the link geometry is encoded in link transforms the are interposed between the joints, such as the matrices A_j in the formulation of the 6R kinematics in § 3.1.2.

If c is the number of independent constraints imposed by a joint, then the number of freedoms of body 2 relative to body 1 allowed by the joint is $6-c$, as summarized in Table 1. When modeling the joints with a low number of freedoms (R, P, C) it is usually more convenient to use an intrinsic formulation, while S joints are best modeled extrinsically.

4.3 Mechanism Types, Families, and Spaces

To fit a wide variety of kinematics problems into a common format, we need the definitions of a mechanism type and a mechanism family.

Definition 4.1 *A* mechanism type *is defined by the number of links, n_L, and a symmetric $n_L \times n_L$ adjacency matrix T whose (i, j)th element denotes the type of joint between links i and j, one of R, P, H, C, E, S, or $\emptyset$, where $\emptyset$ indicates no connection. By convention, all diagonal elements are $\emptyset$.*

(Each joint appears twice in the matrix: $T_{i,j} = T_{j,i}$ are the same joint.) We assume here that the joints are limited to the lower-order pairs, but the list of possibilities could be extended. The enumeration of all possible mechanism types for each value of n_L without double counting mechanisms that are isomorphic under renumbering of the links is a problem in discrete mathematics. Choosing a prospective mechanism type is the first step in a mechanism design effort, and methods for guiding the enumeration of promising alternatives fall into the category of *type synthesis*. In this paper, we assume that this crucial step is already done so that we begin with a mechanism type.

Each mechanism type has an associated parameter space. We have seen in § 4.2 one way to model each of the algebraic lower-order pairs, R, P, C, E, and S, in terms of feature points and unit vectors. The cross-product space of all these geometric features forms a universal parameter space for the mechanism type. One may choose to model the joints in a more parsimonious way, but we assume that in the alternative model there still exists a parameterization for each joint and an associated parameter space for all the joints taken together.

Definition 4.2 *A* universal mechanism family (T, Q) *is a mechanism type T with an associated parameter space Q describing the geometry of the joints. We assume that Q is irreducible.*

If one has a parameter space Q that is not irreducible, each irreducible component should be considered to define a separate universal mechanism family.

Definition 4.3 *A* mechanism family (T, Q') *is a subset of a universal mechanism family* (T, Q) *restricted to an irreducible algebraic subset* $Q' \subset Q$.

Examples of the common sorts of algebraic restrictions that define a mechanism family include the condition that the axes of two R joints in a certain link must be parallel, perpendicular, or intersecting, etc. As a particular example, consider that the universal family of spatial 3R serial-link chains includes the family of 3R planar robots of § 3.1.1 wherein the R joints are all parallel. One should appreciate that there can be subfamilies inside of families, and so on.

For certain mechanisms, all points of the links move in parallel planes, hence the links move in $SE(2)$ and the mechanism is said to be *planar*. In particular, a mechanism family wherein all joints are either rotational R with axis parallel to the $\mathbf{z}$-direction or prismatic P with axis perpendicular to the $\mathbf{z}$-direction is planar.

Definition 4.4 *The* link space Z *for an* n *link mechanism is* $SE(3)^{n-1}$, *where one of the links is designated as ground* $(\mathbf{p}, C) = (0, I)$. *Any of the isomorphic representations of* $SE(3)$ *from § 4.1 can be used as models of* $SE(3)$. *If the mechanism family is planar, then* $Z = SE(2)^{n-1}$ *in any of its isomorphic representations from § 4.1.*

Definition 4.5 *The* mechanism space M *of a mechanism family* (T, Q) *is the subset of* $Z \times Q$ *that satisfies the joint constraints.*

Proposition 4.1 *If a mechanism family is built with only the algebraic joints R, P, C, E, and S, then its mechanism space is algebraic.*

Proof. Section 4.1 with Appendix A shows that Z is algebraic and Q is algebraic by assumption. That is, Z and Q are sets defined by algebraic equations. Section 4.2 shows that the algebraic joints impose algebraic constraints on the coordinates of Z and Q, and hence all the defining equations for M are algebraic. □

Definition 4.6 *A* mechanism *is a member of a mechanism family* (T, Q) *given by a set of parameters* $q \in Q$.

4.4 Kinematic Problems in a Nutshell

In this section, we present an abstract formulation that summarizes all the main types of geometric problems that arise in kinematics. In the next section, we will discuss more concretely how to map a mechanism into this formulation.

The key to our formulation is the following diagram:

$$\begin{tikzcd} X & & M \arrow[ll, "J"'] \arrow[rr, "K"] \arrow[dll, "\hat{J}"'] \arrow[d, "\pi_M"] \arrow[drr, "\hat{K}"] & & Y \\ X \times Q \arrow[u, "\pi_1"] \arrow[rr, "\pi_2"'] & & Q & & Y \times Q \arrow[ll, "\pi_3"] \arrow[u, "\pi_4"'] \end{tikzcd} \tag{38}$$

The main elements of the diagram are four sets X, M, Y, Q and three maps J, K, π_M. The four sets are as follows.

- X is the *input space* of the mechanism. In robotics, it is usually called the "joint space." Its coordinates are typically quantities that we command by controlling motors or other actuators.

- Y is the *output space*, often called the "operational space" in robotics. Its coordinates are the final output(s) we wish to obtain from the mechanism, such as the location of a robot's hand.

- Q is the *parameter space* of a family of mechanisms. It is the set of parameters necessary to describe the geometry of the joints in each link. Each point in Q is therefore a specific mechanism with designated link lengths, etc. The whole set Q constitutes a family of mechanisms, such as the set of all 6R robot arms, with the coordinates of Q representing all possible link lengths, etc. We assume that Q is an irreducible algebraic subset of some $\mathbb{C}^m$, that is, it is an irreducible component of $\mathcal{V}(G)$ for some system of analytic functions $G : \mathbb{C}^m \to \mathbb{C}^{m'}$. If $\mathcal{V}(G)$ has more than one irreducible component, then each such component is considered a different family of mechanisms.

- M is the *mechanism space*, which describes all possible configurations of the mechanism for all possible parameters. Let Z be the space of all possible locations of the links when they are disconnected. That is, for an N-link spatial mechanism with one link designated as ground, $Z = SE(3)^{N-1}$. Then, M is the subset of $Z \times Q$ where the link locations satisfy the constraints imposed by the joints between them. Let $F : Z \times Q \to \mathbb{C}^c$ be a set of polynomials defining the joint constraints. Then, $M = \mathcal{V}(F) \cap \mathcal{V}(G)$ is an extrinsic representation of M. Each point $(z, q) \in M$ is a specific mechanism $q \in Q$ in one of its assembly configurations $z \in Z$. In some cases, it is more natural to describe M intrinsically via an irreducible set, say Θ, that parameterizes the freedoms of the joints of the mechanism, so that Z becomes $\Theta \times SE(3)^{N-1}$. We will use this, for example, to describe M for 6R serial-link robots. In such a representation, F includes the equations that define Θ along with the equations relating link poses to joint freedoms and equations for the constraints imposed by closing kinematic loops. Formulating such equations is part of the art of kinematics, and we will not delve into it in this paper beyond what is necessary to present specific examples.

After choosing a representation for $SE(3)$, and if present, for the joint freedom space Θ, the space Z is a subspace of some Euclidean space, $Z \subset \mathbb{C}^\nu$, and $z \in Z$ has coordinates $z = (z_1, \ldots, z_\nu)$.

Three maps are defined on M, as follows.

- $J : M \to X$ is the *input map*, which extracts from M the input values. The symbol J acknowledges that the inputs are usually a set of joint displacements.

- $K : M \to Y$ is the *output map*, which extracts from M the output values.

- $\pi_M : M \to Q$ is a projection that extracts the parameters from M. It is the natural projection operator on $Z \times Q$ restricted to M given by $\pi_M : (z, q) \mapsto q$.

For the moment, we assume only that F, G, J, K are analytic so that the spaces X, M, Y are analytic sets. Although later we will restrict further to algebraic maps and algebraic sets, the analytic setting allows a somewhat wider range of mechanisms into the framework. Specifically, H joints are analytic but not algebraic.

The commutative diagram is completed by defining $\hat{J} := (J, \pi_M)$ and $\hat{K} := (K, \pi_M)$ and the associated natural projections $\pi_1, \pi_2, \pi_3, \pi_4$.

It should be understood that M characterizes a family of mechanisms, such as the family of spatial 6R serial-link robots, the family of planar four-bar linkages, or the family of Stewart-Gough platforms. Maps J and K are tailored to an application of the mechanism. For a four-bar function generator, J gives the input angle and K gives the output angle, while for a four-bar path generator, K gives instead the position of the coupler point.

Using the diagram of Eq. 38, we may succinctly recast all the problems mentioned in the motivating examples of § 3. The problems broadly classified into three types of problems:

- Analysis (mobility analysis, forward and inverse kinematics, workspace analysis),
- Synthesis (precision point problems), and
- Exceptional mechanisms.

We describe each of these in more detail next.

4.4.1 Analysis

In analysis problems, one has a specific mechanism, say $q^* \in Q$, and one wishes to analyze some aspect of its motion.

Definition 4.7 *The* motion of a mechanism *given by parameters $q^* \in Q$ in a family with mechanism space M is $\pi_M^{-1}(q^*) = M \cap \mathcal{V}(q - q^*)$ where (z, q) are coordinates in $Z \times Q$. This can also be called the* motion fiber over q^*.

In the following, it is also convenient to define the inverses of J and K:

$$J^{-1}(x) = \{(z, q) \in M \mid J(z, q) = x\}, \qquad K^{-1}(y) = \{(z, q) \in M \mid K(z, q) = y\}.$$

These are defined for $x \in X$ and $y \in Y$, respectively. In the set $J^{-1}(x)$ for a particular $x \in X$, q is not fixed, so this inverse applies across a whole mechanism family. When we wish to address just one particular mechanism, we want to consider the inverse of $\hat{J}$ instead:

$$\hat{J}^{-1}(x, q) = \{(z, q) \in M \mid \hat{J}(z, q) = (x, q)\}.$$

Similarly, we have:

$$\hat{K}^{-1}(y, q) = \{(z, q) \in M \mid \hat{K}(z, q) = (y, q)\}.$$

The basic problems in analysis are as follows.

- **Motion decomposition** of a *mechanism* breaks $\pi_M^{-1}(q^*)$ into its irreducible components, often called assembly modes by kinematicians. (See § 7 for a description of irreducible components.) The numerical irreducible decomposition of $\pi_M^{-1}(q^*)$ (§ 7.5) finds the dimension and degree of each assembly mode and provides a set of witness points on each.

- **Motion decomposition** of a *mechanism family* breaks M into its irreducible components. If $A \subset M$ is one of these components, then $\pi_M(A) \subset Q$ is the subfamily of mechanisms that can be assembled in that mode, $\dim \pi_M(A)$ is the dimension of the subfamily, and $\dim A - \dim \pi_M(A)$ is the mobility of that mode.

- **Mobility analysis** seeks to find the degrees of freedom (DOFs) of the mechanism, which is equivalent to
$$\text{Mobility} := \dim \pi_M^{-1}(q^*). \tag{39}$$
As the dimension of an algebraic set is always taken to be the largest dimension of any of its components, this definition of mobility picks out the assembly mode (or modes) having the largest number of DOFs. There are simple formulas, known as the Gruebler-Kutzbach formulas, that correctly estimate the mobility for a wide range of mechanisms, and even more mechanisms submit to refined analysis based on displacement group theory, but there exist so-called "paradoxical" mechanisms that have higher mobility than these methods predict. To handle all cases, one needs to analyze the equations defining M in more detail taking into account that q^* may be on a subset of Q having exceptional mobility.

- **Local mobility analysis** finds the mobility of a mechanism in a given assembly configuration. That is, given $(z^*, q^*) \in Z \times Q$, one wishes to find
$$\text{Local mobility} := \dim_{(z^*,q^*)} \pi_M^{-1}(q^*). \tag{40}$$
A mechanism can have more than one assembly mode, corresponding to the irreducible components of $\pi_M^{-1}(q^*)$. The local mobility is the dimension of the assembly mode that contains the given configuration, z^*, or if there is more than one such mode, the largest dimension among these.

- **Forward kinematics** seeks to find the output that corresponds to a given input x^* for a mechanism q^*. That is, for $x^* \in X$ and $q^* \in Q$, one wishes to find
$$FK(x^*, q^*) := K(\hat{J}^{-1}(x^*, q^*)). \tag{41}$$
Example: given the joint angles of a particular 6R serial-link robot, find its hand pose.

- **Inverse kinematics** is similar to forward kinematics but goes from output to input. For $y^* \in Y$ and $q^* \in Q$ find
$$IK(y^*, q^*) := J(\hat{K}^{-1}(y^*, q^*)). \tag{42}$$
Example: given the hand pose of a particular 6R serial-link robot, find all sets of joint angles that reach that pose.

- **Singularity analysis** finds configurations where the maps lose rank. If we have found a motion decomposition of the mechanism, then for each assembly mode $A \subset \pi_M^{-1}(q^*)$ there is an associated input space $J(A)$ and an output space $K(A)$. The input and output maps have Jacobian matrices $\partial J/\partial z$ and $\partial K/\partial z$. Assume for the moment that A is a reduced algebraic set. (For example, $\mathcal{V}(x-y)$ is a reduced line in the (x,y)-plane, while the double line $\mathcal{V}((x-y)^2)$ is non-reduced.) For generic points $(z,q^*) \in A$, the Jacobian matrices have a constant rank, say $\mathrm{rank}[\partial J/\partial z(z,q^*)] = r_J$ and $\mathrm{rank}[\partial K/\partial z(z,q^*)] = r_K$. Then, there may be input and output singularities, as follows.

Input Singularities: $\{(z,q^*) \in A \mid \mathrm{rank}\,\frac{\partial J}{\partial z}(z,q^*) < r_J\}$. In the common case that $\partial J/\partial z$ is square and generically full rank, these are the special configurations where, to first order, the mechanism can move without any change in its input.

Output Singularities: $\{(z,q^*) \in A \mid \mathrm{rank}\,\frac{\partial K}{\partial z}(z,q^*) < r_K\}$. In the common case that $\partial K/\partial z$ is square and generically full rank, these are the special configurations where, to first order, the mechanism can move without any change in its output.

If A is a non-reduced assembly mode, a kinematician might consider the whole set to be singular, as the defining equations for A are degenerate in the sense that the column rank of the Jacobian for the system of equations that define $\pi_M^{-1}(q^*)$ is less than the codimension of the set A. Alternatively, one might wish to consider the input and output singularities of the reduction of A, which can be analyzed via a deflation of A. (See § 6.3 for a discussion of deflation.)

- **Workspace analysis** seeks to find all possible outputs of a robot or mechanism. Ignoring limits on the inputs, this is just the set $K(\pi_M^{-1}(q^*))$. The main concern in practice is the set of outputs for real assembly configurations, so letting $A_{\mathbb{R}}$ denote the real points in an assembly mode A, the corresponding workspace is $K(A_{\mathbb{R}})$. As we discussed in § 3.1.1, the boundaries of the real workspace $K(A_{\mathbb{R}})$ are given by the real output singularities. When joint limits are included, these may also induce boundaries in the workspace. If $B \subset X$ is a joint limit boundary, then $K(J^{-1}(B) \cap \pi_M^{-1}(q^*))$ may form part of the workspace boundary. Positional workspaces and orientational workspaces fit into the same picture by reformulating K to output just position or orientation.

  Example 1: for a 6R serial-link robot, find all possible poses that the hand can reach.

  Example 2: for a given four-bar linkage with output defined as the position of its coupler point, find the coupler curve.

  Example 3: for a 6-SPS (Stewart-Gough) platform robot with limits on the leg lengths, find all possible poses of the moving platform.

The motion of a mechanism over the complexes contains its real motion, and but the extraction of the real motion from the complex one can be difficult, all the more so as the dimensionality of the motion grows. See § 11 for a discussion. The real motion may break up into smaller pieces than the complex one: for example, the coupler curve illustrated in Figure 4(b) has two real branches but they both belong to the same irreducible curve in the complexes. Some branches of a real motion may even have a smaller dimension than the complex motion that contains them. See [115] for further discussion.

The problems presented above mainly concern the geometry of a mechanism's motion, where principally angles, positions, and poses enter the picture. As indicated by the questions of singularity analysis, one may also be concerned with differential relations between these, so that joint rates, linear velocity, and angular velocity may become objects of study, as might quantities related to acceleration. Once one has a point on M, differentiation is a straightforward process as is the treatment of the linear maps defined by the Jacobian matrices for the input and output maps. Even so, it can be of interest to find computationally efficient procedures for evaluating the derivative relations, particularly in real-time control of robots. Singularity analysis is important as the robot must either be banned from moving too near its singularities or else special control methods must be employed to avoid degenerate, and potentially dangerous, behaviors.

4.4.2 Synthesis

While analysis determines how a mechanism moves, synthesis finds mechanisms that move in a specified way. Synthesis problems begin with a set of desired outputs or a set of input/output pairs and seek to find the mechanisms that will produce these. Synthesis tends to be harder than analysis because one must consider the ability of the mechanism to reach each desired state. In essence, we must consider multiple copies of M simultaneously. The relevant construction in algebraic geometry is called the *fiber product*.

Definition 4.8 *For algebraic sets M and Q with map $\pi : M \to Q$, the* fiber product *of M over Q is*

$$M \times_Q M := \{(u, v) \in M \times M \mid \pi(u) = \pi(v)\}.$$

This just means that u and v are two assembly configurations of the same mechanism q, where $q = \pi(u) = \pi(v)$. If $F(z; q) = 0$ is a set of polynomial equations that defines M, that is, $M = \mathcal{V}(F(z; q))$, then $M \times_Q M$ simply repeats the equations keeping the parameters the same in both copies:

$$M \times_Q M = \mathcal{V}(F(z_1; q), F(z_2; q)).$$

Clearly, the fiber product operation can be extended to triple fiber products and higher, with the k-fold fiber product defined as

$$\prod_Q^k M = \underbrace{M \times_Q \cdots \times_Q M}_{k \text{ times}} := \{(u_1, \ldots, u_k) \in M^k \mid \pi(u_1) = \cdots = \pi(u_k)\}.$$

We may define a map $\Pi : \prod_Q^k M \to Q$ as the composition of $\underbrace{\pi \times \cdots \times \pi}_{k \text{ times}}$ with the map $(q, \ldots, q) \mapsto q$. The input and output maps J and K also extend naturally to fiber products, and we can also define a combined input/output map JK as follows.

$$\begin{aligned} K_k &: \prod_Q^k M \to Y^k, \text{ where } K_k(u_1, \ldots, u_k) = (K(u_1), \ldots, K(u_k)), \text{ and} \\ JK_k &: \prod_Q^k M \to (X \times Y)^k, \text{ where } JK_k(u_1, \ldots, u_k) = \{(J(u_1), K(u_1)), \ldots, (J(u_k), K(u_k))\}. \end{aligned} \tag{43}$$

Note that $K_1 \equiv K$.

With these maps, we may define several kinds of synthesis problems. The following problems are known as *precision point* problems, since there is a set of specified points which the mechanism must interpolate exactly.

- **Output synthesis** seeks mechanisms that can reach a set of specified outputs. For $(y_1, \dots, y_k) \in Y^k$, we wish to find the set

  $$\{q \in Q \mid K_k(\Pi^{-1}(q)) = (y_1, \dots, y_k)\}.$$

  Kinematicians distinguish between different types of output synthesis.

  Path synthesis finds mechanisms where the path of a point of the mechanism interpolates a set of given points. In this case, K is defined on M such that $Y \subset \mathbb{C}^3$.

  Body guidance In this case, the output is the pose of one body of the mechanism, that is, $Y \subset SE(3)$. The purpose of the mechanism is to guide that body through a set of specified poses.

- **Input/output synthesis** seeks mechanisms that produce a coordinated input/output relationship specified by a set of input/output pairs. For $((x_1, y_1), \dots, (x_k, y_k)) \in (X \times Y)^k$, we wish to find the set

  $$\{q \in Q \mid JK_k(\Pi^{-1}(q)) = ((x_1, y_1), \dots, (x_k, y_k)).$$

  A common case is a 1DOF mechanism, such as a four-bar, with the input being the angle of one link with respect to ground. Then, with K defined as in a path synthesis problem, the input/output problem becomes *path synthesis with timing*. Similarly, one can have *body guidance with timing*. (The nomenclature derives from an assumption that the input moves at a constant rate.) If the input and output are both angles of the mechanism, then input/output synthesis becomes *function generation*, as the precision points approximate some desired functional relationship between input and output.

What makes these problems difficult is that the whole system of equations defining M is repeated k times, increasing the total degree of the system exponentially in k.

For any precision point problem, there is a maximum number of precision points that can be specified. Roughly speaking, this is the total number of independent parameters in the mechanism family under consideration divided by the number of constraints placed on the parameters by each precision point. If more than the maximum number of precision points is specified, then there will in general be no mechanism that interpolates them exactly. One may then reformulate the problem by defining an error metric and seek mechanisms whose motion best fits the specified approximation points. This is analogous to finding a best-fit line that approximates three or more points.

We should note that all these synthesis problems have been formulated only at the geometric level. It is also possible to specify motions at the level of velocity or acceleration or to mix specifications at several levels. For a 1DOF motion, differential relations can be approximated by limits as precision points approach each other. For this reason, classical synthesis theory sometimes distinguishes between *finitely-separated precision points* and *infinitesimally-separated precision points*. We will not discuss synthesis problems involving differential relations further here.

4.4.3 Exceptional Mechanisms

While M describes the motion of an entire family of mechanisms, $\pi_M^{-1}(q^*)$ is the motion of a particular mechanism in the family. For any generic $q \in Q$, attributes such as the mobility of the mechanism or the local mobilities of its assembly modes all stay constant. However, there may be algebraic subsets of Q where mobilities increase. These exceptional mechanisms are often called "overconstrained mechanisms," as a slight perturbation of the parameters off of the exceptional set into a generic position suddenly brings in extra constraints that reduce mobility. One may define subsets of M where the local mobility is constant, that is,

$$\mathcal{D}_k^* = \{(z,q) \in M \mid \dim_{(z,q)} \pi_M^{-1}(q) = k\}. \tag{44}$$

The closures of these, $\mathcal{D}_k = \overline{\mathcal{D}_k^*}$, are algebraic sets. When $\mathcal{D}_j \subset \mathcal{D}_k$, $j > k$, we say that $\mathcal{D}_j$ is an exceptional set of mechanisms, a family of overconstrained mechanisms.

