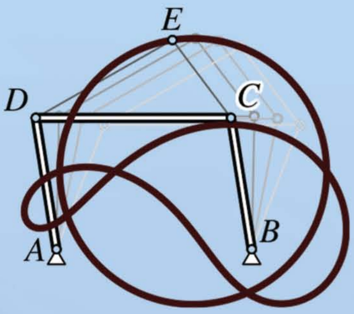
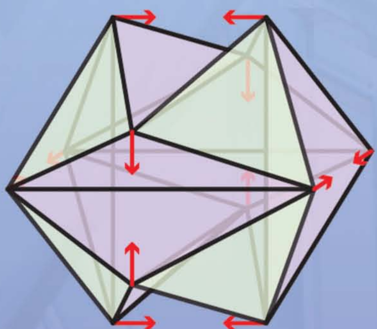
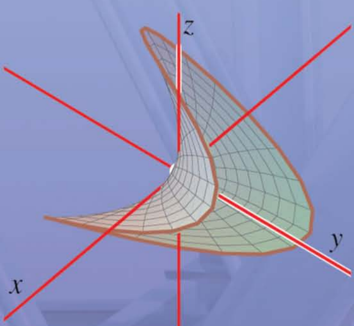
DISCRETE MATHEMATICS AND ITS APPLICATIONS



HANDBOOK OF GEOMETRIC CONSTRAINT SYSTEMS PRINCIPLES



EDITED BY
Meera Sitharam
Audrey St. John
Jessica Sidman



CPD CERTIFIED
The CPD Certification Service

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Handbook of Geometric Constraint Systems Principles

Discrete Mathematics and Its Applications

Series Editors
Miklos Bona
Donald L. Kreher
Douglas West
Patrice Ossona de Mendez

Handbook of Product Graphs, Second Edition
Richard Hammack, Wilfried Imrich, and Sandi Klavžar

Handbook of Mathematical Induction:
Theory and Applications
David S. Gunderson

Graph Polynomials
Yongtang Shi, Matthias Dehmer, Xueliang Li, and Ivan Gutman

Introduction to Combinatorics, Second Edition
W. D. Wallis and J. C. George

Representation Theory of Symmetric Groups
Pierre-Loïc Méliot

Advanced Number Theory with Applications
Richard A. Mollin

A Multidisciplinary Introduction to Information Security
Stig F. Mjølsnes

Combinatorics of Compositions and Words
Silvia Heubach and Toufik Mansour

Handbook of Linear Algebra, Second Edition
Leslie Hogben

Combinatorics, Second Edition
Nicholas A. Loehr

Handbook of Discrete and Computational Geometry, Third Edition
C. Toth, Jacob E. Goodman and Joseph O'Rourke

Handbook of Discrete and Combinatorial Mathematics, Second Edition
Kenneth H. Rosen

Crossing Numbers of Graphs
Marcus Schaefer

Graph Searching Games and Probabilistic Methods
Anthony Bonato and Paweł Prałat

Handbook of Geometric Constraint Systems Principles
Meera Sitharam, Audrey St. John, and Jessica Sidman,

https://www.crcpress.com/Discrete-Mathematics-and-Its-Applications/book-series/CHDISMTHAPP?page=1&order=dtitle&size=12&view=list&status=published,forthcoming

Handbook of Geometric Constraint Systems Principles

Meera Sitharam
Audrey St. John
Jessica Sidman

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed on acid-free paper
Version Date: 20180528

International Standard Book Number-13: 978-1-4987-3891-0 (Hardback)

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

To the works of:

Hilda Polaczek Geiringer 1893–1973

Neil White 1945–2014

Michel Marie Deza 1939–2016

Wenjun Wu 1919–2017

There is no royal road to geometry.
– Euclid

Geometry is not true, it is advantageous.
– Henri Poincare

Geometry, inasmuch as it is concerned with real space, is no longer considered a part of pure mathematics; like mechanics and physics, it belongs among the applications of mathematics.
– Hermann Weyl

Think geometrically, prove algebraically.
– John Tate

Contents

Foreword

Geometric constraint systems arise in a diverse range of applications including: computer aided engineering and architectural design, molecular and materials modeling, robotics and animation, sensor networks, machine learning, and dimension reduction. Broadly, a geometric constraint system (GCS) is defined on a set of geometric primitives (e.g., points, lines, rigid bodies) by specifying geometric relationships (such as distances, angles, or incidences). The core GCS foundations come from at least four interwoven topic areas and research communities: (i) combinatorial and geometric rigidity, (ii) automated geometric theorem proving, (iii) geometrically constrained configuration spaces and, (iv) distance geometry. Indeed, the principles, tools, and techniques rely on invariant theory, combinatorial and discrete geometry, algebraic geometry and topology, convex/semidefinite analysis, with algorithmic foundations and complexity going back to Cauchy, Cayley, Hilbert, Klein, and Maxwell.

With such a rich array of communities working on GCS research, this handbook is intended as an entry point to the principal mathematical and computational tools, techniques and results currently in use. It was born out of continued requests for a single source containing the core principles and results that would be accessible to beginners and experts alike (from the graduate student starting research to the algebraic geometer interested in applications to the roboticist seeking to engineer a swarm of autonomous agents). Recognizing that readers may come from a wide variety of backgrounds, we hope that this book will be a useful tool for navigating the concepts, approaches, and results found in GCS research.

We are grateful to the authors of the chapters that follow; their expertise provides the roadmap for developing a unified view of the varied perspectives. We pledge any royalties toward supporting the activities of the four research communities represented by the four parts of this handbook, especially the activities of young researchers. We would like to thank Louis Theran for his feedback. We thank Rahul Prabhu for his kind and timely expert help with LaTeX. And, finally, we cannot put into words the debt of gratitude owed to our families for their unconditional support and patience during this process.

Meera Sitharam
Audrey St. John
Jessica Sidman

Preface

The goal of this book is to provide a resource for those aiming to become acquainted with the fundamentals as well as experts looking to pinpoint specific results or approaches in the broad landscape. The flow of the handbook is intended to take readers from the general algebro-geometric approaches to more specialized contexts permitting combinatorial analysis and efficient algorithms. Chapters are grouped by the main techniques being deployed, in the hopes that readers can find the material best-suited to their expertise. Of course, the overlapping nature of the material being presented prevents a neat partitioning of chapters by topic area, but we hope the juxtapositioning of the chapters helps the reader to see how the subject is connected.

Chapter 1 provides an overview of the book as a more detailed starting point and is expected to help the reader navigate the book effectively. It includes a basic introduction, some preliminaries, and an overview of the various topics and methods. We also give an alternative pathway through the book, intended to help a newcomer become acquainted with the domain. We hope this is a first step toward a unifying foundation for the rich set of GCS problems.

Contributors

Abdo Y. Alfakih
University of Windsor
Windsor Ontario, Canada

Troy Baker
University of Florida
Gainesville, Florida

Robert Connelly
Cornell University
Ithaca, New York

Michel-Marie Deza
École Normale Supérieure
Paris, France

Elena I. Deza
Moscow State Pedagogical University
Moscow, Russia

Mathieu Dutour Sikirić
Rudjer Bosković Institute
Zagreb, Croatia

Jacques Fleuriot
University of Edinburgh
Scotland, United Kingdom

Ioannis Fudos
University of Ioannina
Ioannina, Greece

Steven J. Gortler
Harvard University
Cambridge, Massachusetts

Timothy F. Havel
Energy Compression Inc.
Boston, Massachusetts

Christoph M. Hoffmann
Purdue University
West Lafayette, Indiana

Bill Jackson
Queen Mary University of London
London, England

Predrag Janičić
University of Belgrade
Belgrade, Serbia

Robert Joan-Arinyo
Universitat Politècnica de Catalunya
Barcelona, Catalonia

Tibor Jordán
ELTE Eötvös Loránd University and
MTA-ELTE Egerváry Research Group on
Combinatorial Optimization
Budapest, Hungary

Oleg Karpenkov
University of Liverpool
Liverpool, United Kingdom

Csaba Király
ELTE Eötvös Loránd University and
MTA-ELTE Egerváry Research Group on
Combinatorial Optimization
Budapest, Hungary

Derek Kitson
Lancaster University
Lancaster, United Kingdom

Hongbo Li
Chinese Academy of Sciences
Beijing, China

Julien Narboux
University of Strasbourg
Strasbourg, France

Anthony Nixon
Lancaster University
Lancaster, United Kingdom

John C. Owen
Siemens

Cambridge, England

Martin Pfurner
University of Innsbruck
Innsbruck, Austria

Rahul Prabhu
University of Florida
Gainesville, Florida

Jürgen Richter-Gebert
Technical University of Munich
Munich, Germany

Elissa Ross
MESH Consultants Inc., Fields Institute
Toronto, Canada

Josef Schadlbauer
University of Innsbruck
Innsbruck, Austria

Hans-Peter Schröcker
University of Innsbruck
Innsbruck, Austria

Bernd Schulze
Lancaster University
Lancaster, United Kingdom

Brigitte Servatius
Worcester Polytechnic Institute
Worcester, Massachusetts

Herman Servatius
Worcester Polytechnic Institute
Worcester, Massachusetts

Jessica Sidman
Mount Holyoke College
South Hadley, Massachusetts

Meera Sitharam
University of Florida
Gainesville, Florida

Frank Sottile
Texas A&M University
College Station, Texas

Audrey St. John
Mount Holyoke College
South Hadley, Massachusetts

Shin-ichi Tanigawa
The University of Tokyo
Tokyo, Japan

William Traves
US Naval Academy
Annapolis, Maryland

Menghan Wang
University of Florida
Gainesville, Florida

Walter Whiteley
York University
Toronto, Canada

Joel Willoughby
University of Florida
Gainesville, Florida

Chapter 1

Overview and Preliminaries

Meera Sitharam
University of Florida

Troy Baker
University of Florida

CONTENTS

In this chapter, we begin with a generalized introduction to geometric constraint systems before giving an overview of the book's contents. We conclude with a section reconciling terminology and concepts that have arisen in different communities and as well as an alternative pathway through the book especially for the novice reader.

1.1 Introduction

A *geometric constraint system* is generally defined on a finite collection of *geometric primitives* (e.g., 0-dimensional points, 1-dimensional lines, general d-dimensional hyperplanes, d-dimensional rigid bodies, conics, cylinders). The setting is a given Euclidean or non-Euclidean geometry over the reals, generally of fixed dimension, and constraints specify geometric n-ary relationships among the primitives. These constraints can be logical (e.g., incidence, perpendicularity, tangency) or metric (e.g., distance, angle, orientation) and may be either equalities or inequalities. Typically, a constraint can be expressed as a set of quadratic polynomials with real (often rational or even integer) coefficients. The combinatorics of a GCS are usually captured separately in a *constraint graph*: a (hyper)graph where each vertex represents a geometric primitive and each (hyper)edge represents a constraint on the corresponding primitives.

A *realization* (or *solution*) of a GCS is a placement (or configuration) of the geometric primitives that satisfies the constraints. The realizations of a GCS can be found algebraically by solving a system of polynomial equations corresponding to the GCS, where the variables are the coordinates of the geometric primitives. Thus, the set of realizations of such a system consists of the solutions to a finite collection of polynomial equations and is hence a variety. Typically, our primary interest is in the real points of this variety, but if we consider solutions over $\mathbb{C}$, then the full power of algebro-geometric methods may be brought to bear.

In the geometric setting it is generally implied (as it is implied throughout this chapter, unless explicitly stated otherwise) that we are concerned with the realization space modulo some group of *trivial motions* that is designated a priori. For example, in Euclidean space the trivial motions are comprised of translation and rotation; in d-dimensional Euclidean space, there is a d-dimensional space of translations and $\binom{d}{2}$-dimensional space of rotations giving a $\binom{d+1}{2}$-dimensional space of trivial motions. Embedding $\mathbb{R}^d$ into projective space can help us to see unifying principles in incidence and other constraint systems. Sometimes, the realization is *pinned* (or *grounded*), i.e., the trivial group is chosen to be the empty group.

1.1.1 Specifying a GCS

To illustrate these core concepts, consider specifying the most common GCS for classical rigidity theory: the *Euclidean bar-and-joint*, or *Euclidean distance constraint* system. The geometric primitives are 0-dimensional points (called "joints"), the constraints are specified distances between points (called "bars") and the ambient space is $\mathbb{R}^d$. $G=(V,E)$ associates a vertex to each joint and an edge (u,v) to each bar constraining the joints represented by vertices u and v. Then a bar-and-joint constraint system of G can be defined as a tuple (G,δ) where $\delta : E \to \mathbb{R}$ assigns distance values to the bars. A bar-and-joint constraint system is also called a *linkage*. A configuration of the joints in $\mathbb{R}^d$ is given by a map $p : V \to \mathbb{R}^d$ and is a realization of (G,δ) if the distance between $p(u)$ and $p(v)$ is $\delta(u,v)$ for all $(u,v) \in E$. For example, let $(G=(\{1,2\},\{(1,2)\}),\delta)$ be a Euclidean bar-and-joint system consisting of two joints with one bar between them specifying a distance of 4; i.e., $\delta((1,2))=4$. Then, if (x_1,y_1) and (x_2,y_2) are the variables for the coordinates of joints 1 and 2, respectively, realizations of this linkage are the solutions to the single constraint equation

$$(x_1-x_2)^2+(y_1-y_2)^2=\delta^2.$$

Notice that some geometric constraints may lead to multiple equations; if we were to place an incidence constraint between two points, it would give the equations:

$$x_1=x_2 \text{ and } y_1=y_2.$$

Now consider specifying a GCS whose geometric primitives are rigid bodies in Euclidean space; a "bar" constraint can be placed between two bodies by picking a point on each and constraining the distance between them. Such a system describes a *Euclidean body-and-bar constraint system*. Realizations are solutions to the quadratic system of distance equations, i.e., placements of the bodies (e.g., by assigning elements of the special Euclidean group $SE(d)$) that satisfy the bar lengths.

