

Francisco Botana
Pedro Quaresma (Eds.)

LNAI 9201

Automated Deduction in Geometry

10th International Workshop, ADG 2014
Coimbra, Portugal, July 9–11, 2014
Revised Selected Papers

 Springer

Lecture Notes in Artificial Intelligence

9201

Subseries of Lecture Notes in Computer Science

LNAI Series Editors

Randy Goebel

University of Alberta, Edmonton, Canada

Yuzuru Tanaka

Hokkaido University, Sapporo, Japan

Wolfgang Wahlster

DFKI and Saarland University, Saarbrücken, Germany

LNAI Founding Series Editor

Joerg Siekmann

DFKI and Saarland University, Saarbrücken, Germany

More information about this series at <http://www.springer.com/series/1244>

Francisco Botana · Pedro Quaresma (Eds.)

Automated Deduction in Geometry

10th International Workshop, ADG 2014
Coimbra, Portugal, July 9–11, 2014
Revised Selected Papers

Editors

Francisco Botana
University of Vigo
Pontevedra
Spain

Pedro Quaresma
University of Coimbra
Coimbra
Portugal

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-319-21361-3 ISBN 978-3-319-21362-0 (eBook)
DOI 10.1007/978-3-319-21362-0

Library of Congress Control Number: 2015944202

LNCS Sublibrary: SL7 – Artificial Intelligence

Springer Cham Heidelberg New York Dordrecht London
© Springer International Publishing Switzerland 2015

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media
(www.springer.com)

Preface

The tenth edition of the series of biennial international workshops on Automated Deduction in Geometry (ADG) took place at the University of Coimbra (Portugal) during July 9–11, 2014. ADG is a well-reputed conference where researchers and software developers working on geometry and automated deduction meet and discuss topics and applications related to automated reasoning in geometry. We acknowledge the support for ADG 2014 provided by the Centre for Mathematics of the University of Porto, the Center for Informatics and Systems, Center for Mathematics and the Science Museum of the University of Coimbra, the Portuguese Foundation for Science and Technology Portugal, and the City Hall of Coimbra.

The ADG 2014 workshop was a fruitful meeting, where four invited talks and 13 ordinary communications were given. The guest speakers were James Davenport, University of Bath, UK, António Leal Duarte, University of Coimbra, Portugal, Deepak Kapur, University of New Mexico, USA, and Tomás Recio, University of Cantabria, Spain.

From the beginning of the ADG meetings, it has been customary to launch an open call for papers inviting workshop participants and other involved people in the ADG community to participate in a proceedings volume in the LNAI series of LNCS. After a detailed peer-review process, we selected 11 articles, which show the trend set of current research in automated reasoning in geometry.

This volume includes a paper by Md. Ashraful Alam and Ileana Streinu describing an initial study of geodesic star unfoldings, a generalization of shortest-path star unfoldings of 3D convex polyhedra with a very simple characterization. In their contribution, Ciprian Borcea and Ileana Streinu study several properties of deformation spaces, including singularities, for families of volume frameworks associated with polygons. James H. Davenport and Matthew England discuss the recent major advances in Collins method for the real quantifier elimination, first proposed by Tarski in the 1930s. In the context of the new version 5 of GeoGebra, Zoltán Kovács presents the Relation Tool, an intuitive graphical user interface between various geometry automatic theorem provers and GeoGebra. In the paper by Vesna Marinković, Predrag Janičić, and Pascal Schreck, the authors present a formal logical framework describing the traditional four-phase process of geometric construction solving, leading to automated production of constructions with corresponding human readable correctness proofs. Shuichi Moritsugu describes computations of the relations between the circumradius R and area S of cyclic polygons given by the lengths of the sides, succeeding in computing integrated formulae of the circumradius and the area for cyclic pentagons and hexagons. The paper by Pavel Pech is on the extension of the well-known Simson–Wallace theorem on skew quadrilaterals in E^3 , investigating the locus of a point P whose orthogonal projections K, L, M, N onto the sides of a skew quadrilateral form a tetrahedron of a constant volume s . Pedro Quaresma and Nuno Baeta report the current status of the I2GATP format and its accompanying components. Meera Sitharam and

Joel Willoughby consider a generalization of the concept of d -flattenability of graphs, introduced for the l_2 norm by Belk and Connelly, to general l_p norms, with integer p , $1 \leq p < \infty$. Dan Song, Dongming Wang, and Xiaoyu Chen describe how they adopted techniques of Hough transform and randomized detection algorithms to detect geometric objects from scanned and photographed images, then use methods of image matching to recognize labels for the detected geometric objects, and finally employ numerical-computation-based methods to mine geometric relations among the objects. Finally, the paper by Menghan Wang and Meera Sitharam extends the combinatorial characterization of pinned subspace-incidence systems (H, X) that are minimally rigid to general pinned subspace-incidence systems, with H being a non-uniform hypergraph and pins in X being subspaces with arbitrary dimension.

We would like to thank the reviewers, both from the Program Committee and outside, for their reviews and revisions. We would also like to acknowledge the interest and help from Springer's LNAI editorial team.

July 2015

Francisco Botana
Pedro Quaresma

Organization

Invited Speakers

James Davenport	University of Bath, UK
António Leal Duarte	University of Coimbra, Portugal
Deepak Kapur	University of New Mexico, USA
Tomás Recio	University of Cantabria, Spain

Organizing Committee

Pedro Quaresma	University of Coimbra, Portugal, Chair
Miguel Abánades	Complutense University of Madrid, Spain
Nuno Baeta	Coimbra Institute of Engineering, Portugal
Nelma Moreira	University of Porto, Portugal
Jaime Carvalho e Silva	University of Coimbra, Portugal
Carlota Simões	University of Coimbra, Portugal

Program Committee

Francisco Botana	University of Vigo, Spain, (Chair)
Hirokazu Anai	Fujitsu Laboratories Ltd./Kyushu University, Japan
Xiaoyu Chen	Beihang University, China
Giorgio Dalzotto	ISI N. Machiavelli, Italy
Jacques Fleuriot	University of Edinburgh, UK
Xiao-Shan Gao	Chinese Academy of Sciences, China
Tetsuo Ida	University of Tsukuba, Japan
Predrag Janičić	University of Belgrade, Serbia
Ulrich Kortenkamp	Martin Luther University of Halle-Wittenberg, Germany
Shuichi Moritsugu	University of Tsukuba, Japan
Julien Narboux	University of Strasbourg, France
Pavel Pech	University of South Bohemia, Czech Republic
Pedro Quaresma	University of Coimbra, Portugal
Eugenio Roanes-Lozano	Complutense University of Madrid, Spain
Pascal Schreck	University of Strasbourg, France
Meera Sitharam	University of Florida, USA
Thomas Sturm	Max Planck Institute, Germany
Dingkang Wang	Chinese Academy of Sciences, China
Dongming Wang	Beihang University, China and UPMC-CNRS, France

Additional Reviewers

Fadoua Ghourabi
Hidenao Iwane
Robert Joan-Arinyo
Marek Kosta
Francisco Santos

Dan Song
Shin-Ichi Tanigawa
Menghan Wang
Bican Xia
Jing Yang

Contents

Star-Unfolding Polygons	1
<i>Md. Ashrafal Alam and Ileana Streinu</i>	
Volume Frameworks and Deformation Varieties	21
<i>Ciprian S. Borcea and Ileana Streinu</i>	
Recent Advances in Real Geometric Reasoning	37
<i>James H. Davenport and Matthew England</i>	
The Relation Tool in GeoGebra 5	53
<i>Zoltán Kovács</i>	
Computer Theorem Proving for Verifiable Solving of Geometric Construction Problems	72
<i>Vesna Marinković, Predrag Janičić, and Pascal Schreck</i>	
Integrated Circumradius and Area Formulae for Cyclic Pentagons and Hexagons	94
<i>Shuichi Moritsugu</i>	
Extension of Simson–Wallace Theorem on Skew Quadrilaterals and Further Properties	108
<i>Pavel Pech</i>	
Current Status of the I2GATP Common Format	119
<i>Pedro Quaresma and Nuno Baeta</i>	
On Flattenability of Graphs	129
<i>Meera Sitharam and Joel Willoughby</i>	
Discovering Geometric Theorems from Scanned and Photographed Images of Diagrams	149
<i>Dan Song, Dongming Wang, and Xiaoyu Chen</i>	
Combinatorial Rigidity and Independence of Generalized Pinned Subspace-Incidence Constraint Systems	166
<i>Menghan Wang and Meera Sitharam</i>	
Author Index	181

Star-Unfolding Polygons

Md. Ashraful Alam¹ and Ileana Streinu^{1,2}(✉)

¹ Computer Science Department, University of Massachusetts Amherst,
Amherst, MA 01003, USA

streinu@cs.umass.edu

² Computer Science Department, Smith College, Northampton, MA 01063, USA

Abstract. In this paper we initiate the study of *geodesic star unfoldings*. They are a generalization of shortest-path star unfoldings of 3D convex polyhedra and have a very simple characterization. We also address several problems concerning the existence of shortest-path star unfoldings on specified source point sets, and of reconstructing shortest-path star unfoldings with given ridge tree combinatorics.

1 Introduction

Let us consider the surface of a 3D convex polyhedron P , on which we draw a tree T whose edges are non-crossing geodesic arcs and whose leaves are the vertices of the polyhedron (Fig. 1(a)).

Polyhedral unfolding. By cutting the polyhedral surface along the tree edges we obtain a disk-like, intrinsically flat surface D with a polygonal boundary B . An *unfolding* of the polyhedron P is obtained by immersing the surface D into the 2D plane such that it is locally non-overlapping. Globally, the surface may self-overlap: its boundary is a planar polygon B , which, in general, may not be simple, i.e. may self-intersect in the 2D immersion. The tree T is called the *cut tree* of the unfolding.

Star unfolding. In this paper we restrict our attention to unfoldings where the cut tree is a *star*. A special case, called *star-unfolding*, is well known in the literature [2, 4, 9]: we fix a point s (the *source vertex*) on the surface of the polyhedron P and cut along the shortest paths [8] from s to all vertices v_1, v_2, \dots, v_n of P (Fig. 1). The resulting unfolding is called a *star-unfolding polygon*, and is known to be non-overlapping.

Our results. In this paper, we initiate the study of more general types of star-unfoldings, where the cut edges of the star-tree are non-crossing geodesics (i.e. not necessarily the *shortest* geodesics) from a given source vertex (Fig. 2). We refer to this more general setting as *geodesic star unfolding* (shortly, *star unfolding* when there is no risk of confusion) and use *shortest-paths star unfolding* (shortly, *sp-star unfolding*) for the special case previously studied in the literature, when all the cuts are along the *shortest* geodesics to all vertices.

We show that these more general unfoldings arising from (geodesic) star-trees have a very simple *characterization* as a family of planar, not necessarily simple

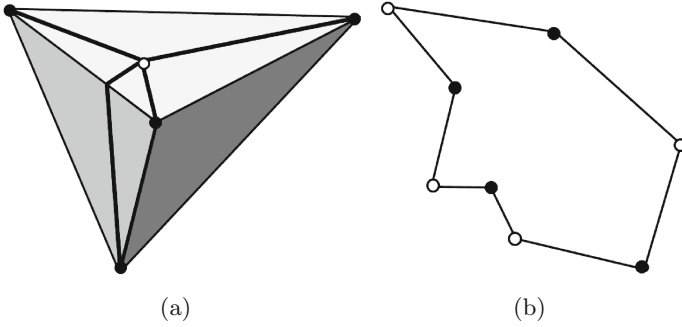


Fig. 1. (a) A tetrahedron and the tree of shortest paths (thick lines) from a source vertex placed on a face. (b) The corresponding sp-star unfolding polygon, with source and polyhedral vertices colored in white, resp. black (Color figure online).

polygons which we call *su-polygons* (“su”, naturally, has been chosen to indicate that they come from *star-unfoldings*). But only a subfamily of the su-polygons arise as shortest-path star unfoldings. Our first result is to prove that certain known necessary conditions (already present in the literature on shortest-paths star unfoldings) are also sufficient to distinguish them in the general class of (geodesic) star unfoldings.

We then study the problem of *constructing polyhedra via star unfolding polygons*, and address two questions: (a) given an ordered point set S in the plane, is there a (shortest path) su-polygon whose source vertices coincide with S ? and (b) are all combinatorial types of topologically embedded trees represented by the ridge trees of convex polyhedra from some source vertex?

The precise statements of the results require technical definitions which, along with further background on unfolding problems, are given in the next Sect. 2.

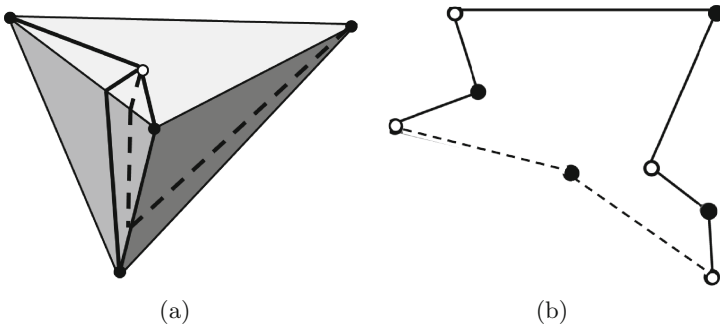


Fig. 2. (a) One of the shortest paths on the tetrahedron from Fig. 1(a) is replaced by a geodesic (dashed line). (b) The corresponding star unfolding polygon.

2 Definitions and Overview of Results

Different types of unfoldings are known in the literature, and they are distinguished by the choice of the cut tree. The example in Fig. 3 is an *edge unfolding*, where the cut tree is a spanning tree of the 1-skeleton¹ of the polyhedral surface, and the uncut edges of the polyhedron appear as diagonals of the unfolding polygon.

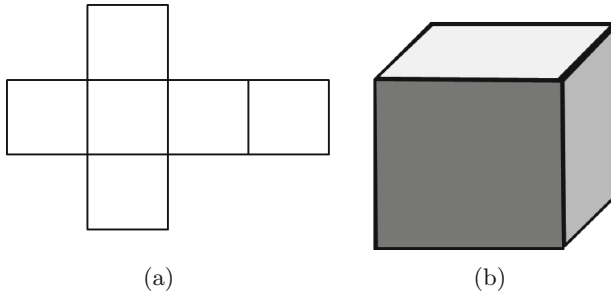


Fig. 3. A classical example of an edge unfolding (a), obtained by cutting the cube (b) along a spanning tree of its 1-skeleton.

Polyhedral edges are, in particular, shortest paths (on the polyhedral surface and in 3D) between their endpoint vertices, but general cut trees may have edges which are not globally shortest. *In this paper we work with geodesic unfoldings* where the tree edges are arbitrary geodesics. They are not, in general, along the polyhedral edges and may not even be along the shortest geodesics between their endpoints.

It is known that both self-overlapping and non-overlapping edge unfoldings exist. It is not known if each convex polyhedral surface admits a non-overlapping edge unfolding (see [5]). In general, understanding which cut trees induce non-overlapping unfoldings remains an elusive, long standing open question. However, the *shortest path star unfoldings* have the remarkable property of always being non-overlapping.

Basic properties of star unfoldings. The star-unfolding polygon S_s from a source point s is not guaranteed, *a priori*, to be non-overlapping. If the polyhedron P has n vertices (also referred to as *corners*), then S_s is a planar polygon with $2n$ vertices, if the source is not placed at one of the corners of the polyhedron, and $2(n - 1)$ otherwise. To streamline the presentation we work under the assumption that the first case holds. Along the boundary of the unfolding, the n vertices corresponding to the polyhedral corners appear in alternation with n vertices which are copies of the source vertex. The latter are known as *source images*, and are often, but not always in *convex position*. To fix the notation, we

¹ The 1-skeleton is the graph of polyhedral edges.

label the source images as $(s_i)_{i=1,\dots,n}$ and color them in white and the polyhedral vertices as $(v_i)_{i=1,\dots,n}$, and color them in black. See Figs. 1 and 2. The sum of all the angles, interior to the polygon, at the source images is exactly 2π under our assumption that the source is not placed at a vertex of the polyhedron. Another immediate property is that, in the unfolding, each polyhedral vertex v_i is placed on the perpendicular bisector of its adjacent two source images s_i and s_{i+1} .

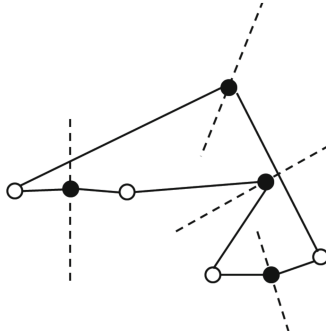


Fig. 4. An su-polygon. The white points are the s -vertices, with the v -vertices placed on the perpendicular bisectors (dashed lines) of consecutive s -vertices.

Su-polygons. By definition, an *su-polygon* (short for *abstract star-unfolding polygon*²) is the boundary of a flat disk-like polyhedral surface immersed in the plane, satisfying the abstract version of the previously stated properties: (a) it has $2n$ vertices, alternately labeled as s_i and v_i and referred to as s -, resp. v -vertices. The angle (interior to the surface) at an s -vertex is referred to as an s -angle; (b) the sum of all the s -angles is 2π , and (c) the v -vertices are placed on the perpendicular bisectors of the two neighboring s -vertices. In particular, this last property implies that the two edges incident to a v -vertex have equal lengths. Intuitively, the s -vertices, v -vertices and s -angles are analogous to source images, polyhedral vertices and source angles for star-unfolding polygons.

Since the boundary edges of sp-star-unfolding polygons must be shortest paths on the corresponding polyhedral surface, not all su-polygons arise as sp-star-unfoldings. However, we argue now that all su-polygons arise as (geodesic) star unfoldings of convex polyhedra, since they are *convex polyhedral metrics*. We remark that the definition allows for the self-overlap of the unfolded surface, i.e. the su-polygon is not necessarily a simple polygon.

Polyhedral metrics. A *convex polyhedral metric* is a presentation of the surface of a convex polyhedron as a collection of (one or more) planar polygonal pieces together with rules for glueing them, along pieces of their boundaries, into a

² The modifier “*abstract*” is used to emphasize that, *a priori*, such polygons are not guaranteed to arise from star unfoldings of 3D convex polyhedra.

surface that is topologically a sphere. The glueing may result in a finite set of points of non-zero Gaussian curvature, called the vertices of the surface. The metric is convex if the sum of the surface angles at each vertex is at most 2π , i.e. if the curvature is positive. Alexandrov's Theorem [3] guarantees that any convex polyhedral metric has an isometric realization as a convex polyhedron. As a side note we remark that the edges of the polygonal pieces need not be related in any way to the edges of the convex polyhedral realization. Su-polygons are special cases of polyhedral metrics, under the rule that the two equal sized edges incident to a v -vertex are glued together.

Geodesic and shortest paths star-unfoldings. Our goal is to characterize the polygons which arise as (geodesic or sp-) star-unfoldings of convex polyhedra. If furthermore the characterization leads to good algorithms, then we could in principle generate convex polyhedral metrics without going through their 3D polyhedral realizations. As an immediate consequence of the definition, of the discussion in the previous paragraph and of Alexandrov's theorem, we obtain:

Theorem 1. *Any su-polygon is a (geodesic) star unfolding of some convex polyhedron.*

This naturally leads to the following:

Problem 1. *Identify intrinsic properties of sp-star unfoldings that distinguish them in the general class of (geodesic) star unfoldings.*

We proceed now to identify such properties.

Ridge tree. Given a convex polyhedron P and a source vertex s , a *ridge point* is a point on the surface of P connected to s by multiple shortest paths. The collection of all such points is a tree T_s , known as the *ridge tree* from s . In an sp-star unfolding of P on the plane, this ridge tree becomes the part of the Voronoi diagram of the source images contained inside the sp-star unfolding polygon, which is, in this case, non-overlapping (see [4, 9] for details of these properties). Our first result is:

Theorem 2. *A simple su-polygon is an sp-star unfolding polygon if and only if (a) the perpendicular bisectors of the pairs of consecutive s -vertices are all present in the Voronoi diagram of all the s -vertices; and (b) any v -vertex lies precisely on the Voronoi edge corresponding to its two s -vertex neighbors.*

The relationship between the ridge tree and the Voronoi diagram of the source vertices has been identified and used in [4, 9], in the context of sp-star unfoldings. Our contribution is to identify the larger class of (geodesic) star-unfoldings and prove that these conditions are sufficient to characterize the class of sp-star unfoldings *within this larger class*. The fact that (in our formulation) they are also necessary is implicit in [4]. We show that if either one of the (a) or (b) conditions is violated in an su-polygon S , then S is *not* an sp-star unfolding.

Point sets supporting sp-star unfolding. We want to use Theorem 2 to construct examples of su-polygons which are, or are not, sp-star unfolding polygons.

We say that a cyclically ordered point set *supports* an su-polygon, resp. an sp-star unfolding if there exists an su-polygon, resp. an sp-star unfolding polygon using these points as s -vertices. We are asking:

Problem 2. *Given a cyclically ordered point set (not necessarily in convex position) with the property that the perpendicular bisectors of the pairs of consecutive vertices are all present in its Voronoi diagram. Does it support an sp-star unfolding?*

In other words, we want to know whether it is always possible to place v -vertices on the corresponding Voronoi edges to satisfy the *source angle sum* property of a su-polygon. We present an algorithm to decide the question.

Theorem 3. *There exist cyclically ordered point sets which do not support sp-star unfoldings, and others that do. For a given point set, the corresponding decision problem can be answered in $O(n \log n)$ time.*

Source vertices in convex position. A special situation appears when all the s -vertices of an su-polygon are in convex position. Equivalently, this means that the source images of a geodesic star-unfolding of some convex polyhedron lie in convex position. In this case, condition (a) in Theorem 2 always holds. The Voronoi diagram is combinatorially a tree, and its leaf edges are infinite rays separating the Voronoi regions of the s -vertices. We are now asking:

Problem 3. *If a given point set in convex position supports an su-polygon, does it support an sp-star unfolding?*

In other words, is it true that a *geodesic* star-unfolding whose source image vertices are in convex position is in fact a *shortest path* star-unfolding? We will answer this problem in the negative. The decision problem can be answered by an algorithm whose structure is similar to the one for general position, but whose individual components are computationally faster.

Theorem 4. *There exist cyclically ordered point sets in convex position which do not support sp-star unfoldings, and others that do. For a given convex point set, the corresponding decision problem can be answered in $O(n)$ time.*

Motivated by the important role played by ridge trees in understanding and characterizing star-unfoldings, we turn now to questions of *realizability* of their combinatorial types.

Combinatorics of ridge trees. For source vertices in convex position, the Voronoi diagram is a tree. The ridge tree inherits this combinatorics, captured by what we call a *topologically embedded tree*: a tree together with circular rotations of the edges incident to each interior node. We would like to know whether all the combinatorial types of such trees are represented among sp-star ridge trees. We prove the following *geometric reconstruction theorem* for combinatorial types of ridge trees.

Theorem 5. *Any combinatorial type of a topologically embedded tree can be realized by the ridge tree of a convex polyhedron together with a source vertex. Moreover, this can be arranged so that the sp-star unfolding of the polyhedron has all the source vertices in convex position.*

Outline. The rest of the paper is organized as follows. Theorem 2 is proven in Sect. 3. Section 4 treats the problem of deciding whether a given point set supports an sp-star unfolding and contains the proofs of Theorems 3 and 4. Section 5 deals with source points in convex position and contains the proof of Theorem 5.

3 Geodesic and Sp-Star Unfoldings

In this section we prove Theorem 2. One direction is already implicit in [4]. We only need to prove the other direction:

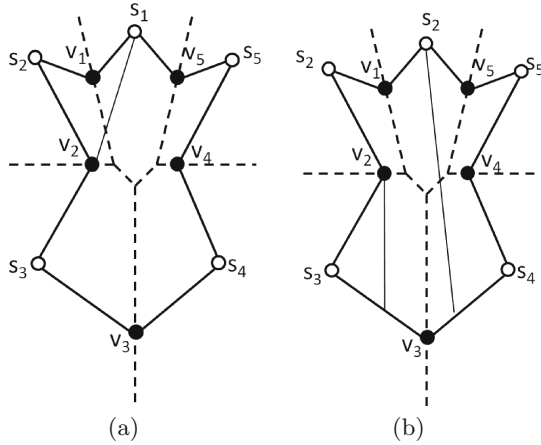


Fig. 5. The two cases appearing in the proof of Theorem 2.

Theorem 6. *Let S be a simple su-polygon satisfying two conditions: (a) the perpendicular bisectors of the pairs of consecutive s -vertices are all present in the Voronoi diagram of all the s -vertices; and (b) any v -vertex lies precisely on the Voronoi edge corresponding to its two s -vertex neighbors. Then S is an sp-star unfolding polygon.*

Proof. By Theorem 1, there exists a convex polyhedron P and a source vertex s such that S is a (geodesic) star-unfolding polygon for P . The proof now proceeds by contradiction. Let v_i be a v -vertex with two adjacent s -vertices s_{i-1} and s_i . We assume that the segment $v_i s_i$ (and its glueing mate of equal length $v_i s_{i-1}$)

does not correspond to the shortest geodesic on the polyhedral surface P . Then, on the polyhedral surface, the shortest path is located (in the cut-star rotation around the source point s) between some pair of cut edges from the source to v_j and v_{j+1} . In the su-polygon S , the shortest path induces a line segment starting from the s -vertex s_j and going towards the interior of the su-polygon.

Two scenarios are possible (Fig. 5): (a) Either the shortest path lies fully inside the su-polygon and it is a straight line joining v_i and s_j , or (b) the shortest path is *broken* and represented by a series of k line segments that exit the su-polygon through an edge and re-enters it again through its corresponding gluing edge.

In case (a) the proof is straightforward. Since v_i is on the Voronoi edge of s_{i-1} and s_i , then v_i is closer to s_i or s_{i-1} than any other s -vertices on the plane. Thus $v_i s_j$ cannot be the shortest path. See Fig. 5(a) for an example illustrating this case.

In case (b), we use the fact that if $v_k s_k$ is an edge of the su-polygon, then any point on this edge lies in the Voronoi region of s_k . Let the first segment of the broken shortest path be $v_i y_1$, where y_1 is the point through which it exits the polygon; similarly, let the last segment be $y_n s_j$. We know that y_n is on the polygonal edge $v_k s_k$, for $k \neq i$. But then we must have $s_k \neq s_j$, because otherwise $y_n s_j$ will make zero angle with the polygonal edge $v_k s_k$ (each line segment of the shortest path makes a non-zero angle when it exits and enters the polygon). Since y_n is on the edge of $v_k s_k$, it is on the Voronoi region of s_k , hence $|y_n s_k| < |y_n s_j|$. Therefore, there exists another s -vertex s_k which has a shorter length path from v_i . Thus the path from v_i to s_j is not the shortest. See Fig. 5(b) for an example illustrating this case.

The derived contradictions conclude the proof that each polygonal edge of the su-polygon comes from the shortest path unfolding of some convex polyhedron with respect to some source vertex. \square

We turn now to applications for this characterization of sp-star unfolding polygons.

4 Points Supporting Star-Unfoldings

In this section we assume that a cyclically ordered point set has been given. We want to decide if it supports an sp-star unfolding polygon and, if so, to construct one.

Flap polygon. This technical concept, needed to formulate and prove the results in this section, is obtained by relaxing the third condition defining an su-polygon: we no longer ask for the s -angle sum to be at most 2π (it can be anything).

4.1 Source Points in Convex Position

We start with the simpler situation when the s -vertices lie in convex position. In this case, their Voronoi diagram is a tree, and the leaf edges are infinite rays

separating consecutive s -vertices s_i and s_{i+1} . The conditions in Theorem 2 are violated only when some v -vertex is placed on the *extension* of a Voronoi ray and not on the ray itself.