The discovery of exceptional mechanisms is perhaps the most difficult kind of kinematics problem. One may think of these as a kind of synthesis problem where the only thing that is specified about the motion is its mobility. As in the precision-point synthesis problems, it turns out that fiber products play a central role. We leave further discussion of this to § 12.

4.5 Fitting into the Nutshell

The examples of § 3 illustrated how a range of problems from kinematics can be written as systems of algebraic equations. Moreover, § 4.1 and § 4.2 show that any mechanism composed of n rigid links connected by any combination of R, P, C, E, or S joints leads to a set of constraint equations that is algebraic in the link locations $(\mathbf{p}_j, C_j) \in SE(3)$, $j = 1, \ldots, n$ and is also algebraic in the parameters defining the joints. In § 4.3,4.4, we put forward a schema that formulates a wide variety of kinematics problems in terms of spaces X, M, Y, Q and maps J, K, π between them. In this section, we will detail how the example mechanism types of § 3 fit into this schema.

4.5.1 Planar 3R Robots

Consider first the 3R planar serial-link robots. These have $n_L = 4$ links, one of which is ground. The adjacency matrix has R in each element of the super- and sub-diagonals and $\emptyset$ everywhere else. Since the mechanism is planar, the link space is $Z = SE(2)^3$. Using the reference frames as indicated in Figure 4.5.1, we have coordinates for Z as

$$z_1 = (P_x, P_y, x_1, y_1), \quad z_2 = (Q_x, Q_y, x_2, y_2), \quad z_3 = (R_x, R_y, x_3, y_3), \tag{45}$$

where (P_x, P_y) are the coordinates of point P, etc., and $x_j = \cos\phi_j$, $y_j = \sin\phi_j$, $j = 1, 2, 3$. (Recall from Figure 1(c) that ϕ_1, ϕ_2, ϕ_3 are the absolute rotation angles of the links.) Accordingly, the algebraic equations defining the link space Z are

$$x_j^2 + y_j^2 - 1 = 0, \qquad j = 1, 2, 3. \tag{46}$$

The parameters of the mechanism are just the link lengths (a, b, c), so the parameter space Q is $\mathbb{C}^3$. In the plane, the constraint imposed on two links by a rotational joint is the coincidence of

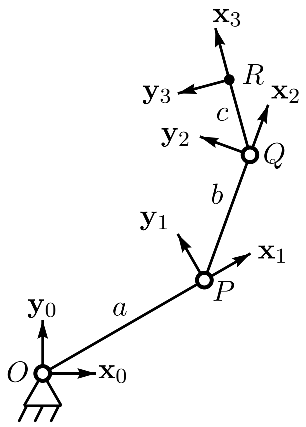


Figure 10: Planar 3R robot with reference frames.

the point of connection. Point $O = (0,0)$ in the ground must coincide with point $(-a, 0)$ in the reference frame of link 1:

$$(0,0) = (P_x - ax_1, P_y - ay_1). \tag{47}$$

Similarly, the other two joints impose the constraints

$$(P_x, P_y) = (Q_x - bx_2, Q_y - by_2) \text{ and } (Q_x, Q_y) = (R_x - cx_3, R_y - cy_3) \tag{48}$$

Accordingly, Eqs. 46–48 define the mechanism space M.

To complete the picture, we need the maps π, J, K. The projection $\pi : Z \to Q$ simply picks out the parameters:

$$\pi : (z_1, z_2, z_3, a, b, c) \mapsto (a, b, c). \tag{49}$$

Assuming the input space $X = T^3$ is the relative rotation angles $(\theta_1, \theta_2, \theta_3)$ (see Figure 1(b)) represented by cosine/sine pairs, the difference formulas for cosine and sine give

$$J : (z_1, z_2, z_3, a, b, c) \mapsto ((x_1, y_1), (x_2x_1 + y_2y_1, y_2x_1 - x_2y_1), (x_3x_2 + y_3y_2, y_3x_2 - x_3y_2)). \tag{50}$$

Finally, assuming the output space $Y = SE(2)$ is the location of reference frame 3, the output map is

$$K : (z_1, z_2, z_3, a, b, c) \mapsto (z_3). \tag{51}$$

If instead the robot is applied to just positioning point R in the plane, we have $Y = \mathbb{C}^2$ with the output map

$$K' : (z_1, z_2, z_3, a, b, c) \mapsto (R_x, R_y). \tag{52}$$

With these definitions, all the problems posed in § 3.1.1 fit neatly into the nutshell scheme.

4.5.2 Spatial 6R robots

The case of spatial 6R robots is quite similar to the 3R planar case, but we shall choose to handle the joint constraints by introducing variables implicitly modeling the freedom of the joints rather than explicitly writing constraint equations. A 6R serial-link chain has $n_L = 7$ links, one of which is ground. The adjacency matrix has entries of R on the super- and sub-diagonals and $\emptyset$ elsewhere. Let the link space be $Z = T^6 \times SE(3)^6$, with unit circle representations of the joint angles and 4×4 homogeneous transforms for the link locations, so that Z is represented as $z = \{(c_j, s_j, T_j), \quad j = 1, \ldots, 6\}$ with

$$c_j^2 + s_j^2 - 1 = 0, \qquad j = 1, \ldots, 6. \tag{53}$$

The first factor of Z is precisely the joint space $X = T^6$ and the output is the location of the "hand," the last frame in the chain, T_6. The link parameters are 4×4 transforms $A_j \in SE(3)$, $j = 0, 1, \ldots, 6$. One can use general transforms, but the Denavit-Hartenberg formalism shows that by choosing reference directions aligned with joint axes and their common normals as indicated in Figure 3, it suffices to parameterize the A_j as

$$A_j = \begin{bmatrix} 1 & 0 & 0 & a_j \\ 0 & \alpha_j & -\beta_j & 0 \\ 0 & \beta_j & \alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad j = 0, \ldots, 6. \tag{54}$$

In this expression, (α_j, β_j) are a cosine/sine pair for the twist of link j, a_j is the length of the link (along its **x**-direction), and d_j is the link offset distance (along its **z**-direction). To keep A_j in $SE(3)$, these must satisfy

$$\alpha_j^2 + \beta_j^2 - 1 = 0, \qquad j = 0, \ldots, 6. \tag{55}$$

With this parameterization, the parameter space is $Q = T^7 \times \mathbb{C}^{14}$ with coordinates $q = (\alpha_j, \beta_j, a_j, d_j)$, $j = 0, \ldots, 7$. Joint rotations $R_z(c_j, s_j)$ of the form

$$R_z(c, s) = \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{56}$$

alternate with relative link displacements A_j to give the transforms of the link locations as

$$T_1 = A_1 R_z(c_1, s_1) A_2, \qquad T_j = T_{j-1} R(c_j, s_j) A_j, \qquad j = 2, \ldots, 6. \tag{57}$$

It is easy to see how Eq. 57 leads to Eq. 13 when one eliminates all but the last transform, T_6. Equations 53–56 define the mechanism space M in terms of coordinates (z, q). The associated maps from M to Q, $X = T^6$, and $Y = SE(3)$ are

$$\pi : (z, q) \mapsto q, \qquad J : (z, q) \mapsto ((c_j, s_j), j = 1, \ldots, 6), \qquad K : (z, q) \mapsto T_6. \tag{58}$$

4.5.3 Four-Bar Linkages

The four-bar has four links with four R joints. If we call the ground link 0, the two links connected to ground as links 1 and 2, and the coupler as link 3, then the adjacency matrix has entries $T_{1,3} = T_{2,3} = T_{0,1} = T_{0,2} = \mathrm{R}$. We have already detailed all the necessary equations to describe the four-bar mechanism family in § 3.2 using isotropic coordinates. The link space Z is given by coordinates $z = (p, \bar{p}, \phi, \bar{\phi}, \theta_1, \bar{\theta}_1, \theta_2, \bar{\theta}_2)$ subject to the unit length conditions of Eq. 20 ($\phi\bar{\phi} = \theta_1\bar{\theta}_1 = \theta_2, \bar{\theta}_2 = 1$). The parameter space is $Q = \mathbb{C}^{10}$ with coordinates $q = (a_1, \bar{a}_1, a_2, \bar{a}_2, b_1, \bar{b}_1, b_2, \bar{b}_2, \ell_1, \ell_2)$. Equations 16–19 express the conditions for connecting the coupler link and thereby define the mechanism space M.

The input and output spaces and their associated maps depend on the application of the mechanism. For path generation, we have output $Y = \mathbb{C}^2$ with map $K : (z, q) \mapsto (p, \bar{p})$. For body guidance, $Y = SE(2)$ with $K : (z, q) \mapsto (p, \bar{p}, \phi, \bar{\phi})$. If timing along the coupler curve or timing of the body motion are of concern, we may name the angle of one of the links connected to ground as input, say $X = T^1$ given by $J : (z, q) \mapsto (\theta_1, \bar{\theta}_1)$. For function generation, the input is the angle of link 1 and the output is the angle of link 2, so $J : (z, q) \mapsto (\theta_1, \bar{\theta}_1)$ and $K : (z, q) \mapsto (\theta_2, \bar{\theta}_2)$.

4.5.4 Planar 3-RPR Platforms

Coordinates for the mechanism space M of the 3-RPR planar platform are an extension of those for the four-bar with $a_3, \bar{a}_3, \theta_3, \bar{\theta}_3, \ell_3$ appended. However, there is a shuffle in which coordinates are parameters and which are variables of the motion. The new maps are

$$\begin{array}{ll} \pi & : (z, q) \mapsto (a_1, \bar{a}_1, a_2, \bar{a}_2, a_3, \bar{a}_3, b_1, \bar{b}_1, b_2, \bar{b}_2) \\ J & : (z, q) \mapsto (\ell_1, \ell_2, \ell_3) \\ K & : (z, q) \mapsto (p, \bar{p}, \phi, \bar{\phi}) \end{array} \tag{59}$$

4.5.5 Spatial 6-SPS Platforms

For the forward and inverse kinematics problems of the 6-SPS platform, we do not need to explicitly represent transforms for the upper and lower leg segments. It is enough to use the leg lengths and the transform for the moving platform. Hence, the link space is $Z = \mathbb{C}^6 \times SE(3)$, and if we use Study coordinates for $SE(3)$, the space is $Z = \mathbb{C}^6 \times S_6^2$, where $S_6^2 \subset \mathbb{P}^7$ is the Study quadric given by Eq. 33. By the notation of § 3.3.2, the coordinates of Z are $(L_1, \ldots, L_6), [e, g]$. The parameter space Q consists of the vectors $\mathbf{a}_j, \mathbf{b}_j \in \mathbb{C}^3$, $j = 1, \ldots, 6$. The mechanism space M is given by the Study quadric along with the leg-length equations Eq. 35. The input space $X = \mathbb{C}^6$ is the set of leg lengths $L_1, \ldots, L_6$, and the output space is $Y = S_6^2$ is the Study quadric for the transform of the moving platform. The maps J, K are the associated natural projections.

5 History of Numerical Algebraic Geometry and Kinematics

The interaction between algebraic geometry and kinematics dates back to the nineteenth century, when many leading mathematicians were quite familiar with kinematics. In this section, we briefly trace the arc from those beginnings to the current state of the interaction between numerical algebraic geometry and kinematics. Symbolic computer algebra with application to kinematics, often called *computational algebraic geometry*[4], shares the same roots, but the scope of this account is limited to the numerical branch of developments. As our goal is to give a sense of the history, not a thorough literature review, we cite only representative articles here.

In the late nineteenth century, mathematicians such as Chebyshev and Cayley [23], made direct contributions to kinematics, while Sylvester is known to have followed developments, having presented news of the Peaucellier straight-line mechanism to the Royal Institution in January, 1874 [106]. Offshoots of his investigations into straight-line motion are the Sylvester plagiograph and the Sylvester quadruplanar inversor [34, 56]. Sylvester's interest in kinematics was reportedly spurred by Chebyshev's urging for him to "Take to kinematics, it will repay you; it is more fecund than geometry; it adds a fourth dimension to space" [44]. J. de Groot's *Bibliography on Kinematics* [31] lists 36 entries authored by Chebyshev, one of which, *Théorie des mécanismes connus sous le nom de parallélogrammes* [24] contains the first appearance of what are now known as Chebyshev polynomials. Another famous contribution is the Chebyshev approximate straight-line mechanism [56, 34]. Other mathematical notables with an interest in kinematics include Kempe [56], Schönflies [86], Study [102], and Clifford [26].

The strong interaction between kinematics and algebraic geometry was suspended in the early 20th century when the algebraic geometry took a turn towards abstraction under the Hilbert program. At this time, the consideration of concrete examples of polynomial systems fell out of favor as mathematicians clamored to obtain results using the power of the new abstract formalisms. In addition, the shift in focus was at least partially spurred on by the increasing difficulty of carrying out by hand the calculations required to study new and more challenging systems. However, kinematicians persevered, relying on the old methodologies, such as the Sylvester "dialytic elimination" method, a generalization of the Sylvester resultant. Bottema and Roth [20] and Hunt [49] give thorough discourses on the theoretical results in algebraic kinematics up to the late 1970s, while Dijksman [34] covers more details of the planar case. The mathematical techniques used in these classic books would have been familiar to mathematicians from the turn of the century. The same can also be said of most later contributions such as Angeles [5] and McCarthy [68], although over time emphasis shifted from hand derivations and graphical methods to computer algorithms based on the same underlying mathematics.

As computers first became available, kinematicians showed a strong interest in applying them to solving polynomial systems, a drive that would eventually bring the kinematics community back in touch with the applied mathematics community. In 1959, one of the first such publications [38] derived a quartic equation for synthesizing a four-bar coupler curve to interpolate five precision points. The expressions were derived by hand and solved on a computer using the quartic formulas. Just a few years later, Roth and Freudenstein [85] attacked the much more difficult problem of interpolating nine general points, the most possible with a four-bar coupler curve. This work introduced the technique of continuation, which they called the "bootstrap method" and subsequently the "parameter-perturbation procedure" [37], and applied it for finding solutions to the associated system of eight seventh-degree equations. From that start onward, problems from kinematics have played a strong role in motivating the development of continuation into the

[4]Good general references in the area are [27, 28]. While most work in the area assumes exact (rational) coefficients, Stetter has pioneered the consideration of numerical issues introduced by inexact (floating point) data [101].

powerful tool that it has become today.

Roth and Freudenstein were motivated by the fact that Newton's method from a very rough initial guess rarely succeeded in the nine-point problem. Continuation overcomes this problem by defining a parameterized family of systems wherein there exists a starting system with a known solution. This solution may then be updated by Newton's method as the parameters are stepped in small increments to arrive at the target system. In Roth and Freudenstein's formulation, there was no guarantee that the solution path would safely arrive at a solution to the target system. In particular, they worked over the real numbers, hence there was always the possibility that the solution path would encounter a singularity where the real solution branches out into the complexes. Although they introduced procedures to evade such trouble spots, these still gave no guarantee of finding a solution. Even so, they unquestionably advanced the art by showing that they could compute some solutions to a problem that had previously seemed intractable.

The development of continuation into a technique that could find all solutions to a system of polynomials required new mathematical ideas. The first such algorithm was published in 1977 by Drexler [35], followed soon by Garcia and Zangwill [39], Garcia and Li [40], and Chow, Mallet-Paret, and Yorke [25]. These latter works all used what today is known as a *total degree homotopy* (see § 6.1).

In the beginning, continuation was rather computationally expensive, so its wider acceptance as a convenient tool depended on more efficient algorithms and more powerful computers. However, early on it was recognized as being more robust than techniques based on algebraic elimination, since an elimination procedure more readily fails on encountering special cases, such as when certain leading coefficients are zero or nearly so. (Consider that Gaussian elimination without pivoting will fail on encountering a zero leading coefficient.) Due to its robustness, Morgan implemented continuation as a back-up method to elimination for finding intersections in GMSOLID, an early solid-modeling program [21].

Morgan's work attracted the attention of coworker L.W. Tsai, a kinematician who thought that continuation might provide a solution to the problem of the inverse kinematics of general six-revolute (6R) robot arms (see § 3.1.2). At the time, this was considered the most important unsolved problem in the area, having once been declared the "Mount Everest of kinematics" by F. Freudenstein. Although an elimination procedure had been found in 1980 that reduced the problem to a single univariate polynomial of degree 32 [36], Tsai and Morgan's [108] solution by continuation was considered to be the first robust algorithm for solving the problem across all special cases. Their work established that the problem has at most 16 isolated solutions and the ASME recognized its importance to the robotics community by awarding them the Melville Medal. Although Tsai was a student of Roth and therefore knew about the earlier bootstrap method, the Tsai-Morgan paper was the one which alerted the wider kinematics community that continuation is a powerful technique for solving their problems. This message was further cemented in Morgan's book [77], which made continuation accessible to the engineering community.

Around this same time, Morgan teamed up with one of us (Sommese) to incorporate ideas from algebraic geometry into the formation of homotopies for continuation. This allowed the field to progress beyond the total degree homotopy to a multihomogeneous formulation that in many instances reduces the number of continuation paths required [70]. Another major advance was the coefficient-parameter technique [72] (see §10.1), which justifies homotopies in the parameter space of a family of problems. Once a single generic problem in the family has been solved, using the total degree homotopy, a multihomogeneous homotopy, or any other method that gives all isolated solutions, coefficient-parameter homotopy solves all subsequent problems in the same family with the number of continuation paths equal to the generic root count of the family. This idea lies behind the justification of all continuation methods. These works were at least in part motivated as improvements to the application of continuation to the 6R problem.

To test the power of the new techniques, Morgan, Sommese, and Wampler went back to the nine-point synthesis problem that Roth had worked on nearly twenty years earlier. Instead of the $7^8 = 5,764,801$ paths that would be required in a total degree homotopy, they used a multihomogeneous homotopy and symmetry to reduce the number of paths to be tracked to only 143,360. In the end, the problem was found to have generically 1442 triplets of solutions, obeying a three-way symmetry discovered by Roberts through an indirect algebraic argument in 1875 [83] and first explained by a geometric construction due to Cayley [23]. For any subsequent nine points, the problem can be solved using a parameter homotopy using just 1442 paths.

Some other notable applications of continuation to kinematics include Raghavan's early demonstration that the forward kinematics problem for general Stewart-Gough parallel-link platforms has 40 roots [81]. (See § 3.3.2.) Su, McCarthy, and Watson solved synthesis problems for all of the basic two degree of freedom spatial mechanisms [105].

The synergy between kinematics and numerical algebraic geometry continues today. Problems from kinematics continue to serve as test cases for further developments in algorithms, while those algorithms are increasingly becoming a standard tool for kinematics research.

Part II

Numerical Algebraic Geometry

The fundamental problem in numerical algebraic geometry is to numerically compute and manipulate the solution set of a system of a system of polynomials

$$f(x) := \begin{bmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{bmatrix} \tag{60}$$

where $x = (x_1, \ldots, x_N) \in \mathbb{C}^N$.

In this survey we have seen that many kinematics questions reduce to questions about systems of polynomial equations. Systems of polynomials naturally arise in many other engineering and science disciplines, e.g., chemical systems, game theory, and the discretization of differential equations. Algebraic geometry, the mathematical discipline that investigates systems of polynomials, concerns itself not only with polynomials with real and complex coefficients, but polynomials with coefficients from arbitrary commutative rings. Algebraic geometry over the complex numbers, which is classically called transcendental algebraic geometry, is special because the powerful tools of topology, complex analysis, and differential equations are applicable and yield deep results about systems of polynomials. In fact, the development of these tools drew motivation in large part out of transcendental algebraic geometry. Furthermore, many of the tools for algebraic geometry over general commutative rings have been developed by analogy with the geometric results developed for complex algebraic geometry. The book [45] and [43] are excellent introductions for abstract algebraic geometry and transcendental algebraic geometry respectively.

Different approaches to algebraic geometry engender different computational tools. Corresponding to the abstract approach to algebraic geometry is the symbolic approach (including in particular resultants and Gröbner bases). Corresponding to transcendental algebraic geometry is numerical algebraic geometry.

Historically, within numerical algebraic geometry, the problem of finding isolated solutions for *square systems*, i.e., systems such as Eq. 60 in the case $n = N$, came first. The most basic tool is homotopy continuation (or continuation for short), which consists of studying a square system of polynomials $f(x)$ by starting with a simpler system $g(x)$ that we know how to solve and deforming $g(x)$ and the solutions of $g(x) = 0$ to $f(x)$ and the solutions of $f(x) = 0$.

The book [77] is an excellent explication of the numerical solution of square systems by continuation as practiced right at the end of the period where the main tools were from differential topology with few ideas from algebraic geometry. The references [63, 99] are good references for the numerical approach to solving systems of polynomials up to the end of 2004.

The solution of non-square systems ($n \neq N$ in Eq. 60) came considerably later than methods for square systems. The techniques we employ always reformulate such problems to reduce them once again to finding isolated solutions.

At first sight, the case of $n > N$ (more equations than unknowns) would seem to be numerically unstable, as a small perturbation of the system can obliterate a solution. Even for square systems, singular solutions and near-singular solutions raise serious questions about stability with respect to perturbations in the parameters of the homotopy.

To address concerns of numerical stability, our approach to the numerical solution of algebraic problems is based on two intertwined principles:

1. extra digits may be used as needed; and

2. the formulation of the problem is exact.

We dedicate the next few paragraphs to explaining these principles

Though the first assumption will be discussed in further below (see §6.5), we would like to discuss next several situations where mere double precision does not suffice. Consequently, to obtain reliable results, enough extra digits must be used to ensure that computational processes in floating point accurately reflect the underlying principles of algebraic geometry.