1.1.2 Fundamental GCS Questions

Given a GCS C with n equality constraints, we seek approaches for finding realizations and/or giving structural characterization of C based on properties of the resulting *set* S of geometrically constrained configurations or realizations. That is, when the solution space S has co-dimension n (independent C); is finite (locally rigid C); has the singleton property (globally rigid C). Other properties of S such as dimension (degrees of freedom), connectedness, singularities (deformation paths and extreme configurations) are also of interest. More generally, many of these properties can be deduced by deriving dependent (often inconsistent) constraints that are locally or globally implied by the given GCS, or by ascertaining its independence.

1.1.3 Tractability and Computational Complexity

In its full generality, the fundamental questions encompass the first order theory of the real closed fields. This theory is complete, and automated theorem proving over the reals (RCF) is decidable as shown by Tarski [36] (i.e., does not suffer from Gödel's incompleteness of Peano's first order theory of natural numbers). That said, its algorithmic complexity is essentially that of polynomial ideal membership, commonly using Gröbner bases or cylindrical algebraic decomposition [4, 2], which is prohibitive, being complete for the class EXPSPACE [25]. Even the existential theory of the reals is NP-hard, with the best-known algorithms requiring doubly exponential time [25].

1.2 Parts and Chapters of the Handbook

This handbook contains a sampling of a wide range of theories and methods that attempt to circumvent the above intractability by taking advantage of properties specific to various types of geometric constraint systems. We have organized it into parts based on the main techniques underlying each chapter, starting with the algebro-geometric techniques and concluding with combinatorial approaches. This essentially provides a flow from approaches that address the general (generic and non-generic) GCS setting to those that work under restricted (generic) settings. Indeed, assumptions of genericity appear throughout Part IV, while Part I contains approaches to address non-generic situations.

Part I **Geometric Reasoning Techniques:** Chapters 2–7 address more general geometric constraint systems, with many of the approaches based on algebraic methods.

Part II **Distance Geometry, Configuration Space, and Real Algebraic Geometry Techniques:** Chapters 8–12 span the underlying topics of distance geometry, configuration spaces, and (real) algebraic geometry.

Part III **Geometric Rigidity Techniques:** Chapters 13–17 (mostly based in Rigidity Theory), while often restricted to generic assumptions, require geometric analyses.

Part IV **Combinatorial Rigidity Techniques:** Chapters 18–25 conclude the book with the Rigidity Theory settings that permit combinatorial approaches.

1.2.1 Part I: Geometric Reasoning Techniques

For the specific case of deriving dependent geometric constraints from a GCS, i.e., automated geometry theorem proving, an algorithm that is significantly more efficient than either of the previously mentioned (exponential time and polynomial space) approaches is the Wu-Ritt characteristic set or triangle decomposition method [9, 35]. This and other techniques for automated geometric theorem proving are discussed in Chapter 2.

Since we are restricting ourselves to geometric constraints, the relevant polynomials are typically invariants of transformation or trivial motion groups that define the underlying geometry (Euclidean, Projective, etc.). Invariant polynomials permit synthetic, coordinate-free, and even metric-free computational approaches to deriving dependent constraints, e.g., using bracket algebras. For example, the Grassman-Cayley algebra [40] yields a synthetic computational approach to deriving dependent constraints in projective and incidence geometry. In fact, such invariant theoretic methods even extend to finite geometries [33, 32].

Chapter 5 introduces the bracket algebra and Grassmann-Cayley algebra for the plane with a view toward proving theorems in Projective and Euclidean geometry. The bracket algebra and Grassmann-Cayley algebra appear again in Chapter 4, this time in the context of nongeneric or special realizations of a GCS. The goal is to provide a geometric meaning to the algebraic condition (so-called *pure condition*) that makes the realization special, and this involves the technique of *multilinear Cayley factorization*. The chapter includes introductions to projective space, homogeneous coordinates, the Grassmannian, and Plücker coordinates and ends with examples applying the theory to body-and-bar GCS.

As another example of a similar approach, Chapter 3 develops the theory and applications of a variety of types of Euclidean invariants in deriving constraint dependences. The $\binom{n}{2}$ pairwise distance polynomials between n points in real Euclidean space are Euclidean invariants that are related by the Cayley-Menger syzygies. These are described in the following classical theorem on Euclidean distance matrices.

A *Euclidean distance matrix (EDM)* for $\mathbb{R}^d$ is an $n \times n$ square matrix of pairwise (squared) distances between n points in $\mathbb{R}^d$. It is denoted $\Delta_{[n]}$ with distance entries δ_{ij} for $1 \leq i, j \leq n$. For $S \subseteq [n]$, the submatrix Δ_S has entries δ_{ij} for $i, j \in S$. The volume matrix $\hat{\Delta}_S$ is the $|S|+1 \times |S|+1$ matrix obtained from Δ_S by bordering Δ_S with a top row $(0, 1, \ldots, 1)$ and a left column $(0, 1, \ldots, 1)^T$. Now $det(\hat{\Delta}_S)$ computes the volume of the simplex with points in S, and is called a Cayley-Menger determinant [34]. The next theorem effectively says that the volumes of simplices formed by $d+2$ points in $\mathbb{R}^d$ is 0.

Theorem 1.1 (Cayley-Menger Relations) *A real symmetric matrix $\Delta_{[n]}$ with 0 diagonal and positive entries is a Euclidean distance matrix for $\mathbb{R}^d$ only if $det(\hat{\Delta}_S) = 0$ for all $S \subseteq [n]$ with $|S| \geq d+2$.*

A more direct approach for both Euclidean distance or Projective constraint systems is to simply solve for the set of realizations of the constraint system to determine (in)dependent constraints. For example, some GCS permit a ruler and compass type construction of the (finite set of) solutions, which is equivalent to solving a recursive triangular block decomposition of the constraint system. A broad class of GCS occurring in computer aided design are of this type, i.e., their underlying combinatorial structure, or constraint graph is *triangle-decomposable* or *tree-decomposable* [7].

It is known that such GCS are Quadratically Radically Solvable (QRS), i.e., the coordinates of the solutions are in nested quadratic extensions of the coefficient field. As to whether all *generic*

QRS systems have triangle-decomposable constraint graphs is still an open problem, with the equivalence being shown only when the constraint graph is topologically planar [26]. Solving these systems entails two stages: the recursive triangle decomposition stage, a combinatorial procedure on the constraint graph, and a solution or realization stage, that obtains the solution of the corresponding recursively decomposed system through a bottom-up process of assembling or recombining the (generically finitely many) solutions of subsystems.

Chapter 6 gives many natural examples of triangle-decomposable constraint systems and formal algorithms for recursive decomposition and recombination. In general, the time complexity of solving or realization is bottle-necked by the largest subsystem that must be solved simultaneously and a polynomial time preprocessing algorithm to identify the subsystems can be beneficial. For non-triangle-decomposable GCS, the above method can be generalized to obtain recursive decompositions into subgraphs that approximate generically rigid subsystems that have finitely many isolated solutions or realizations (see Section 1.3 for Terminology and Basic Concepts).

The process of identifying these subsystems and finding a partial order in which to solve them is called *decomposition-recombination (DR-) planning*, the topic of Chapter 7. Such algorithms vary in their generality but often leverage geometric properties of specific constraint and primitive types or use a priori knowledge of patterns in their arrangement. However, in general, at some point an algebraic system must be solved for recombination of the decomposed subsystem solutions. Chapter 7 surveys several such methods for decomposition and recombination of more general constraint systems.

However, most of the above-mentioned approaches (including those that rely on generic rigidity or finiteness of the solution set for decomposition) do not differentiate between real and complex solution spaces. In particular to specialize the invariant-theoretic approach to the reals requires imposing additional inequality constraints beyond the Cayley-Menger conditions in the previous theorem, by asserting that all simplices have positive volumes. For 1- and 2-dimensional simplices (line segments and triangles), this gives exactly the metric condition on real Euclidean distances.

Theorem 1.2 (Cayley-Menger Inequalities) *A real symmetric matrix with 0 diagonal and positive entries $\Delta_{[n]}$ is a Euclidean distance matrix for $\mathbb{R}^d$ if and only if $det(\hat{\Delta}_S) \geq 0$ for all $S \subseteq [n], |S| \geq 2$; and $det(\hat{\Delta}_S) = 0$ for all $S \subseteq [n]$ with $|S| \geq d+2$.*

Note that the Euclidean invariant approaches described in Chapter 5 use all the Cayley-Menger conditions including the above inequalities.

1.2.2 Part II: Distance Geometry, Configuration Space, and Real Algebraic Geometry Techniques

The inequalitites in Theorem 1.2 can be viewed as partly arising from the metric property of real Euclidean space. This leads to another tool for dealing with geometric constraints with an underlying metric, namely distance or metric geometry. The classical theorem of Schoenberg [30, 31] (which generalizes to infinite dimensional Hilbert spaces) is stated for finite dimensional real Euclidean distance matrices below. It is equivalent to the conjunction of the two Cayley-Menger theorems above.

Theorem 1.3 (Schoenberg's Theorem) *A real symmetric matrix with 0 diagonal and positive entries $\Delta_{[n]}$ is a Euclidean distance matrix for $\mathbb{R}^d$ if and only if it is negative semidefinite on the subspace of all vectors orthogonal to the all 1's vector and rank$(\Delta_{[n]}) \leq d+1$.*

The convexity and face structure of the Euclidean distance cone yield powerful techniques for understanding distance constraint systems, including implied or dependent constraints, and different types of rigidity. Chapter 8 surveys some of these techniques. Chapter 12 introduces the tools of real algebraic geometry, specifically semialgebraic sets that involve polynomial equalities and inequalities (such as the Cayley-Menger determinantal equalities and inequalities above) and the positivenstellensatz as tools for defining generic realizations and dealing with distance constraint systems. It starts with a brief introduction to the correspondence between ideals and varieties over $\mathbb{C}$, and then turns to a discussion of varieties defined by polynomials with real coefficients, which may be viewed as either real or complex varieties. It culminates with a view of the projection of the d-dimensional stratum of the Euclidean (squared) distance cone as a semialgebraic set and its application to rigidity.

Chapter 9 explores the structure of general metric cones. Chapter 10 employs properties of projections and fibers of rank d strata of the Euclidean distance cones to characterize distance constraint systems (their underlying graphs) whose configuration spaces generically map finitely-many-to-one to a convex set and whose singular configurations have a simple description. These characterizations extend to when the distance constraints are inequalities and the distances are l_p^p norms. The techniques yield interesting configuration space properties of a common class of plane linkage mechanisms (Euclidean distance constraint systems with one degree of freedom in $\mathbb{R}^2$) arising from the QRS or triangle decomposable constraint graphs mentioned above.

However, questions about linkage mechanisms in general are inherently difficult: Kempe's universality theorem [20] states that the space of configurations can trace out any desired algebraic curve. Chapter 11 explores constraint varieties of mechanisms and describes how so-called study parameters and dual quaternions are used in kinematics.

1.2.3 Part III: Geometric Rigidity Techniques

The question of local and global uniqueness of polyhedra whose faces have a given combinatorics has been studied by a long line of researchers starting from Cauchy in the early 1800s to Alexandrov in the 1950s. These results concern the geometric rigidity of *polyhedral frameworks*, or just polyhedra, namely 3-dimensional polytopes that are composed of planar rigid panel faces; face panels can rotate about the edge or hinge on which they are incident with another panel. A *triangulated* polyhedron's faces are all triangular.

Although these results concern the rigidity-related properties of *framework*s, i.e., specific realizations of a combinatorial structure of constraints (from which the GCS can be extracted), as opposed to rigidity-related properties of GCSs (as discussed so far), in the latter part of this chapter we will reconcile the slight differences in these two ways of thinking.

Cauchy [5] showed that all convex polyhedra are *rigid* and in fact globally rigid if convexity is stipulated. (Mistakes in Cauchy's proof were fixed and the result extended by a series of subsequent researchers.) Despite a long standing conjecture that the result extended to all triangulated polyhedra (convex or not), verified for many subclasses [10], the general statement was disproven by counterexample [6]. These results and further developments are discussed in Chapter 13.

Let $G = (V,E)$ be the graph associated to the edge skeleton (or 1-skeleton) of a triangulated polyhedron and let $p : V \to \mathbb{R}^3$ be the map that assigns the coordinates of each vertex of the polyhedron to a vertex of G. Then (G, p) is a *bar-joint framework* that is a realization of a bar-joint constraint system in 3D, with a distance constraint graph G. Thus, Cauchy's theorem shows that bar-joint frameworks arising from convex triangulated polyhedra are rigid (and globally rigid if convexity of the framework is stipulated).

Chapter 16 discusses geometric conditions for global rigidity of *generic* bar-joint frameworks in arbitrary dimensions [11]. Recall that a framework is globally rigid if it is the unique realization of its underlying GCS. A framework (G, p) is *generic w.r.t. a property P* (such as global rigidity) if for some neighborhood $\mathcal{N}(p)$, for all frameworks (G,q) with $q \in \mathcal{N}(p)$ (G,q) satisfies P if and only if (G, p) satisfies P. When the context, namely the property P is clear, we simply say the framework is generic.

The result employs a feature of the framework's equilibrium self-stress (defined in Section 1.3) and further shows that global rigidity is in fact a *generic property*, i.e., either all generic frameworks (G,p) of a graph G are globally rigid or none are. In other words, the property of the framework only depends on the graph G, but is given a geometric characterization in Chapter 16. Chapters that additionally give combinatorial as opposed to geometric characterizations of such properties that depend only on the constraint graph are described in the next section on combinatorial rigidity.