To prove Theorem 4 we will construct a flap polygon with s -vertices in convex position and show that there is no readjustment of its v -vertices, keeping them on the Voronoi rays, so that the s -angle sum condition is satisfied. We need the following simple lemma:

Lemma 7. *Given a flap polygon, if a v -vertex moves towards the open end of the Voronoi ray, the sum of the s -angles increases and if it moves towards the closed end, the sum decreases.*

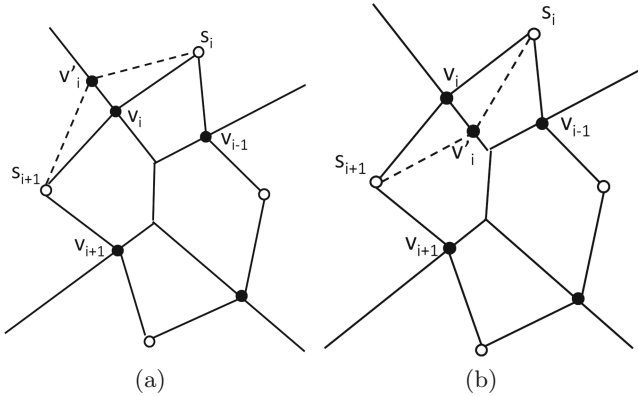


Fig. 6. Dashed lines indicate the increase/decrease behavior of the two source angles incident to a v -vertex v_i , when displaced from its position: (Left) increase, when the displacement is towards the open end of the infinite Voronoi edge, and (Right) decrease, when moved towards the closed end.

Proof. Straightforward, and illustrated in Fig. 6. □

When the s -vertices of a flap polygon are in convex position, the bisector of any consecutive s -vertices is an infinite edge (a ray) of the Voronoi diagram of the s -vertices. The v -vertices are placed on these rays. An *extreme flap polygon* arises when the v -vertices are placed on the Voronoi vertices at the closed end of the corresponding Voronoi rays. Note that an extreme flap polygon always has at least two pairs of adjacent overlapping edges, as in Fig. 7. This can be seen on the dual Delaunay triangulation, which is in this case a maximal outerplanar graph; thus, it has at least two vertices of degree two. One such vertex gives rise to overlapping adjacent edges in the extreme flap polygon.

The following is a straightforward consequence of Lemma 7:

Corollary 8. *For a given set of s -points in convex position and variable v -vertices placed on the Voronoi rays, the sum of the source angles is minimized when the corresponding flap polygon is extreme.*

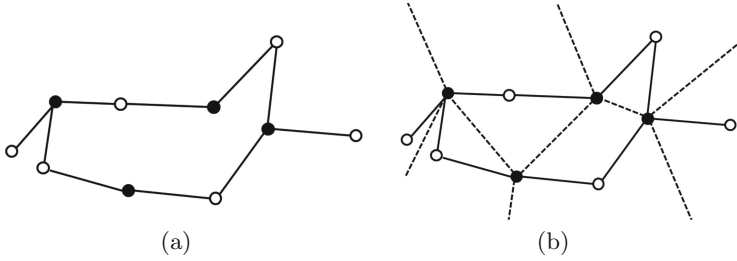


Fig. 7. An extreme flap polygon (a) together with the Voronoi diagram of the s -vertices (dashed) (b)

Lemma 9. *If the sum of the s -angles of an extreme flap polygon with s -vertices in convex position is larger than 2π , then there are no valid placements for the v -vertices that would result in an su -polygon supporting an sp -star unfolding.*

Proof. A flap polygon does not support a sp -star unfolding if the sum of its s -angles is larger than 2π or if the v -vertices are not placed on the Voronoi rays. But in an extreme flap polygon, the sum of the s -angles cannot be reduced (Corollary 8) without moving the v -vertices further inward, thus violating the condition that they must lie on the Voronoi rays. \square

Proof of Theorem 4. Using Lemma 9, we only need to find a set of points in convex position whose extreme flap polygon has total s -angle sum larger than 2π . This is not unique, and not rare either. An example where the sum of the s -angles is larger than 2π is illustrated in Fig. 7.

Algorithm. Given a set of points in convex position, the following simple algorithm decides if they support an sp -star unfolding. We first compute the Voronoi diagram, then we generate the extreme flap polygon by placing the v -vertices at the corresponding Voronoi vertices on the infinite rays. If the sum of the s -angles exceeds 2π , the given point set does not support an sp -star unfolding; otherwise, it does.

Computing the Voronoi diagram of n points in convex position can be done in $O(n)$ time using the algorithm in [1]. Constructing the extreme flap polygon takes linear time. The whole algorithm is therefore linear. \square

4.2 Source Points in Arbitrary Position

Not all sp -star unfolding polygons have s -vertices in convex position. An example is given in Fig. 8(a). We assume now that we are given a circularly ordered point set in non-convex position. The polygon induced by the ordering need not be simple. Such a point set may not even support a flap polygon, but when the perpendicular bisectors of consecutive pairs of points are present in the Voronoi diagram of the point set, there always exists at least one, namely the extreme flap polygon.

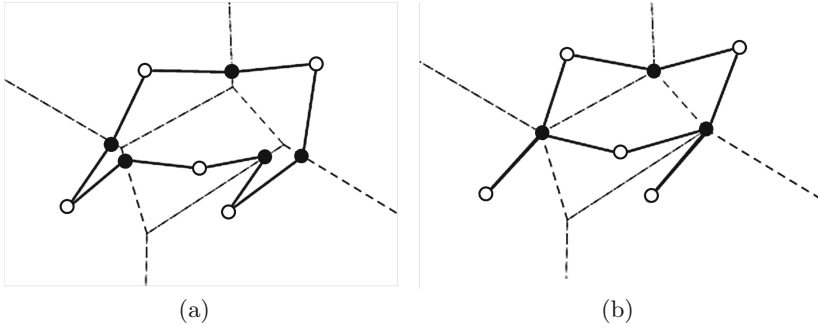


Fig. 8. (a) An *sp*-star unfolding polygon (thick) together with the Voronoi diagram of its *s*-vertices (dashed). The *s*-vertices are in non-convex position. (b) The corresponding extreme flap polygon on the *s*-vertices.

The definition of an *extreme flap polygon* can be extended naturally from the convex case: instead of requiring that the *v*-vertices be on the closed end of a Voronoi ray, we ask that it be placed on the endpoint of the Voronoi edge that is towards the interior of the flap polygon. Figure 8(b) illustrates an extreme flap polygon on *s*-vertices in non-convex position. Lemmas 7 and 8 above can be extended in a straightforward manner to the non-convex case.

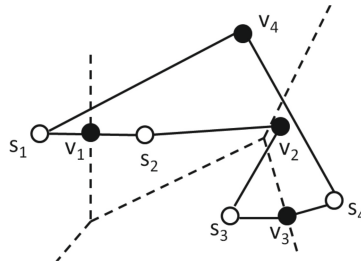


Fig. 9. An example of an *su*-polygon which violates condition (a): the bisector of $\{s_4, s_1\}$ is not in the Voronoi diagram.

The example illustrated in Fig. 9 has the property that *not all* of the consecutive perpendicular bisectors of the *s*-vertices are part of the Voronoi diagram. This proves that:

Lemma 10. *There exist circularly given point sets in non-convex position that do not support *sp*-star unfolding polygons by violating condition (a) of Theorem 6.*

For the example in Fig. 10, the consecutive perpendicular bisectors are present in the Voronoi diagram of the *s*-vertices, but the extreme flap polygon does not satisfy the *s*-angle condition: its *s*-angle sum is $380^\circ > 2\pi$. This proves:

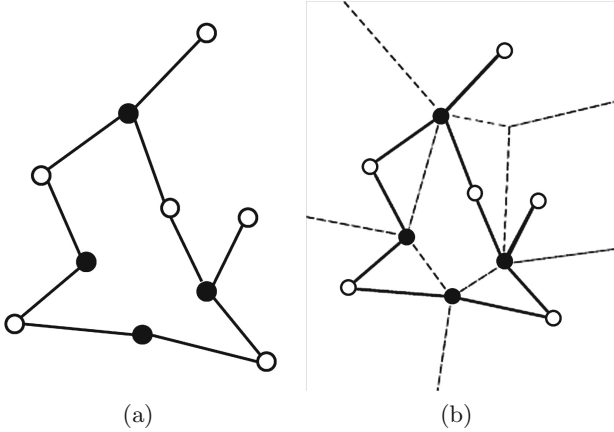


Fig. 10. (a) An extreme flap polygon with sum of the s -angles greater than 2π . (b) The polygon together with the Voronoi diagram of the s -vertices.

Lemma 11. *There exist cyclically given point sets in non-convex position that satisfy condition (a) but violate condition (b) of Theorem 6: they do not support sp -star unfolding polygons.*

We complete the proof of Theorem 3 by giving an algorithm to decide whether a cyclically ordered point set in non-convex position supports an sp -star unfolding polygon. This is a straightforward extension of the one previously presented for the convex case.

Algorithm. Given a cyclically ordered point set $(s_i)_{i=1,\dots,n}$, construct its Voronoi diagram in $O(n \log n)$ time and verify, in linear time, whether the perpendicular bisectors of consecutive pairs of points appear in the Voronoi diagram. If not, stop and return *False*. Otherwise, construct (in linear time) the extreme flap polygon and compute the sum of the s -angles. If they exceed 2π , return *False*, otherwise return *True*. The entire calculation takes $O(n \log n)$ time. \square

5 Realizations of Combinatorial Ridge Trees

In this section, we turn to realizations of sp -star unfolding polygons with prescribed combinatorics for their ridge trees, and prove Theorem 5. The techniques described in the previous sections reduce it to the following reformulation.

Theorem 12. *Any combinatorial tree T arises as the Voronoi diagram of a point set in convex position whose extreme flap polygon can be relaxed to an su -polygon that supports an sp -star unfolding.*

Proof overview. We first consider the maximal outerplanar graph which is the graph dual of the given tree. We realize this graph as the Delaunay triangulation

D_T using Dillencourt's algorithm [6]. The dual of this Delaunay triangulation D_T gives the realization of the tree as the Voronoi diagram V_D . The vertices of D_T are the sites of V_D . Since the sites of V_D are in convex position, there is an infinite Voronoi segment in between two consecutive sites. We place one new vertex on the Voronoi vertex located on the closed end of each of these infinite segments. If we alternately join the sites and new vertices, we obtain an extreme flap polygon. We show that this flap polygon supports an sp-star unfolding. For this, we establish a relationship between the s -angles of this extreme flap polygon and the angles of the Delaunay triangulation. We then use this relationship to prove that the sum of s -angles of this extreme flap polygon is always smaller than 2π .

The proof details occupy the rest of this Section.

5.1 Source Angles and the Angles of the Delaunay Triangulation

We start by establishing a relationship between the sum of the source angles of an extreme flap polygon and the angles of the Delaunay triangulation of its s -vertices.

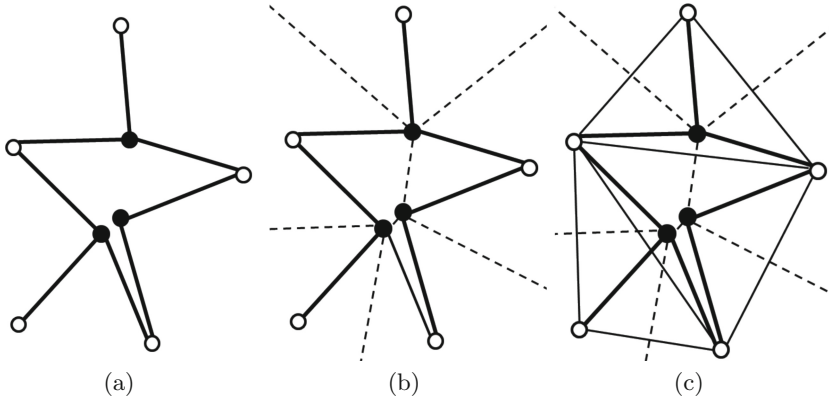


Fig. 11. An extreme flap polygon (a) together with the Voronoi diagram of the s -vertices (dashed) (b) and the Delaunay triangulation of the s -vertices (thin) (c).

Let D_T be the Delaunay triangulation of the s -vertices and let SA be the sum of the s -angles of the extreme flap polygon. If V_D is the Voronoi diagram of the s -vertices, its internal edges are duals of the diagonals of D_T .

Each s -angle in a extreme flap polygon is spanned by one or more internal edges of the Voronoi diagram. Therefore, the sum of the s -angles SA is actually the sum of the angles spanned by each internal edge of the Voronoi diagram. We establish the relationship between s -angles and the diagonals of the D_T , which are dual of the internal edges of V_D . See Fig. 11.

Given a Delaunay realization D_T of a maximal outerplanar graph, we can classify the angles of D_T into two sets. One is the set of angles opposite to all diagonals, known as internal angles and the other is the set of angles opposite to the boundary edges of D_T , known as boundary angles. Let α and β denote the sum of each of these two sets of angles respectively. Since these two sets of angles comprise all the angles of the convex polygon formed by the boundary edges of D_T , $\alpha + \beta = (n - 2)\pi$.

Lemma 13. *Let D_T be the Delaunay triangulation of n s -vertices and let SA_n be the sum of the s -angles of its extreme flap polygon. Then:*

$$SA_n = (n - 3)2\pi - 2\alpha_n$$

where α_n is the sum of the angles opposite to the diagonals of D_T .

Proof. By induction on the number n of s -vertices.

Base case, $n = 4$: In this case we have one diagonal in D_T and one internal edge in V_D . Out of four s -angles, two of them are zero and the remaining two are spanned by the internal edge of V_D . We have to show that:

$$SA_4 = 2\pi - 2\alpha_4$$

We use the notation from Fig. 12: the four s -vertices are A, B, C and D and XY is the Voronoi edge dual to the diagonal BD ; the source angles at A and B are both zero, the other two are not. We show that $\angle XBY + \angle XDY = 2\pi - 2(A+C)$, using equality of angles, such as $\angle BAX = \angle ABX$, resulting from points (such as X) being on the perpendicular bisector of others (such as A and B). The three main cases are illustrated in Fig. 12: (a) $\angle XBY$ is contained in, (b) is partially contained in, and (c) is outside of $\angle ABC$.

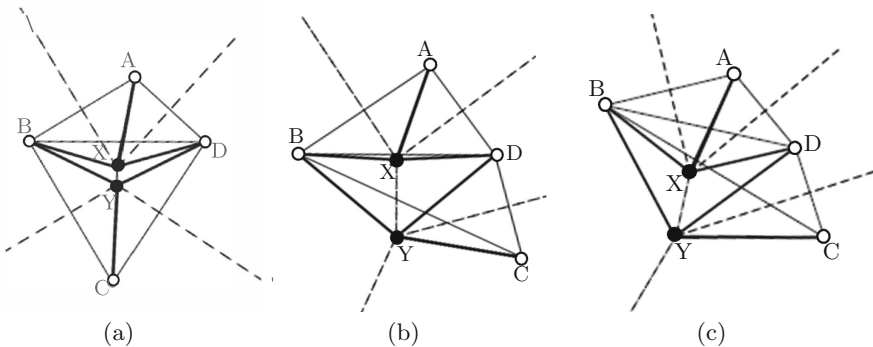


Fig. 12. The base case analysis for the inductive proof of Lemma 13. Shown is an extreme flap polygon for four convex points A, B, C and D . The triangles $\triangle ABD$ and $\triangle BCD$ are Delaunay.

Case 1: Using the angle sum of the quadrilateral $ABCD$ (which is 2π) and decomposing the angles $\angle ABC$ and $\angle CDA$ we obtain:
 $\angle ABX + \angle XBY + \angle YBC + \angle CDY + \angle XDY + \angle ADX = 2\pi - (\angle DAB + \angle BCD)$
 The bisector properties in the isosceles triangles $\triangle AXB$ and $\triangle BYC$ and simple manipulation imply the desired equality: $\angle XBY + \angle XDY = 2\pi - 2(\angle DAB + \angle BCD)$.

Cases 2 and 3: They are similarly treated, using $\angle ABC = \angle ABX + \angle XBY - \angle YBC$, bisector properties and simple manipulation to obtain the desired equality $\angle XBY + \angle XDY = 2\pi - 2(\angle DAB + \angle BCD)$. Since A and C are the opposite angles of the diagonal, we obtain:

$$SA_4 = 2\pi - 2\alpha_4$$

Inductive step. The induction hypothesis for $n - 1$ s -vertices is that:

$$SA_{n-1} = (n - 4)2\pi - 2\alpha_{n-1}$$

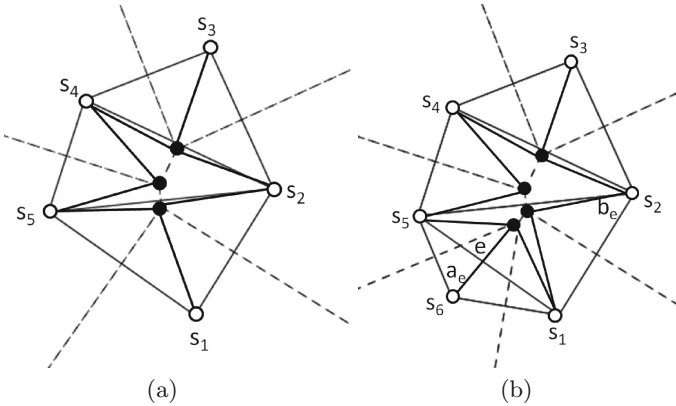


Fig. 13. The inductive step in the proof of Lemma 13. (a) An extreme flap polygon with 5 s -vertices. (b) A new s -vertex is added.

We now add a new s -vertex in convex position. As a result, a new triangle will be formed and one of the boundary edges will become a diagonal. Let e be this new diagonal. One of the two opposite angles of e was a boundary angle which has now become an internal angle. The other opposite angle is a newly formed internal angle of D_T . Let us denote these angles by a_e and b_e . All other internal angles of D_T remain the same.

The addition of the new s -vertex leads to a new diagonal or a new internal Voronoi edge. The sum of s -angles increases by SA' , the sum of angles spanned by the new Voronoi edge. See Fig. 13. Using a similar argument as in the base case, we obtain:

$$SA' = 2\pi - 2(a_e + b_e)$$

The final SA_n is:

$$\begin{aligned} SA_n &= SA_{n-1} + SA' \\ &= (n-3)2\pi - 2\alpha_n, \end{aligned} \tag{1}$$

where $\alpha_n = \alpha_{n-1} + a_e + b_e$

Corollary 14. $SA = 2\beta - 2\pi$, where β is the sum of boundary angles of the Delaunay triangulation of D_T .

Proof. In a Delaunay triangulation we have $\alpha + \beta = (n-2)\pi$. From Lemma 13 we infer that $SA = (n-3)2\pi - 2\alpha$. Substituting $\alpha = (n-2)\pi - \beta$ gives the result.

5.2 Realization of a Delaunay Triangulation

The set $(a_i)_i$ of all the angles in all the triangles in a Delaunay realization of an outerplanar graph satisfies a set of constraints [6]. Viewing each such angle as a variable a_i , the constraints on these variables are:

1. **Positive:** All angles are positive : $\forall i, a_i \geq 0$.
2. **Face angles:** The three angles a_i, a_j, a_k of a triangle add up to π :

$$a_i + a_j + a_k = \pi$$

3. **Convex:** All the points are in convex position, hence the sum of the angles $(a_{i_j})_{1 \leq j \leq k}$ at a vertex does not exceed π :

$$\sum_{j=1}^k a_{i_j} \leq \pi$$

4. **Locally Delaunay:** an edge shared by two triangles is locally Delaunay, i.e. the sum of the two angles opposite to the edge does not exceed π . If a_i and a_j are two angles opposite of an edge e , then $a_i + a_j \leq \pi$.

These constraints can be turned into a linear programming system on variables a_i . If the system has a feasible solution, it gives a set of angles from which the Delaunay triangulation can be realized. Dillencourt [6] showed that, for any maximal outerplanar graph, the linear programming system always has a feasible solution. However, a maximal outerplanar graph may have multiple Delaunay triangulation realizations, but not all of them yield valid sp-star unfoldings. We now show that in fact Dillencourt's algorithm always yields a valid sp-star unfolding. Before proceeding with the proof we outline the algorithm.

Dillencourt's Algorithm. A variable s is assigned to the whole triangulation T and a variable b_i to each angle of the triangulation, such that the following conditions are satisfied:

1. Each $b_i \geq 1$.
2. For each triangle, the sum of its angles is s : $b_i + b_j + b_k = s$.
3. For each vertex, the sum of all angles incident on it is less than or equal to s : $\sum_{j=1}^k b_{i_j} \leq s$.
4. If b_i and b_j are two opposite angles of an internal edge ij , then: $b_i + b_j \leq s - 1$.

By setting each $a_i = \frac{\pi \cdot b_i}{s}$, these conditions become equivalent to the angle constraints. An inductive argument now shows that such an assignment always exists.

Lemma 15. *Dillencourt’s algorithm yields a valid sp -star unfolding, with vertices in convex position.*

Proof. If a triangulation T has only one triangle, then we assign 1 to each angle and set $s = 3$. This assignment satisfies all of the four conditions. We now proceed inductively, visiting new triangles adjacent with the already visited ones. Each time a new triangle is visited, we assign values to its angles and update s . Then we traverse again the already visited triangles and modify their angles to reinstate the above conditions. For example, if we have assigned values to the angles of the triangle ijk , we may be moving to one of its adjacent triangle ijl whose angles will have to be initialized (see Fig. 14(a)). Let x, y and z be, respectively, the angles at the vertices i, j, k of $\triangle ijk$. Then we assign angle values of $z + 1, z + 1$ and $s - z - 1$ to the vertices i, j , and l of $\triangle ijl$. Then we update the value of s by setting $s := s + z + 1$. Since this new value of s causes the second condition to fail (where sum of angles of a triangle is equal to s), we traverse each of the visited triangles from our current $\triangle ijl$. When we enter a visited triangle by crossing an internal edge (i.e. a diagonal of the outerplanar graph), we increase the angle opposite to the internal edge by $z + 1$. For example, when we move from the triangle ijl to ijk , we cross the edge ij and increase the value of the angle at the vertex k by $z + 1$ (see Fig. 14(b)). We continue in this fashion until we re-visit

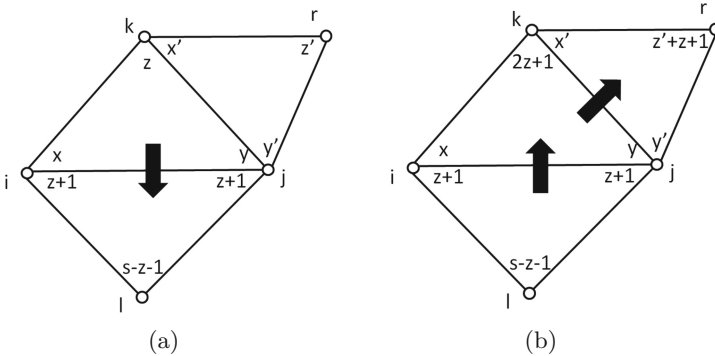


Fig. 14. Illustration of the steps in Dillencourt’s algorithm: (a) First visit of a triangle; (b) Revisiting previously visited triangles and readjusting the variables.

all the previously visited triangles. This whole process is repeated each time we visit and initialize the angles of a new triangle.

These assignments guarantee that all four conditions are satisfied. Indeed, all assignments are ≥ 1 . Every time we assign angles to a new triangle, s is increased by $z + 1$. We increase exactly one angle of each visited triangle by $z + 1$ and second condition is satisfied. When we update the angles of all visited triangles, the sum of the angles at each vertex is increased by $z + 1$ and at the same time s is also increased by $z + 1$. So, the inequality of condition 3 still holds. Finally, when we visit a new triangle, one of its angle is opposite to a diagonal (like angle at vertex l of triangle ijl). This angle is assigned $s - z - 1$ and its opposite angle in the just visited triangle (like angle at vertex k of triangle ijk) is assigned $z + (z + 1) = 2z + 1$. So, their sum $s + z \leq (s + z + 1) - 1$. After these assignments and modifications of these two triangles, we increase the angles of each visited triangle and this inequality remains valid. \square

We retain for further use the following properties of this algorithm:

1. Each time a new triangle is visited, the value of s is increased by $z + 1$.
2. After each iteration, one boundary angle (the one opposite to a boundary edge) becomes an internal angle (i.e. opposite to a diagonal), and two new boundary angles are introduced.
3. During the update phase, only internal angles are updated. Boundary angles remain unchanged.
4. The time complexity of this algorithm is $O(n)$ [7]

5.3 Proof of Theorem 12

In this section, we show that if a maximal outerplanar graph is realized as a Delaunay triangulation using Dillencourt's algorithm, then the sum of the s -angles in the extreme flap polygon is less than 2π .

Let D_T , SA and β denote the resulting Delaunay triangulation using Dillencourt's algorithm, the sum of the s -angles in the extreme flap polygon and the sum of the boundary angles (whose opposite edges are the boundary edges of the Delaunay triangulation) respectively. We know from Corollary 14 that, $SA = 2\beta - 2\pi$. If β is larger than π but smaller than 2π , then the resulting extreme flap polygon will support an sp-star unfolding.

Let us assume that Dillencourt's algorithm assigns a total integer value a to the boundary angles of D_T . Let $\beta = \frac{a-\pi}{s}$. We remind the reader that s is the sum of integer values assigned to any triangle.

Lemma 16. *Let a_n be the sum of the integer values assigned to the boundary angles and s_n be the sum of the integer values assigned to the angles of any triangle, where n is the number of the triangles in D_T . Then, $s_n < a_n < 2s_n$.*

Proof. By induction on the number of triangles of D_T .

Base case. The conditions are satisfied when $n = 2$, since $s_2 = 5$ and $a_2 = 6$.

Induction Step. We assume as induction hypothesis that $s_{n-1} < a_{n-1} < 2s_{n-1}$. Whenever we add a new triangle, one boundary angle becomes internal

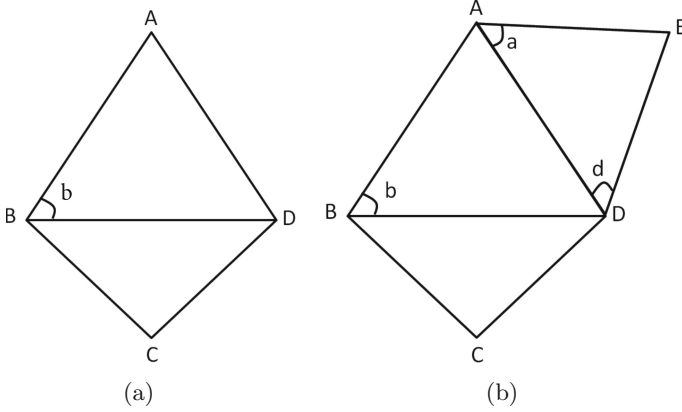


Fig. 15. Illustration of the inductive step in the proof of Lemma 16. (a) In this triangulation, b is a boundary angle. (b) At the next iteration, a new triangle AED is added and b becomes an internal angle; a and d are new boundary angles.

(whose opposite edge is a diagonal shared by the new triangle) and two new boundary angles are added to the triangulation. See Fig. 15. Each of the two new boundary angles is assigned $(z + 1)$ where z was the value of the now-converted internal angle. New value of s_n is obtained by adding $(z + 1)$ to s_{n-1} . All other boundary angles remain unchanged. Therefore:

$$a_n = a_{n-1} - z + 2(z + 1) = a_{n-1} + z + 2$$

Similarly, $s_n = s_{n-1} + z + 1$. Using the induction hypothesis that $s_{n-1} < a_{n-1} < 2s_{n-1}$ and replacing s_{n-1} and a_{n-1} in the above equation, we obtain:

$$\begin{aligned} s_n - z - 1 &< a_n - z - 2 < 2s_n - 2z - 2 \\ s_n + 1 &< a_n < 2s_n - z \\ s_n &< a_n < 2s_n \end{aligned}$$

This completes the proof. \square

Algorithm. The s -angles obtained from the Dillencourt's algorithm sum up to less than 2π . In a valid sp-star unfolding, the sum of these angles will have to be exactly 2π . In this section, we show how we can increase the angles such that they add up to 2π .