The first situation concerns the concepts of *generic*, *general*, and *probability one*. The numerical approach to the solution of polynomial systems relies heavily on these notions. Extended discussions of these concepts maybe found in [89] and [99, Chap. 4]. The fundamental idea is that for algebraic problems depending on parameters, with variables and parameters taken over the complex numbers, there is usually a dense open subset of the parameter space on which some sort of generic behavior occurs. For example, for most choices of coefficients, two polynomials of degree d_1 and d_2 on $\mathbb{C}^2$ have exactly $d_1 d_2$ nonsingular isolated solutions. We often can use genericity, which in practice is implemented using a random number generator, to very quickly check something that would otherwise be computationally expensive. For example, let $X \subset \mathbb{C}^N$

be a solution set of a polynomial system. Assume X is irreducible, i.e., that the manifold points of X are connected. Let $p(x)$ denote a polynomial on $\mathbb{C}^N$. Either $p(x)$ is identically zero on X or else the set $B = \mathcal{V}(p) \cap X$ is very thin. In the latter case, B is contained in a set of real dimension two less than that of X, and in particular, it is of measure zero. Thus, with probability one, one can check if $p(x)$ is identically zero on X by evaluating $p(x)$ at a random point of X. The problem is that even if a point x^* is a solution of $p(x)$ accurate to a large number of decimal points, $p(x^*)$ may be far from zero, e.g., see [99, §5.3]. One needs to evaluate $p(x)$ with enough digits of precision to reliably decide if $p(x^*)$ is zero.

Another basic example of this comes up in pathtracking. The gamma trick [71], which is discussed briefly in § 6.1, is a probability-one method of guaranteeing that different paths in continuation do not cross. In practice, two paths, say A and B, may come so close to crossing that the pathtracking algorithm jumps from A to B. This is often not a problem for finding isolated solutions, since any permutation of the solution list is acceptable (for example, a pair of jumps, one from path A to path B and one from path B to path A, still gives all endpoints). However for modern efficient algorithms, this does not suffice: the algorithms need to be sure which start point leads to which endpoint. Even for finding isolated solutions, a pathtracker that is highly resistant to path jumping minimizes the need to check and correct for such faults. Consider the nine-point path-synthesis problem for four-bars mentioned in § 3.2. The homotopy used to solve this problem in [113] requires $143,360$ paths to be tracked, of which all but 4326 end on various degenerate sets. The 4326 roots of interest appear in a three-way symmetry known at the outset. The original computations on this problem were conducted in double precision followed by a check for any points missing from the expected symmetry groups. Reruns in extended precision cleaned up any paths having questionable numerical stability, filling in the missing points and thereby establishing with high confidence that the solution list was complete. Experiments with a more sophisticated pathtracker having adaptive multiprecision found that in order to be tracked accurately, 0.83% of the paths required precision higher than double precision somewhere in the middle of the paths before returning to double precision (see [8, §5.3]). This approach consistently finds the entire solution set without requiring any reruns or other corrective actions. Although in the early 1990s, this was an extraordinarily difficult computation, we now use this problem as a moderately difficult test system.

Now, let us turn our attention to the second principle, the assumption that the formulation of the problem is exact. At first sight, this would seem to run counter to the approximate nature of problems from engineering and science. However, for the majority of polynomial systems that arise in actual problems, the coefficients are small integers when the natural parameters are incorporated into the systems. The uncertainties are in the parameter values not in the coefficients. Consider a system of polynomials

$$f(x,q) := \begin{bmatrix} f_1(x,q) \\ \vdots \\ f_n(x,q) \end{bmatrix} \tag{61}$$

where $x = (x_1, \ldots, x_N, q_1, \ldots, q_M) \in \mathbb{C}^{N+M}$. We regard this as a family of polynomial systems on $\mathbb{C}^N$ with $q \in \mathbb{C}^M$ regarded as parameters: completely general complex analytic parameter spaces are easily fit within this framework. We have already seen in the motivating kinematics examples of Part I that in addition to complex Euclidean space, $\mathbb{C}^M$, other typical parameter spaces from that field are $SE(3)$ and the unit circle, for which the defining polynomials have coefficients of ± 1. We may not know the twist angle of a link exactly, but we do know that its cosine/sine pair must lie on the unit circle.

When discussing parameter spaces, it is useful to have the concept of a Zariski open set. By a Zariski open set of $\mathbb{C}^M$ we mean a set that is the complement of a subset defined by the vanishing of a system of polynomials on $\mathbb{C}^M$. A Zariski open set of $\mathbb{C}^M$ is (except for the case when it is empty) dense and path connected. The same is true for a Zariski open set of any irreducible algebraic set, such as a parameter space Q associated to a mechanism family in Part I.

For any parameterized family of systems as in Eq. 61, there is a Zariski open set $\mathcal{U} \subset \mathbb{C}^M$ of points q^* such that the solution sets of $f(x, q^*) = 0$ are very similar, see [72] and [99, §A14]. For example, the systems $f(x, q^*) = 0$ have the same number of isolated solutions with the same multiplicities and the same number of positive dimensional components with corresponding multiplicities and degrees. In essence, these are all structural properties that have integer values and must remain constant unless some extra algebraic conditions are met. In particular, if parameter values $q^* \in \mathcal{U}$ are changed slightly, a multiple solution moves just slightly to a solution of the same multiplicity. On the complement of $\mathcal{U}$, that is, on the algebraic set, say A, whose complement defines $\mathcal{U} = \mathbb{C}^M \setminus A$, the integers may change in specific ways. For example, along a path in $\mathcal{U}$ that terminates at a point in A, several isolated solutions might merge to a single isolated solution whose multiplicity is the sum of those for the incoming solutions. Now, A is a new parameter space and the structural properties (isolated root count, etc.) remain constant on a Zariski open set of A.

To make ideas concrete, consider the inverse kinematics problem for 6R serial-link robots from § 3.1.2, formulated into the kinematics nutshell schema in § 4.5.2. From this starting point, one can derive a square system of equations, say $f(x; y, q) = 0$, whose solutions are sought as in [108] or [71]. In the nutshell schema, the problem is to compute $\hat{K}^{-1}(y, q)$, $y \in SE(3)$, $q \in Q$, so the parameter space of the problem is $SE(3) \times Q$, where $y \in SE(3)$ is the desired location of the robot hand and where the definition of Q includes the unit circle relations associated to the link twists, Eq. 55. If one lets the hand orientation be a general 3×3 matrix instead of a member of $SO(3)$ and lets the cosine/sine pairs (α_j, β_j) for the twist angles range over all of $\mathbb{C}^2$ instead of the unit circle, one creates a new family of systems with a Euclidean parameter space that contains the family of 6R inverse kinematics problems. This larger family, whose equations are still $f(x; y, q) = 0$ but with (y, q) in Euclidean space instead of restricted to $SE(3) \times Q$, has a bigger root count than the inverse kinematics problem. As the parameters approach $SE(3) \times Q$, some solutions diverge to infinity. (We will mention in § 6.2 how one may compute the endpoints of such paths even as they diverge.)

When one tries to solve the 6R inverse kinematics problem numerically, one must use finite precision, and so numerical round-off almost inevitably perturbs the parameters off of $SE(3) \times Q$ into the ambient Euclidean space. Standard methods of numerical analysis can be used to evaluate how far the solutions of the inverse kinematics problem will be moved from their true values by the parameter perturbations. What might seem more problematic however is that the perturbations also make extra solutions appear—the ones that would diverge to infinity as the perturbations away from the unit circle go to zero. If improperly interpreted, one might erroneously attribute too large a root count to the problem.

The saving grace is that we know the conditions for the true parameter space exactly, and we can compute using values for the parameters as close as we wish to it. It is just a matter of using enough digits of precision. Increasing the precision of both the parameters and the endpoint computations pushes the extra solutions ever nearer to infinity, providing clear evidence that they will be at infinity for the nearby exact problem.

If one does not take the knowledge of the parameter space into account, one might indeed come to wrong conclusions. In particular, an incautious person might take a version of the problem in which coefficients have been computed (or printed) in limited precision as being a true member of the kinematics problem. No matter how high a precision is applied subsequently in solving the problem, the parameter perturbations have already been locked in and the extra solutions persist without approaching infinity. This clouds the correct interpretation of the solution set. By properly applying the knowledge of the true parameter space, one can avoid this pitfall. For example, when solving the inverse kinematics problem for an actual 6R robot, one cannot know the twist angles of the links exactly, but one does know that whatever these angles may be, the associated cosine/sine pairs fall exactly on the unit circle. Similarly, if one takes an example "IPP" problem from [71] ("inverse position problem" is another moniker for the 6R inverse kinematics problem) using a hand orientation as printed there to only single precision, one gets extra solutions that would disappear to infinity if one corrected the data to make it lie on $SE(3)$. For a very concrete example of this, one may compare solutions obtained from the version of the IPP problem posted at www.math.uic.edu/∼jan/, which uses the data from [71] without correction, to the solutions obtained from the version of the problem posted at www.nd.edu/∼sommese/bertini/ExampleSystems.html, which with a bit of overkill has coefficients computed from parameters that lie on the requisite parameter space to within 300 digits of precision.

As an aside, we note that the fact that systems with parameters often have small integer coefficients makes it possible to use exotic characteristic p methods to solve some real world problems, e.g., the construction in [41] of a family of exceptional Stewart-Gough mechanisms.

With our two basic assumptions in place, we are ready to survey the techniques that make up the field of numerical algebraic geometry. In this survey, we follow the historical order of developments.

We start with a quick review of the basic mechanics of continuation for finding isolated solutions of square systems in § 6. This starts with an introduction to forming homotopies in § 6.1 and a review of how to track solution paths in § 6.2. For nonsingular roots, basic pathtracking methods follow all the way to the end of the homotopy paths and report the solution there. For singular solutions, it is beneficial to bring in a deeper understanding of multiplicities and to introduce the notion of deflation, as discussed in § 6.3. This leads to special methods, called endgames, for computing path endpoints to obtain both the nonsingular and the singular solutions (§ 6.4). Path tracking and endgames are often impossible to carry out with only double precision, especially when systems become large or the degrees of the equations become high. Adaptive precision, an important advance that has happened since 2004, addresses this problem (§ 6.5).

In §7, we discuss the computation of positive dimensional sets. We begin in §7.1 by discussing how the solution set of a polynomial system compares to the solution set of a randomization of the polynomial system. In §7.2, we discuss the behavior of the solution set of a polynomial system under intersections with general affine linear subspaces of the ambient Euclidean space. These results are important because the data structure we use to represent an irreducible component

X of the solution set of $\mathcal{V}(f)$ of a polynomial system f is a triple (f, W, L), where L is a random affine linear subspace of dimension $N - \dim X$ and $W = X \cap L$. In §7.3, we discuss membership tests that decide which irreducible component of $\mathcal{V}(f)$, a given solution point of $f(x) = 0$ lies on. In §7.4, we discuss how the deflation operation of §6.3 applies to irreducible component: often this operation allows us to reduce to the case of a multiplicity-one component. In §7.5, we discuss the numerical irreducible decomposition of $\mathcal{V}(f)$. This decomposition of $\mathcal{V}(f)$ is often the starting point in applications.

6 Finding Isolated Roots

In this section we briefly discuss the continuation method of finding isolated solutions of a polynomial system

$$f(x) := \begin{bmatrix} f_1(x) \\ \vdots \\ f_N(x) \end{bmatrix}$$

where $x = (x_1, \ldots, x_N) \in \mathbb{C}^N$. We refer to [63, 99] for more details.

6.1 Homotopy

By a homotopy for $f(x) = 0$, we mean a system of polynomials

$$H(x,t) := \begin{bmatrix} H_1(x,t) \\ \vdots \\ H_N(x,t) \end{bmatrix} \tag{62}$$

in $(x_1, \ldots, x_N, t) \in \mathbb{C}^{N+1}$ with $H(x, 0) = f(x)$.

One of the most classical homotopies is the total-degree homotopy

$$H(x,t) := (1-t) \begin{bmatrix} f_1(x) \\ \vdots \\ f_N(x) \end{bmatrix} + t \begin{bmatrix} g_1(x) \\ \vdots \\ g_N(x) \end{bmatrix} \tag{63}$$

where g_j is a polynomial of degree $\deg g_j = \deg f_j = d_j$. For example, g_j might be $x_j^{d_j} - 1$.

Assume that we have an isolated solution y^* of $H(x, 1) = 0$. Imagine for a moment that there is a function $x(t) : (0, 1] \to \mathbb{C}^N$ such that

1. $x(1) = y^*$;
2. $H(x(t), t) = 0$;
3. The Jacobian $J_x H(x, t)$ of $H(x, t)$ with respect to the variables x is of maximal rank at $(x(t), t)$ for $t \in (0, 1]$.

If this happens then the path satisfies the Davidenko differential equation

$$\sum_{j=1}^{N} \frac{\partial H(x,t)}{\partial x_j}\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} + \frac{\partial H(x,t)}{\partial t} = 0,$$

and we say that we have a *good* path starting at y^*.

Given the initial condition $x(1) = y^*$, we can solve the differential equation numerically. A typical approach is to move an appropriate predictor step in the t variable and then use Newton's method on $H(x,t) = 0$ in the x variables to make a correction step. This sort of procedure, which is called pathtracking, has been deeply examined. Given the tracking of the path, some estimation procedure may be attempted to compute a solution

$$x^* := \lim_{t\to 0} H(x(t),t)$$

of $f(x) = 0$.

Several things can go wrong:

1. there might be no such path $x(t)$ for the values $t \in (0,1]$;
2. the path $x(t)$ might exist, but the Jacobian $J_x H(x,t)$ might not be of maximal rank at all points $(x(t),t)$ for $t \in (0,1]$; and
3. it might we be that $\lim\limits_{t\to 0} H(x(t),t)$ is not finite.

Bézout's Theorem says that given general polynomials g_j of degree d_j, the solution set of the system $(g_1(x),\ldots,g_N(x)) = 0$ consists of $d_1 \cdots d_N$ nonsingular isolated solutions. A classical result [77] is that the total-degree homotopy starting at a system $g(x)$, with the g_j being sufficiently general degree d_j polynomials, has good paths for the $d_1 \cdots d_N$ nonsingular isolated solutions of $(g_1(x),\ldots,g_N(x)) = 0$ and that the finite limits $\lim\limits_{t\to 0} H(x(t),t)$ of the $d_1 \cdots d_N$ paths contain all the isolated solutions of $f(x) = 0$, though they often contain as well some points lying on positive-dimensional components of the solution set of $f = 0$. Finding the exact set of isolated solutions requires considerably deeper theory: the first algorithm to do this is given in [91].

The most general polynomials of degree d_j in N variables can have a large number of terms; to be exact, there are $\binom{N+d_j}{N}$ different possible monomials. The γ trick of [71] allows one to define a good homotopy without introducing that level of complexity. Suppose that polynomials g_j of degree d_j are such that the solution set of the system $(g_1(x),\ldots,g_N(x)) = 0$ consists of $d_1 \cdots d_N$ nonsingular isolated solutions. Then the γ trick states that with probability one, the homotopy

$$H(x,t) := (1-t)\begin{bmatrix} f_1(x) \\ \vdots \\ f_N(x) \end{bmatrix} + \gamma t \begin{bmatrix} g_1(x) \\ \vdots \\ g_N(x) \end{bmatrix} \tag{64}$$

with γ a random complex number satisfies the properties:

1. $\big\{(x,t)\big| t \in (0,1]; x \in \mathbb{C}^N; H(x,t) = 0\big\}$ is a union of $d_1 \ldots d_N$ good paths $x_1(t),\ldots,x_{d_1\cdots d_N}(t)$ starting at the solutions of $H(x,1) = 0$; and

2. the set of limits $\lim_{t \to 0} x_j(t)$ that are finite include all the isolated solutions of $H(x, 0) = 0$.

This theory justifies the use of the very simple start system defined by $g_j(x) = x_j^{d_j} - 1$, $j = 1, \ldots, N$.

If we have an isolated solution x^* of $H(x, 0) = 0$ and if

1. there is at least one good path with x^* as a limit as $t \to 0$; and

2. every path over $(0, 1]$ with limit x^* as $t \to 0$ is good,

then we say that $H(x, t)$ is a *good homotopy* for x^*. As will be discussed in § 6.3, general theory tells us that if $H(x, t)$ is a good homotopy for a solution x^* of $H(x, 0) = 0$, then the number of paths going to x^* equals the multiplicity of x^* as a solution of $H(x, 0) = 0$.

Constructing good homotopies with the number of paths not too different from the number of isolated solutions of f was an important research topic at the end of the 20th century. There is detailed discussion of this topic in [99, Chap. 8]. For systems that are not too large, which includes many mechanism systems, polyhedral homotopies are excellent (see [64] for a report on this by the inventor of the leading algorithm for this method). For larger systems, the regeneration approach discussed in § 10.2 is more favorable [46, 47].

For a hint of what difference the selection of homotopy can make, consider the 3-RPR forward kinematics problem of § 3.3.1, which is equivalent to intersecting a four-bar coupler curve with a circle, as shown in Figure 6. We saw at Eq. 23 that a four-bar coupler curve $f_{cc}(p, \bar{p}, q) = 0$ is degree six in $(p, \bar{p})$ but only degree three in p or $\bar{p}$ considered independently. That is, it is a bi-cubic sextic. On the other hand, the circle is degree two, but bi-linear in p and $\bar{p}$, see Eq. 30. Accordingly, for this problem, a total degree homotopy has $6 \cdot 2 = 12$ paths, while a two-homogeneous homotopy has only 6 paths, the same as the generic number of solutions. (Multihomogeneous homotopies were first proposed in [70] and discussed in [99, § 8.4].) For a given system, a polyhedral homotopy will never have more paths than the best possible multihomogeneous one. In some cases, the differences can be extreme. For example, the generalized eigenvalue problem $(A + \lambda B)\mathbf{v} = 0$ for given $N \times N$ matrices A and B has total degree 2^N but a two-homogeneous root count of just N. Due to the existence of highly efficient software specialized for eigenvalue problems, one wouldn't use homotopy to solve these, but the root count is illustrative of why much effort has gone into finding efficient formulations of good homotopies.

6.2 Path Tracking

Path tracking for not-necessarily polynomial systems is highly developed (see [1, 2, 3]). An overarching advantage of homotopy methods is that each path may be followed on a different node of a computer cluster, leading to highly parallel algorithms well-designed for modern supercomputers.

Not all algorithms used in continuation are inherently parallel. The lack of parallelism is also a major mark against polyhedral methods. Indeed, they so far have not parallelized well: for systems of moderate size, the computation of the polyhedral start system (done of necessity on a single node) dwarfs the actual pathtracking. Equation-by-equation methods on the other, not only surpass polyhedral methods on a single core for moderate systems, but are highly parallel (see §8 and §10.2).

As will be discussed below, multiple paths go to singular solutions. Since singular solutions are considerably more expensive than nonsingular isolated solutions, it is lucky that the Cauchy endgame allows us to only have to do an endgame for a fraction of the paths going into the singular solution (see §6.4).

Polynomial systems are much better behaved than general systems. As mentioned above, the γ trick ensures that paths of a homotopy are well behaved. Even so, there are numerical difficulties if the path gets near a branchpoint. Bertini [13] circumvents these difficulties by means of adaptive multiprecision [7, 8] (see § 6.5).

Another difficulty are paths going to infinity. Tracking them may be computationally expensive as the path in $\mathbb{C}^N$ is infinitely long and numerical conditioning may be poor as the magnitudes of the solution variables grow. Morgan's projective transformation trick [69] is to work on a random coordinate patch in the projective space containing $\mathbb{C}^N$. This means that instead of solving a given system $f(x) = (f_1(x), \ldots, f_N(x))$, of polynomials in $x = (x_1, \ldots, x_N) \in \mathbb{C}^N$, one shifts to solving the related system on $\mathbb{C}^{N+1}$

$$\begin{bmatrix} \widehat{f}_1(\widehat{x}_0, \ldots, \widehat{x}_N) \\ \vdots \\ \widehat{f}_N(\widehat{x}_0, \ldots, \widehat{x}_N) \\ a_0\widehat{x}_0 + \cdots a_N\widehat{x}_N - 1 \end{bmatrix} = 0 \tag{65}$$

where

$$\widehat{f}_j(\widehat{x}_0, \ldots, \widehat{x}_N) = \widehat{x}_0^{\deg f_j} f_j \left(\frac{\widehat{x}_1}{\widehat{x}_0}, \ldots, \frac{\widehat{x}_N}{\widehat{x}_0} \right)$$

is the homogenization of f_j and $(a_0, \ldots, a_N)$ are random constants (near 1 in absolute value). Note that the solutions $(\widehat{x}_0^*, \ldots, \widehat{x}_N^*)$ of the system Eq. 65 with $\widehat{x}_0^* \neq 0$ correspond to the finite solutions $(\widehat{x}_1^*/\widehat{x}_0^*, \ldots, \widehat{x}_N^*/\widehat{x}_0^*)$ of $f = 0$.

The random coordinate patch trick works generally: there is a straightforward prescription [10] for how to write down the random coordinate patch system for homogeneous compactifications of $\mathbb{C}^N$, e.g., products of Grassmannians.

6.3 Multiplicities and Deflation

For polynomials in one variable, the multiplicity of a solution z^* of a polynomial $p(z) = 0$ is easy to understand. It is the positive integer μ such that $p(z) = (z - z^*)^\mu g(z)$ with $g(z)$ a polynomial not zero at z^*. Of course, numerically the question of whether a polynomial is zero at a point is a real issue, but this is discussed in § 6.5. In several variables, multiplicity still makes sense, but it is a much weaker measure of the local structure of a solution of a polynomial system than multiplicity is in one dimension. In this subsection we consider only isolated solutions of a polynomial system. The much more difficult question of how to decide if a solution is isolated is put off to §9.2.