Chapter 14 considers *tensegrity frameworks* [28] in which the underlying GCS involves inequality as opposed to equality constraints. Some edges of the constraint graph, called struts, have distance lower bounds and others, called ties have distance upper bounds which restrict the sign of the equilibrium self-stress they can carry. Tensegrity frameworks that represent packed incompressible spheres contain only struts. Bar-joint frameworks are special cases of tensegrity frameworks where all edges have both distance upper and lower bounds (fixed distances).

Chapter 14 gives geometric, equilibrium self-stress based characterizations for rigidity and other related properties of tensegrity frameworks both in general and generic settings, and connects them to rigidity properties of bar frameworks.

Chapter 15 specifically considers nongeneric tensegrity frameworks and uses extended Cayley algebra (discussed earlier under the Geometric Reasoning Section of the handbook) to give geometric conditions for rigidity related properties.

Chapter 17 deals with properties related to rigidity that are invariant under various transformations (beyond the trivial motion group of the underlying geometry). Using characterizations of properties related to rigidity theory, techniques (such as Coning or Maxwell-Cremona diagrams for understanding stresses) for GCSs and frameworks in one geometry can be extended to another. For example, techniques and characterizations from Euclidean geometry can be extended to say Affine, Projective, Spherical, Minkowski, and Hyperbolic geometries that are defined using Cayley-Klein metrics or using the trivial motion groups under which the metrics are invariant.

1.2.4 Part IV: Combinatorial Rigidity Techniques

As mentioned earlier, (in)dependence and other properties related to rigidity are often generic (under appropriate, careful definitions of genericity), i.e., they hold for all generic frameworks and/or GCSs with a given constraint graph G, or for none of them. They depend only on the underlying constraint graph G; (in)dependence is captured by a geometric or algebraic matroid such as the (generic) rigidity matroid defined formally in the second part of this chapter, whose ground set is related to the algebraic constraints represented by the edges (and nonedges) of G.

This section of the handbook deals with such (in)dependence properties that are equivalently characterized by purely combinatorial sparsity or graphic matroids, which do not use the algebraic structure of the constraint polynomials or brackets over the reals (or of the coefficient field of the constraint polynomials). The book [12] has so far been the trusted source on combinatorial rigidity, but significant progress has been made since it was published.

A classic example of a purely combinatorial rigidity characterization is a celebrated result of Laman [21] published in 1970, though recently it has also been found in a forgotten work of Hilda Pollaczek-Beiringer [27] from 1927.

Theorem 1.4 ([27, 21]) *A 2D bar-joint graph $G=(V,E)$ is rigid (all its generic frameworks/GCSs are rigid) if and only if $|E|=2|V|-3$ and for any subsystem $G'=(V',E')$ where $|V'|>1$, $|E'|\leq 2|V'|-3$.*

Graphs satisfying such counting conditions – which keep track of the internal degrees of freedom (dof) of the system – are often referred to as *Laman graphs*. This combinatorial characterization of bar-joint rigidity in 2D led to a series of increasingly refined algorithms to detect rigidity, as well as maximal rigid subgraphs in flexible constraint graphs. Chronologically, there was a network flow based algorithm [16], a matroid sums algorithm [8], a bipartite matching algorithm [14], and finally what is known as the pebble game [18]. A version of the pebble game (a special case of network flow) is used for recursive decomposition into approximately rigid subgraphs in the DR-planning algorithms mentioned earlier. The idea of pebble games has since been extended to the class (k,l)-sparse graphs [22] where $l < 2k$, as discussed in Section 1.3.4.

It turns out that the latter condition in the above theorem defines independence in a sparsity matroid on $V \times V$. The analogous condition $d|V| - \binom{d+1}{2}$ formulated by Maxwell in the nineteenth century [24] is necessary but not sufficient for $d \geq 3$ as discussed later in this chapter, see the famous "double banana" graph, Figure 1.4. Typically, the so-called "Maxwell direction" is showing that independence in the combinatorial matroid is necessary for independence in the algebraic rigidity matroid, and is the easier one. The converse direction – that completes the equivalence of the two types of matroids – is the challenging one.

However, there are combinatorial characterizations of independence and local rigidity of bar-joint frameworks in 2D which extend to other types of 2D frameworks such as body-bar, body-hyperpin, 2D bar-joint on the sphere (or 3D line-angle), 2D point-line-incidence-direction, etc. Often there are more than one equivalent characterization. For example, Lovász and Yemini [23] found an alternate characterization of 2D bar-joint rigidity that is superficially quite different from Laman-Pollaczek-Beiringer's characterization. Chapter 18 and Chapter 21 discuss, respectively, such local and global rigidity characterizations. The latter chapter begins with the first combinatorial characterization of generic global rigidity, namely that of 2D bar-joint frameworks [17]. Chapter 22 gives a combinatorial matroid that captures the rigidity matroid of more challenging frameworks in 2D involving angles between lines with distances between points and lines.

1.2.4.1 Inductive Constructions

In 1911, Henneberg [15] gave the following constructive definition of the class of what later became Laman-Pollaczek-Beiringer graphs.

Definition 1.1 Henneberg Construction A *Henneberg construction* of a graph G is a sequence of the following operations which, beginning with a single edge, results in G.

(a) Add a new vertex and two edges connecting it to two existing vertices.

(b) Subdivide an existing edge and add an additional edge from the new vertex to another existing vertex.

Laman [21] used this class in the proof of Theorem 1.4. The basic structure of the proof was to show that the class described in the theorem was exactly the class of graphs with Henneberg constructions. Then he proved that for any Henneberg construction there is a positioning of the vertices that will have no infinitesimal motions. The earlier proof of Pollaczek-Beiringer was stronger, pointing out that almost all (or generic positionings) that avoid certain algebraic conditions will have no infinitesimal motions.

Inductive constructions are one of the mainstays of results that show equivalence between an algebraic rigidity matroid and a combinatorial or graphic matroid, specifically for the difficult direction, i.e., showing that independence in the combinatorial matroid implies independence in the algebraic rigidity matroid. Chapter 19 systematically surveys such inductive constructions in many proofs of combinatorial rigidity characterizations.

1.2.4.2 Body Frameworks

A *body* geometric primitive is a finite n-dimensional rigid object; this is rather general, including points, line segments, plane segments, etc. but also any other rigid free-form shape of the same dimension as the space. A *body-hinge* system has body primitives and *hinge* constraints, which are incidences between two primitives in d-dimensions where d is less than the dimension of either object about which the primitives can rotate. In 2D, this must be a bar-joint system. A *body-bar* system also has body primitives (as discussed in Section 1.2.4.2), but the constraints are distances between generic points on the body. Unlike the higher dimensional bar-joint systems, body-bar-hinge systems have a combinatorial characterization of independence and local rigidity in arbitrary dimensions, first proved in [38, 39] who pioneered the use of so-called pure conditions (where certain determinants vanish) to describe genericity. The characterizations extend to special classes of bar-joint systems that can be cast as body-bar-hinge systems. Combinatorial characterizations of global rigidity also exist.

Polyhedra (see Section 1.2.3) are a subclass of body-hinge structures. The bodies are panels (polygonal faces) and the hinges connect the panels; moreover, the system must completely enclose a volume. The so-called *molecular conjecture* [37] (referring to the ability to model protein backbones as body-hinge structures) stated that the rigidity of coplanar hinge and panel hinge frameworks obeyed the same combinatorial characterization as generic body-hinge structures. It was proven for general dimension in [19].

Chapter 20 surveys combinatorial characterizations of both local and global rigidity for body-hinge structures in arbitrary dimensions.

1.2.4.3 Body-Cad, and Point-Line Frameworks

The set of body-cad frameworks is a catch-all category for 3D constraint systems [13]. Motivated by CAD design software, it includes many of the common constraints and primitives seen in the industry. The above-mentioned categories are in fact specializations of this class. Primitives include points, lines, and planes, and constraints include **c**oincidence, **a**ngular (parallel, perpendicular, or arbitrary fixed angles), and **d**istance (cad). Chapter 23 discusses a combinatorial characterization of (infinitesimal) rigidity for such systems.

1.2.4.4 Symmetric and Periodic Frameworks and Frameworks under Polyhedral Norms

Chapter 25 develops the set up and techniques for extending combinatorial rigidity characterizations to symmetric and periodic frameworks (for different symmetry groups). Chapter 24 does the same for extending from Euclidean distance to polyhedral norms.

1.2.5 Missing Topics and Chapters

The sampling of topics in this handbook would have been more comprehensive with chapters on (a) Wu-Ritt's characteristic set method for automated geometry theorem proving mentioned earlier, (b) the topology (homology and cohomology) of linkage configuration spaces, related to Walker's problem, (c) combinatorial and algorithmic studies on expansive motions of linkages and origami related to the Carpenter's rule problem, (d) sphere-packing rigidity with results arising from analytic perspectives besides the tensegrity perspective, (e) conjectures and progress on characterizing generic bar-joint rigidity in 3D, and (f) the exploration of genericity, rigidity, and configuration spaces of periodic and infinite frameworks. Of these, there is extensive expository literature on (a),(b),(c). We hope that the next revision of the handbook may include chapters condensing the substantial amount of work on (d), (e), and (f). Finally the area is rich in subtopics and directions arising from numerous applications, from computer aided engineering and architectural design, molecular and materials modeling, machine learning and complexity. These however would be outside the scope of this handbook on geometric constraint principles, being more appropriate for a handbook on geometric constraints applications.

1.3 Terminology Reconciliation and Basic Concepts

In this section we clarify slightly different types of terminology used to talk about GCS and frameworks, and their relationship. We additionally introduce the overall program of combinatorial characterizations of GCS and frameworks: define notions of *genericity,* introduce the concept of a rigidity matrix and the notion of *infinitesimal rigidity*, which is a linearization of local rigidity that is generically equivalent and is used to define the so-called (generic) *rigidity matroid*.

1.3.1 Constrainedness

The notion of *constrainedness* exists to discuss the characteristics of the solution space of a GCS. An *over-constrained* system is one that has no solutions. A *well-constrained* system is one that has a finite number of solutions. An *under-constrained* system is one that has infinitely many solutions. Each variable in a system has an *approximate degree-of-freedom* (dof). Each constraint contributes at least one equation.

The definitions above apply to a system of equations with sufficiently general parameters. However, some specific assignments of values to parameters (e.g., length and angle measures) in the equations corresponding to the same underlying constraint graph may result in a different classification. That is, these properties are not structural (or combinatorial) A system is called *generic* over a ground field K (usually $\mathbb{Q}$ or $\mathbb{R}$ in our setting) if the designated constraint parameters are *algebraically independent* over K, i.e., they are not the solutions of any nontrivial polynomial equation with coefficients in K. Weaker definitions of genericity of GCS exist, for example, stating that the parameters are not the zeroes of a given set of polynomials, or some finite but unspecified set of polynomials. In all these cases, the set of nongeneric parameter values is measure-zero, i.e., choosing the parameters at random will result in a generic system with probability of one. Unless otherwise specified, the strongest definition of algebraic independence of parameter values is implied.

A generic constraint system being under-, well-, or over-constrained implies that all generic parameter assignments to the same underlying constraint graph result in an under-, well-, or over-constrained system, respectively. Therefore, being generically *-constrained is a combinatorial property. Thus, generic constrainedness terminology can be extended to the underlying constraint graph. Generic *-constrainedness does not imply *-constrainedness of a nongeneric system, and similarly the converse does not hold either. Therefore, classifying nongeneric constraint systems is an intractable problem.

However, in the *generically over-constrained* setting, a specific class of non-generic systems is of interest. Such a system with a generic assignment to the parameters will have no solution, but those non-generic systems that do have a solution are called *consistently over-constrained*. The sets of *generically well-over-constrained* and *generically under-over-constrained* systems are disjoint, complementary subsets of generically over-constrained. A generically well-over-constrained system is one with a spanning generically well-constrained subsystem. A generically under-over-constrained is any other over-constrained system.

Note that traditionally generically under-constrained was taken to mean the union of generically under-constrained and generically under-over-constrained as defined here. The current definition is cleaner since it ensures that the sets of generically under-, well-, and over-constrained are disjoint.

1.3.2 Rigidity of Frameworks

Two frameworks (G,p) and (G,q) are congruent if p and q are congruent modulo trivial motions. They are equivalent if the underlying GCS of (G,p) is identical to that of (G,q).

A framework (G,p) is *rigid* if there exists a nonempty neighborhood $\mathcal{N}(p)$ (in the Euclidean

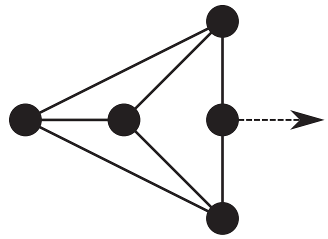


Figure 1.1
A 2D bar-joint framework that is rigid but not infinitesimally rigid. A nontrivial infinitesimal motion is indicated by the dashed arrow.

topology) such that for all $p' \in \mathcal{N}(p)$, congruence of frameworks (G,p) and (G,p') implies equivalence. If the framework is not rigid, it is *flexible*. A rigid framework is *minimally rigid* (or *isostatic*) if the removal of any hyperedge of G results in a flexible framework. The framework is *globally rigid* (or *strongly rigid*) if, for all p' for which the framework (G,p') is equivalent to (G,p), it also holds that they are congruent.

Given a constraint graph G, there is a system of polynomial equations $F = \{f_1,\ldots,f_m\}$ and variables $X = \{x_1,\ldots,x_n\}$ (as explained in Section 1.1.) The *rigidity matrix* $R(G)$ is the Jacobian of this system with respect to X, i.e., the $m \times n$ matrix with element (i,j) equal to $\partial f_i/\partial x_j$. The rigidity matrix of a framework (G,p) (written as $R(G,p)$) is $R(G)$ with all variables x_i replaced by $p(x_i)$. When working with most constraint systems, the equations are quadratic and this process is often referred to as *linearization* of the polynomial system.