We define the slack angle s_a as the difference between 2π and the sum of the s -angles SA obtained from Dillencourt's algorithm, i.e., $s_a = 2\pi - SA$. All the v -vertices V are placed on the Voronoi vertices when we apply Dillencourt's algorithm. Now we move each v -vertex $v \in V$ along the infinite edge of the Voronoi diagram away from the corresponding Voronoi vertex (towards the open end). When we move v , each of the two s -angles at its neighboring s -vertices increases equally. Therefore, we move each v -vertex v such that each of the s -angles at its two neighboring s -vertices is increased by exactly $\frac{s_a}{2n}$.

The construction of the Delaunay triangulation from a maximal outerplanar graph takes $O(n)$ time [7]. The time complexity of moving the v -vertices to make the sum of the s -angles equal to 2π is also $O(n)$. Therefore, the whole algorithm runs in $O(n)$ time.

6 Conclusion

We have presented several results on geodesic star-unfolding, differentiating them from the special case when the geodesics are all shortest paths. We studied sp-star unfoldings with source vertices in convex position and showed that not all convex point sets support an sp-star unfolding. We gave a reconstruction theorem, showing that any maximal outerplanar graph arises as the dual of the ridge tree of a sp-star unfolding. Finally, we presented algorithms to check whether a given set of convex points can be realized as the source images of an sp-star unfolding and to realize any given outerplanar graph as a Delaunay triangulation whose vertices are the source images of a sp-star unfolding. Characterizing the pointsets which cannot be realized as source images of an sp-star unfolding remains an open problem.

Acknowledgement. This research was supported by the NSF grants CCF-1016988 and CCF-1319366.

References

1. Aggarwal, A., Guibas, L.J., Saxe, J., Shor, P.W.: A linear-time algorithm for computing the Voronoi diagram of a convex polygon. *Discret. Comput. Geom.* **4**(1), 591–604 (1989)
2. Agarwal, P.K., Aronov, B., O’Rourke, J., Schevon, C.A.: Star unfolding of a polytope with applications. *SIAM J. Comput.* **26**, 1689–1713 (1997)
3. Alexandrov, A.D.: *Convex Polyhedra*. Springer Monographs in Mathematics. Springer Verlag, Berlin Heidelberg (2005). English Translation of Russian edition, Gosudarstv. Izdat. Tekhn.-Teor.Lit, Moscow-Leningrad (1950)
4. Aronov, B., O’Rourke, J.: Non-overlap of the star unfolding. *Discret. Comput. Geom.* **8**, 219–250 (1992)
5. Demaine, E.D., O’Rourke, J.: *Geometric Folding Algorithms: Linkages, Origami, and Polyhedra*. Cambridge University Press, Cambridge (2007)
6. Dillencourt, M.B.: Realizability of Delaunay triangulations. *Inf. Process. Lett.* **33**(6), 283–287 (1990)
7. Lambert, T.: An optimal algorithm for realizing a Delaunay triangulation. *Inf. Process. Lett.* **62**(5), 245–250 (1997)
8. Mount, D.: On finding shortest paths on convex polyhedra. Technical report 1495, University of Maryland (1985)
9. Sharir, M., Schorr, A.: On shortest paths in polyhedral spaces. *SIAM J. Comput.* **15**(1), 193–215 (1986)

Volume Frameworks and Deformation Varieties

Ciprian S. Borcea¹ and Ileana Streinu²(✉)

¹ Department of Mathematics, Rider University,
Lawrenceville, NJ 08648, USA

² Computer Science Department, Smith College,
Northampton, MA 01063, USA
`streinu@cs.umass.edu`

Abstract. A volume framework is a $(d + 1)$ -uniform hypergraph with real numbers associated to its hyperedges. A realization is given by placing the vertices as points in \mathbb{R}^d in such a way that the volumes of the simplices induced by the hyperedges have the assigned values. A framework realization is rigid if its underlying point set is determined locally up to a volume-preserving transformation, otherwise it is flexible and has a non-trivial deformation space. The study of deformation spaces is a challenging problem requiring techniques from real algebraic geometry. Complementing a previous paper on *Realizations of volume frameworks*, we study several properties of deformation spaces, including singularities, for families of volume frameworks associated to polygons.

Keywords: Volume framework · Singularity · Deformation space
classification: 52C25 and 14N10

1 Introduction

In this paper we investigate *deformation varieties* (or *configuration spaces*) for volume frameworks associated to polygons.

Volume Frameworks. A bar-and-joint framework in \mathbb{R}^d may be conceived as a configuration of n labeled points respecting a system of distance constraints: certain pairs of points have a prescribed distance. Similarly, a volume framework in \mathbb{R}^d may be conceived as a configuration of n points respecting a system of volume constraints: certain ordered $(d + 1)$ subsets of the configuration span simplices of prescribed (signed) volume. In the first case, the system of constraints is given abstractly as a weighted simple graph on n vertices and equivalence of configurations is up to distance-preserving transformations of \mathbb{R}^d . In the second case, the system of constraints is given as a weighted $(d + 1)$ -uniform hypergraph on n vertices and equivalence of configurations is up to affine volume-preserving transformation.

Rigid and Flexible Structures. Classical questions in rigidity theory concern the characterization of *minimally rigid structures* [28], computing their *number of distinct realizations* [6, 7], or, for flexible structures, obtaining descriptions of

their space of *deformations*. These are notoriously challenging problems even for the well studied bar-and-joint case, where a full characterization is known only in dimension two [28, 29]. Generalizing finite frameworks to a periodic setting [8–10] has led to many advances in understanding bar-and-joint frameworks. In a similar spirit, we have initiated in [7] the study of rigidity theoretic properties of volume frameworks, where the generalization goes from graphs with fixed edge lengths to hypergraphs with fixed hyperedge volumes.

Deformation Spaces of Polygons. Deformation spaces of bar-and-joint structures have been characterized only in special cases [23, 30]. Particularly studied, and in different geometries, is the polygonal case [5, 24–27]. In this paper we initiate the study of *deformation varieties* or *configuration spaces* for volume frameworks associated to *polygons*. More precisely, for given $d \geq 2$ and $n \geq d + 3$, we examine volume frameworks corresponding to hypergraphs on vertices $1, \dots, n$ with n (cyclically marked) hyperedges: $12\dots(d + 1)$, $23\dots(d + 2)$, ..., $n12\dots d$. Such *cyclic volume frameworks* provide interesting deformation spaces, especially when considered from the *complex projective* point of view.

Related Work. In [7] we considered minimally rigid volume frameworks and obtained bounds on the number of non-equivalent realizations. We also showed that *sparsity* conditions on the hypergraph are *not* sufficient for characterizing minimal rigidity for $d \geq 2$, just as Maxwell’s sparsity conditions [29] for bar-and-joint frameworks are *not* sufficient for characterizing minimal rigidity for $d \geq 3$.

Here, we give particular attention to the *planar case* not only for comparison and contrast purposes with other instances of area constraint problems [15, 17, 18, 20], but mostly for the explicit and geometrically revealing character of *singular configurations*.

Our Contribution. In this paper we show that, when comparing deformation varieties of polygonal bar-and-joint and volume frameworks, there are significant similarities, but also noticeable distinctions, already in dimension two. While singular configurations for bar-and-joint frameworks remain singular under *projective* transformations, this *is not the case for singular configurations of area frameworks*. This disproves an expectation formulated by Whiteley in [2].

Nevertheless, singular configurations of cyclic area frameworks are remarkable in their own right and are best understood in relation to classical theorems of Desargues and Brianchon. The more general phenomenon of singular configurations on cyclic volume frameworks is related to points on *rational normal curves*.

The planar family may be compared to bar-and-joint polygon spaces described in [5], since both scenarios lead to complex deformation spaces of a distinctive type: elliptic curves, $K3$ surfaces and higher dimensional varieties related to Calabi-Yau manifolds. Varieties of this type play a key role in Mirror Symmetry [12].

Our investigation advances the theory of volume frameworks in the context of a more general but natural theory of frameworks associated to various *classical groups* [31]. Volume frameworks correspond to special linear groups in the same way that bar-and-joint frameworks correspond to orthogonal groups.

2 Preliminaries

We review some basic definitions and formulations involving Grassmanians. See also [7].

Let $H = (V, \mathcal{E})$ be a $(d + 1)$ -uniform hypergraph with vertex set V and hyperedge set \mathcal{E} made of certain ordered subsets I of cardinality $(d+1)$. Actually, we may take $V = \{1, 2, \dots, n\}$, and refer to hyperedges $I \in \mathcal{E}$ as multi-indices $I = (i_0, \dots, i_d)$, with distinct $i_k \in V$, $k = 0, \dots, d < n$.

A placement $p : V \rightarrow \mathbb{R}^d$ of the vertices in \mathbb{R}^d allows the interpretation of the hyperedges $I \in \mathcal{E}$ as (possibly degenerated) marked oriented simplices in \mathbb{R}^d and retaining their signed volume provides weights for \mathcal{E} . Assuming that at least one of the volumes is non-zero, the pair (H, p) gives a *volume framework* in \mathbb{R}^d .

Other placements $\tilde{p} : V \rightarrow \mathbb{R}^d$ which induce the same weights as (H, p) will be called *realizations* of the framework (H, p) . Two realizations which differ by an affine volume-preserving transformation of \mathbb{R}^d are considered equivalent. Equivalence classes of realizations define the *configuration space* of the framework. The realization space can be envisaged as a real algebraic variety and has thereby a topological structure. The configuration space is considered with the quotient topology. The connected component of the configuration represented by (H, p) is called the *deformation space* of the framework (H, p) .

Since placements are assumed to have at least one full dimensional marked simplex, equivalence under affine volume-preserving transformations can be factored out by *pinning* one such simplex, that is, the configuration space can be represented by all realizations maintaining the initial placement for the chosen simplex. This shows that the configuration space of a volume framework can be represented as (the real points) of an affine real algebraic variety.

Another approach, used in [7] and adopted in the present paper, factors out equivalence by resorting to Grassmann manifolds and Plücker embeddings [16]. We denote by $G(k, m)$ the Grassmannian consisting of all k -dimensional vector subspaces in a vector space of dimension m . The ground field will be \mathbb{R} or \mathbb{C} , according to context.

Let $p_1, \dots, p_n \in \mathbb{R}^d$ denote the placements of the vertices by p . With p_i as column vectors, we consider the $(d + 1) \times n$ matrix:

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ p_1 & p_2 & \dots & p_n \end{pmatrix} \tag{1}$$

The $(d + 1)$ rows of this matrix are independent and define a $(d + 1)$ -dimensional vector subspace in \mathbb{R}^n and thereby a point of the Grassmannian $G(d + 1, n)$. The Plücker (projective) coordinates of this point are given by all $(d + 1) \times (d + 1)$ minors:

$$V_I(p) = \det \begin{pmatrix} 1 & 1 & \dots & 1 \\ p_{i_0} & p_{i_1} & \dots & p_{i_d} \end{pmatrix} = \det (p_{i_1} - p_{i_0} \dots p_{i_d} - p_{i_0}) \tag{2}$$

for $I = (i_0, \dots, i_d)$, $1 \leq i_0 < \dots < i_d \leq n$. It follows immediately from (1) and (2) that replacing p_i , $i = 1, \dots, n$ by their images under an affine transformation does not change the corresponding point of the Grassmannian.

With obvious adaptation of signs, we may assume the hyperedges given as increasing subsets of $(d+1)$ indices, hence the volume constraints take the ratio form:

$$\frac{V_I(p)}{\delta_I} = \frac{V_J(p)}{\delta_J} \quad (3)$$

that is:

$$\delta_J V_I(p) = \delta_I V_J(p), \quad \text{for } I, J \in \mathcal{E}$$

where δ_I is the prescribed signed volume for hyperedge I . Note that the scaling aspect is resolved through the use of the projective setting. The essential parameters are the prescribed volumes up to proportionality.

Operating as above in $G(d+1, n)$ makes immediate the correspondence between prescribed volumes and marked Plücker coordinates. However, since the matrix (1) contains the row (11...1) the points under consideration can be identified with the points of the Grassmannian $G(d, n-1)$, in the quotient vector space $\mathbb{R}^n / \mathbb{R}(11\dots 1)$.

Thus, for \mathcal{E} of cardinality m and $\delta = (\delta_1 : \delta_2 : \dots : \delta_m) \in \mathbb{P}_{m-1}$, an independent system of volume constraints (3) gives a projective linear section of the Grassmannian $G(d, n-1)$ of codimension $m-1$.

Since the dimension of $G(d, n-1)$ is $d(n-d-1)$, a necessary *sparsity* condition for *minimally rigid volume frameworks* is that $m = d(n-d-1) + 1$ and on any subset of $n' > d$ vertices there are at most $d(n' - d - 1) + 1$ hyperedges.

3 Area Frameworks on Five Vertices

We start with some elementary examples and identify their singular configurations.

For three or four vertices there is, up to relabeling, a unique minimally rigid area framework hypergraph. With simplified notation, \mathcal{E} is made of 123, for three vertices and 123, 234, 341 for four vertices. The latter is obtained from the former by an elementary operation of ‘*vertex addition*’. This can be defined in arbitrary dimension d and consists in adding a vertex and marking d new simplices which have the new point as a vertex. Clearly, vertex addition takes minimally rigid graphs to minimally rigid graphs.

Proposition 1. *Up to relabeling, a minimally rigid area framework on five vertices $1, \dots, 5$ is one of the following:*

- (1) 512, 523, 534, 541, 524
- (2) 541, 542, 543, 531, 421
- (3) 345, 145, 245, 125, 123
- (4) 512, 523, 534, 541, 123
- (5) 123, 234, 345, 451, 512

Proof: It is more convenient to mark a triangle by the pair of vertices (edge) it does not contain. Sparsity permits at most three edges from any vertex. Since five edges on five vertices must form at least one cycle, cases (1), (2) and (3) result from a triangular cycle, (4) from a quadrangular cycle and (5) from a full pentagonal cycle. \square

Remark: (1) to (4) can be obtained from a triangle by vertex additions and have, generically, unique realizations. Thus (5), which can be obtained from a minimally rigid graph on four vertices by a vertex splitting operation [7] is the more interesting case.

We determine here the *infinitesimally flexible* configurations for area frameworks on five vertices indexed 1,...,5, with marked triangles 123, 234,..., 512. There is no loss of generality if we ‘pin’ the triangle 123 with vertex 2 at the origin and vertices 1 and 3 at (1, 0), respectively (0, 1). Then 4 and 5 have coordinates at (β, y) , respectively (x, α) , with α and β fixed by the areas of 512 and 234.

The two remaining areas 345 and 451 impose conditions of the form:

$$\det \begin{pmatrix} 1 & 1 & 1 \\ 0 & \beta & x \\ 1 & y & \alpha \end{pmatrix} = \alpha\beta - \beta - xy + x = b$$

$$\det \begin{pmatrix} 1 & 1 & 1 \\ \beta & x & 1 \\ y & \alpha & 0 \end{pmatrix} = \alpha\beta - \alpha - xy + y = a$$

This gives:

$$y = x + c \quad \text{with } c = (a + \alpha) - (b + \beta)$$

$$x^2 + (c - 1)x + (b + \beta - \alpha\beta) = 0$$

Infinitesimal flexibility means a double root in the last equation, which results in:

$$x = \frac{1 - c}{2}, \quad y = \frac{1 + c}{2} \quad \text{i.e. } x + y = 1$$

Geometrically, this means that the parallel to edge (1, 2) through vertex 4 meets the parallel to edge (23) through vertex 5 on the line (13). We’ll mark this point as 2’.

Since infinitesimal flexibility is not affected by cyclic permutations of the indices, the above considerations yield the following:

Condition (\ast_5): For all indices $i \in Z_5$, taken mod 5, the parallel to edge $(i, i + 1)$ through vertex $i + 3$ meets the parallel to edge $(i + 1, i + 2)$ through vertex $i + 4$ on the line $(i, i + 2)$. We denote this point as $(i + 1)'$.

Infinitesimally Flexible Configurations: The pentagon 1’2’3’4’5’ has edges parallel to the corresponding edges of the pentagon 12345 and is mutually inscribed with the pentagon 13524. Together, the vertices and edges of the pentagons 1’2’3’4’5’ and 13524 determine a configuration of type 10_3 , with each point

on three lines and each line passing through three points [13,21,22]. In fact, as shown in Fig. 1, it is a *Desargues configuration*. Two triangles in perspective are $41'5'$ and $3'54'$, with center of perspective at 2 and pairs of corresponding edges meeting at the three collinear points $2'$, 3 and 1.

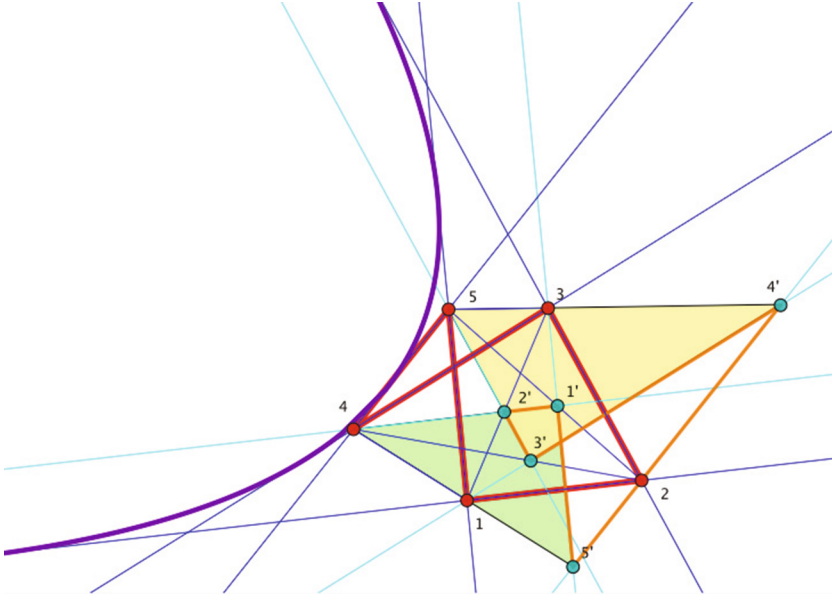


Fig. 1. Infinitesimal flexibility of a pentagon leads to a Desargues configuration. The lines supporting the pentagon edges are tangent to a parabola.

Remark: Since arbitrary projective transformations won't preserve this special class of affine polygons 12345, we see that the property of *infinitesimal flexibility* of volume frameworks is not invariant under projective transformations, in contrast to the bar-and-joint case. This aspect is emphasized in the following equivalent formulation:

Proposition 2. *A cyclic area framework on five vertices, with marked triangles $(i - 1, i, i + 1)$, $i \in Z_5$, is infinitesimally flexible if and only if all edges $(i, i + 1)$ are tangent to a parabola.*

Proof: Infinitesimal flexibility amounts to condition (\ast_5) identified above, namely: the parallel to edge $(i, i + 1)$ through vertex $i + 3$ meets the parallel to edge $(i + 1, i + 2)$ through vertex $i + 4$ on the line $(i, i + 2)$, for all $i \in Z_5$. We have to prove its equivalence with the fact that the pentagon has edges tangent to a parabola. For one implication, we may observe that an affine parabola may be treated as a projective conic with a distinguished tangent ℓ_∞ (that tangent being the line at infinity). Suppose our pentagon has edges tangent to the conic

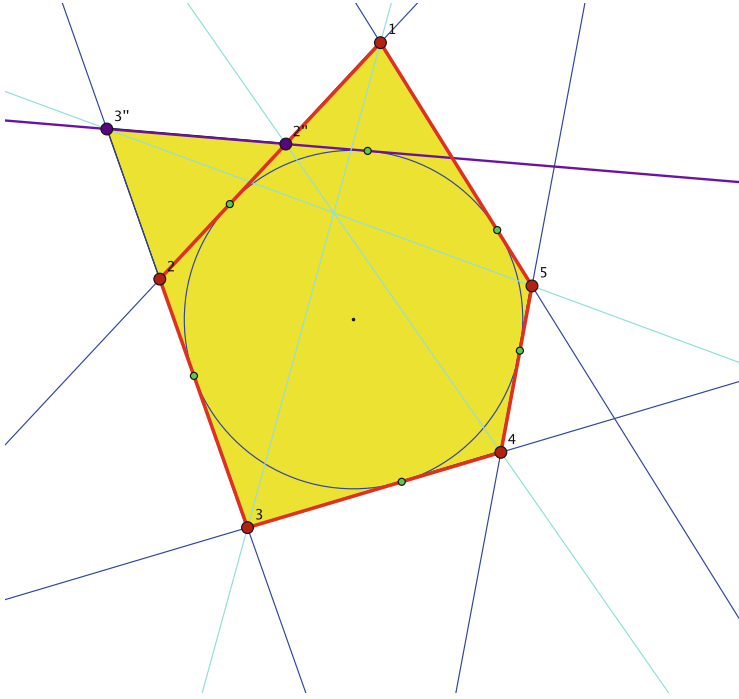


Fig. 2. Singular configuration of a pentagon. The line at infinity ℓ_∞ passes through points $2''$ and $3''$.

as illustrated in Fig. 2 and we want to establish condition (\ast_5) for $i = 1$. We mark the following intersections of lines:

$$(1, 2) \cap \ell_\infty = 2'', \quad (2, 3) \cap \ell_\infty = 3''$$

Then, $15433''2''$ is a hexagon with edges tangent to the conic. Thus, by Brianchon’s theorem [21], the diagonals $(1, 3), (5, 3''), (4, 2'')$ are concurrent. Affinely, this is condition (\ast_5) .

The other implication amounts to using Brianchon’s converse. □

4 Cyclic volume frameworks

In this section we outline a relation between configuration spaces for certain volume frameworks and varieties expected to allow natural desingularizations to Calabi-Yau manifolds. It is analogous to the relation established in [5] between polygon spaces and Darboux varieties (which have natural resolutions to Calabi-Yau manifolds).

Definition. A $(d + 1)$ -uniform hypergraph will be called *cyclic* when defined on $n \geq d + 3$ vertices $1, \dots, n$ by marking (cyclically) the n hyperedges: $12\dots(d + 1), 23\dots(d + 2), \dots, n12\dots d$.

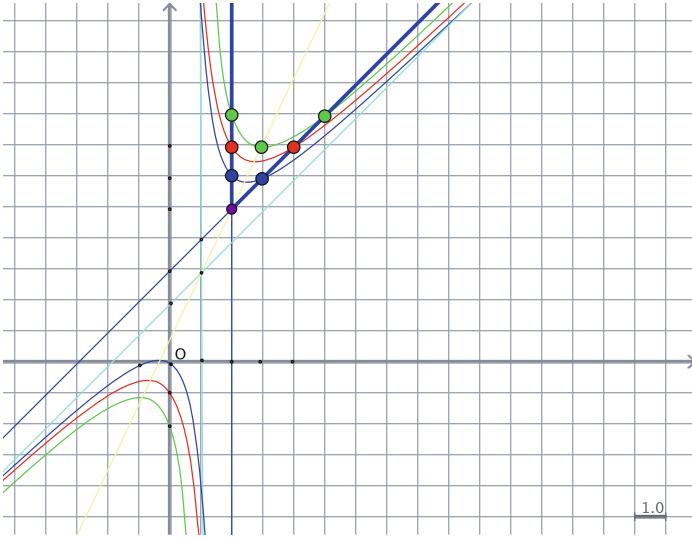


Fig. 3. The (d, n) -plane with hyperbolas $d(n - d - 1) - (n - 1) = D$ depicted for $D = 1, 2, 3$. There is an affine involution $(d, n) \mapsto (n - d - 1, n)$ preserving all hyperbolas in this pencil.

The presentation will be streamlined if we adopt directly a *projective set-up*. Since the scale factor is determined by any non-zero volume, we may consider the equivalence of point configurations up to affine transformations and impose the *volume constraints as ratios of two volumes*. Thus, as seen above, point configurations modulo affine equivalence correspond to $G(d, n - 1) \subset \mathbb{P}^{\binom{n-1}{d}-1}$ and constraints become hyperplane sections.

For cyclic frameworks we have $n - 1$ (ratio) prescriptions resulting in a codimension $(n - 1)$ linear section of the Grassmannian. While generic linear sections of this codimension do yield, over the complex field, smooth submanifolds with trivial canonical class i.e. *Calabi-Yau manifolds*, we have yet to determine particular properties of the sections dictated by framework constraints.

For n vertices in dimension d the space of parameters (the chosen volume ratios) is \mathbb{P}_{n-1} and the complex configuration spaces have dimension: $D(d, n) = d(n - d - 1) - (n - 1) = (d - 1)(n - 1) - d^2$.

Recalling that $d \geq 2$ and $n \geq d + 3$, one may notice in Fig. 3 the symmetry of this array of dimensions under the involution $(d, n) \mapsto (n - d - 1, n)$ which exchanges the two border lines $d = 2$ and $n = d + 3$:

$$D(d, n) = d(n - d - 1) - (n - 1) = D(n - d - 1, n) \tag{4}$$

4.1 Association

Formula (4) is an indication that *association* is implicated in this pairing. The notion of association goes back to Castelnuovo and Coble [11]. It expresses the

fact that the projective invariants of configurations of $(n + 1)$ ordered points in \mathbb{P}_d can be identified with the projective invariants of associated configurations of $(n + 1)$ points in \mathbb{P}_{n-d-1} . See also [4, 14].

We note first that a cyclic volume configuration of type (d, n) is completely described by the affine hyperplanes $[12\dots d], [23\dots(d + 1)], \dots, [n1\dots(d - 1)]$. This gives a *projective* configuration of $(n + 1)$ hyperplanes when we retain the hyperplane at infinity as the last element of this ordered list. Thus, we have $(n + 1)$ points in the dual projective space \mathbb{P}_d^* . For general configurations, association provides, up to projective equivalence, corresponding configurations of $(n + 1)$ points in \mathbb{P}_{n-d-1}^* , which give $(n + 1)$ hyperplanes in \mathbb{P}_{n-d-1} . With the last hyperplane interpreted as the hyperplane at infinity, we have a configuration of n affine hyperplanes in C^{n-d-1} , up to affine equivalence i.e. a cyclic volume configuration of type $(n - d - 1, n)$.

4.2 Cyclic Area Frameworks

Here the emphasis will be on $d = 2$ i.e. *cyclic area frameworks*. We are going to use the following abbreviations:

$$\Delta_{ijk} = \det \begin{pmatrix} 1 & 1 & 1 \\ p_i & p_j & p_k \end{pmatrix}, \quad \Delta_{(i-1)i(i+1)} = \Delta_i$$

With indices considered *mod n*, the equations defining the (projective) configuration space of a cyclic area framework on n vertices are:

$$\frac{\Delta_1}{\delta_1} = \frac{\Delta_2}{\delta_2} = \dots = \frac{\Delta_n}{\delta_n} \tag{5}$$

that is:

$$\delta_i \Delta_j = \delta_j \Delta_i, \quad 1 \leq i < j \leq n, \quad \delta = (\delta_1 : \dots : \delta_n) \in \mathbb{P}_{n-1}$$

Solutions with $\Delta_1 = \dots = \Delta_n = 0$ will be called *degenerate solutions*. For any $\delta_i \neq 0$, the hyperplane section $\Delta_i = 0$ consists of precisely these ‘degenerate’ solutions. As a divisor, or divisor class, it will be called the *degeneracy divisor* or the *divisor at infinity*.