The definition of multiplicity for a system of polynomials in several variables is more technical than in the single-variable case. From the definition above of multiplicity in the single variable case, one can see that derivatives at a solution z^* of multiplicity μ vanish up to order $\mu - 1$, i.e., $d^j p(z^*)/dz^j = 0$, $j = 0, 1, \ldots, \mu - 1$. The somewhat technical way to state the equivalent property in the multivariate case is as follows. Let z^* be an isolated solution of a polynomial system f as in Eq. 60. Let $\mathcal{I}_{z^*}$ denote the ideal generated by $f_1, \ldots, f_n$ in the ring $\mathcal{O}_{z^*}$ of convergent power series in variables $z_1, \ldots, z_N$ centered at z^*. The multiplicity $\mu(f, z^*)$ of z^* as a solution of the system f is defined as the dimension of

$$A := \mathcal{O}_{z^*}/\mathcal{I}_{z^*}$$

considered as a complex vector space. Note that A contains much more information than $\mu(f, z^*)$. Indeed, taking into account the vanishing derivatives, the polynomial system vanishes on an infinitesimal "chunk" of the $\mathbb{C}^N$. As a set this "chunk" consists of just the single point z^*, but the fine structure of this point is defined by A. One may regard A as the algebraic functions defined an infinitesimal neighborhood of z^*. The local ring A is is the nonreduced structure of the algebraic set defined by $\mathcal{I}_{z^*}$ at z^*. If $n = N$ and we computed z^* using a good homotopy in the sense of § 6.2, then the number of paths ending in z^* equals the multiplicity, e.g., [99, Theorem A14.11].

When the multiplicity μ of solution z^* is greater than one, z^* is said to be a singular solution. Such solutions are difficult to work with numerically. A primary problem is that the vanishing derivatives ruin the convergence properties of Newton's method near the singular point. For this reason, tracking paths to z^* from a good homotopy for z^* is computationally expensive and often impossible in double precision. To deal with these points effectively, we use endgames (see §6.4) and adaptive precision (see §6.5).

Deflation is another approach for dealing with singular points [80, 79, 61, 62, 46]. Consider that in the univariate case, z^* is a nonsingular root of $d^{\mu-1}p(z)/dz^{\mu-1} = 0$. If we can determine the correct multiplicity, we can restore the quadratic convergence of Newton's method by searching for zeros of this derivative instead of zeros of p. Deflation is a generalization of this maneuver to the multivariate case. For reasons we discuss at the end of this subsection, this approach is not very beneficial for treating isolated singular solutions, but it turns out to be important for components, as we shall discuss later in § 7.4. To prepare for that eventuality, we discuss deflation for isolated solutions next.

Let z^* denote an isolated solution of a system $f(z) = 0$ of N polynomials in the N variables $z = (z_1, \ldots, z_N)$. Let $F(z, \xi)$ denote a system of $N + k$ polynomials in the variables z and $\xi = (\xi_1, \ldots, \xi_k)$. $F(z, \xi) = 0$ is called a deflation of $f(z) = 0$ at z^* if there is a nonsingular isolated solution (z^*, ξ^*) of $F(z, \xi) = 0$. The usual deflations are linear in the variables ξ.

Leykin, Verschelde, and Zhao [61] presented a procedure to find a deflation of $f(z) = 0$ at a multiplicity μ isolated solution z^* that terminates in less than μ steps. A step in the procedure consists of replacing $f(z) = 0$ with

$$\begin{bmatrix} f(z) \\ Df(z) \cdot \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_N \end{bmatrix} \\ \begin{bmatrix} I_k & A \end{bmatrix} \cdot \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_N \end{bmatrix} - \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \end{bmatrix} = 0,$$

where Df is the Jacobian matrix of $f(z)$, $N-k$ is the rank of Df evaluated at z^*, I_k is the $k\times k$ identity matrix, and the A is a random $k\times N$ complex matrix. Dayton and Zeng [29] made a significant improvement in the understanding of how many steps it requires for the procedure to terminate.

The main difficulty with this procedure lies in determining the rank of Df evaluated at z^*. For a specific root z^* one may investigate and make good guesses, but a polynomial system of interest may have millions or even billions of solutions. In the face of this, one needs to have a highly reliable, automated procedure to determine the rank. The trouble is that the rank of $Df(z)$ is falling as z approaches z^*, so one needs an accurate value of z^* to get the correct rank of $Df(z^*)$. This leads to a vicious circle, since computing z^* accurately is the initial objective anyway. The upshot is that for isolated solution points the cost of computing a deflation system dwarfs the cost of computing the point accurately using the endgame methods in the next subsection.

6.4 Endgames

Let $H(x,t)=0$ be a homotopy as in Eq. 62 and let $z(t)$ with $t\in(0,t]$ be a path as in §6.2 satisfying $H(z(t),t)=0$. Endgames refer to the process of computing $x^*:=\lim_{t\to 0} z(t)$. Assume (as we may, by using Morgan's projective transformation trick, Eq. 65), that x^* is finite, i.e., $x^*\in\mathbb{C}^N$.

The key to endgames is to realize that regarding t as a complex variable, there is an algebraic curve $\mathcal{C}\subset\mathbb{C}^N\times\mathbb{C}$ such that $(z(t),t)\in\mathcal{C}$ for all $t\in(0,1]$. Note $\mathcal{C}$ is an irreducible component of $\mathcal{V}(H)$ and thus by the uniformization theorem [99, Thm. A.2.2], there is a closed disc

$$\Delta(0,\delta):=\{s\in\mathbb{C}\mid |s|\le\delta\}$$

of radius δ around the origin and a homomorphic map

$$\phi:\Delta\to\mathbb{C},$$

such that

1. $\phi(0)=x^*$; and
2. ϕ is an isomorphism restricted to the punctured disc $\Delta^*(0,\delta):=\Delta(0,\delta)\setminus 0$.

By reparameterizing, we can assume that $s^c=t$ for a positive integer c, called the *winding number* of the path $z(t)$. We set $\epsilon=\delta^c$.

Let $\pi:\mathbb{C}^N\times\mathbb{C}\to\mathbb{C}$ be the projection $(x,t)\to t$.

Note that c is at least 1. Theory tells us that in a small enough open ball B around $(x^*,0)$, the one dimensional components of $\mathcal{V}(H)\cap B$ that surject onto $\mathbb{C}$ under π have winding numbers adding up to the multiplicity of x^* as a solution of $H(x,0)=0$. Note that the coordinate functions of ϕ expanded in power series in fractional powers of t are classically called Puiseux series.

The first endgames using the structure inherent in the above uniformization were in the articles [73, 74, 75], where the term endgame was given for such algorithms. Of the three related endgames from those articles, we only discuss the Cauchy endgame [73], which has certain special properties making it stand above the others.

By Cauchy's integral theorem

$$x^* = \frac{1}{2\pi i}\oint_{|s|=\delta} \frac{z(s)}{s}\mathrm{d}s,$$

or in terms of $s^c = t = \epsilon e^{i\theta}$

$$x^* = \frac{1}{2\pi c}\int_0^{2\pi c} z\left(\epsilon^{1/c}e^{i\theta/c}\right)\mathrm{d}\theta.$$

Note this last integral can easily and accurately be computed using continuation on $t = \epsilon e^{i\theta}$, going around the circle c times. One does not need to know c ahead of time, because it is the lowest integer such that along the continuation path $z(\epsilon e^{2\pi i c}) = z(\epsilon)$, that is, one continues around the circle until $z(t)$ repeats its initial value. There might be a worry that one could have numerical trouble deciding when the repeat has occurred, but this is not a problem. Indeed the distances between the values of $z(t)$ over ϵ are of the order $\sqrt[c]{\epsilon}$.

There are several difficulties with the Cauchy endgame that will prevent it (or any endgame) from working if c is large enough and only double precision is available. Indeed for double precision, even $c = 4$ can be numerically challenging. Happily these difficulties disappear when we have adaptive precision.

To understand the difficulty with such solutions, consider

$$H(x,t) = \begin{bmatrix} x_1 - t \\ x_2^6 - t \end{bmatrix} = 0$$

and the path $z(t) = (t, t^{\frac{1}{6}})$. At $t = 10^{-6}$, $(x_1, x_2) = (10^{-6}, 10^{-1})$ is quite far from $(0,0) = \lim_{t\to 0} z(t)$. Putting aside the cost of tracking $z(t)$ to still smaller values of t, we have the condition number of the D_xH is of the order of 10^5. By Wilkinson's Theorem, we may potentially be losing 5 digits of accuracy. The closer we go to $t = 0$, the larger the condition number and the more digits we lose. Thus we need to stay far from $t = 0$.

But we might not be able to stay as far from $t = 0$ as we would like! There may be multiple branch points near 0, i.e., the ϵ for which the uniformization theorem holds may need to be chosen very small. Thus we have two competing demands. The ϵ needs to be chosen small enough so that the uniformization theorem holds, but not so small that ill-conditioning of D_xH precludes pathtracking. The region where this is possible is called the *endgame operating region.* Unfortunately, that region may be empty in double precision. However, using adaptive precision, this region always exists, because one can carry enough digits to overcome the ill conditioning. Paying attention to how the values of the path should behave when t is in the region where the uniformization theorem holds yields heuristics that signal when one is in the endgame operating region.

A big advantage of the Cauchy endgame is that it parallelizes nicely [11]. Indeed, each time we complete a pathtracking loop around the small circle, we find another point over $t = \epsilon$, giving c points in all. Tracking these back to $t = 1$, we find which starting points of the homotopy lead to the same endpoint. Having already calculated that endpoint, we need not do any further computations, and thereby save running an endgame for $c-1$ other startpoints. Each trip around the endgame circle can cost on the same order as tracking from $t = 1$ to the endgame region, so as c grows, the endgame begins to dominate the computation. By running the endgame on only one of c startpoints, the computation is reduced considerably, and the net cost for the group of c paths is approximately the same as if they each had cycle number 1.

6.5 Adaptive Precision

Using extra digits of accuracy solves many problems, but comes with real costs. The first cost is the programming effort to implement high precision arithmetic on machines with different architectures. Happily, two packages the "GNU MPFR Library" (www.mpfr.org) and the "GNU Multiple Precision Arithmetic Library" (gmplib.org), which come with standard Linux distributions, remove the need to develop high-precision arithmetic on different machines and make it straightforward to port software (such as Bertini [13]) to different machines. The price for this standardization is that the high-precision arithmetic makes little use of the hardware double precision arithmetic, so there is a significant cost to using it. Even if the computational cost of high precision matched the theoretical cost—$O(n \ln n)$, where n is the number of digits—the cost would be significant. For these reasons, it is important to use at least as many digits as needed, but not too many more than that.

In homotopy algorithms, the most crucial area where high-precision may be needed is in pathtracking. However, the number of digits needed may fluctuate along a path, so it is desirable to scale the precision up and down in response. In the article [7] an adaptive multiprecision algorithm is introduced that tries to ensure that the Newton corrector part of a pathtracker has enough precision. In [8], adaptive multiprecision and adaptive stepsize are combined under the assumption that the predictor part of the pathtracking uses the Euler ODE method. In [12], modifications are made to combine the algorithms when higher order ODE methods are used for the predictor step.

The basic idea (without the technical details that make it a practical method) recognizes that pathtracking solves linear systems involving the Jacobian matrix in both the ODE predictor and in the Newton corrections. The condition number of the matrix gives an estimate on how many significant digits these processes yield. There must be enough digits to make Newton's method converge within the tolerance set for pathtracking. As the accuracy of the step also depends on the accuracy with which the polynomials are evaluated, the sizes of the coefficients of the polynomial system need to be part of the estimates (see [7, 8]). [As discussed in the introduction to Part II, it also means that the polynomials themselves be given with enough accuracy to match the precision required in their evaluation.] To do all this with enough margin that calculations are secure, but not so much as to make the cost prohibitive, requires a facility for scaling precision up and down on the fly along with fast estimates of how much precision is needed at any time. As noted earlier, using this sort of algorithm to solve the nine-point path-synthesis problem for four-bars, we found that nearly one percent of the paths required extra digits beyond double precision somewhere in the middle of the t-interval $(0, 1]$ before returning back to double precision.

There are other uses of multiprecision besides pathtracking. At times, one may wish to check the vanishing of a polynomial at a given point or test the rank of the Jacobian matrix. It can be useful to do these checks at several different precisions to see if the magnitude of the polynomial or of the putatively zero singular values of the Jacobian matrix actually approach zero in proportion to the precision used. Though no numerical method is absolutely certain, these moves are effective.

Yet another use of multiprecision is to produce points of a witness set (see § 7.2) to a very high number of digits so that various maneuvers to construct exact equations may be carried out [16].

7 Computing Positive-Dimensional Sets

As we saw in §6.1, the polynomial systems to which continuation naturally applies are square, i.e., they have the same number of equations as variables. What can we say about non-square systems? How do we deal with them numerically? The answers to these questions lead quickly to positive dimensional algebraic sets. But before we go any further, we need a few definitions from algebraic geometry.

A complex affine algebraic set is defined to be the solution set $X := \mathcal{V}(f)$ of a system of polynomials as in Eq. 60 on some complex Euclidean set $\mathbb{C}^N$. Unless we say otherwise, algebraic sets in this article are complex affine algebraic sets. Here we drop all multiplicity information, e.g., $\mathcal{V}(x) = \mathcal{V}(x^2)$ where x is the coordinate on the complex line. We say that X is irreducible if the set of smooth points $X_{\rm reg}$ of X is connected. In this case we define $\dim X = \dim X_{\rm reg}$, where we are using complex dimensions unless we say otherwise, which are one-half of the real dimension, e.g., $\mathbb{C}^N$ is N-dimensional. Theory tells us that $X_{\rm reg}$ is dense in X and that a single polynomial $p(x)$ being irreducible, i.e., not factorable into polynomials of strictly lower degrees, is equivalent to $\mathcal{V}(p)$ being irreducible.

Let $X = \mathcal{V}(f)$ for a system of polynomials $f = 0$ on $\mathbb{C}^N$. The irreducible decomposition of X is the decomposition of X into the union of closures of the connected components of $X_{\rm reg}$. Each of these closures is an algebraic set in its own right. We index the i-dimensional irreducible components of this decomposition by X_{ij} for $j \in \mathcal{I}_i$, where $\mathcal{I}_i$ is a possibly empty finite set. Thus the i-dimensional algebraic subset of X is defined to be

$$X_i := \cup_{j\in\mathcal{I}_i} X_{ij},$$

and X is the union of these:

$$X = \cup_{i=0}^N X_i.$$

The irreducible decomposition is the typical starting point in most computations involving algebraic sets. The analogue of the irreducible decomposition is the numerical irreducible decomposition discussed in §7.5.

We say that an algebraic set $X \subset \mathbb{C}^N$ is pure i-dimensional if all the irreducible components in the irreducible decomposition of X are i-dimensional, i.e., $X = X_i$. Note that such an X has a well-defined integer $\deg X$ defined as the number of points in the set $X \cap \mathcal{L}_i$ where $\mathcal{L}_i$ is a generic i-codimensional linear space, in other words, $\mathcal{L}_i$ is a general $(N - i)$-dimensional linear subspace of $\mathbb{C}^N$.

Remark 7.1 We can define a complex quasialgebraic sets more generally as a solution set of a system of homogeneous polynomials on $\mathbb{P}^N$ minus the solution set of a different such system. There is an analogous irreducible decomposition. These concepts are discussed extensively in [99], but for the sake of simplicity we restrict ourselves to affine algebraic sets.

Before we consider positive dimensional solution sets, let's consider nonsquare systems when we want to compute isolated solutions. To this end, fix a system f of n polynomials in N variables. The system is said to be overdetermined if $n > N$ and underdetermined if $n < N$.

If $n \leq N$, then general theory tells us, e.g., [99, Cor. A.4.7] and [99, Cor. A.4.12], that if $\mathcal{V}(f)$ is not empty, then the complex dimension of $\mathcal{V}(f)$ is at least $N - n$. Regarding f as defining a map from $\mathbb{C}^N$ to $\mathbb{C}^n$, then the image is Zariski dense in an algebraic set B of a dimension b such that

1. for y in the complement of an algebraic subset of B of lower dimension than B, $\dim f^{-1}(y) = N - b$; and

2. for all $y \in f(\mathbb{C}^N)$, $\dim f^{-1}(y) \geq N - b$.

In other words, for almost all points in the image of f, the pre-image is a set of dimension $N - b$, but there may be an algebraic subset of the image for which the pre-image has a higher dimension. By Sard's theorem [99, Thm. A.4.10], b equals the rank of the Jacobian of f evaluated at a general point of $\mathbb{C}^N$. The integer b is called the rank of the polynomial system.

Thus if we are interested in isolated solutions, we need that the rank of the polynomial system is N, which requires that $n \geq N$. Suppose that we were interested in isolated solutions of a rank N system with $n > N$ equations. In a numerical approach, where equations are only evaluated approximately, this situation can present some challenges.

Consider the polynomial system

$$\begin{bmatrix} x^2 \\ y^2 \\ xy \end{bmatrix} = 0. \tag{66}$$

This system has a unique root, $(0, 0)$. However, if we change the equations slightly, say we perturb the last one to $xy - \epsilon = 0$, the root disappears. In this sense, the root is unstable under numerical perturbation. We observe that the multiplicity of the root $(0, 0)$ is 3.

One way to stabilize the root is to replace the system in Eq. 66 with the system

$$\begin{bmatrix} x^2 + \alpha xy \\ y^2 + \beta xy \end{bmatrix} = 0, \tag{67}$$

where α and β are random, preferably of magnitude near 1. This new system has a unique root $(0, 0)$ of multiplicity 4. Moreover if the equations in the system Eq. 67 are changed slightly, the isolated solutions of the new system will lie in a small neighborhood of $(0, 0)$, and their multiplicities will add up to 4. Although this maneuver does increase the root's multiplicity, it has the substantial benefit that it stabilizes the root.

Let's consider another simple example:

$$\begin{bmatrix} (x-1)y \\ (x-1) \end{bmatrix} = 0. \tag{68}$$

The solution set consists of the vertical line $x = 1$. This time, if we perturb one of the equations slightly, e.g., if we replace the second equation with $x - 1.00001$, the solution set changes from a one-dimensional set to just the point $(x, y) = (1.00001, 0)$. As in the previous example, the solution set is numerically unstable. One could say that the root cause is the same, because in a sense a system of two equations that defines a one-dimensional solution set in $\mathbb{C}^2$ is overdetermined in the same way a system of three equations is overdetermined for defining a zero-dimensional set (a point). To stabilize the solution, we may take a similar action as before: use randomization. To do so, we form the single equation

$$(x-1)y + \beta(x-1) = 0, \tag{69}$$

with β a random number near 1 in absolute value. The solution set of this new equation consists not just of $x = 1$, but also of an extra component $y = -\beta$. Changing the system in Eq. 68 slightly gives a perturbed version of Eq. 69 whose solution set is a curve lying near the union of these two lines. So the randomization has stabilized the solution of the original system, but it has introduced an extraneous component.

In either of the two preceding examples, it is tempting to find the solution set by solving the system with one equation omitted, and then checking if these preliminary solutions satisfy the extra equation. This works in these examples, but not for the system

$$\begin{bmatrix} (x+y-1)y \\ (x+y-1)x \\ xy \end{bmatrix} = 0. \tag{70}$$

This system has three isolated roots, $\{(0,0),(1,0),(0,1)\}$, but any system formed by two out of three of these equations has just one of these as an isolated root. In contrast, the randomized system

$$\begin{bmatrix} (x+y-1)y + \alpha xy \\ (x+y-1)x + \beta xy \end{bmatrix} = 0, \tag{71}$$

with random α and β has four isolated roots. These consist of the three roots of the original system and one extraneous root that depends on α and β.

Randomization is our approach to stabilizing solutions in any overdetermined situation, and it is key for dealing with positive dimensional sets. Accordingly, we devote the next section to better understanding this procedure.

7.1 Randomization

Fix a system f of n polynomials in N variables. As we saw above, a natural move is to replace f by a new system consisting of k linear combinations of the equations making up f, e.g.,

$$\mathfrak{R}_k(M,f)(x) := M \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \tag{72}$$

where M is a $k \times n$ matrix. For reasons that will be made clear in what follows, we suppress M and write $\mathfrak{R}_k f(x)$ instead of $\mathfrak{R}_k(M,f)(x)$ when M is chosen generically, i.e., when the entries of M are chosen randomly. For numerical stability, it is typically advantageous to choose M close to unitary. We refer to $\mathfrak{R}_k f(x)$ as a randomization of f.

How does $\mathcal{V}(\mathfrak{R}_k(M, f))$ compare with $\mathcal{V}(f)$? Without some restriction on M, all we can say is that

$$\mathcal{V}(f) \subset \mathcal{V}(\mathfrak{R}_k(M, f)).$$

A remarkable consequence of Bertini's theorem [99, Chap. 13.5] is that for most M, the relation is strong and simple.

Theorem 7.2 (A Version of Bertini's Theorem) *Fix a system f of n polynomials in N variables. Fix an integer $k \leq N$. There is a nonempty Zariski open set $\mathcal{U} \subset \mathbb{C}^{k \times n}$, such that for $M \in \mathcal{U}$, every irreducible component of $\mathcal{V}(f)$ of dimension equal to $N - k$ is an irreducible component of $\mathcal{V}(\mathfrak{R}_k(M, f))$; and the irreducible components of $\mathcal{V}(\mathfrak{R}_k(M, f))$ of dimension greater than $N - k$ coincide with the irreducible components of $\mathcal{V}(f)$.*

In other words, a general randomization of f to $k \leq N$ equations preserves all solution components of dimension $N - k$ or higher, but may introduce some extraneous components at dimension $N - k$.

Bertini's theorem implies some additional useful facts concerning general randomizations $\mathcal{V}(\mathfrak{R}_k f(x))$ for $k \leq N$.

- The extraneous solutions $\mathcal{V}(\mathfrak{R}_k f(x)) \setminus \mathcal{V}(f)$ are smooth of dimension $N - k$.
- The multiplicity of an irreducible component of $\mathcal{V}(f)$ of dimension $N - k$ has multiplicity with respect to $f = 0$ no greater than its multiplicity with respect to $\mathfrak{R}_k f = 0$.
- If the multiplicity of an irreducible component of $\mathcal{V}(f)$ of dimension $N - k$ has multiplicity one with respect to $f = 0$, then it has multiplicity one with respect to $\mathfrak{R}_k f = 0$.

Using the fact that a general $k \times k$ matrix is invertible it follows that if $k \leq n$, then we may use M of the form $[I_k \;\; A]$, where I_k is the $k \times k$ identity matrix and A is a random $k \times (n - k)$ matrix, i.e., we may us $\mathfrak{R}_k f(x)$ of the form

$$\begin{bmatrix} I_k & A \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

7.2 Slicing

As we saw from the last section, if we want to investigate the k-dimensional components of $\mathcal{V}(f)$, it is natural to randomize down to $N - k$ equations. Moreover, there will be no k-dimensional components unless we have a system f whose rank is at least $N - k$.