As explained earlier, the solution space of a geometric constraint system is a variety in K^n (where we may let $K = \mathbb{C}$ to make full use of the algebraic theory); it is the set of zeros of the corresponding polynomial system. The row span of the rigidity matrix $R(G,p)$ is the space of normals to this variety at point $(p(x_1), p(x_2), \ldots, p(x_n))$. Any infinitesimal movement on the tangent space, orthogonal to the space of normals (i.e., along the variety), will give another solution. The tangent space is the right nullspace (or kernel) of $R(G,p)$. Geometrically, any infinitesimal vector in the right nullspace represents an infinitesimal change to each primitive such that the resulting framework still satisfies all of the constraints.

The left nullspace of the rigidity matrix also has a geometric interpretation. The left nullspace only has nonzero vectors if there is a linear dependency in the rigidity matrix, which corresponds to an *equilibrium self-stress* of the system.

The *degree-of-freedom* (*dof*) of the framework is the nullity of $R(G,p)$ (i.e., the rank of the right nullspace.) The framework is *infinitesimally rigid* if the dof is equal to the number of trivial motions. Geometrically, this means that all infinitesimal motions arise from trivial motions of the space. By the rank-nullity theorem, a framework is also infinitesimally rigid exactly when the rank of $R(G,p)$ is $n-k$, where n is the number of variables and k is the number of trivial motions. In the case of d-dimensional Euclidean space, the space of trivial motions has dimension $\binom{d+1}{2}$, which are the d translations plus $\binom{d}{2}$ rotations. If the right nullspace of $R(G,p)$ has dimension greater than $\binom{d+1}{2}$, then the framework (G,p) is said to have *infinitesimal motions* (or *infinitesimal flexes*).

It is clear that infinitesimal rigidity implies rigidity, and we give an example of a bar-joint framework that is rigid but not infinitesimally rigid to show that the converse is false. A comprehensive introductory treatment of the rigidity of graphs (bar-joint systems) can be found in Ref. [12].

To prove that rigidity does not imply infinitesimal rigidity, consider Figure 1.1. This is a rigid bar-joint graph that has an infinitesimal motion that is zero at all places except at the single vertex in the direction of the arrow. A rigid framework with a nontrivial infinitesimal motion is called *degenerate* and must be in a nongeneric realization. A combinatorial characterization of rigidity is

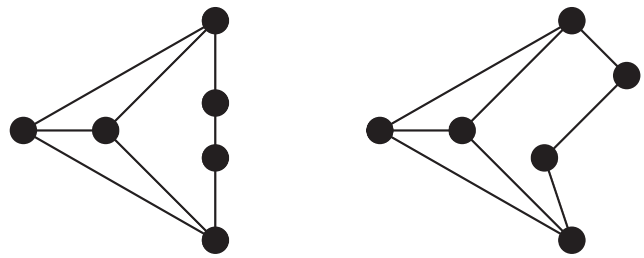


Figure 1.2
Two different 2D frameworks of the same generically flexible bar-joint constraint graph. The degenerate framework on the left is rigid, whereas the generic configuration on the right is flexible.

only guaranteed to hold for a certain generic class of realizations, and we formalize what we mean by genericity in the next section.

The rigidity matrix has a natural notion of dependence, based on the linear dependence of the rows (or columns) of the matrix. As such, the *rigidity matroid* of a framework [12] is simply the linear matroid of the rows of its rigidity matrix. That is, the row vectors of the matrix comprise the ground set, and the linearly independent subsets of rows comprise the family of independent sets. Therefore, the matroid rank (the maximum cardinality of an element in the matroid) is exactly the rank of the matrix. Since each row corresponds to some constraint, a dependent row corresponds to a dependent constraint. The framework as a whole is *independent* if there are no dependent constraints and is *dependent* otherwise.

1.3.3 Generic Rigidity of Frameworks

Determining rigidity of frameworks is difficult; deciding global [29] and local rigidity [1] are both strongly NP-hard for bar-joint systems. However, if a certain measure-zero set of primitive arrangements is excluded, determining rigidity can become much easier for certain constraints and primitives. This is the set of degenerate frameworks, which will be discussed in this section.

In Section 1.3.1, the notion of genericity was discussed in the context of geometric constraint systems. When considering frameworks, genericity has a different meaning. A framework (G, p) is said to be *generic with respect to property* $\mathcal{P}$ (G, p) satisfies $\mathcal{N}P$ if and only if there exists some neighborhood $\mathcal{N}(p)$ around p such that for all points $p' \in \mathcal{N}(p)$ the framework (G, p') satisfies $\mathcal{P}$.

A property $\mathcal{P}$ is said to be *generic* when, for all constraint graphs G, either all generic frameworks of G (w.r.t. $\mathcal{P}$) satisfy $\mathcal{P}$ or none satisfy $\mathcal{P}$. If a property of frameworks (G, p) is generic, then it is a combinatorial property of the underlying constraint graph alone. Intuitively, a generic property of a framework is one that is maintained if primitives were "wiggled" by small amounts in any direction. For example, the independence of the rigidity matrix is a generic property of frameworks.

To illustrate the importance of considering generic frameworks, consider the following examples. See Figure 1.2 which depicts two frameworks of the same constraint graph. The nongeneric framework on the left is rigid while being generically flexible. The three bar "chain" is taut, disallowing finite flexes; with only slightly different lengths, this would no longer be rigid. See also Figure 1.3 which also depicts two frameworks corresponding to a single constraint graph. The nongeneric framework on the left is flexible while being generically rigid. The three vertical bars are the same length, permitting a vertical shear; with different lengths, there would be no infinitesimal motions.

Whereas rigidity does not imply infinitesimal rigidity of a framework, in the generic case it does. The Implicit Function Theorem from multivariate calculus, as in Ref. [3], shows that when a framework is generic w.r.t. infinitesimal rigidity, every infinitesimal flex can be converted into a

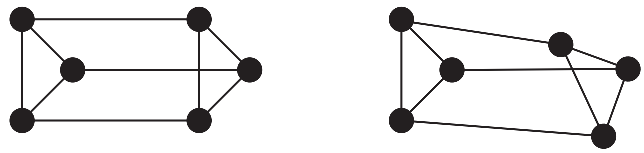


Figure 1.3
Two different 2D frameworks of the same generically rigid bar-joint constraint graph (known as $C_2 \times C_3$). The degenerate framework on the left is flexible, whereas the generic configuration on the right is rigid.

Table 1.1
Correspondence between constrainedness terminology when used in the context of generic systems and generic frameworks.

Generic Systems	**Realizations**	**Generic Framework**
Under-constrained	Infinite solutions	Independent and flexible
Well-constrained	Finite solutions	Independent and rigid
Over-constrained	No solutions	Dependent
Under-over-constrained	No solutions	Dependent and flexible
Well-over-constrained	No solutions	Dependent and rigid

finite flex, i.e., rigidity in fact implies infinitesimal rigidity. Since infinitesimal rigidity is a generic property, this shows that rigidity is also a generic property. As mentioned, this property can be thought of as a combinatorial property of the underlying constraint graph. Therefore, a constraint graph is called rigid if some generic framework of the graph is infinitesimally rigid. The notions of flexibility and minimal rigidity have obvious meanings in the generic sense as well.

The *generic rigidity matroid* of a geometric constraint system with constraint graph G can be defined in two ways: (1) the rigidity matroid of any generic framework (G, p), or (2) the rigidity matroid formed by the rows of the rigidity matrix $R(G)$ of indeterminates. This leads to combinatorial notions of independence among constraints.

Table 1.1 establishes a rough correspondence between the genericity of a GCS, a framework, and rigidity. Traditionally, generically under-constrained was used to describe all flexible frameworks and was therefore a union of under- and under-over-constrained as defined here. The symmetry of the new definitions displayed above is another argument for the new terminology.

1.3.4 Approximate Degree-of-Freedom and Sparsity

There is a real, algebraic notion of degree-of-freedom that is discussed in Section 1.3.2. This section introduces a different, combinatorial idea of degree-of-freedom that can be used to approximate rigidity. This was briefly mentioned in Section 1.3.1, and has an obvious relationship with constrainedness. The following methods are very general, working for any type of primitive or constraint. However, a positive result for rigidity by this characterization is often only a necessary condition for true rigidity. Many systems require additional considerations, if there is a combinatorial characterization at all.

In the constraint graph, each primitive has some *degrees-of-freedom (dofs)* and each constraint eliminates some dofs between participating primitives. In some of the literature, particularly that which uses network flow based algorithms, dof corresponds to the negation of the *density* of the constraint graph. Given a constraint graph $G = (P, C)$, with primitives P and constraints C, and a

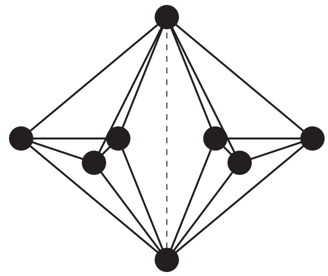


Figure 1.4
A 3D bar-joint framework, known as the double-banana. It is flexible due to a rotation about the dashed line, despite being dof-rigid (i.e., $(3,6)$-tight).

weight function w on P and C, the density of G is

$$d(G) = \sum_{c \in C} w(c) - \sum_{p \in P} w(p).$$

Given some constant k equal to the number of trivial motions of the underlying geometry, a constraint graph is *minimally dof-rigid* if it has k dofs and every subgraph has at least k dofs. In terms of density, the graph is minimally dof-rigid if $d(G) = -k$ and for all subgraphs G', $d(G') \leq -k$. A graph is *dof-rigid* if it contains some minimally dof-rigid subgraph.

For example, consider 2D bar-joint systems: points can be thought of as having 2 dofs (translation but not rotation) and an edge between two points destroys 1 dof, leaving a system with 3 dof (translation and rotation.) In Euclidean 2D space, there are 3 trivial motions and therefore a single edge would be dof-rigid. In fact, as mentioned earlier, this notion of dof-rigid exactly captures generic rigidity of 2D bar-joint systems. We follow the convention of referring to methods using dof analysis as *Laman counts*.

We can understand the rigidity of some systems combinatorially using only dof analysis. However, this combinatorial notion of dof-rigid does not usually imply generic rigidity. Consider 3D bar-joint systems: points instead have 3 dofs, edges still eliminate 1 dof, and the space has 6 trivial motions. By the counts, the famous “double banana” graph in Figure 1.4 is dof-rigid; however, the “bananas” can clearly swivel about the dashed hinge.

As mentioned earlier, the Laman-Pollaczek-Beiringer theorem gives a purely combinatorial property (no algebra, simply counting) to capture the properies of the rigidity matrix and therefore the matroid. These subsystems are exactly the independent sets of the 2D bar-joint rigidity matroid.

This theorem motivated the notion of *sparsity* and *sparsity counts* [22]. This terminology is used to discuss constraint graphs where all primitives have k dofs and all constraints eliminate one dof and are binary; however, the theory does allow loops, effectively permitting unary constraints, and allows for multiedges, so constraints that eliminate n dofs can be represented in as many edges. A graph $G = (V,E)$ is (k,l)-*sparse*, for every induced subgraph $G' = (V',E')$, the inequality $|E'| \leq k|V'| - l$. The graph is (k,l)-*tight* if it additionally satisfies $|E| = k|V| - l$.

For example, Laman graphs would be the set of $(2,3)$-tight graphs. A 2D system of 2D rigid bodies and distance constraints would use $k = 3$ and $l = 3$; in fact, $(3,3)$-tight graphs are exactly the class of 2D rigid body-bar graphs. For a fixed k and l there is often a natural interpretation of a (k,l)-tight graph as a constraint system in which geometric primitives have k dof.

For all $l < 2k$, these counts define a *sparsity matroid* where the basis is the set of edges and the independent sets are the edges in the (k,l)-sparse subgraphs. This allows for an efficient class of algorithms, called *pebble games*, which can detect (k,l)-sparse graphs in polynomial time, if $l < 2k$.

1.4 Alternative Pathway through the Book

Given the interconnected nature of the chapters of this book there are many logical ways of ordering the material. Here, we describe an alternative navigation that might be more accessible to a newcomer.

In this suggested pathway, Group I contains chapters highlighting the different perspectives from developing solvers for general geometric constraint systems to analyzing the rigidity of bar-and-joint structures to constructing purely combinatorial objects arising in rigidity theory. Groups II and III focus on the problems studied in Rigidity Theory, introducing concepts such as local and global rigidity and considering models initially restricted to points before generalizing to other types of geometric primitives (e.g., rigid bodies or lines). Groups IV and V shift the structure to partition by the underlying approaches (metric geometry and algebraic methods).

Group I **Getting Started: Chapters 1, 6, 7, 18, and 19**
To get the lay of the land, start with two sets of chapters highlighting the perspectives of historically distinct communities. Chapters 6 and 7 focus on decomposition-recombination approaches used in computing realizations of a general GCS. In contrast, Chapter 18 restricts its content to the classical structure studied in Rigidity Theory of 2-dimensional bar-and-joint frameworks (introduced in Section 1.1.1); the combinatorial property characterizing generic bar-and-joint rigidity is studied in a generalized setting in Chapter 19.

Group II **Rigidity Theory for Point Primitives: Chapters 21, 16, 8, 14, 15, and 25**
Building upon the fundamentals introduced in Group I, continue the Rigidity Theory perspective, with topics posed in the setting of GCS with points as the geometric primitives.

Group III **Rigidity Theory for Other Primitives: Chapters 20, 13, 22, and 23**
Next, move to the Rigidity Theory for systems defined on geometric primitives beyond simply points (e.g., rigid bodies, lines).

Group IV **Metric Geometry: Chapters 24, 17, 9, and 10**
Shifting to an organization based on underlying techniques, start with approaches relying on distance and metric geometry.

Group V **Algebraic Methods: Chapters 12, 11, 3, 2, 5, and 4**
Finally, conclude with chapters based on algebraic methods.