If we look back at the source of our set-up, we have

$$G(2, n - 1) \subset G(3, n) \subset \mathbb{P}_{\binom{n}{3}-1} \cdots \rightarrow \mathbb{P}_{n-1}$$

where the last map is rational, with indeterminacy locus $\Delta_1 = \dots = \Delta_n = 0$. Thus, our degeneracy locus is where this indeterminacy locus meets the configuration space.

If we denote by $\mathbb{P}_{\binom{n-1}{2}-n}(\delta)$ the codimension $(n - 1)$ projective space defined by equations (5) in the linear span of the Grassmannian $G(2, n - 1)$, the projective configuration space of a cyclic area framework on n vertices is the linear section:

$$X_{n-5} = X_{n-5}(\delta) = G(2, n - 1) \cap P_{\binom{n-1}{2}-n}(\delta) \subset \mathbb{P}_{\binom{n-1}{2}-n} \tag{6}$$

4.3 Other Birational Models

The projective configuration space (6) obtained above involves no particular choice, but one may consider birationally equivalent realizations by ‘pinning’ a certain triangle. Suppose (up to rescaling and relabeling if necessary) that $\delta_1 = 1/2$ and pin triangle $(n12)$ with vertex 1 at the origin, and the two edges from it corresponding to the standard basis. This eliminates the action of affine transformations and the remaining vertices from 3 to $n - 1$ can be parametrized by $(\mathbb{P}_2)^{n-3}$. If we label the hyperplane classes in each factor by h_i , according to the corresponding vertex, the remaining $n - 1$ area prescriptions yield divisors of the following type (pull-back classes being represented by the same symbols):

$$h_3, h_3+h_4, h_3+h_4+h_5, h_4+h_5+h_6, \dots, h_{n-3}+h_{n-2}+h_{n-1}, h_{n-2}+h_{n-1}, h_{n-1}$$

Note that the sum of these divisors on $(\mathbb{P}_2)^{n-3}$ is precisely the canonical divisor $3 \sum_3^{n-1} h_i$, endorsing the expectation of Calabi-Yau birational models for our configuration spaces.

Remark: Triangles formed at the vertices of a polygon have been considered in some other contexts. We mention [17] and [18]. We briefly review here another realization of the configuration space, based on a simple extension of the approach used in [18] for equal areas.

Let $s_i = p_{i+1} - p_i$, $i \in Z_n$ be the edge vectors. Equations (5) can be written as:

$$\frac{1}{\delta_i} s_{i-1} \times s_i = \frac{1}{\delta_{i+1}} s_i \times s_{i+1}$$

which require the existence of scalars x_i such that:

$$s_{i-1} + \frac{\delta_i}{\delta_{i+1}} s_{i+1} + x_i s_i = 0, \quad i \in Z_n \tag{7}$$

With $\mu_i = \delta_i/\delta_{i+1}$, we define the $(n + 1) \times n$ matrix:

$$X_\mu = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 & 1 \\ x_1 & \mu_1 & 0 & \dots & 0 & 1 \\ 1 & x_2 & \mu_2 & \dots & 0 & 0 \\ 0 & 1 & x_3 & \dots & 0 & 0 \\ \cdot & \cdot & \cdot & \dots & \cdot & \cdot \\ 0 & 0 & 0 & \dots & x_{n-1} & \mu_{n-1} \\ \mu_n & 0 & 0 & \dots & 1 & x_n \end{pmatrix}$$

The first row with all entries 1 will take account of the fact that the edge vectors of the polygon have zero sum. If we denote by S the $n \times 2$ matrix with rows s_i , equations (7) take the matrix form:

$$X_\mu S = 0 \tag{8}$$

Thus, a non-degenerate configuration gives a rank two S and therefore a rank $\leq (n - 2)$ matrix X_μ . Conversely, when X_μ has rank $\leq (n - 2)$, the choice

of two independent vectors in its kernel provide a solution for (7). Generically, with rank $(n - 2)$, all solutions are affinely equivalent, hence the affine algebraic variety defined in the n -space with coordinates x_i by the condition:

$$\text{rank}(X_\mu) \leq n - 2 \tag{9}$$

gives another birational model of the configuration space of the area framework. There are some advantages from the point of view of elimination, but the projective completion offered by this realization seems entangled.

4.4 Hexagons and Elliptic Curves

We have seen above that, for generic area prescriptions, the cyclic area framework on five vertices i.e. the case of a generic pentagon, has a configuration space consisting of two points, corresponding to the degree two of the Grassmann-Plücker quadric $G(2, 4) \subset \mathbb{P}_5$.

For a cyclic area framework on a hexagon, we shall determine first the divisor at infinity. In all cases, the degenerate configurations have only three distinct vertices and are completely described by the corresponding coalescence of vertices. They are:

$$(12, 34, 56), (23, 45, 61), (1, 4, 2356), (2, 5, 3461), (3, 6, 4512)$$

Note that they must be simple points on our curve since $\text{deg}(G(2, 5)) = 5$. For the affine part $\Delta_1 \neq 0$ and we may consider the triangle 612 as ‘pinned’. Then 3 runs on a parallel (3) to line (12) at a distance prescribed by the ratio δ_2/δ_1 . Then we put $m = m(3) = (61) \cap (23)$ and with vertices 1m456 we are in the pentagonal case. Thus, our curve is presented as a ramified double covering of \mathbb{P}_1 represented by the completed affine line (3).

Expecting generically a smooth curve, it must be elliptic, since with trivial canonical class. Thus, we should have four ramification points.

Smoothness at affine points can be verified as follows: assume $\delta_1 = 1$ and pin triangle 612 with $6 = e_2, 1 = 0, 2 = e_1$. Then 3 and 5 must run on lines (3) = $\delta_2 e_2 + a e_1$, respectively (5) = $\delta_6 + b e_2$. The locus of $4 = x$ is then obtained by eliminating a and b from:

$$\Delta_k = \delta_k, \quad k = 3, 4, 5$$

This process gives:

$$a = 1 + \frac{1}{x_2}(\delta_3 - \delta_2 + \delta_2 x_1), \quad b = 1 + \frac{1}{x_1}(\delta_5 - \delta_6 + \delta_6 x_2)$$

and results in the affine cubic:

$$x_1^2 x_2 + x_1 x_2^2 - \delta_2 x_1^2 - \delta_6 x_2^2 - (1 + \delta_2 + \delta_6 - \delta_3 - \delta_4 - \delta_5) x_1 x_2 + [\delta_2(\delta_6 - \delta_5) + (\delta_2 - \delta_3)] x_1 + [\delta_6(\delta_2 - \delta_3) + (\delta_6 - \delta_5)] x_2 - (\delta_2 - \delta_3)(\delta_6 - \delta_5) = 0$$

The non-singular example $\delta_2 = \delta_6 = 0, \delta_3 = \delta_5 = 1, \delta_4 = -1$ shows that the cubic is smooth in the generic case. On the other hand, an example of singularity

at $(0, 0)$ is obtained for $\delta_2 - \delta_3 = \delta_6\delta_5 = 0$. For $\delta_2 = \dots = \delta_6 = 0$ the cubic is made of the three supporting lines of the triangle 612.

Remark: An examination of the pencil of projective cubics corresponding to $\delta_2 = \delta_6 = 0$, $\delta_3 = \delta_5 = 1$, illustrates the possibility of one or two loops for the curve of real points, when smooth. The pencil has the transposition symmetry $(x_1 x_2)$ and takes the form:

$$x_1x_2(x_1 + x_2) - x_0^2(x_1 + x_2) - x_0^3 + \lambda x_0x_1x_2 = 0$$

For $\lambda = 0$ the cubic is smooth and the real points form a single loop. For $\lambda = 1$ the cubic decomposes into a line and a conic:

$$(x_0 + x_1 + x_2)(x_1x_2 - x_0^2) = 0$$

The real points avoid the singularities and give two loops. The actual topological transition happens when passing through the nodal case corresponding to the real solution (between 0 and 1) of the equation:

$$3\lambda^3 + 4\lambda^2 + 54\lambda - 33 = 0$$

Returning to the map defined above between the configuration curve $X = X(\delta) \subset G(2, 5)$ and the projective completion of the affine cubic $\mathcal{C}_1 = \mathcal{C}_1(\delta) \subset \mathbb{P}_2$, one finds that the points ‘at infinity’ on \mathcal{C}_1 correspond with the degenerations: $(12, 34, 56)$, $(23, 45, 61)$ and $(1, 4, 2356)$. More precisely, with $(x_0 : x_1 : x_2)$ as homogeneous coordinates on \mathbb{P}_2 , the correspondence is:

$$(0 : 1 : 0) = (12, 34, 56), \quad (0 : 0 : 1) = (23, 45, 61), \quad (0 : 1 : -1) = (1, 4, 2356)$$

Furthermore, by letting 3 go to infinity on line (3), that is $a \rightarrow \infty$, and similarly for 6, we find that:

$$(1 : \delta_6 - \delta_5 : 0) = (2, 5, 3461), \quad (1 : 0 : \delta_2 - \delta_3) = (3, 6, 4512)$$

This explains the singularity observed above for $\delta_2 - \delta_3 = \delta_6 - \delta_5 = 0$ at $(1 : 0 : 0)$ as a self-intersection produced by the map $X \rightarrow \mathcal{C}_1$.

As point 3 moves on line (3), the parallels to line (23) (on which point 4 must lie) run through another intersection of the cubic with $x_2 = 0$, namely $p_{12} = (\delta_2 : \delta_2 - \delta_3 : 0)$. This implies that the involution of X defined by projecting on (3) can be interpreted on the cubic as the exchange of the two points of residual intersection of the cubic with a line running through $(\delta_2 : \delta_2 - \delta_3 : 0)$.

One can be more precise about *singular configurations* and observe that a singularity of the configuration curve must be a ramification point for all maps $X \rightarrow X/\sigma = \mathbb{P}_1$ associated to double coverings (and corresponding involutions σ) as described above for the case where \mathbb{P}_1 was the completed line (3). In fact, being a ramification point for two such maps will be enough. Looking back at the case of the pentagon, the condition for singularity amounts to the following geometric feature:

Condition (\ast_6): the parallel to edge $(i + 1, i + 2)$ through vertex $i + 5$ of the hexagon must meet the parallel to edge $(i + 3, i + 4)$ through vertex $i + 6$ on the diagonal $(i + 1, i + 4)$, $i \in Z_6$.

In agreement with Proposition 2 this condition is equivalent with the following characterization:

Proposition 3. *An affine planar configuration representing $p \in X(\delta)$ is a singular point of the curve if and only if all six lines $(i, i + 1)$ defined by the edges of the hexagon are tangent to a parabola.*

Remark: Again, the strictly affine character of this condition for singularity is in contrast with the projective invariance of singular configurations for bar-and-joint frameworks. Even so, our discussion will be cast in projective language.

Proof: An affine parabola is a projective conic with a distinguished tangent ℓ_∞ (that tangent being the line at infinity). Suppose our hexagon has edges tangent to the conic, as illustrated in Fig. 4. We want to establish condition $(*_6)$ for $i = 0$. We mark the following intersections of lines:

$$(1, 2) \cap \ell_\infty = 2'', \quad (3, 4) \cap \ell_\infty = 3''$$

Then, $12''3''456$ is also a hexagon with edges tangent to the conic. Thus, by Brianchon’s theorem, the diagonals $(1, 4), (2'', 5), (3'', 6)$ are concurrent. Affinely, this is condition $(*_6)$. Brianchon’s converse yields the other implication. \square

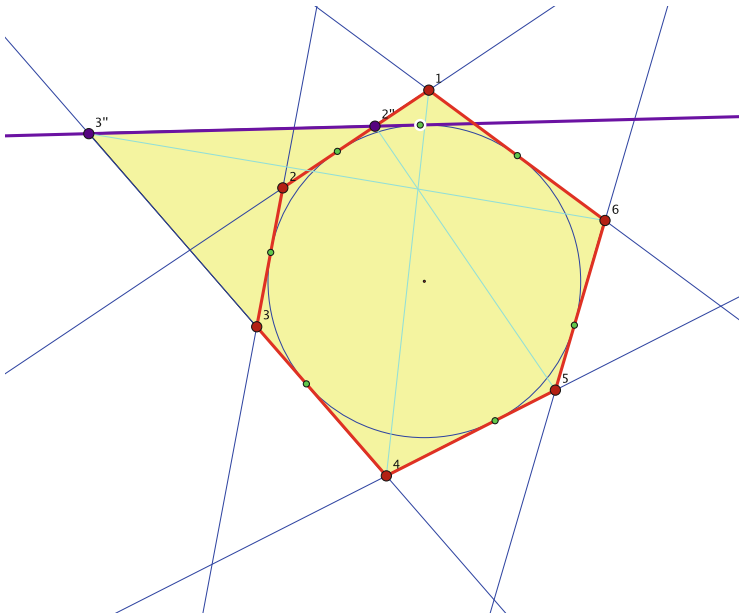


Fig. 4. Singular configuration of a hexagon. The line at infinity ℓ_∞ passes through points $2''$ and $3''$.

4.5 Heptagons and K3 Surfaces

The configuration space for a cyclic area framework on seven vertices will be a codimension six linear section of the Grassmannian $G(2, 6) \subset \mathbb{P}_{14}$, hence a surface $Y = Y(\delta) \subset G(2, 6)$. The parameter space for δ is \mathbb{P}_6 .

The divisor at infinity can be described first by tabulating the degenerations of the heptagon which make all triples of vertices $(i-1, i, i+1)$, $i \in Z_7$, collinear. Degenerations must be triangles and we have two types:

$$a_i \equiv [\bar{i}, (i+1, i+2), (i+5, i+6)](i+3, i+4)$$

$$b_i \equiv i[(i+1, i+2, i+5, i+6), i+3, i+4]$$

where vertices between parentheses (...) coincide, and vertices between brackets [...] are collinear.

Thus, there are fourteen curves of degenerations and they must be all lines since the degree of the Grassmannian $G(2, 6)$ is precisely fourteen. We shall retain the notation a_i, b_i , $i \in Z_7$ for the lines themselves, or for the algebraic cycles they define on the surface Y .

Their pattern of intersection may be summarized as follows:

$$a_i b_i = 1, \quad a_i a_{i-2} = a_i a_{i+2} = 1, \quad b_i b_{i-3} = b_i b_{i+3} = 1$$

and all remaining pairs of lines are disjoint.

5 Conclusion

In this paper we considered deformation varieties for volume frameworks in \mathbb{R}^d associated to cyclic hypergraphs on n labeled vertices. Natural parametrizations were obtained as linear sections of Grassmannians. The resulting (d, n) indexed family is invariant under association. Particular attention was given to the planar case $d = 2$, with geometric characterizations of singular configurations. The affine nature of these singularities contrasts with the projective invariance of infinitesimal rigidity and infinitesimal flexes for bar-and-joint frameworks.

Acknowledgement. This research was supported by the NSF grants CCF-1016988 and CCF-1319366.

References

1. Asimov, L., Roth, B.: The rigidity of graphs. *Trans. Amer. Math. Soc.* **245**, 279–289 (1978)
2. Bobenko, A., Kenyon, R., Sullivan, J.M., Ziegler, G.: *Discrete differential geometry*, Tech. Rep. 12/2006, Mathematisches Forschungsinstitut Oberwolfach (2006)
3. Borcea, C.S.: Point configurations and Cayley-Menger varieties. arXiv: math.AG/0207110

4. Borcea, C.S.: Association for flag configurations, in *Commutative Algebra, Singularities and Computer Algebra*, NATO Science Series, vol. 115, p. 1–8. Kluwer (2003)
5. Borcea, C.S.: Polygon spaces, tangents to quadrics and special Lagrangians, *Oberwolfach Reports*, pp. 2181–2183. European Mathematical Society (2004)
6. Borcea, C.S., Streinu, I.: The number of embeddings of minimally rigid graphs. *Discrete Comput. Geom.* **31**, 287–303 (2004)
7. Borcea, C.S., Streinu, I.: Realizations of volume frameworks. In: Ida, T., Fleuriot, J. (eds.) *ADG 2012*. LNCS, vol. 7993, pp. 110–119. Springer, Heidelberg (2013)
8. Borcea, C.S., Streinu, I.: Periodic frameworks and flexibility. *Proc. Roy. Soc. A* **466**, 2633–2649 (2010)
9. Borcea, C.S., Streinu, I.: Minimally rigid periodic graphs. *Bull. Lond. Math. Soc.* **43**, 1093–1103 (2011). doi:[10.1112/blms/bdr044](https://doi.org/10.1112/blms/bdr044)
10. Borcea, C.S., Streinu, I.: Frameworks with crystallographic symmetry. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **372**, 1317–1321 (2014). doi:[10.1098/rsta.2012.0143](https://doi.org/10.1098/rsta.2012.0143)
11. Coble, A.B.: *Algebraic Geometry and Theta Functions*. AMS Colloquium Publications, New York (1929)
12. Cox, D.A., Katz, S.: Mirror symmetry and algebraic geometry. *Math. Surv. Monogr.* **68**, 469 (1999). Amer. Math. Soc
13. Coxeter, H.S.M.: Self-dual configurations and regular graphs. *Bull. Am. Math. Soc.* **56**, 413–455 (1950)
14. Dolgachev, I., Ortland, D.: Point sets in projective spaces and theta functions. *Astérisque* **165**, 210 (1988)
15. Fischer, J.C., Jamison, R.E.: Properties of affinely regular polygons. *Geom. Dedicata.* **69**, 241–259 (1998)
16. Griffiths, Ph, Harris, J.: *Principles of Algebraic Geometry*. Wiley Classics Library, New York (1994)
17. Gronchi, P., Longinetti, M.: Affinely regular polygons as extremals of area functionals. *Discrete Comput. Geom.* **39**, 273–297 (2008)
18. Harel, G., Rabin, J.M.: Polygons whose vertex triangles have equal area. *Amer. Math. Monthly* **110**(7), 606–619 (2003)
19. Harris, J.: *Algebraic Geometry*, Graduate Texts in Mathematics 133. Springer-Verlag, New York (1993)
20. Hazama, F.: On the moduli space of polygons with area center, [arXiv:1310.0523](https://arxiv.org/abs/1310.0523)
21. Hilbert, D., Cohn-Vossen, S.: *Geometry and the Imagination*. Chelsea Publishing Company, New York (1952)
22. Kantor, S.: Die configurationen $(3, 3)_{10}$, sitzungsberichte der mathematisch-naturwissenschaftliche classe der kaiserlichen akademie der wissenschaften. *Wien* **84**, 1291–1314 (1882)
23. Kapovich, M., Millson, J.: On the moduli spaces of polygons in the Euclidean plane. *J. Differ. Geom.* **42**(1), 133–164 (1995)
24. Kapovich, M., Millson, J.: The symplectic geometry of polygons in the Euclidean space. *J. Differ. Geom.* **44**, 479–513 (1996)
25. Kapovich, M., Millson, J.: Hodge theory and the art of paper folding. *Publi. Res. Inst. Math. Sci.* **33**(1), 1–31 (1997)
26. Kapovich, M., Millson, J.: On the moduli space of a spherical polygonal linkage. *Canad. Math. Bull.* **42**(3), 307–320 (1999)
27. Kapovich, M., Millson, J., Treloar, T.: The symplectic geometry of polygons in hyperbolic 3-space. *Asian J. Math* **4**(1), 123–164 (2000). Kodaira’s 75-th birthday volume

28. Laman, G.: On graphs and rigidity of plane skeletal structures. *J. Eng. Math.* **4**, 331–340 (1970)
29. Maxwell, J.C.: On reciprocal figures. *Frames Diagrams Forces, Trans. R. Soc. Edinb.* **26**, 1–40 (1870)
30. Sitharam, M., Gao, H.: Characterizing graphs with convex and connected cayley configuration spaces. *Discrete Comput. Geom.* **43**(3), 594–625 (2010)
31. Weyl, H.: *The Classical Groups: their Invariants and Representations*. Princeton University Press, Princeton Landmarks in Math (1997)

Recent Advances in Real Geometric Reasoning

James H. Davenport¹ and Matthew England²(✉)

¹ Departments of Computer Science and Mathematical Sciences,
University of Bath, Bath, UK
J.H.Davenport@bath.ac.uk

² Department of Computing, Coventry University, Coventry, UK
Matthew.England@coventry.ac.uk

Abstract. In the 1930s Tarski showed that real quantifier elimination was possible, and in 1975 Collins gave a remotely practicable method, albeit with doubly-exponential complexity, which was later shown to be inherent. We discuss some of the recent major advances in Collins method: such as an alternative approach based on passing via the complexes, and advances which come closer to “solving the question asked” rather than “solving all problems to do with these polynomials”.

1 Introduction

Although methods with better asymptotic complexity are known in theory (e.g. [GV88]), the workhorse of implemented algorithms for real geometric reasoning is Cylindrical Algebraic Decomposition. This was introduced in [Col75] to produce a remotely practicable (complexity “merely” doubly exponential in the number of variables) alternative to Tarski’s original method from 1930 [Tar51], whose complexity could not be bounded by any tower of exponentials. Tarski in fact set out to solve the *quantifier elimination problem* for real algebraic geometry (Sect. 4): given $Q_{k+1}x_{k+1}Q_{k+2}x_{k+2}\dots\Phi(x_1,\dots,x_n)$, where $Q_i \in \{\forall, \exists\}$ and Φ is a Boolean combination of relations involving polynomials $p_i(x_1, \dots, x_n)$, find an equivalent $\Psi(x_1, \dots, x_k)$, where Ψ is a Boolean combination of relations involving polynomials $q_i(x_1, \dots, x_k)$. In fact, we cannot solve this in the language of algebraic geometry: we need *semi-algebraic* geometry, allowing $>$ as well¹ as $=$. The *necessity* of $>$ follows from the fact of the example $\exists y : x = y^2 \Leftrightarrow (x > 0) \vee (x = 0)$; its *sufficiency* is the point of Tarski’s work.

2 Cylindrical Algebraic Decomposition by Projection and Lifting

[Col75] constructs a sampled² *Cylindrical Algebraic Decomposition* (CAD) of \mathbf{R}^n which is *sign-invariant* for the p_i , where these words are defined as follows.

¹ Strictly speaking $>$ is sufficient, but implementations always allow \geq and \neq . In fact, \neq is intrinsic to the regular chains approach discussed in Sect. 3.

² The “sampled” nature is implicit in [Col75, and its successors], but the authors find it helpful to be explicit about this.

Definition 1. (CAD terminology). Note that throughout we are ordering our coordinates/variables, so that x_n is the “last coordinate”.

decomposition: a partition of \mathbf{R}^n into cells $C_{\mathbf{i}}$ indexed by n -tuples of natural numbers (so $\mathbf{R}^n = \bigcup_{\mathbf{i}} C_{\mathbf{i}}$ and $\mathbf{i} \neq \mathbf{j} \Rightarrow C_{\mathbf{i}} \cap C_{\mathbf{j}} = \emptyset$);

(semi-)algebraic: every $C_{\mathbf{i}}$ is defined by a finite set of equalities and inequalities of polynomials in the x_i , including expressions of the form

$$\text{RootOf}_2(f_1(x_1, y)) < x_2 < \text{RootOf}_3(f_2(x_1, y)) \quad (1)$$

(where RootOf_2 means “the second real root, counting from $-\infty$ ”);

cylindrical: for all $k < n$, if π_k is the projection onto the first k coordinates, then, for all \mathbf{i}, \mathbf{j} , $\pi_k(C_{\mathbf{i}})$ and $\pi_k(C_{\mathbf{j}})$ are either equal or disjoint;

sampled: for each cell $C_{\mathbf{i}}$ there is an explicit point $s_{\mathbf{i}} \in C_{\mathbf{i}}$;

sign-invariant: for the polynomials in Φ on each cell, every p_i is identically zero, or everywhere positive, or everywhere negative.

Collins constructed such a decomposition by a process now known (at least by our colleagues) as *CAD by Projection and Lifting* (for more details see [Dav15]). The key property in this approach is the following.

Definition 2. A polynomial $p(x_1, \dots, x_m)$ is delineable³ over a region $C \subset \mathbf{R}^{m-1}$ if:

1. the portion of the real variety of p that lies in the cylinder $C \times \mathbf{R}$ over C consists of the union of the graphs (called sections) of some $k \geq 0$ continuous functions $\theta_1 < \dots < \theta_k$ from C to \mathbf{R} and;
2. there exist integers $m_1, \dots, m_k \geq 1$ such that, for every point (a_1, \dots, a_{m-1}) in C , the multiplicity of the root $\theta_i(a_1, \dots, a_{m-1})$ of $p(a_1, \dots, a_{m-1}, x_m)$, considered as a function of x_m alone, is m_i (and in particular is constant).

A set of polynomials is delineable over C if each is delineable **and** if the sections are either identical or disjoint. This is actually equivalent to saying that the product is delineable.

Intuitively, if the $\{p_i\}$ are delineable over C , their graphs neither fold nor cross.

Let \mathcal{P}_n be the set of polynomials in Φ , with coefficients from some effective⁴ field $K \subset \mathbf{R}$. Then Collins algorithm proceeds as follows:

³ There are various, subtly different, definitions in the literature. This one is from [McC99].

⁴ The literature often stipulates \mathbf{Q} or the algebraic numbers \mathbf{A} . The real requirement is that we can perform all the polynomial algebra we need over K , and that, given expressions $a, b \in K$, we can decide the trichotomy $a < b$ or $a = b$ or $a > b$. Once we start adding transcendental functions to our language, the effectivity of K becomes a major problem, as we run across the usual undecidability results. This is addressed in different ways in [AMW08] and [Vor89, Vor92].

Projection: Given some $\mathcal{P}_k \subset K[x_1, \dots, x_k]$ construct a set $\mathcal{P}_{k-1} \subset K[x_1, \dots, x_{k-1}]$ such that, over each cell of a CAD sign-invariant for \mathcal{P}_{k-1} , the polynomials of \mathcal{P}_k are delineable. Though the details depend on the algorithm, the key ingredients are leading coefficients (where these vanish some θ_i tends to infinity), discriminants (where these vanish some θ_i ceases to have constant multiplicity) and resultants (where these vanish, the θ_i from different polynomials intersect).

Repeat until we have the set of univariate polynomials \mathcal{P}_1 .

Base case: Given \mathcal{P}_1 , isolate the N_1 real roots of these polynomials in \mathbf{R}^1 , and construct a CAD consisting of the N_1 roots, and the $N_1 + 1$ intervals between them (or to the left/right of them all). The sample points for the 0-dimensional cells are the roots themselves: for the 1-dimensional intervals we choose any convenient point, generally rational and with denominator the smallest power of 2 we can find.

Lifting: Given a CAD D_{k-1} of \mathbf{R}^{k-1} , sign-invariant for \mathcal{P}_{k-1} , construct a CAD D_k of \mathbf{R}^k , sign-invariant for \mathcal{P}_k . For each cell C_i , this is done by substituting the sample point s_i into \mathcal{P}_k , and doing the equivalent of the base case for the resulting univariate system (valid across the whole of C_i if the projection operator provides delineable projection polynomials).

Repeat until we have the CAD D_n of \mathbf{R}^n .

If we suppose that \mathcal{P}_n contains m polynomials, of degree (in each variable) bounded by d , and coefficient length bounded by l (coefficients bounded by 2^l), then the time complexity is bounded [Col75, Theorem 16] by

$$O\left(m^{2^{n+6}} (2d)^{2^{2n+8}} l^3\right). \quad (2)$$

This analysis is very sensitive to the details of the sub-algorithms involved, and a more refined analysis of the base case [Dav85] reduces the complexity (though not the actual running time) to

$$O\left(m^{2^{n+4}} (2d)^{2^{2n+6}} l^3\right).$$

This improvement might seem trivial, but in fact implies taking the fourth root of the m, d part of the complexity.