In algebraic geometry, a classical way of understanding a positive dimensional set is to intersect it with linear spaces [17]. This leads us to the approach to positive dimensional solution sets of [89], where we coined the term *numerical algebraic geometry* as the area that corresponds to algebraic geometry in the same way that numerical linear algebra corresponds to linear algebra.

The critical theory underlying the approach is a variant of Bertini's Theorem [99, Chap. 13.2] that covers slicing. In the following theorem, any set of negative dimension is the empty set.

Theorem 7.3 *Let $\mathcal{V}(f)$ be the solution set of a system of n polynomials $\mathbb{C}^N$. Let $\mathcal{L}$ be a generic $N-k$ dimensional affine linear subspace of $\mathbb{C}^N$. Then given any r-dimensional irreducible component X of $\mathcal{V}(f)$, it follows that:*

1. $\dim X \cap \mathcal{L} = r - k$ *and* $\dim \operatorname{Sing}(X \cap \mathcal{L}) = \dim \operatorname{Sing}(X) - k$*; and*
2. *the algebraic set $X \cap \mathcal{L}$ is irreducible if $r > k$.*

Many other properties of X are inherited by $X \cap \mathcal{L}$.

As mentioned in the introduction to §7, the numerical irreducible decomposition is the starting point for many computations with polynomial systems. The numerical irreducible decomposition was first constructed in [91].

Given a k-dimensional irreducible component X_{kj} of $\mathcal{V}(f)$, we define a witness set for X_{kj} to be a triple (f, W_{kj}, L_k) where L_k is a set of k random linear equations and W_{kj} denotes the set of points on X_{kj} satisfying the equations $L_k = 0$. Note that $W_{kj} = X_{kj} \cap \mathcal{V}(L_k)$, thus there are $\deg X_{kj}$ points in the set W_{kj}.

Note that W_{kj} are isolated solutions of the polynomial system

$$\begin{bmatrix} \mathfrak{R}_{N-k}(f) \\ L_k \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} I_{N-k} & A \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \\ L_k \end{bmatrix} = 0, \tag{73}$$

where I_{N-k} is the $(N-k) \times (N-k)$ identity matrix and A is a random $(N-k) \times (n-N+k)$ matrix. Note that the isolated solutions W_{kj} do not have to be of multiplicity one. If they are, then X_{kj} is said to be a generically reduced component of $\mathcal{V}(f)$. We can always reduce to this case (see §9), and so for the rest of this section, we will assume we are in this case.

A witness set W_{kj} is a good proxy for X_{kj}. Indeed, let B be a set of k general linear equations on $\mathbb{C}^N$. By using continuation (for t going from 1 to 0 with W_{kj} as our startpoints) on the system of Eq. 73 with L_k replaced with $(1-t)B + \gamma t L_k$, we get the solutions of $B = 0$ on X_{kj}. This process lets us construct many widely separated points on X_{kj}.

Interpolation of a sufficiently large set of points on X_{kj} constructs polynomials vanishing on it. This was used in [91] to give a probability one *membership test* to determine whether a point x^* is on a k-dimensional irreducible component X_{kj} of the solution set $\mathcal{V}(f)$. Crucially, in the whole process only a finite number of points S on $\mathbb{C}^N$ arise. Unfortunately, for a degree d component, the number of points required to perform the interpolation is one less than the number of monomials of degree at most d in N variables, $\binom{N+d}{N} - 1$, which grows quickly. Although some measures can be taken to reduce this number [88], a completely different approach is needed to avoid the combinatorial complexity inherent in any interpolation scheme. For this reason, interpolation methods have been supplanted by a much more efficient approach based on monodromy, and we visit this next.

7.3 The Monodromy Membership Test

The main membership test used in practice are variants of the following test from [92] (see [99, Chap. 15.4] for details).

Let f be a system on n polynomials on $\mathbb{C}^N$. Let X be a k-dimensional irreducible component of $\mathcal{V}(f)$ and let $(f, \mathcal{W}, L)$ be a witness set for X associated to the system Eq. 73.

Problem 7.3.1 *Decide if $x^* \in \mathcal{V}(f)$ is a point of X.*

To decide this, choose a set B of k linear polynomials that are general except for the constraint that they vanish on x^*. This is easily done by choosing a random matrix $M \in \mathbb{C}^{k\times N}$ and setting $B = M \cdot (x - x^*)$. Use continuation with the homotopy

$$\begin{bmatrix} \begin{bmatrix} I_{N-k} & A \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \\ (1-t)B + tL \end{bmatrix} = 0. \tag{74}$$

to track the points for t going from 1 to 0 starting with $\mathcal{W}$ and ending with a set $\mathcal{E}$. Then $x^* \in X$ if and only if $x^* \in \mathcal{E}$. Moreover if $x^* \in X$, the multiplicity of x^* as a point of X equals the number of paths ending in x^*.

7.4 Deflation Revisited

The monodromy membership test for a component $X \in \mathcal{V}(f)$ and its historical predecessor based on interpolation both require tracking points on X as the system of linear slicing equations is moved in a homotopy. To do this efficiently, the points sliced out of X should be reduced so that the homotopy function has full rank. In general, however, X might be nonreduced, that is, it might be a component of multiplicity greater than one. In such cases, it is desirable to *deflate* X by operating on f to produce a new system for which X is a reduced component. We already saw in § 6.3 how this can be done when X is just an isolated point. This section describes a procedure to handle the case when X is positive dimensional.

Let f be a system on n polynomials on $\mathbb{C}^N$, let X be a k-dimensional irreducible component of $\mathcal{V}(f)$, and let $(f, \mathcal{W}, L)$ be a witness set for X associated to the system Eq. 73. Similar to the definition of multiplicity for an isolated point described in § 6.3, the multiplicity of components may be defined using the local ring associated to the prime ideal of polynomials vanishing on X. This is the same number as the multiplicity of any one of the points $x^* \in \mathcal{W}$ with respect to the restriction of f to $\mathcal{V}(L)$, that is, the multiplicity of $x^* \in \mathcal{W}$ as an isolated point of $\mathcal{V}(f, L)$.

Tracking such a point x^* as the equations L change in Eq. 74 is quite difficult when the multiplicity of X is greater than one [93]. Fortunately, the deflation procedure of §6.3 when done at x^* serves to generically deflate the component X (see [99, Chap. 13.3.2]). To see what the systems for components look like, let us consider the first deflation step as applied to Eq. 73. We need to take the Jacobian with respect to coordinates on $\mathcal{V}(L_k)$, so letting B and b be matrices such that $L_k = B \cdot x - b$ and letting $B^\perp$ be the $N \times (N-k)$ matrix with column vectors an orthonormal basis of the kernel of $x \to B \cdot x = 0$, we have

$$\begin{bmatrix} \begin{bmatrix} I_{N-k} & A \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \\ \begin{bmatrix} I_{N-k} & A \end{bmatrix} \cdot Df(x) \cdot B^{\perp} \cdot \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_{N-k} \end{bmatrix} \\ \begin{bmatrix} I_\ell & C \end{bmatrix} \cdot \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_{N-k} \end{bmatrix} - \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \end{bmatrix} = 0,$$

where $N - k - \ell$ is the rank of

$$\begin{bmatrix} I_{N-k} & A \end{bmatrix} \cdot Df(x^*) \cdot B^{\perp},$$

$Df(x^*)$ is the Jacobian matrix of $f(x)$ evaluated at x^*, I_k is the $k \times k$ identity matrix, and C is a generic $\ell \times (N - k - \ell)$ matrix. With L_k and therefore B fixed, this system is a deflation step for a nonempty Zariski of points $x \in X$. When this procedure yields a system that deflates x^* as a solution of f restricted to L_k, it will also deflate a nonempty Zariski open set of X. When one step of deflation does not suffice to deflate x^*, one may apply a second step of deflation to the above system, and so on until a full reduction to multiplicity one has been obtained. For a multiplicity μ component, at most $\mu - 1$ deflation stages are required. A more controlled deflation is given in [46]. No matter which deflation procedure is used, the end result is as recorded next.

Theorem 7.4 *Let X be an irreducible component of $\mathcal{V}(f) \subset \mathbb{C}^N$, where $f(x)$ is a polynomial system on $\mathbb{C}^N$. Let $(f, \mathcal{W}, L)$ be a witness set for X. There is polynomial system $F(x, u)$ on $\mathbb{C}^N \times \mathbb{C}^M$ and an irreducible component $\mathcal{X}$ of $\mathcal{V}(F)$, such that*

1. *$\mathcal{X}$ is of multiplicity one;*
2. *under the linear projection $\pi : \mathbb{C}^N \times \mathbb{C}^M \to \mathbb{C}^N$, a Zariski open set of $\mathcal{X}$ maps isomorphically onto a Zariski open set of X containing $\mathcal{W}$; and*
3. *$(F, \pi^{-1}(\mathcal{W}) \cap \mathcal{X}, L)$ is a witness set of $\mathcal{X}$.*

7.5 Numerical Irreducible Decomposition

Let $f : \mathbb{C}^N \to \mathbb{C}^n$ be a polynomial system. Let

$$\mathcal{V}(f) := \cup_{i=0}^{\dim \mathcal{V}(f)} X_i = \cup_{i=0}^{\dim \mathcal{V}(f)} \cup_{j \in \mathcal{I}_i} X_{ij}$$

denote the irreducible decomposition. By the numerical irreducible decomposition, we mean a set of witness sets (f, W_{ij}, L_i), one for each X_{ij}. Note we have the same set of linear equations

$$L_i := B_i \cdot x + b$$

for each $j \in \mathcal{I}_i$, where $B \in \mathbb{C}^{i\times N}$ and $b \in \mathbb{C}^{i\times 1}$. Since the multiplicities of any two witness points for the same component are the same, and since a deflation of a component works on a set containing the witness set of an irreducible component, we can assume after renaming that the $\mathcal{W}_i$ are nonsingular solutions of the system Eq. 73.

The computation of the numerical irreducible decomposition proceeds in three main steps:

- computing a witness superset,
- removing "junk" points to obtain a composite witness set, and
- decomposing the composite witness set into irreducible pieces.

First, witness supersets are computed for the X_i. By a witness superset for X_i is meant a triple $(f, \widehat{W}_i, L_i)$ such that L_i is a set of i general linear equations and

$$X_i \cap \mathcal{V}(L_i) \ \subset \widehat{W}_i$$

and

$$\widehat{W}_i \subset \cup_{k=i}^{\mathcal{V}(f)} X_k.$$

This may be done by using continuation to solve the polynomial system given in Eq. 73. This was done first in [89] and improved in [90]. The most efficient algorithm for this is presented in [47].

The witness superset may contain some extra points that we designate as "junk." Specifically, $\widehat{W}_i = W_i + J_i$, where W_i is the witness point set for X_i and J_i is contained in $\cup_{k>i} X_k$. The homotopy that generates $\widehat{W}_i$ does not distinguish between the desired witness points and the junk, so this requires an extra step.

The first approaches to junk removal were based on membership tests [91, 92]. This hinges on the fact that the junk points for X_i all lie on the components of dimension greater than i. Thus, if we start at the top dimension and work down, we can test which points in $\widehat{W}_i$ belong to higher dimensional sets and expunge them. This works with either the interpolation membership test or the more efficient monodromy membership test, both described above. Since one must use witness sets for all the dimensions greater than i before one can attack dimension i, we say that these are global methods. This is in contrast to a local method, which can decide whether a point is junk without any knowledge about solution sets of other dimensions. Although the global methods are now obsolete, the original one in [91] is notable for being the first route discovered for computing the numerical irreducible decomposition. Because of that, it also was the first algorithm for finding X_0, which is the exact set of isolated solutions of $f(x) = 0$. Without junk removal, all previous homotopies provided a superset of X_0.

The global approach to junk removal has been supplanted by a more efficient method based on using the local dimension tests [9] (see §9.2). The idea is that one does not really need to know which higher-dimensional set a junk point is on nor even need to know the precise dimension of the containing set; one only needs to know that a point is on some higher-dimensional set to conclude that it is junk. A local dimension test provides this minimal information with much less computation than is required by global membership testing.

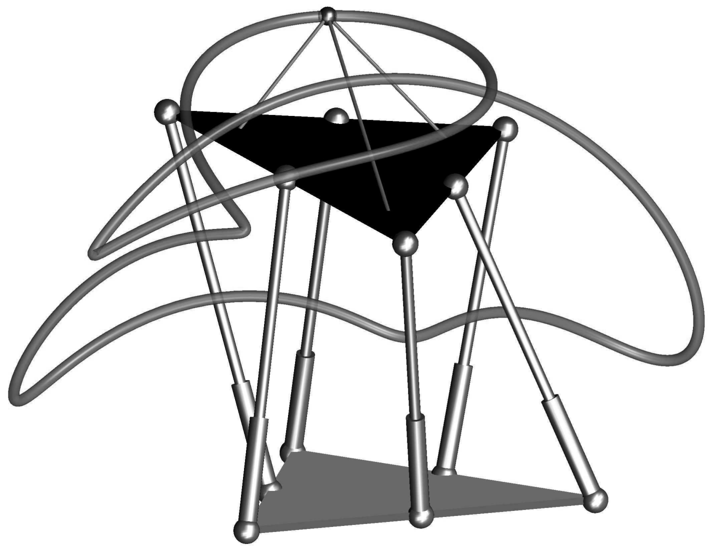


Figure 11: Griffis-Duffy platform of Type I.

With the junk points removed, the next step is the decomposition of W_i into W_{ij}, the witness point sets for the irreducible components of X_i. Recall that irreducible components are the path-connected pieces of the solution set after singularities have been removed. Thus, if we can show that two points are connected by a path that avoids singularities, we know that they are in the same irreducible set. We have $W_i = X_i \cap \mathcal{V}(L_i)$ for L_i being set of i random linear equations. Let us regard

$$L_i(x, y) := B_i \cdot x + y$$

as a family of linear equations in $x \in \mathbb{C}^N$ with parameters $y \in \mathbb{C}^i$. For a Zariski open set U of $\mathbb{C}^i$, the set $X_i \cap \mathcal{V}(L_i(x, y))$ consists of $\deg X_i$ nonsingular points. Any appropriately random one-real-dimensional path in $\mathbb{C}^i$ will lie in U with probability one. Tracking the points $\mathcal{W}_i$ around a closed path in U, we obtain a permutation of the set $\mathcal{W}_i$. Points that are interchanged lie on the same component. Theory tells us that there are enough paths to achieve the breakup into irreducible components [92, 94]. This is the basis for the monodromy approach to decomposing the witness set into irreducibles.

Although we know that paths exist to achieve the breakup, there is no a priori way to know which paths or how many of them are needed. The way we proceed is to take random paths and form groups of points known to belong to the same irreducible component. We then check each group to see if it is in fact a complete witness point set for a component. The test for completeness is based on traces [94], discussed in [99, Chap. 15]. If the monodromy loops have not yet found a complete decomposition but the number of incomplete groups is not too big, exhaustive trace testing can determine which incomplete groups need to be merged to complete the decomposition. In this way, the algorithm is guaranteed to terminate in a finite number of steps.

One interesting example of the application of the numerical irreducible decomposition is a special case of the Stewart-Gough platform called called the Griffis-Duffy Type I architecturally-singular platforms. These have a base and moving platforms that are equilateral triangles, with legs connecting vertices of the base to midpoints of the moving platform and vice versa in a cyclic pattern [51, 95]. No matter what the leg lengths are, a general case of this type of platform has a motion curve in Study coordinates of degree 28. This is illustrated in Figure 11, where the path of a small sphere attached to the moving plate is shown. This path has degree 40 in $\mathbb{R}^3$. For a special case of this in which the two triangles are congruent and the leg lengths are equal, this curve factors into five pieces: four sextics and one quartic. The numerical irreducible decomposition is able to find all of these, providing a witness set on each [95].

8 Software

There are several software packages that compute isolated solutions of polynomial systems, Bertini [13], HOM4PS-2.0 [60], Hompack90 [116] and its extensions [104, 117], and PHCpack [110]. Hompack90 has general parallel tracking facilities. HOM4PS-2.0 has the best implementation of polyhedral methods, but is not a parallel code. Only Bertini and PHCpack implement algorithms of numerical algebraic geometry for positive-dimensional solution sets.

PHCpack and Bertini both allow the user to define their own homotopy and prescribed start points, but HOM4PS-2.0 currently does not.

HOM4PS-2.0 uses only double precision arithmetic to perform computations. To varying degree, both PHCpack and Bertini have the capability to use higher precision arithmetic. PHCpack does not currently have the capability to adapt the precision based on the local conditioning of the homotopy path. This means that more human interaction is needed to verify that the precision is set appropriately to accurately and reliably perform the requested computations.

The more advanced algorithms of numerical algebraic geometry (including the powerful equation-by-equation methods for finding isolated solutions) place strong requirements on the underlying numerical software [14]. For example, without secure path-tracking and adaptive precision, computing the numerical irreducible decomposition for systems that involve more than a few variables is not possible.

Only Bertini gives the numerical irreducible decomposition directly. Exceptional features of Bertini include

- secure path-tracking;
- adaptive multiprecision [7, 8, 12];
- utilities for working with polynomial systems given as straightline programs;
- parallel high-precision linear algebra (in a coming release);
- the numerical irreducible decomposition [91, 99] (see §7.5);
- equation-by-equation methods such as regeneration [46, 47] (see §10.2), which require secure path-tracking for all but the smallest systems;
- local-dimension testing [9] (see §9.2); and

- various endgames (see §6.4) including the power series endgame [75], the Cauchy endgame [73], and a parallel endgame based on it [11].

To put some of this in perspective, consider a sequence of polynomial systems arising from discretizing the Lotka-Volterra population model that is presented in [46]. HOM4PS-2.0 and PHCpack both implement the polyhedral homotopy method for solving sparse polynomial systems, while Bertini implements regeneration [46] (discussed in § 10.2) for solving such systems. For these systems, the number of solutions over $\mathbb{C}$ is the number of paths tracked by the polyhedral homotopy method. Even so, the polyhedral homotopy becomes impractical as the number of variables increases due to the computational complexity of computing a start system via the mixed volume. For example, consider solving the $n = 24$, 32, and 40 variable instances of the Lotka-Volterra model using a single core running 64-bit Linux. Here the number of nonsingular isolated solutions equals $\sqrt{2^n}$ (which equals the mixed volume). For the 24 variable system, PHCpack took over 18 days while HOM4PS-2.0 and Bertini both took around 10 minutes. For the 32 variable system, PHCpack failed to solve the system in 45 days, HOM4PS-2.0 took over 3 days and Bertini took roughly 5 hours. For the 40 variable system, PHCpack and HOM4PS-2.0 both failed to solve the system in 45 days, but Bertini solved the system in under a week. Since regeneration is parallelizable, we also solved the 32 and 40 variable systems using 64 cores (8 nodes each having dual 2.33 GH Xeon 5410 quad-core processors). In parallel, Bertini took less than 8 minutes and 4 hours to solve the 32 and 40 variable polynomial systems, respectively.

Part III
Advanced Topics

9 Nonreduced Components

At several points, we have noted that a component of the solution set $\mathcal{V}(f)$ can be nonreduced, meaning that it has a multiplicity greater than one. How do we compute the multiplicity of a k-dimensional irreducible component of $\mathcal{V}(f) \subset \mathbb{C}^N$, where f is a polynomial system (see Eq. 60) of n polynomials? As mentioned earlier, by slicing with a general $(N-k)$-dimensional affine linear space, this comes down to computing the multiplicity of an isolated solution of a polynomial system. We will see in §9.1, that the Macaulay matrix gives a powerful method for solving this problem. Modifying this method gives us an algorithm § 9.2 for computing the local dimension of an algebraic set.

9.1 Macaulay Matrix

The Macaulay matrix gives a powerful method to compute multiplicities and lay bare the local structure of an algebraic set. This approach to the numerical computation of the multiplicity was presented by Dayton and Zeng [29] (see also [30]). An alternate approach, which played a role in the development of local dimension test [9] of §9.2, was given by Bates, Peterson, and Sommese [15].

We need multiindex notation. Given an s-tuple of nonnegative integers $\alpha = (\alpha_1, \ldots, \alpha_s)$ and an s-tuple of complex numbers $u = (u_1, \ldots, u_s)$, we let

$$\begin{aligned} u^\alpha &:= u_1^{\alpha_1} \cdots u_s^{\alpha_s} \\ \alpha! &:= \alpha_1! \cdots \alpha_s! \\ D^\alpha &:= \frac{1}{\alpha!}\left(\frac{\partial}{\partial x_1}\right)^{\alpha_1} \cdots \left(\frac{\partial}{\partial x_s}\right)^{\alpha_s} \\ |\alpha| &:= \alpha_1 + \cdots + \alpha_s \end{aligned}$$

Let x^* denote a solution of $f(x) = 0$, a system on n polynomials in the variables $x_1, \ldots, x_N$. From this we can construct the k-th Macaulay matrix:

$$M_k(f, x^*) := \left[\ \left(D^\alpha (x - x^*)^\beta f_j(x)\right)(x^*) \ \right] \tag{75}$$

where the rows are indexed by (β, j) with $|\beta| \leq k - 1$, $j \leq n$ and the columns are indexed by α with $|\alpha| \leq k$. Letting $P_{d,N}$ denote the dimension of the vector space of polynomials of degree at most d in N variables, $M_k(f, x^*)$ is an $(nP_{d,N}) \times P_{d,N}$ matrix.

The rows are Taylor series expansions of generators of approximations to the ideal generated by f in the ring of convergent power series at x^*.

Theory (see [29, 9, 30]) implies that the dimensions $\mu_k(f, x^*)$ of the nullspace

$$\left\{\xi \in \mathbb{C}^{P_{d,N}} \,\middle|\, M_k(f, x^*) \cdot \xi = 0\right\}$$

are either strictly increasing with no limit (in which case $\mathcal{V}(f)$ is positive dimensional at x^*) or $\mu_k(f, x^*)$ are strictly increasing until the first nonnegative integer k with $\mu_k(f, x^*) = \mu_{k+1}(f, x^*)$ (in which case x^* is an isolated solution of $f(x) = 0$ of multiplicity $\mu_k(f, x^*)$).