The intention of this pathway is to help to get readers started with the varying perspectives on GCS formulation (Group I). Then, start with topics in Rigidity Theory (Groups II and III) before shifting to an order partitioned more by the underlying machinery (Groups IV and V).

Acknowledgment

We thank Jessica Sidman and Rahul Prabhu for a careful reading, and Audrey St. John for providing the alternative pathway.

References

[1] Timothy Good Abbott. *Generalizations of Kempe's universality theorem*. PhD thesis, Massachusetts Institute of Technology, 2008.

[2] Dennis S Arnon, George E Collins, and Scott McCallum. Cylindrical algebraic decomposition I: The basic algorithm. *SIAM Journal on Computing*, 13(4):865–877, 1984.

[3] L. Asimow and B. Roth. The rigidity of graphs. *Transactions of the American Mathematical Society*, 245:279–289, November 1978.

[4] B. Buchberger. *A note on the complexity of constructing Gröbner-bases*, pages 137–145. Springer Berlin Heidelberg, Berlin, Heidelberg, 1983.

[5] Augustin-Louis Cauchy. Sur les polygones et les polyèdres: second mémoire. *Journal de l'École Polytechnique*, 9:87, 1813.

[6] Robert Connelly. The rigidity of polyhedral surfaces. *Mathematics Magazine*, 52(5):275–283, November 1979.

[7] Ioannis Fudos and Christoph M. Hoffmann. Correctness proof of a geometric constraint solver. *International Journal of Computational Geometry & Applications*, 06(04):405–420, 1996.

[8] Harold N Gabow and Herbert H Westermann. Forests, frames, and games: algorithms for matroid sums and applications. *Algorithmica*, 7(1-6):465–497, 1992.

[9] Giovanni Gallo and Bud Mishra. Wu-Ritt characteristic sets and their complexity. *Discrete and Computational Geometry: Papers from the DIMACS Special Year*, 6:111–136, 1991.

[10] Herman Gluck. Almost all simply connected closed surfaces are rigid. In *Geometric topology*, volume 438 of *Lecture Notes in Mathematics*, pages 225–239. Springer, 1975.

[11] Steven J Gortler, Alexander D Healy, and Dylan P Thurston. Characterizing generic global rigidity. *American Journal of Mathematics*, 132(4):897–939, 2010.

[12] Jack E. Graver, Brigitte Servatius, and Herman Servatius. *Combinatorial Rigidity*, volume 2. Graduate Studies in Math., AMS, 1993.

[13] Kirk Haller, Audrey Lee-St.John, Meera Sitharam, Ileana Streinu, and Neil White. Body-and-cad geometric constraint systems. *Comput. Geom. Theory Appl.*, 45(8):385–405, October 2012.

[14] Bruce Hendrickson. Conditions for unique graph realizations. *SIAM J. Comput.*, 21(1):65–84, February 1992.

[15] Lebrecht Henneberg. *Die graphische Statik der starren Systeme*, volume 31. BG Teubner, Leipzig, 1911.

[16] Hiroshi Imai. On combinatorial structures of line drawings of polyhedra. *Discrete Applied Mathematics*, 10(1):79 – 92, 1985.

[17] B. Jackson and T. Jordán. Connected rigidity matroids and unique realizations of graphs. *J. Combinatorial Theory Ser B*, 94:1–29, 2005.

[18] Donald J. Jacobs and Bruce Hendrickson. An algorithm for two-dimensional rigidity percolation: The pebble game. *Journal of Computational Physics*, 137(2):346 – 365, 1997.

[19] Naoki Katoh and Shin-ichi Tanigawa. A proof of the molecular conjecture. *Discrete & Computational Geometry*, 45(4):647–700, 2011.

[20] Alfred B Kempe. On a general method of describing plane curves of the nth degree by linkwork. *Proceedings of the London Mathematical Society*, 1(1):213–216, 1875.

[21] G. Laman. On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics*, 4(4):331–340, October 1970.

[22] Audrey Lee and Ileana Streinu. Pebble game algorithms and sparse graphs. *Discrete Mathematics*, 308(8):1425–1437, 2008. Third European Conference on Combinatorics, Graph Theory, and Applications.

[23] L. Lovász and Y. Yemini. On generic rigidity in the plane. *SIAM Journal on Algebraic Discrete Methods*, 3(1):91–98, 1982.

[24] James Clerk Maxwell. On reciprocal figures and diagrams of forces. *Philosophical Magazine*, 27:250–261, 1864.

[25] Ernst Mayr and R. Cori. *Membership in polynomial ideals over Q is exponential space complete*, pages 400–406. Springer Berlin Heidelberg, Berlin, Heidelberg, 1989.

[26] J Owen and S Power. The non-solvability by radicals of generic 3-connected planar laman graphs. *Transactions of the American Mathematical Society*, 359(5):2269–2303, 2007.

[27] H. Pollaczek-Geiringer. über die gliederung ebener fachwerk. *Zeitschrift fur Angewandte Mathematik und Mechanik (ZAMM)*, 7:58–72, 1927.

[28] B. Roth and W. Whiteley. Tensegrity frameworks. *Transactions of the American Mathematical Society*, 265(2):419–446, June 1981.

[29] James B Saxe. *Embeddability of weighted graphs in k-space is strongly NP-hard*. Carnegie-Mellon University, Department of Computer Science, 1980.

[30] I. J. Schoenberg. On certain metric spaces arising from euclidean spaces by a change of metric and their imbedding in hilbert space. *Annals of Mathematics*, 38(4):787–793, December 1936.

[31] I. J. Schoenberg. Metric spaces and positive definite functions. *Transactions of the American Mathematical Society*, 44(3):522–526, November 1938.

[32] Neil JA Sloane. Error-correcting codes and invariant theory: new applications of a nineteenth-century technique. *American Mathematical Monthly*, pages 82–107, 1977.

[33] Larry Smith. Polynomial invariants of finite groups. a survey of recent developments. *Bulletin of the American Mathematical Society*, 34(3):211–250, 1997.

[34] D. M. Y. Sommerville. *An Introduction to the Geometry of n Dimensions*. 1958.

[35] Ágnes Szántó. Complexity of the Wu-Ritt decomposition. In *Proceedings of the second international symposium on parallel symbolic computation*, pages 139–149. ACM, 1997.

[36] Alfred Tarski. A decision method for elementary algebra and geometry. In *Quantifier elimination and cylindrical algebraic decomposition*, pages 24–84. Springer, 1998.

[37] T.-S. Tay and W. Whiteley. Recent advances in the generic rigidity of structures. *Structural Topology*, 9:31–38, 1984.

[38] Tiong-Seng Tay. Rigidity of multi-graphs. i. linking rigid bodies in n-space. *Journal of Combinatorial Theory, Series B*, 36(1):95 – 112, 1984.

[39] Neil White and Walter Whiteley. The algebraic geometry of motions of bar-and-body frameworks. *SIAM J. Algebraic Discrete Methods*, 8(1):1–32, 1987.

[40] Neil L. White. *A Tutorial on Grassmann-Cayley Algebra*, pages 93–106. Springer Netherlands, Dordrecht, 1995.

Part I

Geometric Reasoning, Factorization and Decomposition

Chapter 2

Computer-Assisted Theorem Proving in Synthetic Geometry

Julien Narboux

University of Strasbourg

Predrag Janičić

University of Belgrade

Jacques Fleuriot

University of Edinburgh

CONTENTS

2.1 Introduction

Computer-assisted proof in mathematics has been underway since the pioneering times of computers in the 1950s. Starting from early systems with very limited capability, computer-assisted theorem proving has evolved to demonstrate theorems never proved before by humans [156] and assist with monumental efforts spanning several man-years [116]. In this endeavour, geometry plays an important part, just as it has throughout the history of mathematics. This is due to its pervasive role: it is a paradigmatic form of reasoning, with applications to education, mathematical and physical research, but also to many applied areas such as robotics, computer vision, and CAD [48]. Moreover, many of the search techniques and other algorithmic features developed for geometric reasoning have influenced other areas of artificial intelligence.

As for other subfields of computer-assisted proof, the mechanization of geometry spans both *automated* and *interactive* theorem proving. In the former, computers aim to prove theorems completely automatically, while in the latter, the role of the system is to act as a *proof assistant* that verifies the reasoning steps of the user, guides the proving process, and provides some limited automation. These two branches are often connected through methods that can produce geometric proofs automatically, where either the proofs or the methods themselves are fully verified.

Just as for its pen-and-paper counterpart, computer-assisted proof in geometry is subject to different foundations: algebraic or synthetic (axiomatic) ones. This chapter aims to provide a comprehensive survey of the latter and of its applications. In particular, it will mostly deal with *planar Euclidean geometry* [6], which in this case generally means a theory consisting of geometric statements true in $\mathbb{R}^2$. There are several formal systems that aim to axiomatize this theory or its subtheories, including those due to Euclid, Hilbert, and Tarski.

In what follows, to ensure a coherent exposition, we will use a uniform notation (which may differ at times from that used by the original authors). In particular, we will denote points by uppercase letters, lines by lowercase ones, the strict notion of *betweenness* by $A \dot{-} B \dot{-} C$ (i.e., B belongs to segment AC and is different from A and C) while its nonstrict version will be denoted by $A\text{–}B\text{–}C$, collinearity by $\mathrm{Col}\,ABC$, perpendicularity by $\perp$, cyclicity of points, i.e., points lying on the same circle, by $cyclic(A,B,C,D)$, the angle between half-lines AB and AC by $\angle ABC$, the full-angle between *lines* AB and CD by $\angle[AB,CD]$, a triangle with vertices A, B, C by $\triangle ABC$, congruence between segments, triangles, or between angles by $\cong$, and equality over measures of angles or over full-angles by $=$.

2.2 Automated Theorem Proving

Automated theorem proving in geometry is often considered a “classical AI domain.” Its methods can be split into three major families or styles: algebraic, synthetic, and semisynthetic.

The algebraic methods deal with the algebraized formulation of geometric statements and usually involve dealing with the membership of polynomial ideals [45, 217, 224], quantifier eliminations [64, 212], or use coordinate-free approaches based on bracket and Cayley algebras, described in Chapter 3. Although they are powerful, they cannot produce human-readable proofs and, generally, consist of steps that do not have any obvious meaning in synthetic geometry. The second and the third groups of methods, the subjects of this review, focus on proving theorems via geometric axioms (or higher-level geometric lemmas) and often try to automate the traditional theorem proving approaches, while attempting to generate human-readable proofs.

Automated theorem proving is used in various contexts, e.g., for mathematical education [19], in the simplification of geometric axiom systems [69] and in the study of incidence geometry using term rewriting techniques [5].

Unless otherwise stated, the methods presented next only deal with planar geometries assuming the parallel postulate.

Note also that the examples will be presented in a uniform way, although we have taken care to preserve the essence of the methods being applied. Finally, we remark that there are other surveys [48, 90, 97, 131, 218] that cover some of the approaches that we will discuss next.

2.2.1 Foundations

Aside from the algorithmic techniques used, automated theorem proving methods rely on various choices with regard to the underlying logic and geometric knowledge. One issue relates to a set of axioms to be used. Some methods use well-known geometric axioms sets, but most use custom (finite sets of first-order) axioms. In the latter case, the axioms are actually simple theorems of Euclidean geometry whose proofs are not considered and are asserted as facts belonging to common geometric knowledge (hence they are often called "lemmas" or "rules"). The set of axioms is not necessarily minimal as they are often selected to ease the automatic proof of more complex theorems. For many methods, choosing an appropriate set and level of axioms is one of most critical issues when it comes to power and efficiency.

2.2.2 Nondegenerate Conditions

The notion of nondegenerate (NDG) conditions arises for each style of automated geometric reasoning. Namely, it is often the case that the goal (denoted by G) is implied by the configuration (denoted by C) plus some *additional* conditions [46, 48, 55]. For such conditions (denoted by ndg), the following formulae (where $\forall*$ and $\exists*$ denote universal and existential closure) have to be valid: $\exists*(C \wedge ndg)$ and $\forall*(C \wedge ndg \Rightarrow G)$. In many cases, the methods can automatically produce such NDG conditions, although not necessarily the "weakest" ones. There are, however, algorithms for computing the weakest NDG conditions [44].

2.2.3 Purely Synthetic Methods

Purely synthetic methods, or simply synthetic methods do not use coordinates and algebraic forms for the geometric statements. Many of these techniques add auxiliary elements to the geometric configuration under consideration, so that certain axioms can be applied. This usually leads to a combinatorial explosion in the search space. The challenge then rests in controlling this explosion and in developing suitable heuristics in order to avoid unnecessary construction steps. Due to the nature of these problems, such synthetic proof techniques are sometimes called Artificial Intelligence (AI) methods for automated theorem proving in geometry.

The very first AI method was developed by Gelernter et al. Quoting the authors: "In early spring, 1959, an IBM 704 computer, with the assistance of a program comprising some 20000 individual instructions, proved its first theorem in elementary Euclidean plane geometry" [101]. Their program, called the Geometry Machine and written in FORTRAN [99, 100], was not only the first automated theorem prover for geometry, but also one of the very first automated reasoning systems for mathematics. Although its power, from a modern point of view, was very limited, this system is important both for historical reasons and for introducing a number of ideas and techniques that were used by many subsequent reasoning systems. At the time, geometry was viewed as a typical, paradigmatic AI domain but also as a potentially easy domain where simple *ad-hoc* rules, basic forward and backward chaining* applied exhaustively, and simple heuristics could be used to bear easy fruit e.g. such

*Forward and backward chaining are two important forms of inferences within reasoning systems. The former can be viewed as a sequence of applications of modus ponens that derives new facts from existing premises in order to prove the goal, while as cracking the whole area of high school geometry. In the description of the approaches, logical representation, algorithms, implementation details, and even the description of features of the programming languages were often intermixed. Over time, more sophisticated and mature approaches have emerged, with many still using some of the early techniques and ideas though.