A less sensitive property (and one that reflects the cost of *using* such a decomposition) is the number of cells: for the Collins method this is bounded, by an analysis similar to [BDE+14], by

$$O\left(m^{2^n} (2d)^{2 \cdot 3^n}\right). \quad (3)$$

As is often the case in mathematics, we get more insight if we solve an apparently harder problem. [McC84] did this, demanding that the decompositions D_k , $k < n$ be, not just sign-invariant, but

Order-invariant for the polynomials in Φ , i.e. on each cell, every p_i is identically zero, and vanishes to the same order throughout the cell, or everywhere positive, or everywhere negative.

This actually lets his \mathcal{P}_k be much simpler than Collins', with the cost that the lifting procedure might fail if some element p of \mathcal{P}_k *nullifies* (is identically zero) over some cell in D_{k-1} . In this case, McCallum says that \mathcal{P}_k was not *well-oriented*, and has to either:

1. proceed by working around the problem or concluding it not relevant. This is only possible in certain cases (e.g. the cell is dimension 0) [Bro05]. Otherwise;
2. revert to Collins' projection (or a variant due to [Hon90]); or,
3. add the partial derivatives of p to \mathcal{P}_k and resume the projection process from there — an operation that to the best of the authors' knowledge has never been implemented, doubtless because of the complicated backtracking involved, and the fact that, whereas we only *ought* to add this polynomial in the nullifying region, the design of Collins' algorithm and its successors assume a global set of polynomials at each level.

“Randomly”, well-orientedness ought to occur with probability 1, but we have a family of “real-world” examples (simplification/branch cuts, see [BBDP07]) where it often fails. The analogy of (3) is given by [McC85, Theorem 6.1.5] as

$$O\left(m^{2^n}(2d)^{n \cdot 2^n}\right), \quad (4)$$

and a recent improved analysis in [BDE+14, (12)] reduces this to

$$O\left(2^{2^{n-1}}m(m+1)^{2^n-2}d^{2^n-1}\right). \quad (5)$$

3 CAD by Regular Chains

This alternative to the traditional computation scheme of projection and lifting was introduced in [CMXY09], then improved in [CM14a]. The method can be described as “going via the complexes”, since the authors first construct a cylindrical decomposition of \mathbf{C}^n , and then infer a CAD of \mathbf{R}^n . They make use of the well developed body of theory around regular systems [Wan00] for the work over the complexes, and the algorithms are all implemented in the `RegularChains Library`⁵ for MAPLE, hence our designation: *CAD by Regular Chains*.

We first need the following analogue of Definition 2 (not precisely analogous, as Definition 2 allows for non-square-free polynomials and this does not).

Definition 3. *Let $K \subset \mathbf{C}$ be an effective field. Let C be a subset of \mathbf{C}^{n-1} and $P \subset K[x_1, \dots, x_{n-1}, x_n]$ be a finite set of polynomials whose main variable really is x_n . We say that P separates above C if for each $\alpha \in C$:*

1. for each $p \in P$, the polynomial $lc_{x_n}(p)$ does not vanish at α ;
2. the polynomials $p(\alpha, x_n) \in \mathbf{C}[x_n]$, for all $p \in P$, are squarefree and coprime.

Note that the empty set is trivially separable.

⁵ <http://www.regularchains.org>.

We then need an analogue of Definition 1 for the case of complex space. We follow [CM14a] and describe these (complex) cylindrical decompositions in terms of the tree data structure they are stored as.

Definition 4. We define a cylindrical decomposition of \mathbf{C}^n , and its associated tree, by induction on n .

Base Either: There is one set D_1 , the whole of \mathbf{C} and $\mathcal{D} = \{D_1\}$;

Base Or: there are r non-constant square-free relatively prime polynomials p_i such that D_i is the set of zeros of p_i , and D_{r+1} is the complement: $\{x : p_1(x)p_2(x) \cdots p_r(x) \neq 0\}$: $\mathcal{D} = \{D_1, \dots, D_r, D_{r+1}\}$.

Base Tree: The root and all the D_i as leaves of it.

Induction: Let \mathcal{D}' be a cylindrical decomposition of \mathbf{C}^{n-1} . For each $D_i \in \mathcal{D}'$, let r_i be a non-negative integer, and $P_i = \{p_{i,1}, \dots, p_{i,r_i}\}$ be a set of polynomials which separates over D_i .

Induction Either: $r = 0$ and we set $D_{i,1} = D_i \times \mathbf{C}$;

Induction Or: we set $D_{i,j} = \{(\alpha, x) : \alpha \in D_i \wedge p_{i,j}(\alpha, x) = 0\}$;

$$D_{i,r+1} = \left\{ (\alpha, x) : \alpha \in D_i \wedge \prod_j p_{i,j}(\alpha, x) \neq 0 \right\};$$

Then: a cylindrical decomposition of \mathbf{C}^n is given by

$$\mathcal{D} = \{D_{i,j} : 1 \leq i \leq |\mathcal{D}'|; 1 \leq j \leq r_i + 1\}.$$

Induction Tree: If T' is the tree associated to \mathcal{D}' then the tree associated to \mathcal{D} is obtained by adding to each leaf $D_i \in T'$ as children all the $D_{i,j}$ such that $1 \leq j \leq r_i + 1$.

Unlike Definition 1, the different roots of a given polynomial are not separated. Each cell is the zero set of a system of polynomial equations and inequations, where the main variables are all distinct: a triangular system [ALM99].

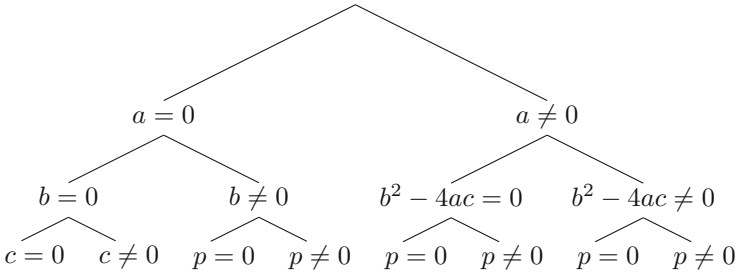
Definition 5. Let F be a set of polynomials in $k = K[x_1, \dots, x_n]$. A cylindrical decomposition \mathcal{D} is F -invariant if, for each cell $D \in \mathcal{D}$ and each $f_i \in F$, either f vanishes at all points of D or f vanishes at no point of D .

The trivial decomposition, obtained by taking the “either” branch each time, with one cell, is \emptyset -invariant. Given a cylindrical decomposition \mathcal{D} which is F -invariant, and supposing $\widehat{F} = F \cup \{f\}$, [CM14a] shows how to refine \mathcal{D} to a cylindrical decomposition $\widehat{\mathcal{D}}$ which is \widehat{F} -invariant, hence the “incremental” in the title of their paper. The key ingredients in this process are again leading coefficients, resultants and discriminants. The paper [CMXY09] shows, assuming that $K \subset \mathbf{R}$, how to construct from \mathcal{D} a cylindrical algebraic decomposition of \mathbf{R}^n which is sign-invariant for F .

The construction of the cylindrical decomposition can be seen, as pointed out in [CM14a], as an analogue of the projection phase of projection and lifting. Indeed, if n is small, it is often the case that the polynomials at level i in the tree corresponding to \mathcal{D} are those in \mathcal{P}_{n-i} . The fundamental difference is that the \mathcal{P}_i are *global* structures: over the whole cylindrical algebraic decomposition of \mathbf{R}^k we need to isolate all the branches of all of \mathcal{P}_{k+1} , whereas there is a tree structure underpinning \mathcal{D} and the cylindrical algebraic decomposition, which means that “polynomials are not considered when they are blatantly not relevant”.

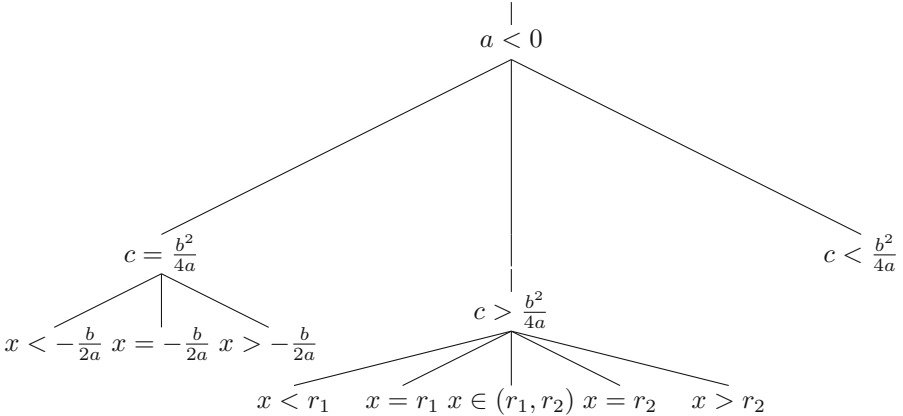
Example: Consider the parabola $p := ax^2 + bx + c$ and assume the variable ordering $x \succ c \succ b \succ a$. Suppose we were to use projection and lifting. Then the first projection identifies the coefficients a, b, c and the discriminant with respect to x : $b^2 - 4ac$. Subsequent projection do not identify any further projection polynomials for this example. Lifting produces CADs sign-invariant for these 4 projection polynomials, as well as p itself.

The regular chains approach would start by building the following tree, representing a cylindrical decomposition of \mathbf{C}^n :



This decomposition was produced to be sign-invariant for p . However, it does not insist on sign-invariance for all the other projection polynomials. In particular, it is not sign-invariant for b . The polynomial b is included in the projection set because its vanishing can determine delineability, but only when the coefficient of the higher degree terms vanish. So, when $a = 0$ it is important to ensure b is sign-invariant, but not otherwise. Hence the tree above is doing only what is necessary to make the final conclusion about p .

The next step is to apply real root isolation, extending this tree to one representing a CAD. At the top level the case $a \neq 0$ must split into the two possibilities: $a < 0$ and $a > 0$. For brevity we display only the branch for $a < 0$ below (where r_1 and r_2 represent the two real roots of p in the case where the leading coefficient is negative and the discriminant positive). The full tree has 27 leaves, thus representing a CAD with 27 cells. This compares with a minimal CAD of 115 cells produced by projection and lifting to be sign invariant for all projection polynomials.



Are there significant savings in general? We refer the reader to [BDE+14, Table 1]. Here PL-CAD refers to our implementation of McCallum’s algorithm of Sect. 2; RC-INC-CAD refers to the algorithm of [CM14a] (Sect. 3); and QEPCAD [Bro03] is another, highly optimised, implementation of McCallum’s algorithm. Where both terminate, QEPCAD and PL-CAD often, though not always, have the same cell count. RC-INC-CAD does sometimes have the same count, but on other examples such as BC-Phisanbut-4, needs only 2007 cells, while both implementations of McCallum’s algorithm need 51,763.

4 Quantifier Elimination

The original motivation for [Col75] was the following problem.

Problem 1. (Quantifier Elimination). Let $Q_i \in \{\exists, \forall\}$, and \mathcal{L}_{RCF} be the language of Boolean-connected equalities and inequalities concerning polynomials in $K[x_1, \dots, x_n]$, where K is an effective field with $\mathbf{Q} \subseteq K \subset \mathbf{R}$. Given a statement (known as a Tarski statement, or a Tarski sentence if $k = 0$)

$$\Phi := Q_{k+1}x_{k+1} \dots Q_n x_n \phi(x_1, \dots, x_n) : \quad \phi \in \mathcal{L}_{\text{RCF}}, \tag{6}$$

the *Quantifier Elimination* problem is that of producing an equivalent

$$\Psi := \psi(x_1, \dots, x_k) : \quad \psi \in \mathcal{L}_{\text{RCF}}. \tag{7}$$

In particular, $k = 0$ is a decision problem: is Φ true?

If we have a CAD $\mathcal{D}^{(n)}$ of \mathbf{R}^n (noting that the x_i must be ordered in the same way in Definition 1 and formula (6)) sign-invariant for the polynomials of Φ , then constructing Ψ is conceptually easy.

1. The truth of ϕ in a cell D_i of $\mathcal{D}^{(n)}$ is that of ϕ at the sample point s_i .
2. $\mathcal{D}^{(n)}$ projects to a CAD $\mathcal{D}^{(k)}$ of \mathbf{R}^k .

3. The truth of Φ in a cell $\widehat{D}_j \in \mathcal{D}^{(k)}$ is then the appropriate (\forall for \exists etc.) Boolean combination of the truth of ϕ in the cells of \mathcal{D} that project to \widehat{D}_j .
4. Ψ is then the disjunction of the defining formulae for all the \widehat{D}_j for which Φ is true.

There is a problem in practice with the last step, first pointed out in [Bro99]. In the lifting stage, we produce branches θ_i of polynomials, with descriptions such as “that branch of $p(x_1, \dots, x_l)$ which, above the sample point $s = (\alpha_1, \dots, \alpha_{l-1})$, has the (unique) root in (β, γ) ”, and this is not a statement of \mathcal{L}_{RCF} . We could equally describe it as “the third real branch of $p(x_1, \dots, x_l)$ above s ”, but again this statement is not in \mathcal{L}_{RCF} . Now by Thom’s Lemma [CR88], we can describe this branch in terms of the signs of p and its derivatives, but, whereas these derivatives are in the Collins projection, they are not in the McCallum projection, or in the tree constructed by the method of Sect. 3. However, when it comes to describing \widehat{D}_j , we can just add these (as described in [Bro99] for projection and lifting and in [CM14b] for regular chains CAD construction). The additional cost is negligible, in particular, we do not need them for projection (Sect. 2), or for tree construction (Sect. 3).

Though it may depend non-linearly on polynomial degree etc., this process is linear in the number of cells in $\mathcal{D}^{(n)}$, and produces a disjunction of at most as many clauses as there are cells in $\mathcal{D}^{(k)}$.

5 Lower Bounds

This last remark is the basis of the complexity lower bounds in [DH88, BD07]. Both constructions use the fact that

$$\exists z_m \forall x_{m-1} \forall y_{m-1} \left(\begin{array}{l} (y_{m-1} = y_m \wedge x_{m-1} = z_m) \\ \vee (y_{m-1} = z_m \wedge x_{m-1} = x_m) \\ \Rightarrow y_{m-1} = F_{m-1}(x_{m-1}) \end{array} \right) \quad (8)$$

encodes $y_m = F_{m-1}(F_{m-1}(x_m))$. Hence applying this construct $m - 1$ times to $y_1 = F_1(x_1)$ gives

$$y_m = \underbrace{F_1(F_1(\dots F_1((x_n))\dots))}_{2^{m-1} \text{ times}}$$

This can then be used to produce expressions with n quantifiers and having $2^{2^{O(n)}}$ isolated point solutions, hence needing $2^{2^{O(n)}}$ cells to describe them (the $O(n)$ terms are $n/3 + O(1)$ in [BD07] and $n/5 + O(1)$ in [DH88]). An example which needs $2^{2^{O(n)}}$ cells for all possible variable orders is also produced in [BD07], along with another which needs $2^{2^{O(n)}}$ cells in one order, but a constant number in another. Hence the great interest in variable order selection methods for CAD [DSS04, EBDW14, HEW+14] to name a few.

The construction in (8) uses both \exists and \forall in a way that cannot be unnested. In fact, it is possible [Gri88] to decide Tarski sentences (i.e. no free variables) with a cost that is singly-exponential in n , but doubly-exponential in a , the number of

alternations of \exists and \forall in (6). These methods, or any methods singly-exponential in n , have, in general, not been implemented, though there has been work on the purely existential case (for example [Hun08]).

6 Equational Constraints

The methods described in the previous sections produce decompositions which are sign- (or order-)invariant for a set of polynomials. In particular, we can apply steps 1–4 of Sect. 4 to the same CAD to solve (6) for any other ϕ involving the same polynomials. Indeed, as long as the x_i stayed in the same order, we could change the Q_i as well. [Col98] suggested that we could do better if ϕ was of the form $p_1 = 0 \wedge \phi'$, as we would not be interested in the behaviour of polynomials in ϕ' except when $p_1 = 0$. This was implemented in [McC99], who produced a CAD which was sign-invariant for p_1 , and **when** $p_1 = 0$, sign-invariant for the polynomials in ϕ' . The main effect of this is to reduce the double exponent n of m in (5) by 1, i.e. to take the square root of this term, as shown recently in [BDE+14] (14).

It is worth seeing how this works. Consider

$$\phi := (f_1 = 0) \wedge ((f_2 > 0) \vee (f_3 > 0)). \quad (9)$$

Then a [McC84]-style projection ignoring the fact that there is an equation constraint would contain⁶ three $\text{disc}(f_i)$ and three $\text{res}(f_i, f_j)$. However, [McC99] observes that we are not interested in f_2, f_3 except when $f_1 = 0$, and hence we need only consider $\text{disc}(f_1)$ and $\text{res}(f_1, f_2), \text{res}(f_1, f_3)$, half as many polynomials.

Consider now

$$\phi_1 := ((g_1 = 0) \wedge (g_2 > 0)) \vee ((g_3 > 0) \wedge (g_4 = 0)). \quad (10)$$

A [McC84]-style projection ignoring the fact that there is an equation constraint would contain four discriminants and six resultants. Although (10) does not contain an *overt* equational constraint, $\phi_1 \Rightarrow (g_1 = 0) \vee (g_4 = 0)$, which is $\phi_1 \Rightarrow (g_1 g_4 = 0)$, and so the equational constraint $g_1 g_4 = 0$ is implicit. If we study $g_1 g_4 = 0 \wedge \phi_1$ in the style of (9), and drop trivial resultants, we consider $\text{disc}(g_1 g_4)$, $\text{res}(g_1 g_4, g_2)$ and $\text{res}(g_1 g_4, g_3)$. Using the multiplicative properties of resultants and discriminants (which we would certainly do in practice!), this is $\text{disc}(g_1)$, $\text{disc}(g_4)$ and all the resultants except $\text{res}(g_2, g_3)$, i.e. two discriminants and five resultants.

Intuitively $\text{res}(g_1, g_3)$ and $\text{res}(g_2, g_4)$ are redundant, but how do we achieve this in general? This was solved in [BDE+13], where, rather than producing a sign-invariant CAD, we compute *truth table invariant* (a TTICAD) for the two propositions $(g_1 = 0) \wedge (g_2 > 0)$ and $(g_3 > 0) \wedge (g_4 = 0)$, i.e. on each cell, each of these two propositions is either identically true, or identically false. This process does indeed remove these two resultants, so we have two discriminants and three resultants.

⁶ It would also have some leading coefficients etc., but these are not the main drivers of the complexity in McCallum's projection.

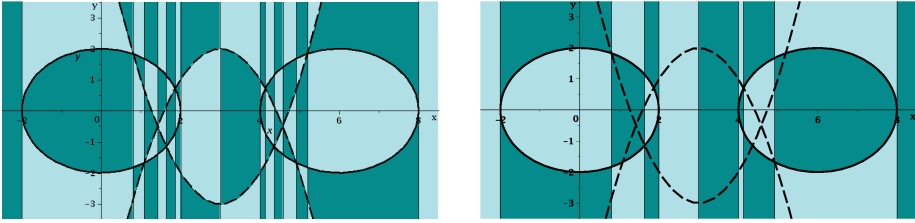


Fig. 1. The left is a sign-invariant CAD, and the right a TTICAD, for (10) with the polynomials from the Example.

Example: Consider (10) with

$$g_1 := x^2 + y^2 - 4, \quad g_2 := (x - 3)^2 - (y + 3),$$

and

$$g_3 := (x - 3)^2 + (y - 2), \quad g_4 := (x - 6)^2 + y^2 - 4.$$

Fig. 1 shows the two dimensional cells produced for both a sign-invariant CAD and a truth-table invariant CAD, built under ordering $x < y$. The sign-invariant CAD has 231 cells (72 full-dimensional but the splitting of the final cylinder is out of view) and the TTICAD 67 (22 full-dimensional).

By comparing the figures we see two types of differences. First, the CAD of the real line is split into fewer cells (there are not as many cylinders in \mathbf{R}^2). This is the effect of the reduction in projection polynomials identified, (less univariate polynomials with real roots to isolate). The second difference is that the full-dimensional cylinders are no longer split over the dashed lines. This came from an improvement in the lifting phase (discussed in detail in [BDE+14]). It leveraged the projection theory to conclude that we usually only need to lift with respect to equational constraints themselves.

More recently [BDE+14] truth-table invariance has been achieved even when there is no implicit equational constraint, as with an example of the form

$$((h_1 = 0) \wedge (h_2 > 0)) \vee (h_3 > 0). \quad (11)$$

The savings that can be achieved depend on the number of equational constraints involved in sub-clauses of the parent formula.

It is also possible to apply equational constraints in the regular chains technology view of CAD [CM14a], again even when there is no global equational constraint, as in (11) [BCD+14].

7 How Reliable Is This?

Cylindrical algebraic decomposition can be used as tool in program verification, as in the MetiTarski tool [Pau12]. This leads to the question: who will verify the CAD, or at least the inferences we draw from it? We note that a positive answer

to a purely existential question (equally, a negative answer to a purely universal question) is easily verified since we have a witness. The converse questions are essentially questions of refutation, see [JdM12]. Questions involving a mixture of quantifiers are much harder.

Almost all current implementations of CAD are based on computer algebra systems, which are generally unverified. We can at least compare, on a fairly level playing field, the implementations in MAPLE of four algorithms: see Table 1. The classification of the amount of mathematics involved is subjective, but we note that [McC84], and hence [BDEW13], relies on [Zar65, Zar75] to justify the smaller projection set compared with [Col75]. [CM14a] and [BCD+14] rely on, *inter alia*, [ALM99].

Table 1. Comparison of algorithms

Algorithm	Implementation	Code Lines (above Maple)	Specialist Mathematics
[Col75]	[EWBD14]	2600	some
[McC84]	[EWBD14]	2500	a lot
[CM14a]	[CM14a]	5000	medium
[BDEW13]	[EWBD14]	3000	a lot
[BCD+14]	[BCD+14]	5500	medium

There are two challenges involved in verifying a CAD algorithm.

1. There is a “program verification” question of ensuring that the algorithms produce the result that they say they will, i.e. that resultants, discriminants, real roots etc. are computed correctly. This is non-trivial, to say the least, sitting on top of an unverified computer algebra system, but should be feasible for an implementation based on a sound kernel, such as Coq or Isabelle.
 2. There is a “mathematics verification” question whether the resulting decomposition is truly sign-/order-/truth table-invariant for the inputs. This is where the column labelled “Mathematics” in Table 1 comes in. The only attempt to produce verified CADs known to the authors, [CM12, in Coq], is based, not on [Col75] and its successors, but rather on [BPR06, chapter 2], itself essentially that of [Tar51].
- 2a. There is an interesting tension here between “precomputed” and *ad hoc* verification. An implementation based in [McC84] would essentially have to verify the relevant theorems from [Zar65, Zar75], but these could be imported as pre-verified lemmas. An implementation based on [CM14a] would verify that *in this case* we had an appropriate cylindrical decomposition of \mathbf{C}^n which *in this case* translated to an appropriate cylindrical algebraic decomposition of \mathbf{R}^n .

8 Final Thoughts

The topics we focussed on in this paper are implemented in MAPLE:

- CAD by Regular Chains is implemented in the REGULARCHAINS Library. A version of this ships with the core MAPLE distribution while the latest version is freely available from <http://www.regularchains.org/>.
- The authors' own work (equational constraints, truth-table invariance, sub-decompositions) is freely available in a MAPLE package PROJECTIONCAD. The latest version is available from: <http://opus.bath.ac.uk/43911/>.

Other implementations of cylindrical algebraic decomposition include:

- MATHEMATICA [Str06]; The commands `CylindricalDecomposition` and `Reduce` can make use of an underlying CAD implementation. These commands can be exceptionally fast but it can be hard to judge the CAD components individually as they are just one of several underlying methods available and the output is in the form of formulae rather than cells.
- QEPCAD [Bro03]; a dedicated interactive command-line program available from <http://www.usna.edu/CS/qepcadweb/B/QEPCAD.html>. One notable feature is the SLFQ program which can simplify large quantifier free formulae giving more readable output. SAGE now has a QEPCAD interface.
- REDLOG [SS03]; this REDUCE package implements CAD along with other quantifier elimination methods such as virtual substitution.
- SYNRAAC [IYAY13]; a MAPLE package notable for its symbolic-numeric approach. An older version is available for free download from: <http://jp.fujitsu.com/group/labs/en/techinfo/freeware/synrac/> with more recent advances part of the wider Todai Robot project.

The only reported experiments to cover all of these implementations were detailed in Sect. 4 of [BCD+14].

Of course, this paper surveyed only a few of the recent advances in cylindrical algebraic decomposition. Others include (but are not limited to):

- The use of certified numerics in the lifting phase to minimise the amount of symbolic computation required [Str06, IYAY13].
- Local projection schemes [Str14], generic projection schemes [SS03] and single CAD cells [Bro13, JdM12].
- Problem formulation for CAD [DSS04, BDEW13, WEDB14] (projection and lifting) [EBDW14, EBC+14] (regular chains). These all develop heuristics to help with choices, while [HEW+14] applies machine learning in the form of support vector machines to pick a heuristic.
- Work on cylindrical algebraic sub-decompositions, which return only a subset of the cells in a full CAD [Sei06]. In [WBDE14] algorithms are given to return cells that lie on a prescribed variety, or have a designated dimension, while in [WDEB13] these techniques are combined to solve a motion planning problem. Note that if restricting to cells of full dimension then sample points can always be chosen to be rational, greatly reducing running time.

There are numerous unsolved problems, both theoretical and practical. Three that stand out to the authors are the following.

1. There is no complexity analysis of the Regular Chains method (though clearly it is subject to the lower bounds in Sect. 5).
2. There has been much progress in the last forty years, but implementations (at least for systems with alternations of quantifiers) are still doubly-exponential in the number of variables while the theory suggests we can do better.
3. Cylindricity is needed in step 3 of quantifier elimination, as \exists translates into \bigvee and \forall into \bigwedge . However, in fact we only need this at the points where \exists and \forall alternate, so we can weaken the definition of cylindricity from being true for all π_k to merely being true for those k where x_k and x_{k+1} are governed by different quantifiers (or where x_k is unquantified but x_{k+1} is quantified, a concept we can call *block-cylindrical*. Unfortunately, we currently know of no way of computing a block-cylindrical algebraic decomposition without computing the full cylindrical algebraic decomposition first.

Acknowledgements. This work was supported by the EPSRC (grant number EP/J003247/1).

The authors thank Russell Bradford, Nicolai Vorobjov, David Wilson (University of Bath), Changbo Chen (Chinese Academy of Sciences, Chongqing), Zongyan Huang (University of Cambridge), Scott McCallum (Macquarie University) and Marc Moreno Maza (Western University).