In particular, assume that we know a positive number $\hat{\mu}$, such that if x^* is an isolated solution of $f(x) = 0$, then $\hat{\mu}$ is a bound for the multiplicity of x^* as an isolated solution of $f(x) = 0$. In this case, we can check whether x^* is isolated or not, because the $\mu_k(f, x^*)$ must either stabilize or eventually surpass $\hat{\mu}$. Since $\mu_k(f, x^*)$ is strictly increasing, if it does not stabilize, it must pass $\hat{\mu}$ no later than by $k = \hat{\mu} + 1$. This last observation leads to the local dimension test [9] explained in the next subsection.

The Macaulay matrices $M_k(f, x^*)$ grow quickly in size, but for computation of the $\mu_k(f, x^*)$, $M_k(f, x)$ may be reduced to a much smaller matrix [118].

9.2 Local Dimension

In computing the numerical irreducible decomposition, we noted that compared with the use of membership tests, a local dimension test can be used to improve the speed of removing junk points. Even outside of this application, the topic of determining local dimension is of interest in its own right. The problem to be addressed is as follows.

Problem 9.0.1 *Given a solution x^* of a polynomial system $f(x) = 0$, determine the local dimension at x^* of the solution set $\mathcal{V}(f)$.*

By *local dimension* at x^*, we simply mean the maximum of the dimensions of the irreducible components of $\mathcal{V}(f)$ that pass through x^*. In junk removal, we do not even need the full determination of the local dimension; it is enough to determine whether or not a point in $\widehat{W}_i$ has a local dimension higher than i.

A global way to find the local dimension is to carry out a full irreducible decomposition of $\mathcal{V}(f)$ and use membership tests to determine which component or components contain x^*. This is a global method requiring a very expensive computation with much of the output of the computation unused.

The test described in the previous section which uses the Macaulay matrix to determine whether or not a point is isolated was put forward in [9] as an efficient method for junk removal in the numerical irreducible decomposition of $\mathcal{V}(f)$. The test requires a positive integer $\hat{\mu}$ which would be a bound for the multiplicity of the point in question. In the irreducible decomposition, suppose we have a point, say x^*, in the witness superset $\widehat{W}_i$ for dimension i. If x^* is an isolated point of $\mathcal{V}(f(x), L_i(x))$, where $L_i(x) = 0$ is the set of i linear equations that slice out the witness set, then it belongs in the witness set. Otherwise, if it is not isolated, it is a junk point to be expunged. In witness superset generation, the bound $\hat{\mu}$ is just the number of incoming paths to x^* in the homotopy that computes the witness superset. With this bound, the Macaulay matrix test for isolation can eliminate junk points. Since we no longer need all the higher-dimensional sets (dimensions $j > i$) to do junk removal at dimension i, the local dimension test allows us to do the numerical irreducible decomposition for each dimension independently.

While junk removal only needs one test for isolation, a procedure described in [9] uses a sequence of such tests to exactly determine local dimension. The procedure uses the following fact. Let $(f(x), L(x)) = 0$ be a new system formed by appending to f a single linear equation $L(x)$ that is general except for the restriction that it vanishes at x^*. If $x^* \in \mathcal{V}(f)$ is not isolated, then the local dimension of $\mathcal{V}(f, L)$ at x^* is precisely one less than the local dimension of $\mathcal{V}(f)$ at x^*. Thus we can compute the dimension by repeatedly slicing until x^* is isolated. For this to work, each time we add another linear equation to the system, we need to have a bound $\hat{\mu}$ on what would be the multiplicity of x^* if it were isolated. The article [9] leaves open the question of how to find these bounds.

A natural way to provide the multiplicity bounds needed to complete the local dimension procedure is to construct a good homotopy $H(x, t) = 0$ (see the end of §6.1) for finding isolated solutions of $(f(x), L_i(x)) = 0$ and count the number of paths coming into x^*. (Here $L_i(x)$ is a system of i general linear polynomials that vanish at x^*.) If there are no paths with limit x^*, then x^* is not an isolated solution of $(f(x), L_i(x)) = 0$. If paths do arrive at x^*, their number give the requisite bound and we may check for isolation. If x^* is an isolated solution of this system, then the local dimension of $\mathcal{V}(f)$ at x^* is i. If it is not isolated, then the local dimension must be greater than i and we may start over, appending some greater number of linear polynomials. Eventually, all the possible dimensions (from 0 to N) will be exhausted, so at some point we must terminate with the correct local dimension. The homotopies used here to get the multiplicity bounds are in fact generating witness supersets dimension-by-dimension, so if one is charged with finding the local dimension at several test points, one might be just as well off to compute an entire witness superset for $\mathcal{V}(f)$. At present, there is no known way to find the local dimension without resorting to these homotopies.

Suppose that one does not have a multiplicity bound in hand and one is unwilling to run homotopies to find one. It can still be useful to investigate local dimension by testing the ranks of the Macaulay matrices. Limiting the tests to $M_k(f, x)$, for $k \leq k^*$, gives the k^*-depth-bounded local dimension [115]. For large enough k^*, this is the correct local dimension, but without a multiplicity bound, one cannot make a definitive conclusion. Still the depth-bounded local dimension yields much of what one needs to know in practice, where the difference between a very high multiplicity and increased dimension can become academic. For example, in the case of a mechanism, a high multiplicity isolated root in the rigid-body model may in fact exhibit substantial motion when small elastic deformations of the links, which are always present in a physical device, enter the picture.

10 Optimal Solving

There is a basic and ever present problem.

Problem 10.0.2 *Assume we have a* large *polynomial system $f(x) = 0$ with* few *solutions. How do we find the solutions without doing the amount of work that the size of the system would suggest?*

The words *large* and *few* and the problem itself may seem imprecise, but the problem is quite real. For simplicity we consider only a square system f, and we seek to find a finite set of points in $\mathcal{V}(f)$ that contains all the isolated solutions. Most problems in numerical algebraic geometry may be reduced to this problem.

In the middle 1980s, a polynomial system with 8 equations and 8 variables could be quite challenging. For example, to solve the inverse kinematics problem for 6R robots (see § 3.1.2,4.5.2), Tsai and Morgan [108] used a total degree homotopy with 256 paths. Each path took several minutes on a large mainframe, and paths that went to infinity were computationally much more expensive than finite isolated nonsingular solutions. The use of multihomogeneous structure [70] allowed the construction of a homotopy tracking 96 paths, which were overall not only fewer in number, but better behaved. Polyhedral homotopies are even better behaved.

The nine-point path-synthesis problem for four-bars (see § 3.2 and [113, 114]) has only 1442 solutions, though a total-degree homotopy for the Roth-Freudenstein formulation of the problem would require tracking $7^8 = 5,764,801$ paths. This would have required much more computational power than was available around 1990, when the problem was solved using a multihomogeneous homotopy. Even that calculation pushed the limits of its day: it tracked 143,360 paths and required over 100 hours of computation on a large mainframe computer.

Over and beyond numerics, finding the homotopy used in that first solution of the nine-point problem required a great deal of insight. It is interesting to consider whether there is a solution method that could proceed efficiently to solve the original Roth-Freudenstein formulation of the system. We will return to this question shortly.

As computer capabilities advance, what large and few mean change, but the problem remains. For example, polyhedral methods are remarkable, but when the number of variables grows, the computation to construct the polyhedral homotopy eventually precludes solving the system (see the examples in [46]). This is because the construction involves a combinatorial calculation, called the mixed volume, that grows quickly with the number of variables and the number of monomials in the problem.

Another aspect of this problem is that many polynomial systems are given in an efficient straight-line form, meaning that each polynomial is evaluated using a sequence of expressions with the later expressions being functions of the earlier expressions. Evaluation for such systems may be much quicker and more stable than if the same polynomials were expanded out into a sum of terms. Consider, for example, the four-bar coupler-curve equation as written in Eq. 23. The efficient way to evaluate this is to first evaluate each of the column matrices of Eq. 24, then the 2×2 determinants in Eq. 23, and then finally evaluate the polynomial using these. Polyhedral homotopies work with the expanded monomials of the problem, so may destroy the straight-line structure.

Computational work is roughly proportional to the number of paths tracked. *Optimally*, one hopes for a method of solution of the problem whose computational cost is a small multiple of the computational cost of tracking, for a system of similar characteristics, a number of paths equal to the number of solutions. We shall see in § 10.1 that this hope can be attained for a parameterized family of problems $f(x; q) = 0$, if only one can first solve any one general example in the family. In §10.2, we will discuss a class of methods with promise to achieve near optimal computational cost even on the first example problem.

10.1 Coefficient Parameter Homotopies

Fix a parameterized family of polynomial systems

$$f(x,q) := \begin{bmatrix} f_1(x,q) \\ \vdots \\ f_N(x,q) \end{bmatrix} \tag{76}$$

where $(x,q) \in \mathbb{C}^N \times \mathbb{C}^M$ with $M \geq 1$. The basic result of [72] (see also [65] and [99, Chap. 7]) is that, though it might computationally cost a great deal to compute solve $f(x,q) = 0$ for a generic q^*, the cost of solving $f(x,q) = 0$ for subsequent q is the optimal cost. Let us make this precise.

Theorem 10.1 ([72]) *Let $f(x,q) = 0$ is a system of polynomials*

$$f(x,q) := \begin{bmatrix} f_1(x,q) \\ \vdots \\ f_N(x,q) \end{bmatrix}$$

where $(x,q) \in \mathbb{C}^N \times \mathbb{C}^M$. There is a nonempty Zariski open set $U \subset \mathbb{C}^M$ such that for each positive integer μ, the set

$$Z_\mu := \left\{(x,q) \in \mathbb{C}^N \times U \;\middle|\; x \text{ is a multiplicity } \mu \text{ isolated solution of } f(x,q) = 0\right\}$$

is either empty or a manifold with the map $\pi_Z : Z \to U$, under the restriction of the projection $\pi : \mathbb{C}^N \times \mathbb{C}^M \to \mathbb{C}^M$, a finite covering. Given any $q^* \in \mathbb{C}^M \setminus U$ and any isolated solution x^* of $f(x,q^*) = 0$, for any open ball B around x^*, there is an open ball D around q^*, such that for every $q \in D \cap U$, $f(x,q) = 0$ has solutions in $B \times \{q\}$ and they are all isolated.

Set

$$Z := \cup_{\mu \geq 1} Z_\mu.$$

Fix a generic $q^* \in \mathbb{C}^M$. Then U may be chosen with the further property, that the deflation of $f(x, q^*)$ at an isolated solution of $f(x, q^*) = 0$ deflates the whole component of Z that (x, q^*) belongs to.

Thus at the expense of computing the isolated solutions of $f(x, q^*) = 0$, and carrying out a set of deflations one time, we can find the isolated solutions of $f(x, q) = 0$ for any other $q \in \mathbb{C}^M$ at the cost of tracking a number of paths equal to the sheet number of $\pi_Z : Z \to U$. This means that if we solve enough examples of the problem family, the amortized cost of solving $f(x, q) = 0$ approaches the cost of tracking a number of solutions equal to the number of isolated solutions of $f(x, q)$ for a generic q^*.

Finally we note, all the results above are true with $\mathbb{C}^M$ replaced by any irreducible and reduced algebraic set which is locally irreducible [99].

The coefficient-parameter homotopy is especially simple to formulate when one only wishes to find Z_1, the multiplicity-one isolated solutions. Suppose we have already solved by some means the system $f(x, q^*) = 0$, with q^* general. Call the set of multiplicity-one isolated solutions Z_1^*. Now suppose that we wish to solve for Z_1^{**} for the problem $f(x, q^{**}) = 0$. If the parameter space is Euclidean, then the path

$$\phi(t) := tq^{**} + (1 - t)q^*, \qquad t = \frac{\gamma\tau}{1 + (\gamma - 1)\tau}, \qquad \tau \in (0, 1],$$

leads from q^* to q^{**} as τ goes from 1 to 0, and it stays general with probability one for random choices of $\gamma \in \mathbb{C}$ [99, Chap. 7]. Accordingly, the coefficient-parameter homotopy is

$$H(x, \tau) = f(x, \phi(t(\tau))). \tag{77}$$

If the parameter space is not Euclidean, one must arrange a path that stays in the parameter space. This is easily done in most cases. For example, if the parameter space is a unit circle, we just replace $\phi(t)$ by $\phi(t)/|\phi(t)|$.

As we saw in Part I, problems from kinematics automatically come with parameter spaces so coefficient-parameter homotopy becomes a powerful tool in the area. As a first example, consider the forward kinematics problem for general Stewart-Gough (6-SPS) platforms, given by Eqs. 33,35. These are 7 equations in $[e, g] \in \mathbb{P}^7$, all quadratic. The parameter space for the problem may be taken as the entries in the 4×4 matrices A_j, B_j, $j = 1, \ldots, 6$. One can solve a general member of this family using a total-degree homotopy having $2^7 = 128$ paths. If one takes general matrices A_j and B_j, the problem has 84 finite solutions. But recall that in the kinematics problem each B_j is an antisymmetric matrix. On this parameter space, the problem has just 40 solutions. One can solve any other example in the family with a coefficient-parameter homotopy that has just 40 paths. Moreover, there are several different subfamilies of interest wherein some of the S joints coincide. One of these is the octahedral family illustrated in Figure 9, for which the problem has only 16 roots, appearing in a two-way symmetry. (Reflection of the mechanism through the plane of the base does not alter the geometry.) Since a coefficient-parameter homotopy respects this symmetry, only eight paths need to be tracked. As discussed in [99, § 7.7], after solving one general member of any Stewart-Gough subfamily, the remaining ones can be solved with an optimal number of paths by coefficient-parameter homotopy. Although these problems are all simple enough that a elimination approach can be devised—and this has been done for most cases—each special case requires a new derivation. In contrast, homotopy methods cover all the cases seamlessly.

A more extreme illustration of the power of the coefficient-parameter homotopy technique is provided by the nine-point path-synthesis problem for four-bars. As we mentioned earlier, the best multihomogeneous formulation found for the problem has 143,360 paths. Of these, only 4326 paths have finite endpoints. So after a one-time execution of that homotopy for a general example, all subsequent examples can be solved with a coefficient-parameter homotopy having only 4326 paths. But the story gets even better, because the 4326 solutions appear in a three-way symmetry called Robert's cognates [83]. The coefficient-parameter homotopy respects this symmetry, so only one path in each symmetry group needs to be tracked, resulting in a homotopy with only 1442 paths. This is nearly a 100-fold decrease in the number of paths compared to the original multihomogeneous homotopy.

10.2 Equation-by-Equation Methods

The idea behind equation-by-equation methods is to process the system adding one equation at a time, that is, intersecting the solution of the new equation with the solution set for all the preceding equations. The hope is that the computations at any stage will remain comparable to what one might expect from an optimal homotopy having one path to each of the final solutions. Specifically, one hopes that the number of paths in each stage of the equation-by-equation approach is at most a small multiple of the number of solutions at the final stage. If that number is small compared to the total degree, or even better small compared to the mixed volume (the number of paths in a polyhedral homotopy), then the multistage equation-by-equation approach will pay off. Although a coefficient-parameter homotopy is efficient for all but the first example in a family, the equation-by-equation approach tries to improve the efficiency of solving the very first example. Also, one might hope that this can be done for a problem in its most basic presentation, without requiring extensive work up front to simplify it. In fact, the equation-by-equation approaches often go a long way towards these goals. For example, the regeneration method below solves the original unaltered Roth-Freudenstein formulation of the nine-point problem [46, §9.3] with computational effort similar to the extensively reworked multihomogeneous version from [113].

We discuss the two equation-by-equation methods below. The diagonal method [96, 97, 98] is important in its own right because it applies to the problem of intersecting general algebraic sets. For the problems that both methods apply to, the regeneration method [46, 47] typically runs a several times more quickly than the diagonal method.

For simplicity we present the special case of both algorithms that apply to find the nonsingular isolated solutions of a system f of N equations $f_1(x), \dots, f_N(x)$ in N variables. Since

$$\mathcal{V}(f_1, \dots, f_N) = \mathcal{V}(f_1) \cap \cdots \cap \mathcal{V}(f_N),$$

an isolated nonsingular solution $x^* \in \mathcal{V}(f)$ must be a smooth point of all the $\mathcal{V}(f_i)$ and they must be transversal at x^*, i.e., have linearly independent normals at x^*. Thus the component of $\mathcal{V}(f_1, \dots, f_k)$ must be smooth of codimension k at x^*.

Fix N linear functions $L_1(x), \dots, L_N(x)$, each of the form

$$L_i(x) = a_{i,1}x_1 + \cdots + a_{i,N}x_N - 1 \tag{78}$$

with the coefficients $a_{i,j}$ random. We will use the notation $L_{i:j}(x)$ to mean the subset of these linears from index i to j:

$$L_{i:j}(x) := \begin{bmatrix} L_i(x) \\ \vdots \\ L_j(x) \end{bmatrix}.$$

It is to be understood that if $i > j$, then $L_{i:j}(x)$ is empty.

We proceed going, for each $k \geq 1$, from $\mathcal{V}(f_1, \dots, f_k)$ to $\mathcal{V}(f_1, \dots, f_k) \cap \mathcal{V}(f_{k+1}) = \mathcal{V}(f_1, \dots, f_{k+1})$. Let $F_k(x) := \{f_1(x), \dots, f_k(x)\}$, and let W_k denote the nonsingular isolated solutions of

$$\begin{bmatrix} F_k(x) \\ L_{k+1:N}(x) \end{bmatrix}.$$

The number paths in stage k of the equation-by-equation approaches is proportional to the number of points in W_k. Typically, the sizes of these sets grows as one proceeds towards W_N, so that the total computational work is governed by the final number of solutions. This is not a hard-and-fast rule; it is possible for the sizes of the intermediate sets to grow large and fall again, but this is not generally seen in practice.

Diagonal Homotopy Let S_{k+1} denote the nonsingular isolated solutions of $G_{k+1}(x) = 0$, where

$$G_{k+1}(x) := \begin{bmatrix} L_{1:k}(x) \\ f_{k+1}(x) \\ L_{k+2:N}(x) \end{bmatrix}.$$

Note that the solution of $G_{k+1}(x) = 0$ is $\mathcal{V}(f_{k+1}) \cap \mathcal{L}$, where $\mathcal{L} := \mathcal{V}(L_{i:k-1}(x), L_{k+1:N}(x))$ is a line. After solving for this line using standard linear algebra, one can restrict f_{k+1} to it, turning the solution of $G_{k+1} = 0$ into a polynomial in one variable. In this way, one finds S_{k+1} easily, and it has at most $\deg f_{k+1}$ points.

Given W_k and S_{k+1}, the goal is to construct the nonsingular isolated solutions W_{k+1} of $\{F_{k+1}(x), L_{k+2:N}(x)\} = 0$, and by progressing in this incremental way arrive at W_N, the nonsingular isolated solutions of $F_N(x) = f(x) = 0$. Consider the homotopy:

$$H_k(x,t) := \begin{bmatrix} F_k(x) \\ f_{k+1}(y) \\ t \begin{bmatrix} L_1(y) \\ \vdots \\ L_k(y) \\ L_{k+1}(x) \end{bmatrix} + (1-t) \begin{bmatrix} x_1 - y_1 \\ \vdots \\ x_k - y_k \\ x_{k+1} - y_{k+1} \end{bmatrix} \\ L_{k+2:N}(x) \\ L_{k+2:N}(y) \end{bmatrix} \tag{79}$$

on $\mathbb{C}^N \times \mathbb{C}^N$. Note that $(a, b) \in W_k \times S_{k+1}$ are nonsingular solutions of $H_k(x, y, 1) = 0$. The isolated nonsingular solutions of $H_k(x, y, 0) = 0$ must satisfy $x = y$ and may be identified with the isolated nonsingular solutions of $F_{k+1}(x) = 0$ using the map $(x, y) \mapsto (x)$. In this manner, we have generated W_{k+1}, a result proven in [98]. All that remains to be done is to recurse on this procedure until we arrive at the final answer W_N.

Notice that in Eq. 79, the trailing equations $L_{k+2:N}(x) = 0$ and $L_{k+2:N}(y) = 0$ do not change during the continuation. Therefore, one can use linear methods to compute a basis for the null space of these at the beginning (they both have the same null space, of course), and use this to substitute for x and y in the other equations. In this way, the homotopy is effectively reformulated to work on $\mathbb{C}^{k+1} \times \mathbb{C}^{k+1}$. When k is small compared to N, this maneuver reduces the computation significantly.

Regeneration Regeneration is an equation-by-equation method, whose ancestry includes m-homogeneous homotopies of [70], the set structure homotopies of [111], and the product homotopies of [76].

The set up for regeneration uses functions $F_k(x)$, $L_i(x)$, and set W_k identical to their use in the diagonal homotopy, but we increment from W_k to W_{k+1} in a new way. To do this, we introduce additional random linear functions of the same form as Eq. 78, but with $\deg f_k$ of them associated to the kth polynomial. Let us denote these linear functions as $\hat{L}_{i,j}(x)$, $i = 1, \ldots, N$, $j = 1, \ldots, \deg f_i$. (It is acceptable to take $\hat{L}_{i,1} \equiv L_i$, but the others must be distinct random linear functions.)

Regeneration proceeds in two stages. First, for $j = 1, \ldots, \deg f_{k+1}$, we move from W_k to $W_{k,j}$ being respectively the nonsingular isolated solutions of $\{F_k(x), L_{k+1:N}(x)\} = 0$ and $\{F_k, \hat{L}_{k+1,j}(x), L_{k+2:N}(x)\} = 0$. This is accomplished with the homotopies

$$H_{k,j}(x, t) = \begin{bmatrix} F_k(x) \\ (1-t)\hat{L}_{k+1,j}(x) + tL_{k+1}(x) \\ L_{k+2:N}(x) \end{bmatrix} = 0, \qquad j = 1, \ldots, \deg f_{k+1}. \tag{80}$$

After completing these, we may gather all the solutions into one set

$$S_k = \cup_{j=1}^{\deg f_{k+1}} W_{k,j}.$$

The second stage of the regeneration procedure uses S_k as the startpoints for the homotopy

$$R_k(x, t) = \begin{bmatrix} F_k(x) \\ (1-t)f_{k+1}(x) + t\hat{L}_{k+1,1}(x) \cdots \hat{L}_{k+1,\deg f_{k+1}}(x) \\ L_{k+2:N}(x) \end{bmatrix}. \tag{81}$$

The nonsingular endpoints of this homotopy as t goes from 1 to 0 are the set W_{k+1}. As in the diagonal homotopy method, we recursively apply the two stages of regeneration until we arrive at the final solution set W_N. The main result of [46] justifies this procedure for finding all the nonsingular isolated solutions of $f(x) = 0$.