2.2.3.1 Early Systems

Gelernter's Geometry Machine implemented several reasoning techniques. These included the use of "diagrams" (concrete models in the Cartesian plane) in an attempt to reject false subgoals before any attempt to prove them, dealing with symmetries, and the use of simplification rules (akin to modern rewrite rules). The system worked by backward reasoning – starting with the given goal and trying to decompose it to simpler provable subgoals. Its basic rules were based on axioms about the congruence of triangles. Thus, proving that two segments were congruent could be done by showing that these two segments were the corresponding edges of two congruent triangles.

Example. Proving the following theorem, in less than 20s, was one of the big triumphs of Geometry Machine: if $\angle ABD \cong \angle DBC$, $AD \perp AB$, $DC \perp BC$, then $AD \cong CD$ (Figure 2.1):

$\angle ABD \cong \angle DBC$	(Premise)
$\angle DAB$ is right angle	(by Definition of perpendicular)
$\angle DCB$ is right angle	(by Definition of perpendicular)
$\angle BAD \cong \angle BCD$	(by All right angles are congruent)
$BD \cong BD$	(by Reflexivity of congruence)
$\triangle ADB \cong \triangle CDB$	(by Congruence of triangles, rule Side-Angle-Angle)
$AD \cong CD$	(by Corresponding elements of congruent triangles are congruent)

There were several subsequent systems improving on Gelernter's ideas, e.g., by combining backward and forward chaining, by trying to model the human solving process more faithfully, or by being designed to serve as support for tutoring systems [2, 109, 56, 57, 89, 103, 105, 136, 166]. However, despite all these efforts, these early systems had a very limited scope and were only able to prove geometric problems of small or moderate complexity. They didn't treat NDG conditions and were not able (or were able only to a limited extent) to add new, "auxiliary" points, necessary in many proofs. So, they typically dealt only with axioms and conjectures of the following form (universal closure is assumed): $A_1(\vec{x}) \wedge \ldots \wedge A_n(\vec{x}) \Rightarrow B(\vec{x})$, where $\vec{x}$ denotes a sequence of variables, A_i and B are atomic formulae or their negations.

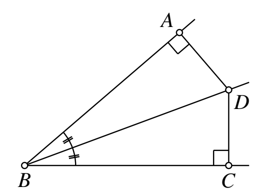


Figure 2.1
Geometry machine diagram.

2.2.3.2 Deductive Database Method, GRAMY, and iGeoTutor

There are several theorem proving methods, including the deductive database (DD) [54] and those used by the systems GRAMY [155] and iGeoTutor [220], that deal with "rules" and conjectures of the form (universal closure is assumed): $A_1(\vec{x}) \wedge \ldots \wedge A_n(\vec{x}) \Rightarrow B_1(\vec{x})$ and also rules of the form: $A_1(\vec{x}) \wedge \ldots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}(B_1(\vec{x},\vec{y}) \wedge \ldots \wedge B_m(\vec{x},\vec{y}))$ where $\vec{x}$ and $\vec{y}$ denote sequences of variables, B_i are atomic formulae and A_j are atomic formulae or their negations. There are no disjunctions either in the rules or in the conjectures. Hence, these methods cannot prove conjectures involving existential quantifiers, but can use new, auxiliary points (or segments), while searching for a proof.[†]

the latter can be viewed as applying modus ponens backward to refine the goal into subgoals that can hopefully be proven from the premises.

[†]Using auxiliary points makes a substantial change compared to the early methods (Section 2.2.3.1), a change that enabled the proof of a wider set of complex theorems.

One of the main challenges, though, lies in controlling the introduction of additional objects since these can lead to a combinatorial explosion.

Unlike for algebraic methods, a common motivation here is the generation of human-readable synthetic proofs that are as close as possible to those taught in schools. Moreover, in the case of GRAMY, the generation of several proofs is attempted, making it suitable for some forms of tutoring.

Scope. The methods deal with formulae containing no function symbols and with fixed sets of predicates symbols. For instance, the DD method uses predicate symbols corresponding to geometric relations (over points), such as Col, $\perp$, $cyclic()$, and equality over full-angles (see Section 2.2.4.2). GRAMY deals not only with points, but also with segments, angles and triangles, and with a set of predicate symbols that includes $\cong$, $\perp$, $\|$, membership, etc. Each system uses a fixed set of axioms, for instance, the DD method uses around 75 axioms of the first form, including:

D41: $cyclic(A,B,P,Q) \Rightarrow \angle[PA,PB] = \angle[QA,QB]$
D42: $\angle[PA,PB] = \angle[QA,QB] \wedge \neg \mathrm{Col}\, PQAB \Rightarrow cyclic(A,B,P,Q)$
D74: $\angle[AB,CD] = \angle[PQ,UV] \wedge PQ \perp UV \Rightarrow AB \perp CD$

and around 20 rules of the second form, including, for example:

X1: $OM \perp MA \wedge \angle[XO,MO] = \angle[MO,AO] \Rightarrow \exists B\, (\mathrm{Col}\, BAM \wedge \mathrm{Col}\, BOX)$

The DD method was reported as being able to prove, in a matter of seconds and via hundreds of derived facts, 160 out of the 600 theorems in the authors' collection of results proved by Wu's method. GRAMY and iGeoTutor were applied on smaller benchmark sets gathered from different sources.

Theorem proving mechanisms. For a given conjecture, atomic formulae from the hypotheses are considered as "facts." All three methods use forward chaining for deriving new facts using the available axioms (the DD method takes some ideas from deductive database theory [96]). There is a number of techniques (e.g., based on symmetries) used for keeping the number of stored derived facts low and the proving process efficient.

Auxiliary points are introduced only in a controlled manner, determined by various strategies, such as introducing new points only if new facts cannot be derived using forward chaining, or introducing points only through a very limited number of *templates*, specific geometry configurations.

GRAMY and iGeoTutor do not handle NDG conditions, while the DD method treats them using a form of negation as failure augmented with some basic diagrammatic model checking.

iGeoTutor can deal with conjectures involving some arithmetical constraints and for that purpose uses external provers that are specialized for theories like linear arithmetic and are available within the SMT (Satisfiability Modulo Theory) solver Z3 [160].

Example. The system iGeoTutor can prove the following theorem: Given a square $ABCD$, a point in its interior such that $AP \cong PD$ and $\angle PAD = 15°$, prove that the triangle $\triangle PBC$ is equilateral. The system, using the fact $AB \cong AD$, decides to use the *congruent triangles template*, and adds a points Q such that $\angle BAQ \cong \angle PAD$ and $AQ \cong AP$ (such that $\triangle AQB \cong \triangle APD$ holds). Later, the system decides to use another template and introduces the segment QP. By the initial forward chaining, the system deduces $BP \cong CP$, $\angle APD = 150°$ (among others facts). The rest of the proof is as follows:

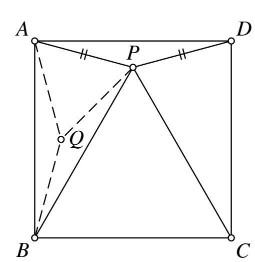


Figure 2.2
iGeoTutor diagram.

$\triangle AQB \cong \triangle APD$	(by Congruence of triangles, rule Side-Angle-Side)
$\angle BAQ = 15°$, $\angle BQA = 150°$	(by Corresponding parts of congruent triangles are congruent)
$\angle QAP = 60°$	($\angle QAP = 90° - \angle BAQ - \angle PAD$)
$\angle AQP = 60°$, $\angle APQ = 60°$	(by $AQ \cong AP$, Isosceles triangle)
$AQ \cong PQ$	(by $\triangle AQP$ is equilateral)
$\angle BQP = 150°$	($\angle BQP = 360° - \angle BQA - \angle AQP$)
$\triangle AQB \cong \triangle PQB$	(by $BQ \cong BQ$, $\angle BQA \cong \angle BQP$ $AQ \cong PQ$, Side-Angle-Side)
$AB \cong BP$	(by Corresponding parts of congruent triangles are congruent)
$BC \cong BP$, $BC \cong CP$	(by $AB \cong BC$, $AB \cong BP$, $BP \cong CP$, Transitivity of congruence)

Properties. These methods are complete, with respect to the set of rules being used, for proofs that do not require the introduction of auxiliary points since the search space is then finite. This feature enables not only proof, but also the deduction of additional facts from a given configuration and hence the discovery of new theorems. If a proof requires auxiliary elements, completeness can be proved only under specific conditions.

2.2.3.3 Logic-Based Approaches

Geometric theorems can be proved not only by dedicated systems but also by general ones, typically by theorem provers for first order logic or some of its fragments such as *Coherent logic* (CL). Such automated provers usually cover rich sets of formulae that include existential quantification. We review some of these approaches next.

Coherent logic consists of formulae of the following form (universal closure is assumed) [15, 85]: $A_1(\vec{x}) \wedge \ldots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}(B_1(\vec{x},\vec{y}) \vee \ldots \vee B_m(\vec{x},\vec{y}))$, $\vec{x}$ and $\vec{y}$ denote sequences of variables, A_i denotes an atomic formula, and B_j denotes a conjunction of atomic formulae. There are no function symbols with arity greater than zero and there is no negation. *Resolution logic* (RL) deals with *clauses*, i.e., formulae of the following form (universal closure is assumed): $A_1(\vec{x}) \vee \ldots \vee A_n(\vec{x})$, where $\vec{x}$ denotes sequences of variables, and A_i denotes an atomic formula or a negation of an atomic formula.

CL conveys a wider range of formulae compared to the DD method, for instance (Section 2.2.3.2) – there can be existential quantification over variables, not only in axioms but also in conjectures and, also, there can be disjunctions. CL can also be considered as an extension of RL, but in contrast to the resolution-based proving, the CL conjecture can be proved directly and unchanged (refutation, Skolemization and transformation to clausal form are not used). The domain of procedures for CL actually covers first-order logic because every first-order theory can be translated into coherent logic, possibly with additional predicate symbols [185, 85]. Checking validity of an arbitrary first-order formula can be replaced by checking unsatisfiability of a corresponding set of clauses (after refutation, Skolemization and transformation to clausal form).

Provability in CL and unsatisfiability in RL are semi-decidable and there is a number of methods and provers for coherent logic (some of them are based on simple forward reasoning and iterative deepening, with a number of techniques for narrowing the search space [15, 208, 209], while some use more advanced techniques, like lemma learning and back-jumping [167, 169]) and much more for RL [196]. CL admits a simple sequent-calculus style proof system, and any corresponding CL proof has a simple structure [208]. Readable proofs in a forward reasoning style can be easily obtained in CL [15]. The existing theorem proving methods for CL and RL do not deal with NDG conditions but can use the same heuristics as for the DD method (Section 2.2.3.2).

CL provers have been used in a variety of settings and domains. In particular, they have been applied to Euclidean geometry using an axiom system similar to Borsuk's [18, 129] and to Hilbert's and Tarski's axiomatics, with proofs exported to Isabelle, Coq, and natural language [208, 209]. They have also been used to prove the correctness of solutions to ruler and compass construction problems [153]. They have been used for projective plane geometry, where a proof of Hessenberg's theorem was carried out with Coq proof objects generated [16]. Recent work has also seen them combined with resolution theorem provers such as Vampire (for filtering relevant axioms) [210].

Example. The following theorem from Tarski's geometry [201] can be proved in CL (using the theorem prover ArgoCLP [209]): Assuming that A–B–C, $(A, B) \cong (A, D)$, and $(C, B) \cong (C, D)$, show that $B = D$. The presented proof is obtained by simplifying and transforming the generated proof, so it reintroduces negation and uses *reductio ad absurdum* [152].

1. It holds that B–A–A (by th_3_1).
2. From the facts that A–B–C, it holds that Col CAB (by ax_4_10_3).
3. From the facts that $(A, B) \cong (A, D)$, it holds that $(A, D) \cong (A, B)$ (by th_2_2).
4. It holds that $A = B$ or $A \neq B$ (by ax_g1).
    5. Assume that $A = B$.
        6. From the facts that $(A, D) \cong (A, B)$ and $A = B$ it holds that $(A, D) \cong (A, A)$.
        7. From the facts that $(A, D) \cong (A, A)$, it holds that $A = D$ (by ax_3).
        8. From the facts that $A = B$ and $A = D$ it holds that $B = D$.

        This proves the conjecture.
    9. Assume that $A \neq B$.

        Let us prove that $A \neq C$ by reductio ad absurdum.

        10. Assume that $A = C$.
            11. From the facts that A–B–C and $A = C$ it holds that A–B–A.
            12. From the facts that A–B–A, and B–A–A, it holds that $A = B$ (by th_3_4).
            13. From the facts that $A \neq B$, and $A = B$ we get a contradiction.

            Contradiction.

        Therefore, it holds that $A \neq C$.
    14. From the fact that $A \neq C$, it holds that $C \neq A$ (by the equality axioms).
    15. From the facts that $C \neq A$, Col CAB, $(C, B) \cong (C, D)$, and $(A, B) \cong (A, D)$, it holds that $B = D$ (by th_4_18).

        This proves the conjecture.

Quaife used the resolution theorem prover OTTER to prove several non-trivial theorems in Tarski's geometry [201], with a slightly modified axiom system [189]. During theorem proving, a number of techniques were employed to guide resolution and, upon success, some post-processing used to translate the resolution proofs into a more readable form. More recently, Beeson and Wos carried out some similar work, but with a newer version of OTTER and with much more success thanks to a number of techniques and strategies that have become available in the meantime [10, 11]. They proved around 200 theorems from the book by Schwabhäuser et al. [201]. Of these theorems, 76% were proved automatically using different custom heuristics and strategies, while for the others heavy human support (in a form of lemmas and hints) was required. This latest work did not involve the production of readable proofs from the resolution ones. Even more recently, other resolution provers with state-of-the-art techniques have led to an even higher percentage of theorems from the same corpus being proved completely automatically, without any guidance by humans [216].