References

- ALM99. Aubry, P., Lazard, D., Moreno Maza, M.: On the theories of triangular sets. *J. Symbolic Comp.* **28**, 105–124 (1999)
- AMW08. Achatz, M., McCallum, S., Weispfenning, V.: Deciding polynomial-exponential problems. In: Jeffrey, D.J. (ed.) *Proceedings ISSAC 2008*, pp. 215–222 (2008)
- BBDP07. Beaumont, J.C., Bradford, R.J., Davenport, J.H., Phisanbut, N.: Testing elementary function identities using CAD. *AAECC* **18**, 513–543 (2007)
- BCD+14. Bradford, R., Chen, C., Davenport, J.H., England, M., Moreno Maza, M., Wilson, D.: Truth table invariant cylindrical algebraic decomposition by regular chains. In: Gerdt, V.P., Koepf, W., Seiler, W.M., Vorozhtsov, E.V. (eds.) *CASC 2014. LNCS*, vol. 8660, pp. 44–58. Springer, Heidelberg (2014)
- BD07. Brown, C.W., Davenport, J.H.: The complexity of quantifier elimination and cylindrical algebraic decomposition. In: Brown, C.W. (ed.) *Proceedings of ISSAC 2007*, pp. 54–60 (2007)
- BDE+13. Bradford, R.J., Davenport, J.H., England, M., McCallum, S., Wilson, D.J.: Cylindrical algebraic decompositions for boolean combinations. In: *Proceedings of ISSAC 2013*, pp. 125–132 (2013)
- BDE+14. Bradford, R.J., Davenport, J.H., England, M., McCallum, S., Wilson, D.J.: Truth Table Invariant Cylindrical Algebraic Decomposition (2014). <http://arxiv.org/abs/1401.0645>

- BDEW13. Bradford, R., Davenport, J.H., England, M., Wilson, D.: Optimising problem formulation for cylindrical algebraic decomposition. In: Carette, J., Aspinall, D., Lange, C., Sojka, P., Windsteiger, W. (eds.) C1CM 2013. LNCS, vol. 7961, pp. 19–34. Springer, Heidelberg (2013)
- BPR06. Basu, S., Pollack, R., Roy, M.-F.: Algorithms in Real Algebraic Geometry, 2nd edn. Springer, Heidelberg (2006)
- Bro99. Brown, C.W.: Guaranteed Solution Formula Construction. In: Dooley, S. (ed.) Proceedings of ISSAC 1999, pp. 137–144 (1999)
- Bro03. Brown, C.W.: QEPCAD B: a program for computing with semi-algebraic sets using CADs. ACM SIGSAM Bull. 4(37), 97–108 (2003)
- Bro05. Brown, C.W.: The McCallum projection, lifting, and order-invariance. Technical report, U.S. Naval Academy, Computer Science Department (2005)
- Bro13. Brown, C.W.: Constructing a single open cell in a cylindrical algebraic decomposition. In: Proceedings of ISSAC 2013, pp. 133–140. ACM (2013)
- CM12. Cohen, C., Mahboubi, A.: Formal proofs in real algebraic geometry: from ordered fields to quantifier elimination. Logical Methods Comput. Sci. 8, 1–40 (2012)
- CM14a. Chen, C., Moreno Maza, M.: An Incremental Algorithm for Computing Cylindrical Algebraic Decompositions. In: Feng, R., Sato, W.-S., Sato, Y. (eds.) Computer Mathematics, pp. 199–221. Springer, Heidelberg (2014)
- CM14b. Chen, C., Moreno Maza, M.: Quantifier Elimination by Cylindrical Algebraic Decomposition Based on Regular Chains. In: Proceedings of ISSAC 2014, pp. 91–98 (2014)
- CMXY09. Chen, C., Moreno Maza, M., Xia, B., Yang, L.: Computing Cylindrical Algebraic Decomposition via Triangular Decomposition. In: May, J. (ed.) Proceedings of ISSAC 2009, pp. 95–102 (2009)
- Col75. Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In: Proceedings 2nd GI Conference Automata Theory & Formal Languages, pp. 134–183 (1975)
- Col98. Collins, G.E.: Quantifier elimination by cylindrical algebraic decomposition – twenty years of progress. In: Caviness, B.F., Johnson, J.R. (eds.) Quantifier Elimination and Cylindrical Algebraic Decomposition, pp. 8–23. Springer, Vienna (1998)
- CR88. Coste, M., Roy, M.-F.: Thom’s lemma, the coding of real algebraic numbers and the computation of the topology of semi-algebraic sets. J. Symbolic Comp. 5, 121–129 (1988)
- Dav85. Davenport, J.H.: Computer algebra for cylindrical algebraic decomposition. Technical report TRITA-NA-8511 NADA KTH Stockholm (Reissued as Bath Computer Science Technical report 88–10) (1985)
- Dav15. Davenport, J.H.: Solving Computational Problems in Real Algebra/Geometry. Annales Mathematicae et Informaticae 44, 35–36 (2015) <http://opus.bath.ac.uk/42826/>
- DH88. Davenport, J.H., Heintz, J.: Real quantifier elimination is doubly exponential. J. Symbolic Comp. 5, 29–35 (1988)
- DSS04. Dolzmann, A., Seidl, A., Sturm, T.: Efficient Projection Orders for CAD. In: Gutierrez, J. (ed.) Proceedings of ISSAC 2004, pp. 111–118 (2004)

- EBC+14. England, M., Bradford, R., Chen, C., Davenport, J.H., Maza, M.M., Wilson, D.: Problem formulation for truth-table invariant cylindrical algebraic decomposition by incremental triangular decomposition. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) C1CM 2014. LNCS, vol. 8543, pp. 45–60. Springer, Heidelberg (2014)
- EBDW14. England, M., Bradford, R., Davenport, J.H., Wilson, D.: Choosing a variable ordering for truth-table invariant cylindrical algebraic decomposition by incremental triangular decomposition. In: Hong, H., Yap, C. (eds.) ICMS 2014. LNCS, vol. 8592, pp. 450–457. Springer, Heidelberg (2014)
- EWBD14. England, M., Wilson, D., Bradford, R., Davenport, J.H.: Using the regular chains library to build cylindrical algebraic decompositions by projecting and lifting. In: Hong, H., Yap, C. (eds.) ICMS 2014. LNCS, vol. 8592, pp. 458–465. Springer, Heidelberg (2014)
- Gri88. Grigoriev, D.Y.: Complexity of deciding Tarski algebra. *J. Symbolic Comp.* **5**, 65–108 (1988)
- GV88. Grigoriev, D.Y., Vorobjov Jr., N.N.: Solving systems of polynomial inequalities in subexponential time. *J. Symbolic Comp.* **5**, 37–64 (1988)
- HEW+14. Huang, Z., England, M., Wilson, D., Davenport, J.H., Paulson, L.C., Bridge, J.: Applying machine learning to the problem of choosing a heuristic to select the variable ordering for cylindrical algebraic decomposition. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) C1CM 2014. LNCS, vol. 8543, pp. 92–107. Springer, Heidelberg (2014)
- Hon90. H. Hong. An improvement of the projection operator in cylindrical algebraic decomposition. In: Watanabe, S., Nagata, M. (eds.) Proceedings of ISSAC 1990, pp. 261–264 (1990)
- Hun08. Huntington, G.B.: Towards an efficient decision procedure for the existential theory of the reals. PhD thesis, University of California at Berkeley (2008)
- IYAY13. Iwane, H., Yanami, H., Anai, H., Yokoyama, K.: An effective implementation of symbolic-numeric cylindrical algebraic decomposition for quantifier elimination. *Theor. Comput. Sci.* **479**, 43–69 (2013)
- JdM12. Jovanović, D., de Moura, L.: Solving non-linear arithmetic. In: Gramlich, B., Miller, D., Sattler, U. (eds.) IJCAR 2012. LNCS, vol. 7364, pp. 339–354. Springer, Heidelberg (2012)
- McC84. McCallum, S.: An Improved projection operation for cylindrical algebraic decomposition. PhD thesis, University of Wisconsin-Madison Computer Science (1984)
- McC85. McCallum, S.: An improved projection operation for cylindrical algebraic decomposition. Technical report 548 Computer Science University Wisconsin at Madison (1985)
- McC99. McCallum, S.: On projection in cad-based quantifier elimination with equational constraints. In: Dooley, S. (ed.) Proceedings of ISSAC 1999, pp. 145–149 (1999)
- Pau12. Paulson, L.C.: MetiTarski: past and future. In: Beringer, L., Felty, A. (eds.) ITP 2012. LNCS, vol. 7406, pp. 1–10. Springer, Heidelberg (2012)
- Sei06. Seidl, A.: Cylindrical decomposition under application-oriented paradigms. PhD thesis University of Passau, Germany (2006)
- SS03. Seidl, A., Sturm, T.: A generic projection operator for partial cylindrical algebraic decomposition. In: Proceedings of ISSAC 2003, pp. 240–247 (2003)
- Str06. Strzeboński, A.: Cylindrical algebraic decomposition using validated numerics. *J. Symbolic Comput.* **41**(9), 1021–1038 (2006)

- Str14. Strzeboński, A.: Cylindrical algebraic decomposition using local projections. In: Proceedings of ISSAC 2014, pp. 389–396. ACM (2014)
- Tar51. Tarski, A.: A decision method for elementary algebra and geometry. In: Caviness, B.F., Johnson, J.R. (eds.) *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pp. 24–84. Springer, Vienna (1998)
- Vor89. Vorobjov, Jr., N.N.: Deciding consistency of systems of polynomial in exponent inequalities in subexponential time. In: *Notes of Science Seminars of Leningrad Department of Mathematical Steklov Institute*, p. 176 (1989)
- Vor92. Vorobjov Jr., N.N.: The complexity of deciding consistency of systems of polynomial in exponent inequalities. *J. Symbolic Comp.* **13**, 139–173 (1992)
- Wan00. Wang, D.: Computing triangular systems and regular systems. *J. Symbolic Comp.* **30**(2), 221–236 (2000)
- WBDE14. Wilson, D., Bradford, R., Davenport, J.H., England, M.: Cylindrical algebraic sub-decompositions. *Math. Comput. Sci.* **8**, 263–288 (2014)
- WDEB13. Wilson, D., Davenport, J.H., England, M., Bradford, R.: A “piano movers” problem reformulated. In: *Proceedings SYNASC 2013*, pp. 53–60. IEEE (2013)
- WEDB14. Wilson, D., England, M., Davenport, J.H., Bradford, R.: Using the distribution of cells by dimension in a cylindrical algebraic decomposition. In: *Proceedings SYNASC 2014*. pp. 53–60. IEEE (2013)
- Zar65. Zariski, O.: Studies in equisingularity II. *Amer. J. Math.* **87**, 972–1006 (1965)
- Zar75. Zariski, O.: On equimultiple subvarieties of algebroid hypersurfaces. *Proc. Nat. Acad. Sci. USA* **72**, 1425–1426 (1975)

The Relation Tool in GeoGebra 5

Zoltán Kovács^(✉)

Department of Mathematics Education, Johannes Kepler University,
Altenberger Strasse 54, 4040 Linz, Austria
zoltan@geogebra.org

Abstract. GeoGebra is open source mathematics education software being used in thousands of schools worldwide. Its new version 5 supports automatic geometry theorem proving by using various methods which are already well known, but not widely used in education software. GeoGebra's new embedded prover system chooses one of the available methods and translates the problem specified by the end user as the input for the selected method, similarly to portfolio solvers. The available methods include Wu's method, the Buchberger-Kapur method, the Area method and Recio's exact check method, some of them as embedded algorithms, others as outsourced computations. These methods can also be hidden from end users who are provided with an intuitive graphical user interface, the Relation Tool. Since GeoGebra maintains the development in an open-sourced way by collaborating with the OpenGeoProver, Singular and Giac projects, further enhancements can be expected by a larger community, including implementing other methods, too.

Keywords: GeoGebra · Portfolio solver · Computer algebra · Computer aided mathematics education · Automated theorem proving

1 Introduction

GeoGebra [1] is educational mathematics software, with millions of users worldwide. Its founder, Markus Hohenwarter broadened its software development into an open source project in 2005. GeoGebra has many features (including dynamic geometry, computer algebra, spreadsheets and function investigation), but it primarily focuses on facilitating student experiments in Euclidean geometry, and not on formal reasoning. Including automated deduction tools in GeoGebra could bring a whole new range of teaching and learning scenarios. Since automated theorem proving (ATP) in geometry has reached a rather mature stage, some ATP experts agreed on starting a project of incorporating and testing a number of different automated provers for geometry in GeoGebra. This collaboration was initiated by Tomás Recio in 2010.

In this paper a general overview is given about the technical details of this joint work, highlighting its latest achievements including the new Relation Tool in GeoGebra. Section 2 presents related work and gives a brief historical overview. In Sect. 3, arguments for the structural decisions of the collaboration

are shown. In Sect. 4, the design of the portfolio prover is demonstrated. Section 5 shows some examples how the implemented system works. Section 6 depicts the new Relation Tool. Section 7 discusses the possible ways of a future enhancement of the joint work.

2 Related Work

It must be emphasized that a number of software systems, which focus on computer aided proving and dynamic geometry, have existed for several years. The most well known include *GeoProof*¹ by Julien Narboux [2, 3], *GDI-Discovery* [4] by José Valcarce and Francisco Botana, the *Geometry Expert*² [5] by Shang-Ching Chou, Xiao-Shan Gao and Zheng Ye, and *GEOTHER* [6] by Dongming Wang. These (and some other) dynamic geometry systems (DGS) with ATP features can efficiently prove many complex geometry theorems, but these ATP features are not primarily designed for applications in education³. They are, in many aspects, still in the prototype phase, not yet well distributed, maintained or not fully operative.

Without any doubts a remarkable amount of work has already been contributed to publicly available algorithms and their implementations. Also open-sourced ATP systems and test databases were introduced, including *Open-GeoProver* (OGP) [7, 8] and *GeoThms* [9], started and maintained by Predrag Janičić, Pedro Quaresma and their colleagues. Collaborative work seemed to be a very important step in widening the availability of ATP algorithms to be used in education. We found that a primary problem is that scientific contributions to DGS and ATP are isolated, and in an open-sourced system the existing efforts could be gathered and continued.

In 2011 Narboux, Yves Bertot and his PhD student Tuan Minh Pham started to change GeoProof’s user interface to GeoGebra [10], but they still used *Coq* [11] as the external formal prover. However, a part of the GeoGebra Team, lead by the author of this paper, advised by Hohenwarter, Recio and Botana, and supported by Janičić, started to work on a different approach at the beginning of 2012—their solution was to use both an embedded system in GeoGebra and also outsourced computations [12–16].

One of the most advanced pieces of software being available for free download is the *Java Geometry Expert* (*JGEX* [17, 18]) developed by Chou, Gao and Ye, but built upon the work of several other experts of the modern Chinese computational mathematics. JGEX’s prover system contains multiple methods to compute proofs of the input being constructed graphically or by a custom programming language. The available methods are the Gröbner basis method (developed by Buchberger, Kapur [19] and others), Wu’s method [20], the Area method [21] and the geometry deductive database (GDD) method [22].

¹ <http://home.gna.org/geoproof/>.

² <http://www.mmrc.iss.ac.cn/gex/>.

³ One example of a system planned for teaching proofs interactively is Jacques Gressier’s *Géométrie*, available at <http://geometrix.free.fr>.

JGEX gives the opportunity for the user to select from the existing methods since in some circumstances any of them can fail: each has its strengths and weaknesses. In GeoGebra’s portfolio prover (GPP) we followed the same approach, but an automated selection will be used to minimize user interaction: we assume that the user is a secondary school student who knows nothing about ATP. That is, GPP will work with default values which can be overridden by only expert users via command line arguments⁴.

Unfortunately, the source code for JGEX is not available for free downloading, and also the GDD method (which probably has the most remarkable reputation on educational use) is a kind of black box. GeoGebra’s long term success in the classroom is not only that it is freely available for end users, but it is also possible to be enhanced and bug-fixed—the open source development model is a requirement for that.

3 Open Source Collaboration

We also wanted to leave the opportunity for other DGS to use the implemented methods in other applications than GeoGebra as well. For such a warranty we agreed to start joint work with Janičić’s research team on the OGP system. In its first version, OGP was capable only for computing geometry proofs by using Wu’s method, but later we managed to get support from the Google Summer of Code for a university student, Damien Desfontaines, who implemented the Area method in OGP in 2012 [23]. The OGP project was considered as a fruitful way for collaboration of other experts as well: the Mass point method and the Full angle method were worked on (an introductory report on the latter was already published by Quaresma and Baeta in 2013 [24]).

Meanwhile, we also started to work on internal implementations of other methods. The first internally implemented method was a new approach suggested by Recio, to use exact coordinates and arbitrary integer arithmetics on testing geometry statements for checking validity for a set of elementary theorems. His algorithm [13], which sets up a bounded number of test cases of the geometry statement with carefully chosen coordinates, was implemented by Simon Weitzhofer in 2012 and published in his master thesis [25] in 2013. A more traditional approach, namely the Buchberger-Kapur method (based on Gröbner basis computation), was also implemented by the author of this paper, by using external computations for solving equation systems. This method was later extended to use the Recio-Vélez algorithm [26] to obtain degeneracy conditions, and the algorithm was later enhanced for better educational use.

The outsourced computations were achieved by using *Singular* [27] as an external web service, running on a dedicated virtual server. This technology was discussed by Botana and the author in the planning stage of GeoGebra’s ATP capabilities. For this reason the method used in GeoGebra is called “Botana’s method” [12].

⁴ http://wiki.geogebra.org/en/Release_Notes_GeoGebra_5.0#New_Command_Line_Arguments.

In GeoGebra we already managed to make the prototype implementation work, many schools and students started to move from the desktop application to a different technology: they preferred to use tablets and smartphones instead of desktop PCs and laptops. Unfortunately, the Java technology and the outsourced computations are not always applicable in the changed way of using computers in the education. That is why we had to find even new technologies to support the HTML5/JavaScript approach of application development, including offline HTML5 applications as well [28]. Fortunately, the GeoGebra Team managed to change the internal computer algebra system from *Reduce* [29] to *Giac* [30] which was a step forward to support faster Gröbner basis computations, also available in a web browser in offline mode on a tablet or a smartphone [31, 32].

Now GPP is no longer a prototype. It is also fully documented not only in its source code and the Developers' Howto⁵ but on GeoGebra's Wiki pages⁶, and a set of demonstrational examples is available on GeoGebraTube⁷. It is an extensible system in both GeoGebra and OGP.

We would like to highlight that the success of GPP is based definitely on its open-sourced roots. Without the existing knowledge in the implementations of several modern algorithms in the used systems—especially in Singular and Giac—GeoGebra would not have the chance to offer competitive ATP features for the mathematics education. These systems have several years of programmers' experience and millions of lines of program code which could not have been reimplemented from scratch within a reasonable time.

4 The Design of GPP

The design of GPP is shown in Fig. 1. At the top of the figure GeoGebra's user interface is shown: the higher the action is drawn, the easier communication for the user is assumed. Also in former versions of GeoGebra the highest level action is to use the *Relation Tool* which purely numerically decides if two geometrical objects have a relation like parallelism, perpendicularity, equal length etc. Prior to GeoGebra 5, the Relation Tool was using floating point (FP) arithmetics to decide if a certain relation holds between the input objects. In some cases, however, FP returns an inaccurate or wrong answer, thus the output for Relation Tool is incorrect in some cases.

Always there may be a need to increase the “verification level” of the investigation which means we collect more or more trustworthy information about the statement to decide (see the top of Fig. 1, moving from left to the right). GeoGebra's new command **Prove** gives an ATP answer instead of FP arithmetics with the possible outputs “yes”, “no” or “undefined” (the last kind of output means that no trustworthy answer was found). What is more, the new **ProveDetails**

⁵ See <http://dev.geogebra.org/trac/wiki/TheoremProving> for more details. This documentation includes detailed description of the applied methods and the supported constructions and statements for them.

⁶ For an example, see <http://wiki.geogebra.org/en/ProveDetails.Command>.

⁷ See <http://www.geogebraTube.org/student/b104296> for an example.

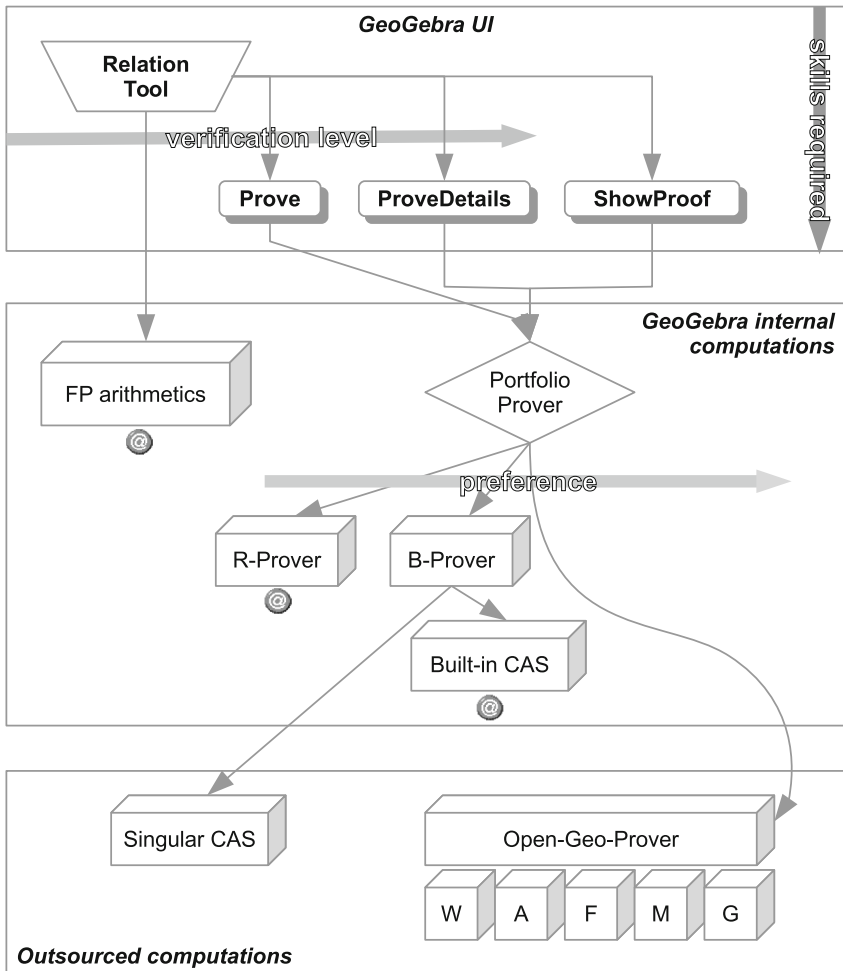


Fig. 1. The design of GeoGebra's portfolio prover. Here W, A, F, M and G describe different computation methods in OpenGeoProver like Wu's method, the Area method, the Full angle method, the Mass point method and the Gröbner basis method

command can give more details on degeneracy conditions, that is, it can refine the statement by adding a sufficient condition if needed. For the moment, the **ShowProof** command is not yet implemented just planned: when a “readable enough” proof will be found by the ATP subsystem, the proof could be shown to the student for ensuring complete certainty.

In the middle of Fig. 1 GeoGebra's internal computations are shown. GPP optionally analyzes the shape of the statement and tries to select the most fitting method to work with: for example, for problems which do not contain circles but lines only and contain at most 3 free points, Recio's method should be used. If

no analysis is used (that is, in AUTO mode), then a simple priority list of the methods is taken: at the moment for the **Prove** command this is

$$\text{Recio} < \text{Botana} < \text{Wu (OGP)} < \text{Area (OGP)},$$

and for the **ProveDetails** command⁸:

$$\text{Botana} < \text{Wu (OGP)}.$$

Recio’s method (R-Prover) is a quick method for proving statements concerning points and lines in a triangle. Since it cannot be applied for constructions containing conics, and for more than 3 free points it may be too slow (i.e., the computation may take several seconds), GeoGebra will consider Botana’s method (B-Prover) as a fallback in such cases. At the moment, some sorts of constructions (including angle bisectors) are not yet implemented in the B-Prover, thus in such cases GeoGebra will go ahead by using OGP’s Wu’s method to obtain an answer. If Wu’s method fails, GeoGebra resends the statement to OGP by requesting the computation via the Area method—for example, to prove Ceva’s theorem this will be the only working way at the moment (see Sect. 5.3).

R-Prover uses arbitrary integer arithmetics internally, but B-Prover requires computing solutions of a polynomial equation system for the **Prove** command and elimination of some variables from a polynomial equation system for the **ProveDetails** command. (In general, B-Prover requires Gröbner basis computations.) Detailed investigation showed that it would be too time consuming to implement an internal algorithm in GeoGebra to efficiently compute polynomial based calculations, thus we use the Singular computer algebra system (CAS) instead. After changing internal CAS of GeoGebra from Reduce to Giac we found that Giac computed Gröbner bases surprisingly efficiently, and its speed was comparable with Singular in many cases. Also Giac can easily be used in a web browser as well since it is written in C++ and by utilizing the *emscripten* [33] C++ to Javascript compiler (or Google’s *NaCl* C++ to Native Client compiler) the Gröbner basis computations are still acceptably fast.⁹

In the bottom of Fig. 1 the externally used systems are shown. At the moment, these external systems cannot be used in HTML5 mode (including online and offline modes)—we used the symbol “@” to mark those subsystems in GPP which are transparent for the technology change. Our future plans include compile both Singular and OGP to be technology transparent, i.e. we plan to make them available also on the web platform.

⁸ Degeneracy conditions can be obtained only by two methods at the moment.

⁹ See <http://test.geogebra.org/~kovzol/data/Prove-20150219b/> for a recent report on benchmarking various theorems with the R-Prover based on computations with Singular and Giac, and compared with OGP’s Wu’s method. The full source code of GPP and its benchmarking system are freely available at <https://dev.geogebra.org/trac/browser/trunk/geogebra> in folders `common/src/main/java/org/geogebra/common/kernel/prover/` and `test/`, respectively.

As already mentioned in Sect. 3, OGP currently supports Wu’s method (W) and the Area method (A), and is subject to be enhanced by additional ATP methods including the Full angle method (F), the Mass point method (M) and the Gröbner basis method (G)—the last is definitely the same as the internal B-Prover in GeoGebra.

At the top of Fig. 1 the Relation Tool is shown as the easiest way to start GPP. For users having advanced skills, GPP is also available via GeoGebra commands.

5 GPP Examples

In this section four theorems are provided as constructed in GeoGebra 5. Most of them can be introduced in many secondary schools and thus they are examples of possible classroom uses of GPP. The first three examples run in the desktop version, and the final example is shown in a web browser.

Despite the interesting part of the log messages being shown after these examples, they are not intended to be displayed neither for the students, nor the teachers. Here they are printed for the researcher’s interest. Students and teachers should be informed via GeoGebra’s user interface only in the Algebra View, or by using the symbolical feature of the Relation Tool.

That is, the output of the **Prove** command in this section is shown on the left hand side of the GeoGebra window (by default) among the Boolean Value entries (see the top-left corner of Figs. 2, 3 and 4). The output of the **ProveDetails** command is shown among the List entries (see the bottom-left corner of Fig. 5).

5.1 Altitudes of a Triangle are Concurrent

Figure 2 was created by using GeoGebra 5¹⁰ by drawing a triangle with its altitudes d , e and f and then the command **Prove[AreConcurrent[d,e,f]]** was entered in the Input Bar (at the bottom). Here GPP selects R-prover to compute $\binom{3+2}{2}$ tests for a trustworthy answer if the altitudes are always concurrent (not considering some degenerate cases). The computation took 8 ms on a typical PC¹¹:

```
12:32:26.218 DEBUG: geogebra.m.o$a.run[-1]: Using AUTO
12:32:26.218 DEBUG: geogebra.common.p.y.a[-1]: Using RECIOS_PROVER
12:32:26.219 DEBUG: geogebra.common.l.q.a.a[-1]: nr of tests: 10
12:32:26.224 DEBUG: geogebra.common.l.q.i.<init>[-1]: Benchmarking: 8 ms
12:32:26.224 DEBUG: geogebra.common.l.q.i.<init>[-1]: Statement is TRUE
```

¹⁰ GeoGebra 5 can be downloaded from its official web site <http://www.geogebra.org/download>.

¹¹ Java method names in the log are obfuscated to ensure faster results and a smaller software package.