In both stages, the trailing equations, $L_{k+2:N}(x) = 0$, do not change during the continuation, so as before one can work on the associated null space and thereby reduce the number of variables to $k + 1$. This is half of the number of variables used in the diagonal homotopy.

To appreciate the power of regeneration, note that solving the Roth-Freudenstein system for the nine-point problem using regeneration is comparable in timing to solving it using a four-homogeneous formulation and faster on one processor than the best polyhedral code [46]. This happens even though the total degree of the Roth-Freudenstein system is about twenty times the number of paths in the four-homogeneous formulation and sixty-five times the number of paths in the polyhedral homotopy. (When run tracking homotopy paths in parallel using a system with 65 processors, the regeneration method was proportionately faster.) More tests comparing regeneration to polyhedral methods are reported in [107] along with a complementary method that helps sort the solution sets more automatically.

Part IV
Frontiers

11 Real Sets

Throughout this article we have dealt almost exclusively with complex solutions, even though isolated and positive dimensional real solutions are the main interest for most applications.

For isolated solutions, we may simply pick the isolated real solutions out of the isolated complex solutions. Typically most complex solutions are not real, but we do not know any algorithms that find all the isolated real solutions without finding all the isolated complex solutions. Indeed, the majority of systems arising in applications depend on parameters and the continuation of a real isolated solution as one goes from one point in the parameter space to another, e.g., using the methodology in the coefficient parameter theorem 10.1, very often does not end up at a real solution. Similarly, the path that leads to a real solution might begin at a complex one. Consequently, continuation of just the real solutions over a real parameter space is problematic and generally will not produce a complete solution list, so we work over the complex numbers instead.

For positive dimensional sets, there is no direct method of computing the real solution set when given the complex solution set. There is a classical description, using symbolic computation, called the cylindrical decomposition [6], which may be approached numerically.

Set $\mathcal{V}_{\mathbb{R}}(f) \subset \mathbb{R}^N$ denote the set of real solutions of the system $f(x) = 0$. Throughout this article, we have used $\mathcal{V}(f)$ to mean the complex solution set, though for emphasis here, we may use $\mathcal{V}_{\mathbb{C}}(f)$.

Real and complex solution sets are not as tightly related to as one might expect, and this shows up in considering the dimension of $\mathcal{V}_{\mathbb{R}}(f)$ versus $\mathcal{V}_{\mathbb{C}}(f)$. For example consider the equation $x^2 + y^2 = 0$. Over the complex numbers the solution set consists of two lines meeting at the origin, i.e., a set of complex dimension one. The real solution set is the origin, an isolated singular point, i.e., a set of real dimension 0. This difference of dimension only happens at real solutions contained in the singular set of the complex solution set.

One-dimensional sets are relatively straightforward [67]. Let $f(x) = 0$ be a polynomial system with real coefficients. Assume that the solution set is one complex dimension, i.e., $\dim \mathcal{V}(f) = 1$. We know that $\dim_{\mathbb{R}} \mathcal{V}_{\mathbb{R}}(f) \le 1$. We assume for simplicity that all the positive dimensional

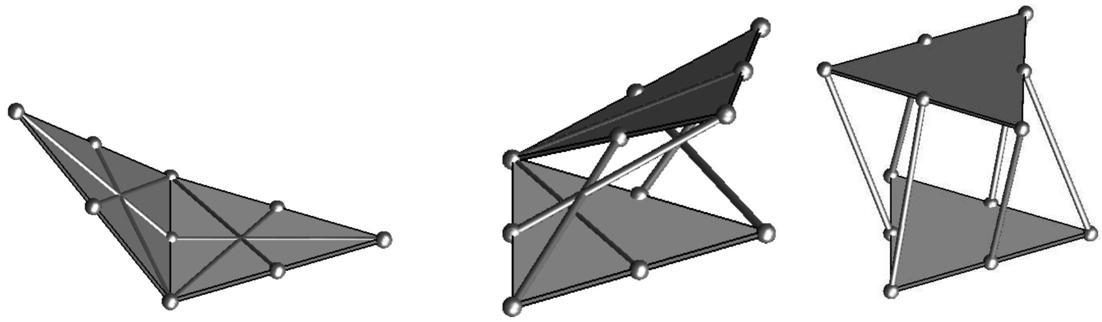


Figure 12: Selected poses of the foldable Stewart-Gough platform. From [67] by permission, ©SIAM.

components are of multiplicity one (by using deflation one can always make this so).

Choose a general linear projection $\pi : \mathbb{C}^N \to \mathbb{C}$. The projection π is of the form

$$A \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} + b$$

where A is a $1 \times N$ generic complex matrix and b is a random complex constant. If we restrict A and b to be real, the set $\mathcal{V}_{\mathbb{R}}(f)$ will map to the real line. We can now compute the set of points $\mathcal{S}$ where $\pi_{\mathcal{V}(f)}$ is not of rank 1. Away from these points $\pi_{\mathcal{V}(f)}$ is a covering map. We know that $\mathcal{V}_{\mathbb{R}}(f)$ is a union of isolated solutions (which must lie in $\mathcal{S}$) plus intervals meeting and ending at points of $\mathcal{S}$. This information is enough to give a decomposition of $\mathcal{V}_{\mathbb{R}}(f)$ that may be used in applications.

The case when $\dim \mathcal{V}_{\mathbb{R}}(f) = 2$ is more difficult. It is natural to choose a random linear projection to $\mathbb{C}^2$ which has real coefficients so that $\mathcal{V}_{\mathbb{R}}(f)$ maps to $\mathbb{R}^2$. A decomposition of $\mathcal{V}_{\mathbb{R}}(f)$ into isolated points, intervals, and cells may be done if all components are multiplicity one. When some components have higher multiplicity, one must introduce deflation, but this brings in some technical issues that are not easily dispatched. Thus the treatment of the two-dimensional case is at present only partially complete.

In § 7.5, we illustrated the Griffis-Duffy Type I platform robot, a special case of the Stewart-Gough (6-SPS) platform and mentioned that the motion for the Griffis-Duffy Type II subcase factors into five pieces: four sextics and one quartic. In [67], an even more special example of a Griffis-Duffy Type II robot is considered, one whose leg lengths are all equal to the altitude of the base and moving triangles (with are congruent equilateral triangles). This robot is unusual in that it can fold up into the plane with both triangles coinciding. Its motion is a curve that factors even more finely than general Type II cases into three double lines, three quadrics, and four quartics. (The sum of the degrees $3 \cdot 2 + 3 \cdot 2 + 4 \cdot 4 = 28$ is the same as the degree of the irreducible curve in the Type I case.) Numerical irreducible decomposition finds this factorization, and the technique sketched above for extracts the real curves inside these complex factors. It turns out that three of the quartics have no real points, but each of the others gives a real curve. One pose on each type of curve is shown in Figure 12. Notice that the double lines are nonreduced, so deflation is needed to sample points on them.

12 Exceptional Sets

Many problems may be rephrased as a problem of finding the set of parameters where some exceptional behavior occurs. An important example that motivated our much of our work is finding overconstrained mechanisms, i.e., mechanisms of a given family that have more degrees of freedom than most of the other mechanisms in the family.

Fiber products give a powerful handle on the computation of exceptional sets. Before we say more, let's give a definition and an example.

Let $p : X \to Y$ and $q : W \to Y$ be two algebraic maps between algebraic sets. Then the fiber product $X \times_Y W$ is the set

$$\left\{(x, w) \in X \times W \;\middle|\; p(x) = q(w)\right\}.$$

Note that $X \times_Y W$ is an algebraic set and has natural maps to X, W, Y. If Y is a point, the fiber product is the usual product.

On the level of systems of polynomials, let $f(x, y) = 0$ and $g(w, y) = 0$ be two systems of polynomials on $\mathbb{C}^{N_1} \times \mathbb{C}^M$ and $\mathbb{C}^{N_2} \times \mathbb{C}^M$ respectively. Letting $X = \mathcal{V}(f(x, y))$ and $W = \mathcal{V}(g(w, y))$, we have the maps $p : X \to Y$ and $q : W \to Y$ induced by the natural projections to $\mathbb{C}^M$. The fiber product $X \times_Y W$ equals $\mathcal{V}(f(x, y), g(w, y))$.

Letting $X = \mathcal{V}(f(x, y))$, the fiber product $X \times_Y X$ of X with itself over Y is the solution set of $(f(x_1, y), f(x_2, y)) = 0$.

The striking property of the fiber product is that exceptional sets correspond to irreducible components of fiber products. As an example, consider the algebraic set $X = \mathcal{V}(x - u, xy - v)$, which we would like to consider as the solution set of a family of systems having (x, y) as variables and (u, v) as parameters. Note the map $(x, y, u, v) \mapsto (u, v)$ restricted to X has a zero-dimensional fiber over general (u, v), given by $(x, y, u, v) = (\lambda, \mu/\lambda, \lambda, \mu)$, but it has a one-dimensional fiber X_1 over $(0, 0)$ given by $(x, y, u, v) = (0, \lambda, 0, 0)$, where $\lambda, \mu \in \mathbb{C}$. [Also, the fiber over $(0, \lambda)$ for $\lambda \neq 0$ is empty, but this is not our main concern at the moment.] Since X_1 has larger dimension than the fiber over general points, it is an exceptional set. Since the whole set X can be parameterized as $(\lambda, \mu, \lambda, \lambda\mu)$, it is isomorphic to $\mathbb{C}^2$. It is therefore one irreducible set that contains X_1, and so X_1 does not stand out as a component on its own in an irreducible decomposition of X. However, the fiber product of X with itself is given by $\mathcal{V}(x_1 - u, x_1y_1 - v, x_2 - u, x_2y_2 - v)$, and it can be shown that the solution set of this system has two components. One is the two-dimensional set of the form $(x_1, y_1, x_2, y_2, u, v) = (\lambda, \mu, \lambda, \mu, \lambda, \lambda\mu)$ that maps onto $\mathbb{C}^2$, and the other is the two dimensional set of the form $(x_1, y_1, x_2, y_2, u, v) = (0, \lambda, 0, \mu, 0, 0)$ that maps onto $(0, 0)$. This latter component is the fiber product of X_1 with itself, and it now stands out as an irreducible component.

In the example above, one fiber product was sufficient to promote the exceptional set to irreducibility. In general, it may take several fiber products, e.g., $X \times_Y X \times_Y X$, and so on, to

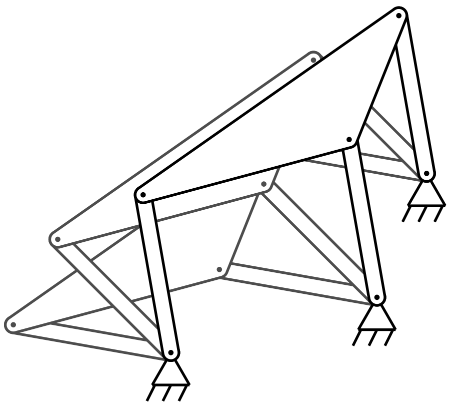


Figure 13: Double-parallelogram linkage.

make the exceptional set stand out. The main results of [100] show that taking higher and higher fiber products will eventually promote any exceptional set to irreducibility. It makes precise how exceptional sets become irreducible components and gives some bounds on how many successive fiber products are needed for different exceptional sets. The polynomial systems that arise from fiber products grow large quickly, but they have much internal structure with many components that are irrelevant for finding exceptional sets. There is promise that this approach will yield effective numerical algorithms to compute overconstrained mechanisms.

We have already touched on several overconstrained mechanisms in this paper. First, and simplest, is the planar four-bar linkage of Figure 4. This is overconstrained in the sense that a 4R spatial closed-loop mechanism (four links connected by rotational joints 1 to 2, 2 to 3, 3 to 4, and 4 to 1) cannot move. For 4×4 link transforms $A_j \in SE(3)$, $j = 1, 2, 3, 4$, the condition for closing the 4R loop is

$$R_z(\Theta_1)A_1R_z(\Theta_2)A_2R_z(\Theta_3)A_3R_z(\Theta_4)A_4 = I_4, \tag{82}$$

where $R_z(\Theta)$ is as in Eq. 12 and I_4 is the 4×4 identity matrix. When the A_j are general, this system is equivalent to six independent equations (the dimension of $SE(3)$)in only four unknowns, so there are no solutions. However, when the A_j are constrained to $SE(2)$, the mechanism becomes planar and we have just three independent equations (the dimension of $SE(3)$) in four unknowns, which allows in general a 1 dimensional motion (1DOF). Similarly, with A_j restricted to $SO(3)$, we obtain the spherical four-bar linkages, which also have 1DOF. These cases are easy to see by merely considering the subgroups $SE(2)$ and $SO(3)$ in $SE(3)$, but these do not tell the whole story. In 1903, Bennett [18] made the surprising discovery that there is another family of moveable four-bar linkages, now called Bennett four-bars, that have special relations between the four link transforms A_j. In 1922, Delassus [32] proved that there are no other families of moveable four-bars.

Outside of four-bar linkages, the theory of overconstrained mechanisms is not so complete. The Griffis-Duffy Type 1 family of 6-SPS mechanisms illustrated in Figure 11 is an example of what is known as an architecturally-singular Stewart-Gough mechanism. Such mechanisms have the property that if one locks the six leg lengths at the values corresponding to any pose of the upper platform, the resulting 6-SS mechanism still moves, whereas a general 6-SS platform can only be assembled in 40 isolated configurations. Karger[54, 55] has classified all the architecturally-singular Stewart-Gough platforms. However, there also exist 6-SS platforms [41] that move but are not covered by Karger's analysis, because they are not architecturally singular: the movement of the associated 6-SPS platform only occurs when locking the leg lengths at special poses of the upper platform. While Geiss and Schreyer found such an example by generalizing to the reals from a search conducted over finite fields of small characteristic, there is not yet any general theory for classifying all such mechanisms.

For a simpler example, consider the planar analog of the spatial 6-SS mechanism, which is the planar 3-RR mechanism, also known as the planar pentad, of Figure 6. It can be shown that the only non-trivial moving pentads (that is, ignoring cases with some link length is equal to zero) are the double-parallelogram linkages, as illustrated in Figure 13. These have upper and ground links that are congruent triangles, and legs that are all equal length. The upper triangle moves in a circular fashion without rotating. In the figure, two poses of the linkage are shown in gray and a final one in black. The question of the existence of a moving pentad is equivalent to asking if a four-bar coupler curve can have a circle as one factor. (See Figure 4(b).)

The relationship between moving pentads and four-bars that trace a circle illustrates the fact that a search for a mechanism with special motion characteristics can often be cast as a search for a related overconstrained mechanism. In this way, we see that methods for finding overconstrained linkages actually have a wider applicability to finding non-overconstrained mechanisms that have special motions. One classical question of this type, discussed briefly in § 5, was the search in the late 1800s to find mechanisms with only rotational joints that draw an exact straight line. The modern version of this is the search for 3DOF spatial parallel-link robots whose output link translates without rotating [22, 33, 42, 48, 57, 58, 66, 109].

To date, investigations of overconstrained mechanisms tend to employ specialized arguments for the specific mechanism family under consideration. The fiber product approach to finding exceptional sets has the potential to provide a general approach applicable to many mechanism families.

Part V

Conclusions

From its roots in methods to find isolated roots of polynomial systems by continuation, numerical algebraic geometry has matured into a set of tools for finding and manipulating solution sets of any dimension. The key step is to use generic linear slices to isolate a finite number of points on a positive dimensional set, thereby forming a finite witness set that becomes a proxy for the entire set. Of central importance are methods to compute the numerical irreducible decomposition, which consists of a witness set for each irreducible component of an algebraic set. Through the use of witness sets to represent intermediate results, a large system can be solved introducing one equation at a time, an approach that often reveals simplifications as the computation progresses and thereby reduces the computations required in the most expensive final stages. While advanced formulations of homotopies reduce the number of continuation paths to be tracked, the numerical techniques required to implement them are of equal importance. Adaptive multiprecision is an important factor in obtaining high reliability, and the use of parallel computing greatly increases speed.

These advances in methods for solving polynomial systems are quite useful in treating problems from algebraic kinematics. We have reviewed some of the specific problems from that field which have provided useful information to kinematicians while also motivating the development of better algorithms. Furthermore, we have shown how problems in kinematics can be formulated as algebraic systems, thereby introducing the concept of a mechanism space and its associated input and output maps. This provides a framework for understanding the definitions of a variety of kinematics problems, including analysis problems, such as the forward and inverse kinematics problems for robots, and synthesis problems that seek to design mechanisms that produce a desired motion.

The current frontiers of work in the area include improving methods for working with real sets, which is of obvious importance in applications, and furthering initial progress that has been made towards algorithms to find exceptional sets. In algebraic kinematics, exceptional sets correspond to overconstrained mechanisms, an area of interest since the earliest days of formal kinematics but which to date has resisted efforts to develop effective general methods.

Acknowledgments

We thank Jonathan Hauenstein for helpful comments.

Part VI

Appendices

A Study coordinates

Study coordinates, also known as dual quaternions and called soma coordinates by E. Study[20], represent $SE(3)$ as points on a quadratic surface, the Study quadric, in seven-dimensional projective space, $\mathbb{P}^7$. Although presentations may be found in numerous places, including [103, 52, 68, 87, 53], for completeness we give a brief description here.

The representation may be developed as follows. First, we note that if $e = e_0\mathbf{1}+e_1\mathbf{i}+e_2\mathbf{j}+e_3\mathbf{k}$ is a quaternion and $\mathbf{v} = v_1\mathbf{i} + v_2\mathbf{j} + v_3\mathbf{k}$ is a pure vector (a quaternion with real part equal to zero), then

$$\mathbf{u} = e * \mathbf{v} * e'/(e * e') \tag{83}$$

is also a pure vector. Moreover, $\mathbf{u}$ and $\mathbf{v}$ have the same length $\mathbf{u} * \mathbf{u}' = \mathbf{v} * \mathbf{v}'$. In fact, casting $\mathbf{u}$ and $\mathbf{v}$ as three-vectors, one may see that Eq. 83 is equivalent to multiplication by a 3×3 rotation matrix $R(e) \in SO(3)$. That is, $\mathbf{u} = R(e)\mathbf{v}$ with

$$R(e) = \frac{1}{\Delta(e)} \begin{bmatrix} e_0^2 + e_1^2 - e_2^2 - e_3^2 & 2(-e_0e_3 + e_1e_2) & 2(e_0e_2 + e_1e_3) \\ 2(e_0e_3 + e_2e_1) & e_0^2 - e_1^2 + e_2^2 - e_3^2 & 2(-e_0e_1 + e_2e_3) \\ 2(-e_0e_2 + e_3e_1) & 2(e_0e_1 + e_3e_2) & e_0^2 - e_1^2 - e_2^2 + e_3^2 \end{bmatrix} \tag{84}$$

where $\Delta(e)$ is the squared length of e:

$$\Delta(e) = e * e' = e_0^2 + e_1^2 + e_2^2 + e_3^2. \tag{85}$$

Of course, we must restrict $R(e)$ to the set of quaternions having nonzero length.

In addition to a quaternion e for rotation, Study coordinates include a second quaternion g related to position. Consider the mapping from Study coordinates (e, g) to $\mathbb{C}^4 \times \mathbb{C}^{3\times 3}$ defined by

$$\mathrm{St} : (e, g) \mapsto (g * e'/(e * e'), R(e)), \quad e * e' \neq 0. \tag{86}$$

In this map, we would like $g * e'$ to represent a translation vector, which is a pure vector. Hence, we require

$$Q(e, g) = g_0e_0 + g_1e_1 + g_2e_2 + g_3e_3 = 0, \tag{87}$$

so that the real part of $g * e'$ is zero. $S_6^2 = \mathcal{V}(Q)$ is called the *Study quadric*. Since St, Q, and Δ are all homogeneous, we may regard (e, g) as homogeneous coordinates on seven-dimensional projective space: $[e, g] \in \mathbb{P}^7$. Restricting St to $(S_6^2 \backslash \mathcal{V}(\Delta)) \subset \mathbb{P}^7$ gives a map from a six-dimensional quasiprojective set to $SE(3)$.

The map St induces an isomorphism. That is, given $(\mathbf{p}, C) \in SE(3)$, $C = R(e)$ has a unique inverse up to scale, and then $\mathbf{p} = g * e'/(e * e')$ may be inverted as $g = \mathbf{p} * e$. (Some authors use $\mathbf{p} = 2g * e'/(e * e')$ and $g = (1/2)\mathbf{p} * e$.) Letting c_{ij} be the i, j-th element of C, the formulas for inverting $C = R(e)$ are

$$\begin{aligned} e &= 1 + c_{11} + c_{22} + c_{33} : c_{32} - c_{23} : c_{13} - c_{31} : c_{21} - c_{12} \\ &= c_{32} - c_{23} : 1 + c_{11} - c_{22} - c_{33} : c_{21} + c_{12} : c_{13} + c_{31} \\ &= c_{13} - c_{31} : c_{21} + c_{12} : 1 - c_{11} + c_{22} - c_{33} : c_{32} + c_{23} \\ &= c_{21} - c_{12} : c_{13} + c_{31} : c_{32} + c_{23} : 1 - c_{11} - c_{22} + c_{33}. \end{aligned} \tag{88}$$

For $C \in SO(3)$, at least one of these four formulas is always nonzero.

References

[1] E. L. Allgower and K. Georg. Continuation and path following. In *Acta numerica*, volume 2, pages 1–64. Cambridge Univ. Press, Cambridge, 1993.

[2] E. L. Allgower and K. Georg. Numerical path following. In *Handbook of numerical analysis, Vol. V*, pages 3–207. North-Holland, Amsterdam, 1997.