Example. The following proof, slightly reformulated for the sake of uniformity, is generated by Quaife's approach: if C is between B and D, each of which is between A and E, then C is between A and E.

37	$U–V–W \Rightarrow W–V–U$	(by Axiom)
45	$U–V–X, V–W–X \Rightarrow U–W–X$	(by Lemma)
46	$U–V–W, U–W–X \Rightarrow U–V–X$	(by Lemma)
74	$U–V–X, U–W–X \Rightarrow U–V–W, U–W–V$	(by Lemma)
77	$A–B–E$	(by Hyp)
78	$B–C–D$	(by Hyp)
79	$A–D–E$	(by Hyp)
80	$\neg A–C–E$	(by negated goal)
91	$\neg A–C–D$	(by 80, 46, 79)
92	$\neg A–C–B$	(by 80, 46, 77)
109	$\neg A–B–D$	(by 91, 45, 78)
127	$\neg B–C–A$	(by 92, 37)
184	$A–D–B$	(by 109, 74, 79, 77)
253	$\neg B–D–A$	(by 127, 46, 78)
309	Contradiction!	(by 184, 37, 253)

2.2.4 Semisynthetic Methods

Semisynthetic methods, sometimes also called coordinate-free methods or geometric invariant methods, do not use algebraic formulation of geometry problems, but express conjectures in terms of certain *geometric quantities* and prove them by manipulating equalities over expressions in these quantities. This approach can also lead to combinatorial explosion, but in many cases can give short and readable proofs.

2.2.4.1 Area Method

The area method is a procedure for a fragment of Euclidean plane geometry [50, 51, 130, 229]. It uses suitably chosen geometry quantities, such as area of triangle, and can efficiently prove many non-trivial theorems and produces proofs that are often very concise and human-readable. The method had been extended to solid Euclidean geometry [52], to non-Euclidean geometries [225, 226] and, in conjunction with Collins algorithm [63], to a system for proving geometry inequalities [197].

Scope. A conjecture consists of a construction and a goal, where the construction is expressed in terms of (five basic) specific construction primitives (or constructions composed of the primitive ones), and the goal is an equality over expressions given in terms of (three basic) specific primitive geometry quantities. Both the construction and the goal are expressed only in terms of points (i.e., cannot involve lines or circles explicitly).

An example of a construction primitive is INTER $Y\ U\ V\ P\ Q$, which indicates that point Y is the intersection of lines UV and PQ. For a construction step to be well-defined, certain NDG conditions may be required. The above construction step has a NDG condition $U \neq V\ \wedge\ P \neq Q\ \wedge\ UV \nparallel PQ$. Intersections of two circles and intersections of a line and a circle are supported by construction primitives only in some special cases. Additional construction steps can be expressed in terms of the basic ones.

The geometric quantities used are: the signed ratio of parallel directed segments, denoted $\frac{\overline{AB}}{\overline{CD}}$, the signed area for a triangle ABC; denoted $\mathcal{S}_{ABC}$ (negated for the triangle with the opposite orientation); the Pythagoras difference, denoted $\mathcal{P}_{ABC}$ (for the points A, B, C, defined as $\mathcal{P}_{ABC} = \overline{AB}^2 + \overline{CB}^2 - \overline{AC}^2$). Using these quantities, a number of geometric predicates can be simply expressed, for instance: $A = B$ iff $\mathcal{P}_{ABA} = 0$; Col $A\,B\,C$ iff $\mathcal{S}_{ABC} = 0$; $AB \perp CD$ iff $\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{CDC} \neq 0 \wedge \mathcal{P}_{ACD} = \mathcal{P}_{BCD}$; $AB \parallel CD$ iff $\mathcal{P}_{ABA} \neq 0 \wedge \mathcal{P}_{CDC} \neq 0 \wedge \mathcal{S}_{ACD} = \mathcal{S}_{BCD}$, etc.

The method implemented by its authors proved 500 theorems from their collection [51].

Theorem proving mechanism. The method works by the elimination of constructed points in reverse order, using a set of specific elimination lemmas.[‡] All the lemmas used by the method can be proved by an elegant, custom axiom system (Section 2.3.1.4).

There is an elimination lemma for each pair of construction step and geometric quantity. For instance, the following lemma is used for eliminating a point constructed on a line at a given ratio from a signed area:

If Y is a point constructed on line PQ, such that $\frac{\overline{PY}}{\overline{PQ}} = \lambda$ then for any points A and B

$$\mathcal{S}_{ABY} = \lambda \mathcal{S}_{ABQ} + (1-\lambda)\mathcal{S}_{ABP}.$$

The combined NDG conditions of the conjecture is the conjunction of those for the corresponding construction steps, of the conditions that the denominators of the ratios of parallel directed segments in the goal equality are not equal to zero, and of the conditions that lines appearing in ratios of segments in the goal are parallel. It is then proved that the goal equality follows from the construction specification and the combined NDG conditions.[§]

Apart from the basic NDG conditions, there are also side conditions in some of the elimination lemmas having two cases — positive (always of the form "A is on PQ") and negative (always of the form "A is not on PQ"). If one side condition can be proved, then that case is applied. Otherwise, in one variation of the method, the proof process branches into two cases, and in another, the negative case is assumed and added to the NDG conditions [130].

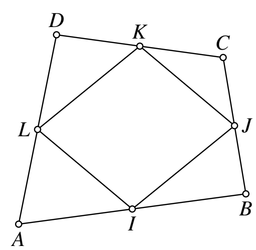


Figure 2.3
Varignon's theorem.

Example. As an example, we give the proof of Varignon's theorem: Given a quadrilateral $ABCD$, let I, J, K and L be the midpoints of AB, BC, CD, DA, then $IJKL$ is a parallelogram. We give below the proof that $IJ \parallel KL$, the proof that $JK \parallel IL$ is similar. Note that a synthetic proof within Coq is given in Figure 2.9.

$$\begin{array}{lll}
 & \mathcal{S}_{KIJ} - \mathcal{S}_{LIJ} & \\
= & \frac{\mathcal{S}_{KIB}}{2} + \frac{\mathcal{S}_{KIC}}{2} - \frac{\mathcal{S}_{LIB}}{2} - \frac{\mathcal{S}_{LIC}}{2} & J \text{ Eliminated} \\
= & \frac{\mathcal{S}_{BKA}}{2} + \frac{\mathcal{S}_{BKB}}{2} + \frac{\mathcal{S}_{CKA}}{2} + \frac{\mathcal{S}_{CKB}}{2} - \frac{\mathcal{S}_{BLA}}{2} - \frac{\mathcal{S}_{BLB}}{2} - \frac{\mathcal{S}_{CLA}}{2} - \frac{\mathcal{S}_{CLB}}{2} & I \text{ Eliminated} \\
= & \frac{1}{2}(\mathcal{S}_{BKA} + \mathcal{S}_{CKA} + \mathcal{S}_{CKB} - \mathcal{S}_{BLA} - \mathcal{S}_{CLA} - \mathcal{S}_{CLB}) & \text{Simplification} \\
= & \frac{1}{2}(\frac{\mathcal{S}_{ABC}}{2} + \frac{\mathcal{S}_{ABD}}{2} + \frac{\mathcal{S}_{ACC}}{2} + \frac{\mathcal{S}_{ACD}}{2} + \frac{\mathcal{S}_{BCC}}{2} + \frac{\mathcal{S}_{BCD}}{2} - \mathcal{S}_{BLA} - \mathcal{S}_{CLA} - \mathcal{S}_{CLB}) & K \text{ Eliminated} \\
= & \frac{1}{2}(\frac{\mathcal{S}_{ABC}}{2} + \frac{\mathcal{S}_{ABD}}{2} + \frac{\mathcal{S}_{ACC}}{2} + \frac{\mathcal{S}_{ACD}}{2} + \frac{\mathcal{S}_{BCC}}{2} + \frac{\mathcal{S}_{BCD}}{2} - \frac{\mathcal{S}_{ABA}}{2} - \frac{\mathcal{S}_{ABD}}{2} - \frac{\mathcal{S}_{ACA}}{2} - \frac{\mathcal{S}_{ACD}}{2} - \frac{\mathcal{S}_{BCA}}{2} - \frac{\mathcal{S}_{BCD}}{2}) & L \text{ Eliminated} \\
= & \frac{1}{4}(\mathcal{S}_{ABC} + \mathcal{S}_{BCA}) & \text{Simplification} \\
= & 0 & \text{Simplification}
\end{array}$$

Properties. The method is terminating, sound, and complete: for each geometric statement in its scope, it can decide whether it is a theorem, i.e., it is a decision procedure for this fragment of geometry. Its complexity is exponential in the number of points involved [229].

[‡] A later variant of the method also deals with nonconstructive statements, described in terms of various geometric predicates [53].

[§] If the negation of some NDG condition of a geometric statement is implied by the remaining construction steps, the left-hand side of the implication is inconsistent and the statement is trivially valid.

2.2.4.2 Full-Angle Method

The full-angle method [47] is, in spirit, closely related to the area method (Section 2.2.4.1) and can also produce elegant proofs for a number of complex theorems. The idea of eliminating points is extended to eliminating lines. The main motivation of the full-angle method is the fact that using "traditional angles" in geometrical proofs typically leads to considering a number of cases. For instance, for four distinct cyclic points A, B, C, D, one can claim that the angles $\angle ABC$ and $\angle ADC$ are congruent if B and D are on the same side of line AC *or* complementary if they are on opposite sides (Figure 2.4). On the other hand, with full-angles one can simply (without using order relation or orientations of plane) state $\angle[AD,CD] = \angle[AB,CB]$. Namely, it holds $\angle[AB,BC] = \angle[DE,EF]$ iff $\angle ABC \cong \angle DEF$ and the two angles have the same orientation or $\angle ABC = 180° - \angle DEF$ and the two angles have opposite orientations.

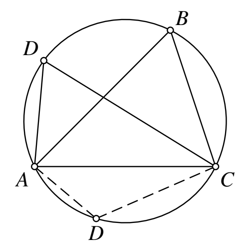


Figure 2.4
Cyclic points and peripheral angles.

A full-angle is defined to be an ordered pair $\angle[m,n]$ of two intersecting lines m and n, such that $\angle[m,n]$ is equal to another full-angle $\angle[u,v]$ if there is a rotation $\mathcal{R}$ such that $\mathcal{R}(m) \parallel u$ and $\mathcal{R}(n) \parallel v$ (therefore, any full-angle can be considered as an equivalence class) [51]. The sum of two full-angles is defined as follows: given four lines m, n, u, and v, and a rotation $\mathcal{R}$ such that $\mathcal{R}(u) \parallel n$, then $\angle[m,n] + \angle[u,v] = \angle[m,\mathcal{R}(v)]$. For arbitrary line m, $\angle[m,m]$ is denoted by $\mathbf{0}$. For arbitrary lines m and n, $\angle[m,n]$ can be denoted also by $-\angle[n,m]$. It can be proved that full-angles form an Abelian group with the operation $+$, the neutral element $\mathbf{0}$, and with inverse element corresponding to the (unary) operator $-$. In addition, $\angle[m,n] + -\angle[u,v]$ is abbreviated by $\angle[m,n] - \angle[u,v]$, and for arbitrary perpendicular lines m and n, $\angle[m,n]$ is denoted by $\mathbf{1}$.

It can be proved that full-angles satisfy around 20 properties useful for transforming goals, including the following ones (where, for each full-angle $\angle[AB,CD]$, it is assumed that $A \neq B$ and $C \neq D$):

R4: $\mathbf{1} + \mathbf{1} = \mathbf{0}$
R6: if Col PQX then $\angle[AB,PX] = \angle[AB,PQ]$
R10: if $cyclic(A,B,C,D)$ then $\angle[AD,CD] = \angle[AB,CB]$
R13: $\angle[AB,CD] = -\angle[CD,AB]$
R14: for any line UV, $\angle[AB,CD] = \angle[AB,UV] + \angle[UV,CD]$

Scope. The full-angle method deals with conjectures consisting of hypotheses, expressed in terms of relevant construction steps (cf. the area method in Section 2.2.4.1) or in terms of other geometric predicates (as in the later variation of the area method), and of a goal that is an equality over full-angles.

Theorem proving mechanism. The proof method uses forward chaining for exhaustively deducing new facts from the existing ones, using lemmas (rules) like:

F1: if $m \parallel n$ and $m \parallel l$, then $n \parallel l$
F5: if $PA \perp PB$ then $QA \perp QB$ iff $cyclic(A,B,P,Q)$
F8: if $AB \parallel AC$, then Col ABC
K2: if $m \perp n$ and $u \perp v$, then $\angle[m,u] = \angle[n,v]$

Some rules have NDG conditions attached and they can be treated as for the area method (see Section 2.2.4.1).

Using derived facts, rules like the ones listed above are used for elimination of points (R6) or lines (R10) from the goal – this is again analogous to the area method although the expressions are now simpler since there are no multiplications or divisions over full-angles.

The configuration is not necessarily expressed in terms of constructive statements and so (unlike in the basic version of the area method) there is no implicit order in which points can be eliminated. So, an ordering has to be imposed over points, which extends to full-angles, in order to control the application of the rules. For instance, the rule: *For any line UV, $\angle[AB,CD] = \angle[AB,UV] + \angle[UV,CD]$* is used only if the two new full-angles can be further reduced to full-angles less than $\angle[AB,CD]$ (in the ordering).

Example. For Simson's theorem (Figure 2.5) the hypotheses are $cyclic(A,B,C,D)$, E is the foot from D to BC (i.e., $\mathrm{Col}\,BCE$ and $DE \perp BC$), F is the foot from D to AC (i.e., $\mathrm{Col}\,ACF$ and $DF \perp AC$), G is the foot from D to AB (i.e., $\mathrm{Col}\,ABG$ and $DG \perp AB$), and the goal is that G, F, E are collinear, i.e., $\angle[GF,GE] = \mathbf{0}$.