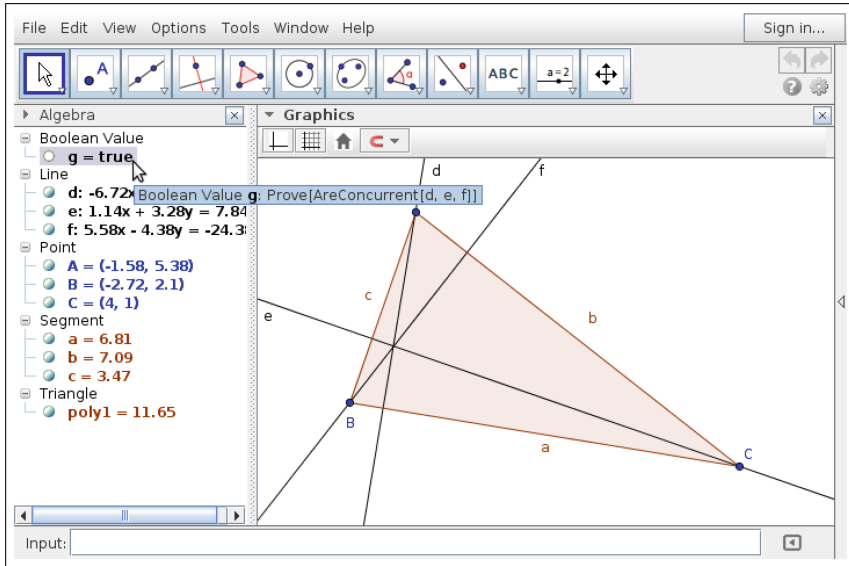


Fig. 2. Altitudes of a triangle are concurrent

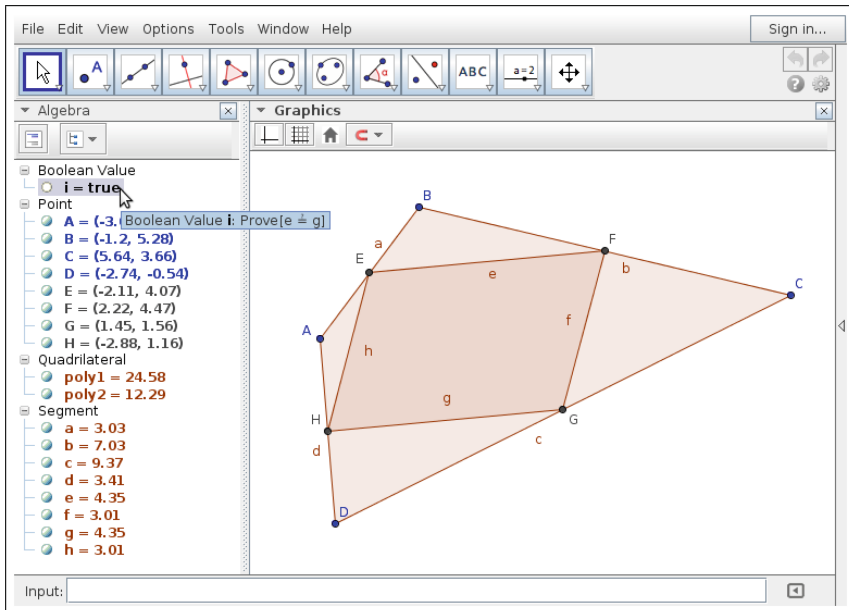


Fig. 3. Varignon's theorem

5.2 Varignon’s Theorem

Figure 3 shows an arbitrary quadrilateral with the midpoints of its sides (E, F, G and H). Now when considering the quadrilateral $EF GH$ we can find that it is a parallelogram. To verify this we need the command **Prove[e==g]**, for example. Since R-Prover cannot deal with Euclidean distances, B-Prover is selected by GPP. The computation took 47 milliseconds on a typical PC (which already contains the HTTP request to the external server and also its background computation—here the uninteresting parts of the log messages have been omitted and substituted by [...]):

```

12:56:23.790 DEBUG: geogebra.m.o$a.run[-1]: Using AUTO
12:56:23.790 DEBUG: geogebra.common.p.y.a[-1]: Using RECIOS_PROVER
12:56:23.791 DEBUG: geogebra.common.p.y.a[-1]: Using BOTANAS_PROVER
[...]
12:56:23.806 DEBUG: geogebra.common.p.y.b[-1]: Thesis reductio ad absurdum (denied statement):
12:56:23.806 DEBUG: geogebra.common.p.y.b[-1]: 9. -1+-1*v17*v16^2+-1*v17*v15^2+2*v17*v16*v14+
-1*v17*v14^2+2*v17*v15*v13+-1*v17*v13^2+v17*v12^2+v17*v11^2+-2*v17*v12*v10+v17*v10^2+
-2*v17*v11*v9+v17*v9^2
12:56:23.807 DEBUG: geogebra.common.l.q.n.a[-1]: ring r=(0,v1,v2,v3,v4,v5,v6,v7,v8),
(v9,v10,v11,v12,v13,v14,v15,v17,v16),dp;ideal i=2*v9+-1*v3+-1*v1,2*v10+-1*v4+-1*v2,
2*v11+-1*v5+-1*v3,2*v12+-1*v6+-1*v4,2*v13+-1*v7+-1*v5,2*v14+-1*v8+-1*v6,2*v15+-1*v7+-1*v1,
2*v16+-1*v8+-1*v2,-1+-1*v17*v16^2+-1*v17*v15^2+2*v17*v16*v14+-1*v17*v14^2+2*v17*v15*v13+
-1*v17*v13^2+v17*v12^2+v17*v11^2+-2*v17*v12*v10+v17*v10^2+-2*v17*v11*v9+v17*v9^2;
i=subst(i,v1,0,v2,0,v3,0,v4,1);groebner(i)!=1; -> singular
12:56:23.835 DEBUG: geogebra.common.l.q.n.a[-1]: singular -> 0
12:56:23.836 DEBUG: geogebra.common.l.q.i.<init>[-1]: Benchmarking: 47 ms
12:56:23.836 DEBUG: geogebra.common.l.q.i.<init>[-1]: Statement is TRUE
    
```

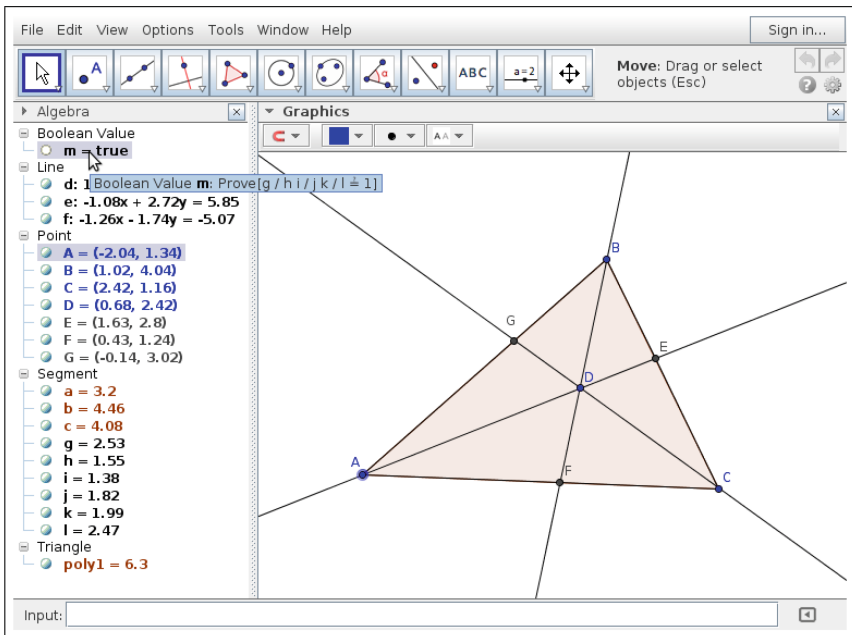


Fig. 4. Ceva’s theorem

5.3 Ceva's Theorem

Triangle ABC and its arbitrary internal point D was constructed in Fig. 4. Now intersection points of lines AD , BD , CD and the appropriate sides are E , F , G , respectively. Now let us define $g = AG$, $h = GB$, $i = BE$, $j = EC$, $k = CF$, $l = FA$, then $g/h \cdot i/j \cdot k/l = 1$. Here OGP's Area method is the only possible way to get a useful ATP answer to decide the statement. To start GPP we enter **Prove**[$g/h \cdot i/j \cdot k/l == 1$]. The result is computed in 4 ms in OGP's area subsystem, but since other methods were also attempted to use, the total time is 95 ms spent in GPP (see Appendix for the full output).

As seen, (after taking 1 ms in the R-Prover and realizing that it is not helpful) B-Prover cannot process the construction since measuring segments are not yet implemented for it. OGP's Wu's method is also unable to read the information provided by GeoGebra, thus finally OGP's Area method converts the statement into an internal object, and then successfully processes it.

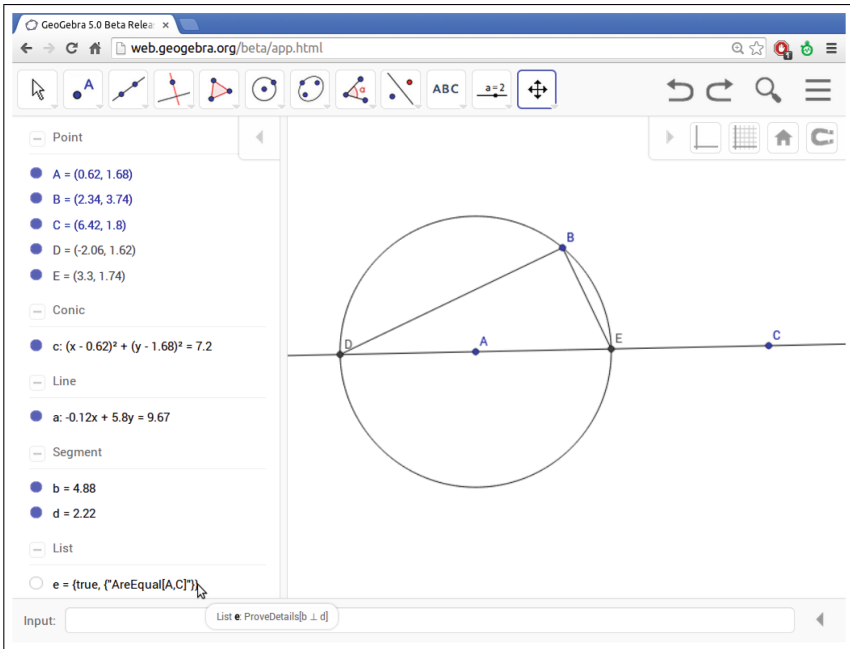


Fig. 5. Thales' circle theorem

5.4 Thales' Circle Theorem

Thales' (circle) theorem states that in a given circle with center A , circum-point B and diameter DE , lines BD and BE are perpendicular. In Fig. 5 we

use free points A , B and C (which is a technical point to define D and E as intersection points of line AC and the circle). Let us denote BD by b and BE by d . Now GeoGebra command **ProveDetails**[$b \perp d$] returns the output list $\{true, \{ "AreEqual[A,C]" \} \}$ which has the following meaning:

- the statement “ $b \perp d$ ” holds in general,
- if $A \neq C$ then the statement surely holds.

Clearly, if $A = C$, points D and E are undefined, thus the statement has no meaning. We emphasize here that Fig. 5 is created by running GeoGebra in a web browser. Thus here the only possible method is B-Prover, and only with the embedded CAS, Giac. In this example the construction is loaded from an external file, thus the Javascript version of Giac (giac.js) must be preloaded before any concrete computations in GPP. This is the technical reason why we can see that GPP ran multiple times (first it reported “undefined” result in 18 ms, then another “undefined” in 3 ms, and the final computation took 2091 ms—“undefined” here is displayed as “Statement is null”). (See Appendix for the full output.)

In the log we can see that Giac returns the elimination ideal for the Recio-Vélez algorithm in the same way as Singular does.

6 The Relation Tool

From the educational point of view, the direct classroom use of the **Prove** and **ProveDetails** commands cannot always be considered to be an adequate approach. First of all, the output of both commands looks static: they are not natural extensions of the traditional “visual” way of dynamic geometry systems. This means that for a given conjecture both commands will compute some results, but after dynamically changing the construction by dragging the free points, the outputs of **Prove** and **ProveDetails** will remain the same. This is, although, mathematically correct and transmits some kind of “theoretical stability”, it may not reflect the importance of the result that we actually proved something for infinitely many cases. The fast computation (i.e. that the result is shown almost immediately on the screen) gives the feeling that something easy is in the background, so in sum using these new commands can yield an opposite effect to the teacher’s plans. Finally, these commands cannot be reached from the GeoGebra toolbar: the keyboard is required to enter them. On the other hand, the output of the **ProveDetails** command can still be inconvenient for many students since it is given in a list type object (which is again not visual enough).

The Relation Tool addresses these problems. In GeoGebra 5 its original numerical version has been extended to use symbolical checks as well. To not confuse experienced users who already assume that a numerical check will be done we ensured backward compatibility, that is, the symbolical check will be executed only on demand when the user presses an extra button.

The following algorithm describes how the Relation Tool in GeoGebra 5 works:

1. Use the Relation Tool like former GeoGebra versions do (i.e., a numerical check is still computed), but add an extra button “More...” on the right to the output statement if the statement was found to be true numerically. (If the statement is not true numerically, then this extra button is not shown.)
2. If the student clicks this button, GeoGebra computes the result for the statement by using the **Prove** command internally. Now the answer can be “generally true” (true), “not generally true” (false) or GeoGebra “cannot decide” (undefined):
 - (a) In the “generically true” and “cannot decide” cases GeoGebra will compute the result for the statement by using the **ProveDetails** command as a second step. Its output will be converted to a more user friendly format:
 - i. If there are no non-degeneracy conditions, the Relation Tool appends “always true” to the statement.
 - ii. If there is a list of readable conjunctions found as non-degeneracy conditions, then the list is displayed on the computer screen as a sufficient condition for the statement.
 - iii. If non-degeneracy conditions were found which are not “human readable”, then the statement is appended with the message “under certain conditions”. (Here “human readable” means such conditions which are easy to formulate in the classroom as well, for example *perpendicularity*, *parallelism* or *equality*.)
 - iv. If the **ProveDetails** command returned undefined,
 - A. but **Prove** returned true then the statement is proven to be “generally true”;
 - B. otherwise GeoGebra appends the text “possibly generally true”.
 - v. If the **ProveDetails** command returns {false} then GeoGebra realizes that an internal error occurred since the result of the **Prove** and **ProveDetails** commands are contradictory. This case should not happen.
 - (b) In the “not generically true” case the Relation Tool appends “but not generally true” to the statement.

This decision process is shown as a flowchart diagram in Fig. 6.

We highlight that we make a difference between cases (a) iii. and iv. A. In the first case the student is informed that there should be conditions given, but they are difficult to explain. In the second case it is not sure that there are any conditions, that is the statement can be always true as well, but GeoGebra cannot compute the detailed answer. Here the teacher may need to explain this situation.

Also an intentional decision is the wording for case (a) iv. B. For the ‘too difficult’ conjectures both the **Prove** and **ProveDetails** commands return undefined which may mean that

1. one or more steps cannot be formulated, algebraized or processed by GeoGebra, or

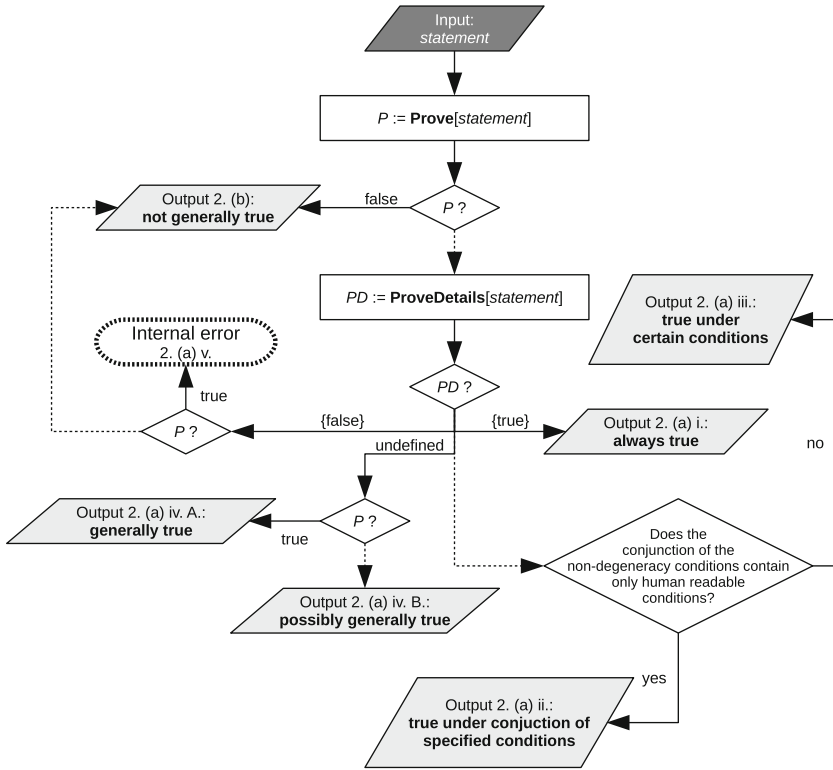


Fig. 6. The decision process in the Relation Tool. Dashed lines show the “otherwise” cases

2. the computation was too difficult (more memory or CPU time would have been needed).

Since the numerical computation about the truth of the statement returned positive, it seems plausible that a symbolical computation with more resources could certify the numerical computation as well. This is why we use the word “possibly” here. Of course this does not tell any certainty about the truth of the statement.

In our opinion, using the Relation Tool is convenient: the student does not require to use GeoGebra’s Input Bar to enter anything—only mouse clicks are sufficient for the investigation.

Let us see an example for case 2. (a) ii. which is the most general case. In Fig. 7 the nine point circle theorem is shown. Given the ABC triangle, we construct feet points of altitudes d , e and f as points D , E and F , respectively. Intersection point of d and e is G . Midpoints of AG , BG and CG are H , I and J , respectively. Finally, midpoints of sides a , b and c are K , L and M , respectively. The statement is that points D , E , F , H , I , J , K , L and M lie on the same circle.

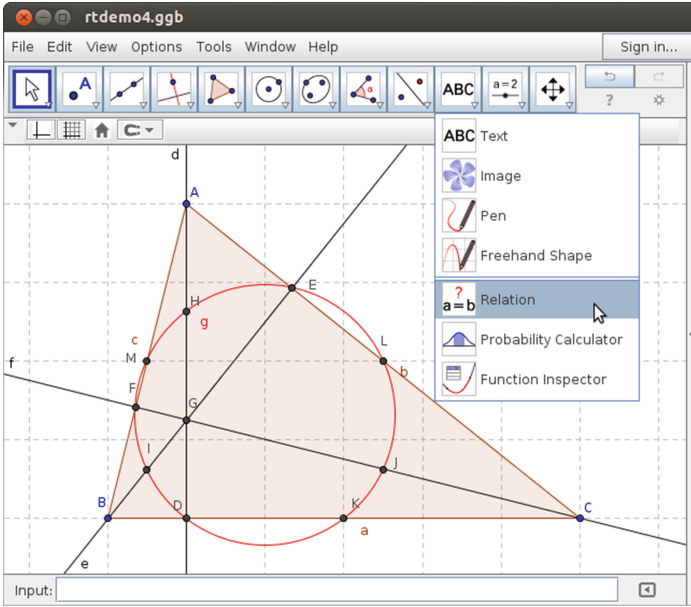


Fig. 7. The nine point circle theorem is to be checked by the Relation Tool

This statement can be formulated by various ways, for example one formulation is to create circle g which is defined by circumference points F , I and M . Now if we are about to check if point L is lying on g , we need to select the Relation Tool from the Toolbar (as shown in Fig. 7), and compare objects L and g by clicking on them. Then GeoGebra initiates a numerical check first (Fig. 8), and since it is positive, an extra symbolical check can be started by pressing the “More...” button. The result, after a short computation¹², can be found in Fig. 9.

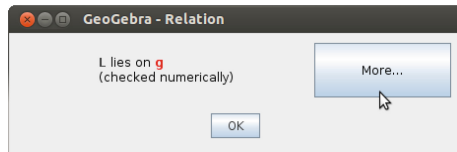


Fig. 8. A numerical check is performed by the Relation Tool

¹² To obtain exactly the same result, the user needs to force using SingularWS which is deactivated by default in GeoGebra 5. Enabling SingularWS can be performed by using the command line `geogebra --singularws=enable:true` e.g. on Linux. When SingularWS is disabled, Giac is used to compute the results: here case 2. (a) (iv.) B. (i.e., “possibly generally true”) will be selected.

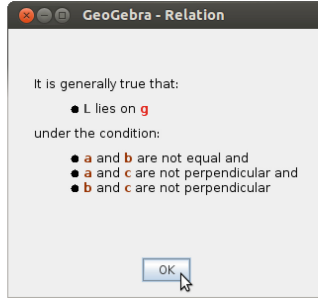


Fig. 9. A symbolical check is performed by the Relation Tool on demand

The Relation Tool in GeoGebra 5 is ready for starting usability tests on students, and when the tests are positive, it may become a tool to help in the teaching of proofs. However, additional enhancements are still possible. In GeoGebra between two objects there may be more kind of relationships: for example, two segments can have same length and/or parallel independently. Now the Relation Tool reports numerical checks for all possible relationships in its output window. This idea could be further improved to collect relationships among a wider set of objects than two, and investigate all possible combinations of these objects to collect the relations. For example, the student could draw a triangle and its medians and select some (or eventually all) objects in the figure. In this case GeoGebra could give a list of all relationships among the selected objects, including the concurrency of the medians. Such a list can be, however, quite long. Thus it would be important to make it possible to filter out some trivial relations, or at least to show first those which seem to be interesting enough.

7 Conclusion and Future Work

The DGS features of GeoGebra are already well known in schools, but also ATP functionality is included in version 5 only since September 2014. Its careful design of the user interface and the applied methods using open source technology ensure, in our opinion, a good start to offer useful new ways to teach and learn Euclidean geometry.

There is, however, still lot of work to do. Other methods could be implemented either in GeoGebra or OGP (or even in both), the current methods could be parallelized and further improved. Also the outsourced computations could be included in GeoGebra by using the newest compilers.

A completely open question is how readable proofs should be shown in GeoGebra without confusing the students. The JGEX application has already many promising ways which should be further discussed not only by ATP experts but teachers as well, especially those who teach in secondary schools. After such a consensus we would like to start to implement the **ShowProof** command in GeoGebra.

We believe that GeoGebra (and also OGP) could be a common platform for other ATP experts to join and since the source code is completely open to the public, there is no technological obstacle to integrate the knowledge base into a single application. A long term collaboration with experts from various countries would be fruitful in each classroom since GeoGebra helps each student in his or her native language to understand mathematics even more.

Acknowledgments. Ivan Petrović, PhD student of University of Belgrade, the primary author of OGP had a great part in making OGP generally work. Gábor Ancsin was the leader of the HTML5 based experiments in the GeoGebra Team. Michael Borchers and Zbyněk Konečný helped regularly to find and eliminate bugs during the development. Last but not least, Bernard Parisse kindly helped in improving Giac to be even more robust for Gröbner basis computations.

Bruno Buchberger kindly invited the author of this paper to present a preliminary version of this paper at the RISC Theorema seminar on 9 April 2014. (The slides for that talk are available to download at <http://ggbl.idm.jku.at/~kovzol/talks/risc2014-2/>.)

Pedro Quaresma and Francisco Botana supported the author to present essential parts of this paper at the ADG 2014 conference on 10 July 2014. (Slides and supplementary materials for this talk can be accessed at <http://ggbl.idm.jku.at/~kovzol/talks/adg2014/>.)

Appendix

Log Output for Ceva's Theorem

```
13:22:01.959 DEBUG: geogebra.m.o$a.run[-1]: Using AUTO
13:22:01.959 DEBUG: geogebra.common.p.y.a[-1]: Using RECIOS_PROVER
13:22:01.960 DEBUG: geogebra.common.p.y.a[-1]: Using BOTANAS_PROVER
[...] not fully implemented
13:22:01.966 DEBUG: geogebra.common.p.y.a[-1]: Using OPENGEOPROVER_WU
[...]
13:22:01.987 INFO: a.b.a.a.a[-1]: Reading input geometry problem...
[...]
13:22:02.022 DEBUG: geogebra.common.p.y.b[-1]: success: false
13:22:02.022 DEBUG: geogebra.common.p.y.b[-1]: failureMessage: Failed in reading input
[...]
13:22:02.024 DEBUG: geogebra.common.p.y.a[-1]: Using OPENGEOPROVER_AREA
[...]
13:22:02.026 INFO: a.b.a.a.a[-1]: Reading input geometry problem...
13:22:02.037 INFO: a.b.a.a.d.e[-1]: Converting equal statement. Arguments :
13:22:02.037 INFO: a.b.a.a.d.e[-1]: ((Segment[A, G] / Segment[G, B] Segment[B, E] /
Segment[E, C]) Segment[C, F] / Segment[F, A])
13:22:02.037 INFO: a.b.a.a.d.e[-1]: 1
[...]
13:22:02.053 DEBUG: geogebra.common.p.y.b[-1]: success: true
[...]
13:22:02.053 DEBUG: geogebra.common.p.y.b[-1]: time: 0.004
[...]
13:22:02.054 DEBUG: geogebra.common.l.q.i.<init>[-1]: Benchmarking: 95 ms
13:22:02.055 DEBUG: geogebra.common.l.q.i.<init>[-1]: Statement is TRUE
```

Log Output for Thales's Circle Theorem

```
14:31:25.918 DEBUG: ?: Using AUTO
```

```

14:31:25.918 DEBUG: ?: Using BOTANAS_PROVER
14:31:25.919 DEBUG: ?: Testing local CAS connection
14:31:25.928 DEBUG: ?: starting CAS
[...]
14:31:25.935 DEBUG: ?: Benchmarking: 18 ms
14:31:25.936 DEBUG: ?: Statement is null
14:31:25.967 DEBUG: ?: Using AUTO
14:31:25.967 DEBUG: ?: Using BOTANAS_PROVER
14:31:25.968 DEBUG: ?: Testing local CAS connection
[...]
14:31:25.971 DEBUG: ?: Benchmarking: 3 ms
14:31:25.971 DEBUG: ?: Statement is null
14:31:26.704 DEBUG: ?: giac.js loading success
14:31:27.273 DEBUG: ?: Using AUTO
14:31:27.274 DEBUG: ?: Using BOTANAS_PROVER
14:31:27.275 DEBUG: ?: Testing local CAS connection
[...]
14:31:27.719 DEBUG: ?: Thesis reductio ad absurdum (denied statement):
14:31:27.720 DEBUG: ?: 6.  $-1+1*v12*v10*v8+1*v12*v9*v7+v12*v10*v4+v12*v8*v4+1*v12*v4^2+v12*v9*v3+v12*v7*v3+1*v12*v3^2$ 
14:31:27.727 DEBUG: ?: Eliminating system in 10 variables (6 dependent)
[...]
14:31:27.817 INFO: ?: [eliminateFactorized] input to cas: [[containsvars(poly,varlist)]:=
{local ii; for (ii:=0; ii<size(varlist); ii++) { if (degree(poly,varlist[ii])>0
{ return true } } return false}], [myeliminate(polylist,varlist)]:={local ii,jj,kk;
kk=[]; jj:=gbasis(polylist,varlist,revlex); for (ii:=0; ii<size(jj); ii++) { if
(!containsvars(jj[ii],varlist)) { kk:=append(kk,jj[ii]) } } return kk }], [ff:=""],
[aa:=myeliminate([-1*v7*v6+v8*v5,-1*v8^2+1*v7^2+v4^2+v3^2,-1*v9*v6+v10*v5,
-1*v10^2+1*v9^2+v4^2+v3^2,-1+v11*v10^2+v11*v9^2+2*v11*v10*v8+v11*v8^2+
-2*v11*v9*v7+v11*v7^2,-1+1*v12*v10*v8+1*v12*v9*v7+v12*v10*v4+v12*v8*v4+1*v12*v4^2
+v12*v9*v3+v12*v7*v3+1*v12*v3^2],[v7,v8,v9,v10,v11,v12])],[bb:=size(aa)], [for ii
from 0 to bb-1 do ff+="{ "+(ii+1)+"": [1]: _[1]=1";cc:=factors(aa[ii]);dd:=size(cc);
for jj from 0 to dd-1 by 2 do ff+="{ "+(jj/2+2)+"": "+cc[jj]"; od; ff+="{ [2]: "+
cc[1]";for kk from 1 to dd-1 by 2 do ff+="{ "+cc[kk]";od;od},ff][6]
[...]
14:31:29.316 DEBUG: ?: giac output:"[1]: [1]: _[1]=1 _[2]=v5 [2]: 1,1[2]: [1]: _[1]=1
_[2]=v6 [2]: 1,1
14:31:29.318 INFO: ?: [eliminateFactorized] output from cas: [1]: [1]: [...]
14:31:29.324 DEBUG: ?: Considering NDG 1...
14:31:29.329 DEBUG: ?: Trying to detect polynomial v5
14:31:29.349 DEBUG: ?: v5 means x-equality for [v1, v5]
14:31:29.350 DEBUG: ?: Not better than previous NDG score (Infinity), this is Infinity
14:31:29.350 DEBUG: ?: Considering NDG 2...
14:31:29.351 DEBUG: ?: Trying to detect polynomial v6
14:31:29.363 DEBUG: ?: v6 means y-equality for [v2, v6]
14:31:29.364 DEBUG: ?: Found a better NDG score (0.5) than Infinity
14:31:29.365 DEBUG: ?: Benchmarking: 2091 ms
14:31:29.365 DEBUG: ?: Statement is true