[3] E. L. Allgower and K. Georg. *Introduction to numerical continuation methods*, volume 45 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003. Reprint of the 1990 edition [Springer-Verlag, Berlin].

[4] H. Alt. Über die Erzeugung gegebener ebener Kurven mit Hilfe des Gelenkvierecks. *Zeitschrift für Angewandte Mathematik und Mechanik*, 3(1):13–19, 1923.

[5] J. Angeles. *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms, 3rd ed.* Springer Science+Business Media, New York, 2007.

[6] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, second edition, 2006.

[7] D. Bates, J.D. Hauenstein, A. J. Sommese, and C. W. Wampler. Adaptive multiprecision path tracking. *SIAM J. Numer. Anal.*, 46:722–746, 2008.

[8] D. Bates, J.D. Hauenstein, A. J. Sommese, and C. W. Wampler. Stepsize control for adaptive multiprecision path tracking. *Contemporary Mathematics*, 496:21–31, 2009.

[9] D.J. Bates, J.D. Hauenstein, C. Peterson, and A.J. Sommese. A numerical local dimension test for points on the solution set of a system of polynomial equations. *SIAM Journal on Numerical Analysis*, 47:3608–3623, 2009.

[10] D.J. Bates, J.D. Hauenstein, C. Peterson, and A.J. Sommese. Numerical decomposition of the rank-deficiency set of a matrix of multivariate polynomials. In *Approximate Commutative Algebra*, volume 14 of *Texts and Monographs in Symbolic Computation*, pages 55–77. Springer, 2010.

[11] D.J. Bates, J.D. Hauenstein, and A.J. Sommese. A parallel endgame. *preprint*, 2010. Available at www.nd.edu/∼sommese/preprints.

[12] D.J. Bates, J.D. Hauenstein, and A.J. Sommese. Efficient pathtracking methods. *Numerical Algorithms*, to appear 2011.

[13] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. Bertini: Software for numerical algebraic geometry. Available at www.nd.edu/∼sommese/bertini.

[14] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. Software for numerical algebraic geometry: a paradigm and progress towards its implementation. In M.E. Stillman, N. Takayama, and J. Verschelde, editors, *Software for Algebraic Geometry*, volume 148 of *IMA Volumes in Mathematics and Its Applications*, pages 1–14. Springer, New York, 2008.

[15] D.J. Bates, C. Peterson, and A.J. Sommese. A numerical-symbolic algorithm for computing the multiplicity of a component of an algebraic set. *Journal of Complexity*, 22:475–489, 2006.

[16] D.J. Bates, C. Peterson, and A.J. Sommese. Applications of a numerical version of Terracini's lemma for secants and joins. In A. Dickenstein, F.-O. F.-O. Schreyer, and A.J. Sommese, editors, *Algorithms in Algebraic Geometry*, pages 1–14. Springer, New York, 2008.

[17] M. C. Beltrametti and A. J. Sommese. *The adjunction theory of complex projective varieties*, volume 16 of *de Gruyter Expositions in Mathematics*. Walter de Gruyter & Co., Berlin, 1995.

[18] G.T. Bennett. A new mechanism. *Engineering*, 76:777–778, 1903.

[19] I. Bonev. The true origins of parallel robots. www.parallemic.org/Reviews/Review007.html, Jan. 2003. accessed Dec. 14, 2010.

[20] O. Bottema and B. Roth. *Theoretical Kinematics*, volume 24 of *North-Holland Series in Applied Mathematics and Mechanics*. North-Holland Publishing Co., Amsterdam, 1979. Reprinted by Dover Publications, New York, 1990.

[21] J.W. Boyse and J.E. Gilchrist. GMSOLID—interactive modeling for design and analysis of solids. *IEEE Comput. Graphics Appl.*, 2:27–40, 1982.

[22] M. Carricato and V. Parenti-Castelli. A family of 3-dof translational parallel manipulators. *Journal of Mechanical Design*, 125(2):302–307, 2003.

[23] A. Cayley. On three-bar motion. *Proc. London Math. Soc.*, VII:136–166, 1876.

[24] P.L. Chebyshev. Théorie des mécanismes connus sous le nom de parallélogrammes. *Mémoires des Savants étrangers présentes à l'Académie de Saint-Pétersbourg*, 7:539–568, 1854.

[25] S. N. Chow, J. Mallet-Paret, and J. A. Yorke. A homotopy method for locating all zeros of a system of polynomials. In *Functional differential equations and approximation of fixed points (Proc. Summer School and Conf., Univ. Bonn, Bonn, 1978)*, volume 730 of *Lecture Notes in Math.*, pages 77–88. Springer, Berlin, 1979.

[26] W.K. Clifford. On the triple generation of three-bar curves. *Proc. London Math. Soc.*, 9:27–28, 1878.

[27] D. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, second edition, 1997. An introduction to computational algebraic geometry and commutative algebra.

[28] D. Cox, J. Little, and D. O'Shea. *Using algebraic geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1998.

[29] B. Dayton and Z. Zeng. Computing the multiplicity structure in solving polynomial systems. In *Proc. of ISSAC 2005*, pages 166–123, New York, 2005. ACM.

[30] B. H. Dayton, T.-Y. Li, and Z. Zeng. Multiple zeros of nonlinear systems. *preprint*, accessed Dec. 2010. Available at orion.neiu.edu/∼zzeng.

[31] J. de Groot. *Bibliography on Kinematics*. Eindhoven Univ. of Tech., Eindhoven-Holland, 1970.

[32] E. Delassus. Les chaînes articulées fermées at déformables à quatre membres. *Bulletin des Sciences Mathématiques et Astonomiques*, 46:283–304, 1922.

[33] R. Di Gregorio and V. Parenti-Castelli. A translational 3-DOF parallel manipulator. In J. Lenarcic and M. L. Husty, editors, *Advances in Robot Kinematics: Analysis and Control*, pages 49–58. Kluwer Academic Publishers, Dordrecht, 1998.

[34] E.A. Dijksman. *Motion Geometry of Mechanisms.* Cambridge Univ. Press, Cambridge, 1976.

[35] F. J. Drexler. Eine Methode zur Berechnung sämtlicher Lösungen von Polynomgleichungssystemen. *Numer. Math.*, 29(1):45–58, 1977.

[36] J. Duffy and C. Crane. A displacement analysis of the general spatial 7-link, 7R mechanism. *Mechanism Machine Theory*, 15(3–A):153–169, 1980.

[37] F. Freudenstein and B. Roth. Numerical solution of systems of nonlinear equations. *J. ACM*, 10(4):550–556, 1963.

[38] F. Freudenstein and G.N. Sandor. Synthesis of path-generating mechanisms by means of a programmed digital computer. *ASME J. Eng. Industry*, 81:159–168, 1959.

[39] C. B. García and W. I. Zangwill. Finding all solutions to polynomial systems and other systems of equations. *Math. Programming*, 16(2):159–176, 1979.

[40] C.B. Garcia and T.Y. Li. On the number of solutions to polynomial systems of equations. *SIAM J. Numer. Anal.*, 17(4):540–546, 1980.

[41] F. Geiss and F.-O. Shreyer. A family of exceptional Stewart-Gough mechanisms of genus 7. In D.J. Bates, G.-M. Besana, S. Di Rocco, and C.W. Wampler, editors, *Interactions of Classical and Numerical Algebraic Geometry*, volume Contemporary Mathematics 496, pages 221–234. Amer. Math. Soc., 2009.

[42] G. Gogu. Structural synthesis of fully-isotropic translational parallel robots via theory of linear transformations. *European Journal of Mechanics - A/Solids*, 23(6):1021 – 1039, 2004.

[43] P.A. Griffths and J. Harris. *Principles of Algebraic Geometry.* Wiley Classics Library. John Wiley & Sons Inc., New York, 1994. Reprint of the 1978 original.

[44] G.B. Halsted. Biography. Pafnutij Lvovitsch Tchebychev. *American Mathematical Monthly*, II(3), Mar. 1895.

[45] R. Hartshorne. *Algebraic Geometry.* Springer-Verlag, New York, 1977. Graduate Texts in Mathematics, No. 52.

[46] J. Hauenstein, A.J. Sommese, and C.W. Wampler. Regeneration homotopies for solving systems of polynomials. *Mathematics of Computation*, 80:345–377, 2011.

[47] J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. Regenerative cascade homotopies for solving polynomial systems. *preprint*, 2010. Available at www.nd.edu/∼sommese/preprints.

[48] T. Huang, Z. Li, M. Li, D.G. Chetwynd, and C.M. Gosselin. Conceptual design and dimensional synthesis of a novel 2-DOF translational parallel robot for pick-and-place operations. *Journal of Mechanical Design*, 126(3):449–455, 2004.

[49] K.H. Hunt. *Kinematic Geometry of Mechanisms.* Clarendon Press (Oxford Univ. Press), Oxford, 1978.

[50] M. L. Husty. An algorithm for solving the direct kinematics of general Stewart-Gough platforms. *Mechanism Machine Theory*, 31(4):365–380, 1996.

[51] M.L. Husty and A. Karger. Self-motions of griffis-duffy type parallel manipulators. In *Proc. 2000 IEEE Int. Conf. Robotics and Automation, CDROM, San Francisco, CA, April 24-28*, 2000.

[52] M.L. Husty, A. Karger, H. Sachs, and W. Steinhilper. *Kinematik und Robotik*. Springer, Berlin, New York, 1997.

[53] M.L. Husty, M. Pfurner, H.-P. Schröcker, and K. Brunnthaler. Algebraic methods in mechanism analysis and synthesis. *Robotica*, 25:661–675, 2007.

[54] A. Karger. Architecture singular planar parallel manipulators. *Mechanism and Machine Theory*, 38(11):1149–1164, Nov. 2003.

[55] A. Karger. Architecturally singular non-planar parallel manipulators. *Mechanism and Machine Theory*, 43(3):335 – 346, 2008.

[56] A.B. Kempe. *How to draw a straight line: A lecture on linkages*. MacMillan and Co., London, 1877. Available at: www2.cddc.vt.edu/gutenberg/2/5/1/5/25155/25155-pdf.pdf.

[57] X. Kong and C.M Gosselin. Kinematics and singularity analysis of a novel type of 3-CRR 3-DOF translational parallel manipulator. *Int. J. Robotics Research*, 21(9):791–798, 2002.

[58] X. Kong and C.M Gosselin. Type synthesis of 3-dof translational parallel manipulators based on screw theory. *Journal of Mechanical Design*, 126(1):83–92, 2004.

[59] D. Lazard. On the representation of rigid-body motions and its application to generalized platform manipulators. In J. Angeles, P. Kovacs, and G. Hommel, editors, *Computational Kinematics*, pages 175–182. Kluwer, 1993.

[60] T.-L. Lee, T.-Y. Li, and C.H. Tsai. HOM4PS-2.0: a software package for solving polynomial systems by the polyhedral homotopy continuation method. *Computing*, 83:109–133, 2008.

[61] A. Leykin, J. Verschelde, and A. Zhao. Newton's method with deflation for isolated singularities of polynomial systems. *Theor. Comp. Sci*, 359:111–122, 2006.

[62] A. Leykin, J. Verschelde, and A. Zhao. Higher-order deflation for polynomial systems with isolated singular solutions. In A. Dickenstein, F.-O. Schreyer, and A.J. Sommese, editors, *Algorithms in Algebraic Geometry*, pages 79–97. Springer, New York, 2008.

[63] T.-Y. Li. Numerical solution of multivariate polynomial systems by homotopy continuation methods. In *Acta numerica*, volume 6, pages 399–436. Cambridge Univ. Press, Cambridge, 1997.

[64] T.-Y. Li. Numerical solution of polynomial systems by homotopy continuation methods. In *Handbook of numerical analysis, Vol. XI*, pages 209–304. North-Holland, Amsterdam, 2003.

[65] T.-Y. Li, T. Sauer, and J. A. Yorke. The cheater's homotopy: an efficient procedure for solving systems of polynomial equations. *SIAM J. Numer. Anal.*, 26(5):1241–1251, 1989.

[66] Y. Li and Q. Xu. Kinematic analysis and design of a new 3-dof translational parallel manipulator. *Journal of Mechanical Design*, 128(4):729–737, 2006.

[67] Y. Lu, D. Bates, A.J. Sommese, and C.W. Wampler. Finding all real points of a complex curve. In *Proceedings of the Midwest Algebra, Geometry and Its Interactions Conference*, volume Contemporary Mathematics 448, pages 183–205. AMS, 2007.

[68] J.M. McCarthy. *Geometric Design of Linkages*. Springer, New York, 2000.

[69] A. P. Morgan. A transformation to avoid solutions at infinity for polynomial systems. *Appl. Math. Comput.*, 18(1):77–86, 1986.

[70] A. P. Morgan and A. J. Sommese. A homotopy for solving general polynomial systems that respects m-homogeneous structures. *Appl. Math. Comput.*, 24:101–113, 1987.

[71] A. P. Morgan and A. J. Sommese. Computing all solutions to polynomial systems using homotopy continuation. *Appl. Math. Comput.*, 24:115–138, 1987. Errata: *Appl. Math. Comput.* 51 (1992), p. 209.

[72] A. P. Morgan and A. J. Sommese. Coefficient-parameter polynomial continuation. *Appl. Math. Comput.*, 29(2):123–160, 1989. Errata: *Appl. Math. Comput.* 51:207(1992).

[73] A. P. Morgan, A. J. Sommese, and C. W. Wampler. Computing singular solutions to nonlinear analytic systems. *Numer. Math.*, 58(7):669–684, 1991.

[74] A. P. Morgan, A. J. Sommese, and C. W. Wampler. Computing singular solutions to polynomial systems. *Adv. in Appl. Math.*, 13(3):305–327, 1992.

[75] A. P. Morgan, A. J. Sommese, and C. W. Wampler. A power series method for computing singular solutions to nonlinear analytic systems. *Numer. Math.*, 63(3):391–409, 1992.

[76] A. P. Morgan, A. J. Sommese, and C. W. Wampler. A product-decomposition bound for Bezout numbers. *SIAM J. Numer. Anal.*, 32(4):1308–1325, 1995.

[77] A.P. Morgan. *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*. Prentice Hall Inc., Englewood Cliffs, NJ, 1987.

[78] B. Mourrain. The 40 generic positions of a parallel robot. In M. Bronstein, editor, *Proc. ISSAC'93 (Kiev)*, pages 173–182. ACM Press, July 1993.

[79] T. Ojika. Modified deflation algorithm for the solution of singular problems. I. A system of nonlinear algebraic equations. *J. Math. Anal. Appl.*, 123:199–221, 1987.

[80] T. Ojika, S. Watanabe, and T. Mitsui. Deflation algorithm for the multiple roots of a system of nonlinear equations. *J. Math. Anal. Appl.*, 96:463–479, 1983.

[81] M. Raghavan. The Stewart platform of general geometry has 40 configurations. In *Proc. ASME Design and Automation Conf., vol. 32-2*, pages 397–402. ASME, 1991.

[82] M. Raghavan. The Stewart platform of general geometry has 40 configurations. *ASME J. Mech. Design*, 115:277–282, 1993.

[83] S. Roberts. On three-bar motion in plane space. *Proc. London Math. Soc.*, VII:14–23, 1875.

[84] F. Ronga and T. Vust. Stewart platforms without computer? In *Real analytic and algebraic geometry (Trento, 1992)*, pages 197–212. de Gruyter, Berlin, 1995.

[85] B. Roth and F. Freudenstein. Synthesis of path-generating mechanisms by numerical means. *J. Eng. Industry*, pages 298–306, 1963. Trans. ASME, vol. 85, Series B.

[86] A. Schoenflies and M. Greubler. Kinematik. In *Enzyclopaedie der Mathematischen Wissenschaften*, volume 3, pages 190–278. Verlag von B.G. Teubner, Leipzig, 1902.

[87] J.M. Selig. *Geometric Fundamentals of Robotics, Second Edition.* Monographs in Computer Science. Springer-Verlag, New York, 2005.

[88] A. J. Sommese, J. Verschelde, and C. W. Wampler. Numerical irreducible decomposition using projections from points on the components. In *Symbolic computation: solving equations in algebra, geometry, and engineering (South Hadley, MA, 2000)*, volume 286 of *Contemp. Math.*, pages 37–51. Amer. Math. Soc., Providence, RI, 2001.

[89] A. J. Sommese and C. W. Wampler. Numerical algebraic geometry. In *The mathematics of numerical analysis (Park City, UT, 1995)*, volume 32 of *Lectures in Appl. Math.*, pages 749–763. Amer. Math. Soc., Providence, RI, 1996.

[90] A.J. Sommese and J. Verschelde. Numerical homotopies to compute generic points on positive dimensional algebraic sets. *J. Complexity*, 16(3):572–602, 2000.

[91] A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical decomposition of the solution sets of polynomial systems into irreducible components. *SIAM J. Numer. Anal.*, 38(6):2022–2046, 2001.

[92] A.J. Sommese, J. Verschelde, and C.W. Wampler. Using monodromy to decompose solution sets of polynomial systems into irreducible components. In *Applications of algebraic geometry to coding theory, physics and computation (Eilat, 2001)*, volume 36 of *NATO Sci. Ser. II Math. Phys. Chem.*, pages 297–315. Kluwer Acad. Publ., Dordrecht, 2001.

[93] A.J. Sommese, J. Verschelde, and C.W. Wampler. A method for tracking singular paths with application to the numerical irreducible decomposition. In *Algebraic geometry*, pages 329–345. de Gruyter, Berlin, 2002.

[94] A.J. Sommese, J. Verschelde, and C.W. Wampler. Symmetric functions applied to decomposing solution sets of polynomial systems. *SIAM J. Numer. Anal.*, 40(6):2026–2046, 2002.

[95] A.J. Sommese, J. Verschelde, and C.W. Wampler. Advances in polynomial continuation for solving problems in kinematics. *Journal of Mechanical Design*, 126(2):262–268, 2004.

[96] A.J. Sommese, J. Verschelde, and C.W. Wampler. Homotopies for intersecting solution components of polynomial systems. *SIAM J. Numer. Anal.*, 42(4):1552–1571, 2004.

[97] A.J. Sommese, J. Verschelde, and C.W. Wampler. An intrinsic homotopy for intersecting algebraic varieties. *J. Complexity*, 21(4):593–608, 2005.

[98] A.J. Sommese, J. Verschelde, and C.W. Wampler. Solving polynomial systems equation by equation. In *Algorithms in algebraic geometry*, volume 146 of *IMA Vol. Math. Appl.*, pages 133–152. Springer, New York, 2008.

[99] A.J. Sommese and C.W. Wampler. *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science.* World Scientific, Singapore, 2005.

[100] A.J. Sommese and C.W. Wampler. Exceptional sets and fiber products. *Foundations of Computational Mathematics*, 28:171–196, 2008.

[101] H. J. Stetter. *Numerical polynomial algebra.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2004.

[102] E. Study. Von den bewegungen und umlegungen. *Mathematische Annalen*, 39:441–556, 1891.

[103] E. Study. *Geometrie der Dynamen.* B.G. Teubner, Leipzig, 1903.

[104] H.-J. Su, J.M. McCarthy, M. Sosonkina, and L.T. Watson. Algorithm 857: POLSYS_GLP—a parallel general linear product homotopy code for solving polynomial systems of equations. *ACM Trans. Math. Software*, 32:561–579, 2006.

[105] H.-J. Su, J.M. McCarthy, and L.T. Watson. Generalized linear product homotopy algorithms and the computation of reachable surfaces. *J. Comput. Inf. Sci. Eng.*, 4(3):226–234, 2004.

[106] J.J. Sylvester. On recent discoveries in mechanical conversion of motion. *Proc. Royal Institution of Great Britain*, 7:179–198, 1874.

[107] H. Tari, H.-J. Su, and T.-Y. Li. A constrained homotopy technique for excluding unwanted solutions from polynomial equations arising in kinematics problems. *Mechanism Machine Theory*, 45(6):898–910, 2010.

[108] L. W. Tsai and A. P. Morgan. Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods. *ASME J. Mech., Trans., Auto. Design*, 107:48–57, 1985.

[109] L.-W. Tsai, G.C. Walsh, and R.E. Stamper. Kinematics of a novel three dof translational platform. In *Proc. 1996 IEEE Int. Conf. Robotics and Automation*, volume 4, pages 3446 –3451, 1996.

[110] J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. on Math. Software*, 25(2):251–276, 1999.

[111] J. Verschelde and R. Cools. Symbolic homotopy construction. *Appl. Algebra Engrg. Comm. Comput.*, 4(3):169–183, 1993.

[112] C. W. Wampler. Forward displacement analysis of general six-in-parallel SPS (Stewart) platform manipulators using soma coordinates. *Mechanism Machine Theory*, 31:331–337, 1996.

[113] C. W. Wampler, A. P. Morgan, and A. J. Sommese. Complete solution of the nine-point path synthesis problem for four-bar linkages. *ASME J. Mech. Design*, 114:153–159, 1992.

[114] C. W. Wampler, A. P. Morgan, and A. J. Sommese. Complete solution of the nine-point path synthesis problem for four-bar linkages - Closure. *ASME J. Mech. Design*, 119:150–152, 1997.

[115] C.W. Wampler, J.D. Hauenstein, and A.J. Sommese. Mechanism mobility and a local dimension test. *Mechanism Machine Theory*, 2011. (to appear).

[116] L. T. Watson, M. Sosonkina, R. C. Melville, A. P. Morgan, and H. F. Walker. Algorithm 777: HOMPACK90: a suite of Fortran 90 codes for globally convergent homotopy algorithms. *ACM Trans. Math. Software*, 23(4):514–549, 1997.

[117] S. M. Wise, A. J. Sommese, and L. T. Watson. Algorithm 801: POLSYS_PLP: A partitioned linear product homotopy code for solving polynomial systems of equations. *ACM Trans. Math. Software*, 26(1):176–200, 2000.

[118] Z. Zeng. The closedness subspace method for computing the multiplicity structure of a polynomial system. In *Interactions of classical and numerical algebraic geometry*, volume 496 of *Contemp. Math.*, pages 347–362. Amer. Math. Soc., Providence, RI, 2009.