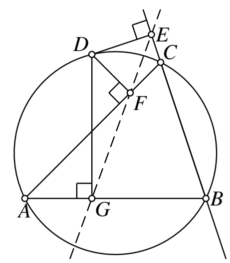


Figure 2.5
Simson's theorem.

The proof assumes that for each full-angle $\angle[XY,UV]$ used it holds that $X \neq U$ and $Y \neq V$. The following point order is used O,A,B,C,D,E,F,G, and the following facts (among others) can be derived from the hypotheses: $cyclic(F,A,D,G)$ (because $FA \perp FD, GA \perp GD$), $cyclic(E,B,D,G)$ (because $EB \perp ED, GB \perp GD$). In the following proof outline, applications of rules related to the symmetry properties of relations are not shown.

$$
\begin{aligned}
\angle[GF,GE] &= \\
&= \angle[GF,GD] + \angle[GD,GE] && \text{(by R14)} \\
&= \angle[AF,AD] + \angle[GD,GE] && \text{(by R10, } cyclic(F,A,D,G)) \\
&= \angle[AF,AD] + \angle[BD,BE] && \text{(by R10, } cyclic(E,B,D,G)) \\
&= \angle[AF,AD] - \angle[BE,BD] && \text{(by R13)} \\
&= \angle[AC,AD] - \angle[BE,BD] && \text{(by R6, Col } AFC) \\
&= \angle[AC,AD] - \angle[BC,BD] && \text{(by R6, Col } BCE) \\
&= \angle[AC,AD] - \angle[AC,AD] && \text{(by R10, } cyclic(A,B,C,D)) \\
&= \mathbf{0}
\end{aligned}
$$

Properties. The method is not complete, but can be used as a complement to the area method. When applied to a conjecture in its scope, if it succeeds, the generated proof is typically short and readable. Otherwise, the goal is transformed into a goal for the area method: an equality $\alpha = \beta$ is transformed[¶] into $\tan(\alpha) = \tan(\beta)$ and then further using the following equations (where $\mathcal{P}_{ABCD} = \mathcal{P}_{ABD} - \mathcal{P}_{CBD}$):

$$\tan(\angle[AB,CD] + \angle[PQ,UV]) = \frac{\tan(\angle[AB,CD]) + \tan(\angle[PQ,UV])}{1 - \tan(\angle[AB,CD])\tan(\angle[PQ,UV])}$$

$$\tan(\angle[AB,CD]) = \frac{4\mathcal{S}_{ACBD}}{\mathcal{P}_{ADBC}}$$

Since the area method is complete, the above gives a decision procedure for formulae belonging to the scope of the full-angle method [47].

2.2.4.3 Vector-Based Method

The idea of using vectors for automating geometric proofs has been proposed by several authors, but probably the most important work in the area is due to Chou, Gao, and Zhang [49]. Their method

[¶]The function tan for the full-angle (corresponding to the usual trigonometric function), $\tan(\angle[AB,CD]) = \frac{4\mathscr{S}_{ACBD}}{\mathscr{P}_{ADBC}}$, is well-defined, thanks to the fact that $\angle[AB,CD] = \angle[PQ,UV]$ iff $\mathscr{S}_{ACBD}\mathscr{P}_{PUQV} = \mathscr{S}_{PUQV}\mathscr{P}_{ACBD}$.

is, in spirit, close to the area method (Section 2.2.4.1), in the way the hypotheses are described constructively and the constructed points are eliminated from the goal one by one using appropriate lemmas.

Scope. The hypotheses are expressed in terms of (four) specific construction primitives. For instance, PRATIO $A\ W\ U\ V\ r$ denotes the construction of a point A such that $\overrightarrow{WA} = r\overrightarrow{UV}$, where r is a rational number, an expression over geometric quantities, or a parameter (the NDG condition is $U \neq V$). Additional construction steps can be suitably expressed in terms of the basic ones. For instance, the construction of the midpoint, denoted MIDPOINT $M\ A\ B$, can be expressed as PRATIO $M\ A\ A\ B$ 1/2.

A goal is either an equality over vectors or an equality involving the inner products ($\langle\overrightarrow{AB}, \overrightarrow{CD}\rangle$) and exterior products ($[\overrightarrow{AB}, \overrightarrow{CD}]$) of vectors over constructed points.

There are two kinds of NDG conditions: those induced by the construction steps and those necessary for the goal to be defined (denominators are not zero). Then, the conjecture is changed by augmenting the hypotheses with these NDG conditions.

Theorem proving mechanism. The method works by the elimination of constructed points in reverse order, using a set of specific elimination lemmas, like for the area method. There are elimination lemmas for each pair (construction step, geometry quantity). For instance, the following lemma is used for the elimination of a point Y constructed by the PRATIO step from the *linear* quantity $G(Y)$ satisfying $G(\alpha Y_1 + \beta Y_2) = \alpha G(Y_1) + \beta G(Y_2)$, for any real numbers α and β:

If Y is introduced by PRATIO $Y\ W\ U\ V\ r$*, then* $G(Y) = G(W) + r(G(V) - G(U))$.

Properties. The method is terminating, sound, and complete: for each geometry statement in its domain, it can decide whether it is a theorem, i.e., the method is a decision procedure for its fragment of geometry. The complexity of the method is exponential in the number of involved points [49].

2.2.4.4 Mass-Point Method

Barycentric coordinates and mass points have been used in geometry at least since 1969 [68], and were introduced in automated theorem proving for geometry by Zou and Zhang [230]. In the non-complex case, the method is similar to the method used by Kimberling for studying triangle centers (Section 2.3.2). A *mass point* is mP, where m ("mass") is a positive real number, and P is a point in a plane. Two mass points mP and nQ are equal iff $m = n$ and $P = Q$.

Scope. The conjecture consists of hypotheses (in the form of a construction) and a goal. Hypotheses are expressed in terms of three free (arbitrary) points and subsequent points are obtained by five basic geometric constructions and some compound ones (that enable a constructed point to be expressed as a linear combination of the three basis points), including:

C3 LRATIO $X\ A\ B\ r$, that gives a point X on the line AB such that $\overrightarrow{AX} = r\overrightarrow{AB}$, where r is a rational number, a rational expression, or a variable. Specially, MIDPOINT $X\ A\ B$ denotes LRATIO $X\ A\ B$ $1/2$. Constructing a point X such that $\overrightarrow{AX} = r\overrightarrow{XB}$ (or $(1+r)X = A + rB$) is denoted by MRATIO $X\ A\ B\ r$.

C5 INTER $X\ U\ V\ A\ B$, that gives the intersection point X of lines UV and AB (the NDG condition is that X is not equal to some of the points U, V, A, B, and that UV and AB are not parallel; otherwise, the prover fails).

The goal is a predicate over constructed points, one from a set that includes, for instance, Col ABC. For this predicate, it can be proved [228]: if P, Q, and R are points of the plane ABC (A, B, C are noncollinear points), and $P = a_pA + b_pB + c_pC$, $Q = a_qA + b_qB + c_qC$, $R = a_rA + b_rB + c_rC$, then P, Q and R are collinear iff

$$\begin{vmatrix} a_p & b_p & c_p \\ a_q & b_q & c_q \\ a_r & b_r & c_r \end{vmatrix} = 0.$$

The method is extended to deal with additional constructions (such as a construction of a circle) and uses complex numbers for convenience. This extended version has a scope strictly wider than the basic version.

The authors of the implementation successfully used it for proving hundreds of nontrivial theorems. Although the generated proofs are understandable, they are still not human-like proofs.

Theorem proving mechanism. The mass point method works by expressing all constructed points as a linear combination, of three (or two, for some simple conjectures) free points, then reformulating the goal the same way and finally proving it as a goal over real numbers.

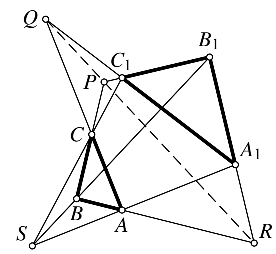


Figure 2.6
Desargues's theorem.

Example. Desargues's theorem (see also Section 2.3.2) states that, given two triangles $\triangle ABC$ and $\triangle A_1B_1C_1$, if the lines AA_1, BB_1, CC_1 intersect in a point S, then the intersection points P of BC and B_1C_1, Q of CA and C_1A_1, and R of AB and A_1B_1 are collinear (Figure 2.6). The theorem has to be slightly reformulated in order to use the mass point method. The construction is as follows:

Let A, B, and C be three free points.

Let S be an arbitrary point of the plane ABC, hence $S = aA + bB + cC$ for some real numbers a, b, c, such that $a+b+c=1$.

MRATIO A_1 S A x, for some x (then it holds $A_1 = \frac{1}{1+x}S + \frac{x}{1+x}A = \frac{1}{1+x}(aA+bB+cC) + \frac{x}{1+x}A = \frac{a+x}{1+x}A + \frac{b}{1+x}B + \frac{c}{1+x}C$).

MRATIO B_1 S B y, for some y (then it holds $B_1 = \frac{a}{1+y}A + \frac{b+y}{1+y}B + \frac{c}{1+y}C$).

MRATIO C_1 S C z, for some z (then it holds $C_1 = \frac{a}{1+z}A + \frac{b}{1+z}B + \frac{c+z}{1+z}C$).

INTER P B C B_1 C_1 (then it holds $yB - zC = (1+y)B_1 - (1+z)C_1 = (y-z)P$, i.e., $P = \frac{y}{y-z}B - \frac{z}{y-z}C$).

INTER Q A C A_1 C_1 (then it holds $Q = \frac{x}{x-z}A - \frac{z}{x-z}C$).

INTER R A B A_1 B_1 (then it holds $R = \frac{x}{x-y}A - \frac{y}{x-y}B$).

The goal is to prove that P, Q, R are collinear, which is done by showing that:

$$\begin{vmatrix} 0 & \frac{y}{y-z} & -\frac{z}{y-z} \\ \frac{x}{x-z} & 0 & -\frac{z}{x-z} \\ \frac{x}{x-y} & -\frac{y}{x-y} & 0 \end{vmatrix} = 0$$

Properties. The mass point method provides a decision procedure for conjectures within its scope [230].

2.2.5 Provers Implementations and Repositories of Theorems

There are a number of tools, typically providing dynamic geometry functionalities, that have support for the automated proof of geometry theorems. We mention the most notable ones next.

GEX/jGEX/MMP/Geometer is a family of systems equipped with provers based on algebraic approaches, the DD method, the area method, the vector method, and the full-angle method [98, 227]. GeoGebra [19], also equipped with several algebraic-based provers and tools based on the area method, is aimed at education. It can also work with the Coq proof assistant to support interactive proofs [180]. GeoProof is another tool linked to Coq that can use provers based on the area method, Wu's method, and the Gröbner basis method for generating machine verifiable proofs [163]. GCLC is a system that supports two algebraic methods and the area method [128]. Theorema [36], built on top of Mathematica, is a general mathematical tool with support for several theorem proving approaches, including the area method. OpenGeoProver is a library with several algebraic provers and one based on the area method [151]. Geometry Explorer uses the full-angle method [223] and provides means of visualizing geometric proofs as graphs.

There are ongoing efforts toward linking dynamic geometry systems with automated theorem proving and also with automated discovery, intelligent management of geometry knowledge, tutoring, eLearning, and so on [43, 142, 191, 219].

Finally, aside from collections of theorems available within the above tools, we note the existence of a dedicated repository of geometry theorems known as TGTP [190].

2.3 Interactive Theorem Proving

A proof assistant is a piece of software that can check mathematical assertions interactively. The main ones that have been used for the formalization of geometry are Coq [14, 66, 67], Isabelle [168, 175, 221], HOL4 [207], HOL-Light [119], and Mizar [215, 222]. They differ in their mathematical foundations (e.g., type theory, higher order logic [HOL], or set theory) and their proof language. In procedural style proof assistants (e.g., Coq and HOL Light), proofs are described as sequences of commands that modify the proof state whereas in proof assistants that use a declarative language (e.g., Mizar and Isabelle), the proofs are structured and contain the intermediate assertions that were given by the user and justified by the system.

2.3.1 Formalization of Foundations of Geometry

There are several ways in which the foundations of geometry can be laid.

In the *synthetic* approach, the geometry theory is built from axioms, with non-logical symbols corresponding to geometric predicates, and sorts corresponding to geometric objects.

The best-known *modern* axiomatic systems along these lines are those of Hilbert [124] and Tarski [214], which we will examine in detail next.

In the *analytic* approach, a field $\mathbb{F}$ is assumed (usually $\mathbb{R}$, the reals), the space is defined as $\mathbb{F}^n$, and the geometric objects and predicates are *defined*.

In the mixed analytic/synthetic approaches, one assumes both the existence of a field and also some geometric axioms. For example, the axiomatic systems for geometry proposed for education in North America by the School Mathematics Study Group in the 1960s are based on Birkhoff's axiomatic system [17] in which the underlying field ($\mathbb{R}$) serves to measure distances and angles. This approach, known as the metric approach, is developed in a number of modern sources [154, 159]. A similar approach is used by Chou, Gao, and Zhang for the foundations of the area method [229] (Section 2.2.4.1), where the underlying field is used to express ratios of signed distances and areas. The axioms and properties of the area method have been formalized in Coq [130]. Geometry can also be defined as a space of objects and a group of transformations acting on it (Erlangen program [135]), and several axiom systems based on this approach have been proposed [86, 170].

Axiom systems based on intuitionistic logic have also been proposed for geometry. Von Plato, for instance, uses the concept of apartness of points and convergence of lines to study plane geometry [183, 184]. Beeson, for his part, introduces a constructive version of Tarski's axiom system [7, 8].