```

References

1. Hohenwarter, M.: GeoGebra: Ein Softwaresystem für dynamische Geometrie und Algebra der Ebene. Master's thesis, Paris Lodron University, Salzburg, Austria (2002). (in German)
2. Narboux, J.: GeoProof, a user interface for formal proofs in geometry. In: Mathematical User-Interfaces Workshop, Electronic Proceedings. Schloss Hagenberg, Linz, Austria (2007). <http://www.activemath.org/workshops/MathUI/07/proceedings/Narboux-Geoproof-MathUI07.html>
3. Narboux, J.: A graphical user interface for formal proofs in geometry. *J. Autom. Reasoning* **39**, 161–180 (2007)
4. Botana, F., Valcarce, J.: A dynamic-symbolic interface for geometric theorem discovery. *Comput. Educ.* **38**, 21–35 (2002)

5. Ye, Z., Chou, S.-C., Gao, X.-S.: An introduction to Java Geometry Expert. In: Sturm, T., Zengler, C. (eds.) ADG 2008. LNCS, vol. 6301, pp. 189–195. Springer, Heidelberg (2011)
6. Wang, D.: GEOTHER 1.1: handling and proving geometric theorems automatically. In: Winkler, F. (ed.) ADG 2002. LNCS (LNAI), vol. 2930, pp. 194–215. Springer, Heidelberg (2004)
7. Petrović, I., Janičić, P.: Integration of OpenGeoProver with GeoGebra (2012). <http://argo.matf.bg.ac.rs/events/2012/fatpa2012/slides/IvanPetrovic.pdf>
8. Janičić, P.: Challenges for the next generation mathematics education software. Keynote Presentation at the CADGME 2014 Conference, Halle (Saale), Germany (2014). <http://cadgme2014.ceremat.org/node/79?width=640&height=450>
9. Quaresma, P., Janičić, P.: GeoThms – a web system for Euclidean constructive geometry. In: Proceedings of the 7th Workshop on User Interfaces for Theorem Provers (UITP 2006). Electronic Notes in Theoretical Computer Science, vol. 174, pp. 35–48 (2007)
10. Pham, T.-M., Bertot, Y., Narboux, J.: A Coq-based library for interactive and automated theorem proving in plane geometry. In: Murgante, B., Gervasi, O., Iglesias, A., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2011, Part IV. LNCS, vol. 6785, pp. 368–383. Springer, Heidelberg (2011). http://dx.doi.org/10.1007/978-3-642-21898-9_32
11. Wikipedia: Coq – Wikipedia, the free encyclopedia (2014). <http://en.wikipedia.org/w/index.php?title=Coq>
12. Botana, F., Kovács, Z., Weitzhofer, S.: Implementing theorem proving in GeoGebra by using a Singular webservice. In: Proceedings EACA 2012, Libro de Resúmenes del XIII Encuentro de Álgebra Computacional y Aplicaciones. Universidad de Alcalá, pp. 67–70 (2012)
13. Kovács, S., Recio, T., Weitzhofer, S.: Implementing theorem proving in GeoGebra by exact check of a statement in a bounded number of test cases. In: Proceedings EACA 2012, Libro de Resúmenes del XIII Encuentro de Álgebra Computacional y Aplicaciones. Universidad de Alcalá, pp. 123–126 (2012)
14. Petrović, I., Kovács, Z., Weitzhofer, S., Hohenwarter, M., Janičić, P.: Extending GeoGebra with automated theorem proving by using OpenGeoProver. Presentation at the CADGME 2012 Conference in Novi Sad, Serbia (2012). <http://ggb1.idm.jku.at/~kovzol/talks/cadgme12/06/06.pdf>
15. Recio, T.: Dynamic Geometry and Mathematics: few trains on a two-way track. Keynote Presentation at the CADGME 2014 Conference, Halle (Saale), Germany (2014). <http://cadgme2014.ceremat.org/node/82?width=640&height=450>
16. Botana, F., Hohenwarter, M., Janičić, P., Kovács, Z., Petrović, I., Recio, T., Weitzhofer, S.: Automated theorem proving in GeoGebra: current achievements. *J. Autom. Reasoning* **55**, 39–59 (2015). Kindly check and confirm the edit made in Ref. [16]
17. Ye, Z., Chou, S., Gao, X.: Visually dynamic presentation of proofs in plane geometry, part 1. Basic features and the manual input method. *J. Autom. Reasoning* **45**, 213–241 (2010)
18. Ye, Z., Chou, S., Gao, X.: Visually dynamic presentation of proofs in plane geometry, part 2. Automated generation of visually dynamic presentations with the full-angle method and the deductive database method. *J. Autom. Reasoning* **45**, 243–266 (2010)
19. Kapur, D.: Using Gröbner bases to reason about geometry problems. *J. Symbolic Comput.* **2**, 399–408 (1986). <http://www.sciencedirect.com/science/article/pii/S0747717186800074>

20. Wu, W.T.: On the decision problem and the mechanization of theorem proving in elementary geometry. *Sci. Sinica* **21**, 157–179 (1978)
21. Janičić, P., Narboux, J., Quaresma, P.: The area method: a recapitulation. *J. Autom. Reasoning* **48**, 489–532 (2012)
22. Chou, S., Gao, X., Zhang, J.: A deductive database approach to automated geometry theorem proving and discovering. *J. Autom. Reasoning* **25**, 219–246 (2000)
23. Desfontaines, D.: Theorem proving in GeoGebra: implementing the area method into OpenGeoProver (internship report) (2012). <http://www.eleves.ens.fr/home/desfonta/InternshipReport.pdf>
24. Baeta, N., Quaresma, P.: The full angle method on the OpenGeoProver. In: Lange, C., Aspinall, D., Carette, J., Davenport, J., Kohlhase, A., Kohlhase, M., Libbrecht, P., Quaresma, P., Rabe, F., Sojka, P., Whiteside, I., Windsteiger, W. (eds.) *MathUI, OpenMath, PLMMS and ThEdu Workshops and Work in Progress at the Conference on Intelligent Computer Mathematics, CEUR Workshop Proceedings*, number 1010, Aachen (2013)
25. Weitzhofer, S.: Mechanic proving of theorems in plane geometry. Master's thesis, Johannes Kepler University, Linz, Austria (2013). <http://test.geogebra.org/~kovzol/guests/SimonWeitzhofer/DiplArbeit.pdf>
26. Recio, T., Vélez, M.: Automatic discovery of theorems in elementary geometry. *J. Autom. Reasoning* **23**, 63–82 (1999)
27. Decker, W., Greuel, G.M., Pfister, G., Schönemann, H.: Singular 3-1-6 – A computer algebra system for polynomial computations (2012). <http://www.singular.uni-kl.de>
28. Ancsin, G., Hohenwarter, M., Kovács, Z.: GeoGebra goes web. *Electron. J. Math. Technol.* **7**, 412–418 (2013)
29. Hearn, A.C.: REDUCE User's Manual Version 3.8 (2004). <http://reduce-algebra.com/docs/reduce.pdf>
30. Parrisé, B.: Giac/Xcas, a free computer algebra system (2013). <http://www-fourier.ujf-grenoble.fr/~parrisé/giac.html>
31. Kovács, Z., Parrisé, B.: Giac and GeoGebra – improved Gröbner basis computations, Special semester on applications of algebra and number theory, workshop 3 on computer algebra and polynomials (2013). <https://www.ricam.oeaw.ac.at/specsem/specsem2013/workshop3/slides/parrisé-kovacs.pdf>
32. Kovács, Z., Parrisé, B.: Giac and GeoGebra – improved Gröbner basis computations. In: Gutierrez, J., Schicho, J., Weimann, M. (eds.) *Computer Algebra and Polynomials*. LNCS, vol. 8942, pp. 126–138. Springer, Heidelberg (2015)
33. Zakai, A.: Emscripten: an LLVM-to-JavaScript compiler (2013). <https://github.com/kripken/emscripten/blob/master/docs/paper.pdf?raw=true>

Computer Theorem Proving for Verifiable Solving of Geometric Construction Problems

Vesna Marinković¹ (✉), Predrag Janičić¹, and Pascal Schreck²

¹ Faculty of Mathematics, University of Belgrade, Belgrade, Serbia
vesnap@matf.bg.ac.rs

² ICube, UMR CNRS 7357, Université de Strasbourg, Strasbourg, France

Abstract. Over the last sixty years, a number of methods for automated theorem proving in geometry, especially Euclidean geometry, have been developed. Almost all of them focus on universally quantified theorems. On the other hand, there are only few studies about logical approaches to geometric constructions. Consequently, automated proving of $\forall\exists$ theorems, that correspond to geometric construction problems, have seldom been studied. In this paper, we present a formal logical framework describing the traditional four phases process of geometric construction solving. It leads to automated production of constructions with corresponding human readable correctness proofs. To our knowledge, this is the first study in that direction. In this paper we also discuss algebraic approaches for solving ruler-and-compass construction problems. There are famous problems showing that it is often difficult to prove non-existence of constructible solutions for some tasks. We show how to put into practice well-known algebra-based methods and, in particular, field theory, to prove RC-constructibility in the case of problems from Wernick's list.

1 Introduction

In spite of a long tradition of straightedge and compass constructions,¹ automation and mechanization of *solving geometric construction problems* has been barely touched in computer science. As far as we know, except for works described in [5, 14, 21, 24, 26], for a geometric treatment, and in [6, 10], for an algebraic point of view, results the closest to this subject are about geometric constraint solving in CAD [1, 4, 8, 23] (see [16] for a recent survey), but there the challenges are quite different [25, 26]. Moreover, most of the existing methods for solving construction problems do not consider proving correctness of solutions. For instance, the method designed by Gulwani [14] is derived from general methods for testing and synthesizing pieces of software. It finds a formal construction by using a probabilistic approach of having a particular solution which serves to guide a

¹ The English word *ruler* designates a tool with measurement in opposition to *straight-edge*. In this paper, however, we will conform to the habits and use the terms *ruler-and-compass constructibility* or resolvability, in short RC-constructibility or RC-resolvability, for straightedge and compass constructibility or resolvability.

search in a big space of formal functional terms. For such a method, a proof of correctness is really needed.

Some studies about the foundations of geometry also consider geometric constructions in order to define constructive geometry through the elimination of quantifiers and the use of functional symbols (see [30,31] for instance). But the contexts are very different: we here consider RC-constructions within the classical Euclidean plane, while in constructive geometry the aim is to define a logical framework where all considered objects correspond to ground functional terms (or in other words, the ground set of its initial model is the set of points RC-constructible from $\{O, I\}$) where O is point with coordinates $(0, 0)$ and I , $(1, 0)$.

This limited interest is even more surprising given that solving geometric construction problems links two important fields of computer science applied to geometry:

- automated theorem proving, with geometry as one of the most successful domains since the seminal work by Gelernter [11];
- dynamic geometry software, which provoked huge changes in educational practices in geometry by effective visualizations and rewarding experimentations.

Both these fields have deep connections with construction problems. A lot of methods for automated theorem proving in geometry rely on basic geometric constructions, and constructions performed within dynamic geometry tools typically correspond to RC-constructions. Still, links between automated solving of construction problems with automated theorem proving or dynamic geometry software have been hardly explored.

From the logical point of view, solving a geometric construction problem requires proving a theorem of the form $\forall X \exists Y. \Psi(X; Y)$ in an intuitionistic way. The witness for Y that is found during such proof represents a *construction of a solution* and must involve only points that are RC-constructible from the set X . In other words, the task is, given a declarative specification of the required figure $\Psi(X; Y)$, to provide a corresponding – possibly equivalent – procedural specification based on available construction steps. Both directions of this equivalence have to be proved as we will discuss in more details in this paper.

This transformation of a declarative statement into a procedural specification within a formal framework relies on formalization of the tools allowed to perform the construction. Usually, ruler and compass are considered and operations like intersection of lines and circles can be used. However, the folklore of geometric constructions also considers many variations, for instance, by forbidding either ruler or compass, by restricting operations by some tools (for example, collapsible compass or blocked compass), or by allowing new tools like tool for tracing the MacLaurin trisectrix or Origami folds. Some of these sets of tools are equivalent to ruler and compass, while MacLaurin trisectrix and Origami folds are more powerful. In this paper, we restrict ourselves to RC-constructibility which definition is recalled here:

Definition 1. *Given a finite set of points $\mathcal{B} = \{B_0, \dots, B_m\}$ in the Euclidean plane, a point P is RC-constructible from the set \mathcal{B} if there is a finite set of*

points $\{P_0, \dots, P_n\}$ such that $P = P_n$, $P_0 \in \mathcal{B}$ and every point P_i ($1 \leq i \leq n$) is either a point from \mathcal{B} or is obtained as the intersection of two lines, or of a line and a circle, or of two circles, themselves obtained as follows:

- any considered circle has its center in the set $\{P_0, \dots, P_{i-1}\}$ and its radius is equal to the distance $P_j P_k$ for some $j < i$ and $k < i$;
- any considered line passes through two points from the set $\{P_0, \dots, P_{i-1}\}$.

For problems involving parameters, the *solution* is in fact a way to construct all solutions. Already in the early 40s, Lebesgue was calling this a *program of construction* [18].

It is important to note that sometimes there is no solution for a given construction problem. The absence of solutions does not necessarily mean that there is no solutions in the Euclidean plane, but no solution of the problem can be constructed using only ruler and compass. Examples are famous classical problems like the circle quadrature problem. It is often difficult to prove such impossibility theorems. Let us note that it suffices to find a counter-example to prove that a problem is RC-unconstructible.

In this paper, we will focus on triangle construction problems and one particular corpus of such problems – Wernick’s list [29]. This corpus consists of triangle construction problems with located points, and for each, the task is, given some points X to construct a triangle ABC such that the points X meet the condition $\Psi(X; A, B, C)$. We first discuss a geometrical approach for solving construction problems, with classical “four-phases solutions” and then algebraic approach, both for proving constructibility and unconstructibility. Within the geometry part, we will comment on our wider project: automation of the solving process, accompanied by machine verifiable proofs.

2 Geometrical Approach

In this section we give a rigorous, first-order logic description of classical form of solutions of construction problems. To our knowledge, surprisingly, this is the first such description, despite the fact that construction problems have been around for two and a half millennia. Our rigorous description serves as a basis for a semi-automatic methodology for solving construction problems and generating their solutions, supported by automated theorem provers and formal proofs within proof assistants. To our knowledge, automated and formal proving in the context of automated solving of construction problems were never treated so far.

2.1 Main Conjecture in a Problem Solution

As said above, for a construction problem, roughly said, the task is to prove constructively a theorem of the following form (where X and Y denote vectors of objects – points, lines, rays, etc.):

$$\forall X \exists Y. \Psi(X; Y) \tag{1}$$

The above subsumes two claims: that the problem is solvable and that a particular construction (that witnesses $\exists Y.\Psi(X; Y)$) is correct.

Within the problem description, there could be given some constraints imposed on the given objects X . Some construction problems do not have solutions and some construction problems have solutions only under some additional conditions, not known in advance. So, instead of (1), one typically has to discover $\Phi(X)$ (for the given $\Psi(X; Y)$) and to prove:

$$\forall X.(\Phi(X) \Rightarrow \exists Y.\Psi(X; Y)) \quad (2)$$

The above claims that solution exists under some conditions. But one may claim even more:

$$\forall X.(\Phi(X) \Rightarrow \exists Y.\Psi(X; Y) \wedge \neg\Phi(X) \Rightarrow \neg\exists Y.\Psi(X; Y)) \quad (3)$$

The above gives a complete characterization of resolvability: it states that solution exists under some conditions Φ and solution does not exist otherwise. The problem is that conditions for resolvability often cannot be expressed only in terms of the given objects X , but have to involve some auxiliary objects (used within the construction).

In solving specific classes of construction problems, some goal conditions may be assumed. For instance, in solving triangle construction problems, an implicit goal condition is that the constructed points A , B , and C are not collinear.

2.2 Constructible Cases and Four-Phases Solutions

Before going to theory, let us bring our esteemed readers back to school and remind them that traditionally, finding a solution of a construction problem passes through the following four phases [7]:

Analysis: One starts from the assumption that certain geometrical objects satisfy the conditions of the problem $\Psi(X; Y)$ (see Sect. 2.1) and proves properties $Plans(X; Y)$ that enable construction (geometric loci and theorems are used for producing candidates for solutions).

Construction: A construction is based on the analysis, that is, on the procedural (ruler-and-compass) counterpart $Plans(X; Y)$ to the specification $\Psi(X; Y)$.

Proof: It has to be proved that, the constructed figure meets the conditions $\Psi(X; Y)$ (possibly under some preconditions).

Discussion: The discussion should state sufficient and necessary conditions for solutions to exist, and should also consider how many possible solutions to the problem there exist. Ideally, the number of solutions should be expressed effectively in the function of mutual relations of the given elements, but sometimes it is sufficient to express it implicitly in the function of the relation of the figures obtained during the construction.

In previous works on geometric construction or geometric constraint solving, the first two phases are often set in the foreground, while the last two are hardly mentioned (while they are seldom easy to achieve).

In the following text, we will give an account of all solution phases while, for illustrating them (in Examples 1–4), we will use one running example (the problem 4 from Wernick’s list): *Given points A , B , and G , construct a triangle ABC , such that G is the centroid of ABC .* For this problem, $\Psi(X; Y)$ is $\neg\text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)$, i.e., the task is to prove:

$$\forall A, B, G. (? \Leftrightarrow \exists C. (\neg\text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)))$$

where $\text{centroid}(G, A, B, C)$ states that G is the centroid of the triangle ABC and $?$ is a condition, not known in advance that characterizes resolvability of the problem.

Analysis. The purpose of analysis is to detect knowledge sufficient for a procedural specification $\text{Plans}(X; Y')$ for a given declarative specification $\Psi(X; Y)$. More precisely, analysis consists of a sequence of proofs of statements of the following form, for $k = 1, \dots, n$:²

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X; Y) \wedge \text{Def}(X; Y' \setminus Y) \wedge \bigwedge_{i=1}^{k-1} \text{Rel}_i(X; Y'_i) \Rightarrow \text{Rel}_k(X; Y'_k)) \quad (4)$$

where:

- Y' is the sequence of variables y_1, \dots, y_n such that $Y \subseteq Y'$ (informally, $Y' \setminus Y$ are auxiliary points to be constructed and used in the construction, along the objects from Y);
- Y'_k is the sequence of variables y_1, \dots, y_k ;
- y_n belongs to Y ;
- $\Phi_a(X)$ represents some constraints on the given objects (possibly \top , if there are no constraints);
- $\text{Def}(X; Y' \setminus Y)$ introduces properties of $Y' \setminus Y$;
- $\text{Rel}_k(X; Y'_k)$ is a formula that corresponds to an effective way for constructing y_k by ruler and compass using X and Y'_{k-1} .³

Let us denote $\bigwedge_{i=1}^n \text{Rel}_i(X; Y'_i)$ by $\text{Cons}(X; Y')$. From the above sequence of theorems, the following theorem follows:

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X; Y) \wedge \text{Def}(X; Y' \setminus Y) \Rightarrow \text{Cons}(X; Y')) \quad (5)$$

In order to enable a construction as an effective procedure, it is needed to turn implicit relationship $\text{Rel}_i(X; Y'_i)$ (for $i = 1$ to n) into the form $\bigvee_{k=1}^{K_i} y_i = \text{RC}_{i,k}(X; Y'_i)$ [9, 25], which expresses the way(s) in which y_i can be obtained

² In later stages of the solution, the given condition $\Phi_a(X)$ may be extended to some condition Φ for which (3) holds.

³ This formula may involve disjunctions corresponding to different “cases” for X and Y . For instance, $(A \neq B \wedge \text{midpoint}(C, A, B)) \vee (A = B \wedge C = A)$.

from X and Y'_i using ruler and compass.⁴ Here, K_i denotes a number of different *ways* in which y_i can be constructed. This number has to be finite, although some *ways* may allow infinite choices. For example, it may be the case that y_i is the intersection point of two lines p and q or an arbitrary point on the line r . It must hold:

$$\forall X, Y'. (Rel_i(X; Y'_i) \Leftrightarrow \bigvee_{k=1}^{K_i} y_i = RC_{i,k}(X; Y'_i)) \tag{6}$$

Since $Cons(X; Y')$ denotes $\bigwedge_{i=1}^n Rel_i(X; Y'_i)$, it also holds:

$$\forall X, Y'. (Cons(X; Y') \Leftrightarrow \bigwedge_{i=1}^n \bigvee_{k=1}^{K_i} y_i = RC_{i,k}(X; Y'_i)) \tag{7}$$

If we denote by $Plan_j(X; Y')$ the conjunction $\bigwedge_{i=1}^n y_i = RC_{i,k_i}(X; Y'_i)$, for some $k \in \{1, \dots, K_i\}$ for each $i = 1, \dots, n$ then, by distributivity, we obtain some J disjuncts as individual construction plans:

$$\forall X, Y'. (Cons(X; Y') \Leftrightarrow \bigvee_{j=1}^J Plan_j(X; Y')) \tag{8}$$

If we denote $\bigvee_{j=1}^J Plan_j(X; Y')$ by $Plans(X; Y')$, from the above formula and (5), we have:

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X; Y) \wedge Def(X; Y' \setminus Y) \Rightarrow Plans(X; Y')) \tag{9}$$

Since we are interested in effective constructions of solutions expressed by $Plans(X; Y')$, and if we are careful to introduce only needed auxiliary objects in $Y' \setminus Y$, then it is necessary that they can be defined for every solution. Expressing this obligation for Def , we have the following requirement:

$$\forall X, Y. (\Phi_a(X) \wedge \Psi(X; Y) \Rightarrow \exists Y' \setminus Y. Def(X; Y' \setminus Y)) \tag{10}$$

There is a subtle issue with $Plans(X; Y')$ – it has to be precise enough to enable the construction, but also it has to be strong enough to prove that the constructed figure indeed meets the specification.

Because of the specific goal, the analysis is more a search process than a proving process. It can be implemented as a search process, while at the end, it can produce a required formula.

⁴ Strictly speaking, functions $RC_{i,k}$ may involve more than only ruler and compass. For instance, it may be the case that only one intersection point of two circles can be picked (e.g. “that is different from the point...”, “that is not on the same side...”, etc.). Also, some of $RC_{i,k}$ may be non-deterministic, for instance “pick a point on the line ...”.

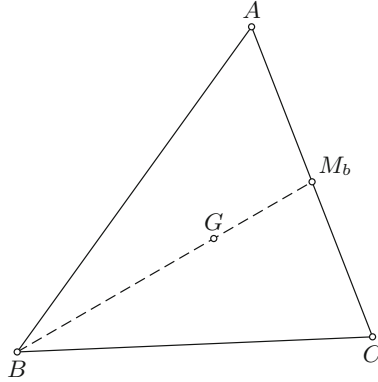


Fig. 1. Illustration for the solution for the running example

Example 1. Let $sratio(P, Q, R, S, m, n)$ denote that the ratio of parallel vectors \overrightarrow{PQ} and \overrightarrow{RS} is equal to the rational number m/n , meaning: $\overrightarrow{PQ}/\overrightarrow{RS} = m/n$. Let $midpoint(P, Q, R)$ denote that P is the midpoint of the segment QR . The first derivation step for our running example is (Fig. 1):

$$\begin{aligned} &\forall A, B, G, M_b, C. \\ &(\neg collinear(A, B, C) \wedge centroid(G, A, B, C) \wedge midpoint(M_b, A, C)) \\ &\Rightarrow sratio(B, M_b, B, G, 3, 2) \end{aligned}$$

and the second derivation step is:

$$\begin{aligned} &\forall A, B, G, M_b, C. \\ &(\neg collinear(A, B, C) \wedge centroid(G, A, B, C) \wedge midpoint(M_b, A, C)) \\ &\wedge sratio(B, M_b, B, G, 3, 2) \\ &\Rightarrow sratio(A, C, A, M_b, 2, 1) \end{aligned}$$

These two steps combined give:

$$\begin{aligned} &\forall A, B, G, M_b, C. \\ &(\neg collinear(A, B, C) \wedge centroid(G, A, B, C) \wedge midpoint(M_b, A, C)) \\ &\Rightarrow sratio(B, M_b, B, G, 3, 2) \wedge sratio(A, C, A, M_b, 2, 1) \end{aligned}$$

Here, $\Phi_a(A, B, G)$ is (there may be some preconditions added within the proof phase) \top (meaning that there are no constraints on A, B, G), $Def(A, B, C; M_b)$ is $midpoint(M_b, A, C)$, and $Cons(A, B, G; M_b, C)$ is $sratio(B, M_b, B, G, 3, 2) \wedge sratio(A, C, A, M_b, 2, 1)$.

Let $sratioF$ be a partial function such that $Q = sratioF(P, R, S, m, n)$ if $\overrightarrow{PQ}/\overrightarrow{RS} = m/n$. In our running example, there are no different ways for

construction, so each K_i equals 1. Then the following holds:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (sratio(B, M_b, B, G, 3, 2) \wedge sratio(A, C, A, M_b, 2, 1)) \\ & \Rightarrow (M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1)) \end{aligned}$$

and the direct consequence of the previous two formulae is:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg collinear(A, B, C) \wedge centroid(G, A, B, C) \wedge midpoint(M_b, A, C)) \quad (11) \\ & \Rightarrow M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \end{aligned}$$

Following the formula (10), we get the following conjecture that is proved easily:

$$\begin{aligned} & \forall A, B, C, G. \\ & (\neg collinear(A, B, C) \wedge centroid(G, A, B, C)) \quad (12) \\ & \Rightarrow \exists M_b.(midpoint(M_b, A, C)) \end{aligned}$$

Construction. The analysis yields the formula enabling effective constructions. For each $j \in \{1, \dots, J\}$, $Plan_j(X; Y')$ yields one *construction plan* of the form:

- Objects X are given (as *free* objects);
- For $i = 1$ to n construct y_i as $y_i = RC_{i,k}(X; Y'_i)$ (for some $k \in \{1, \dots, K_i\}$).

Compound construction steps can also be used (say, construction of the midpoint) so it should be proved that each of $RC_{i,k}$ is expressible using ruler and compass.

Example 2. The construction, derived from the formula $sratio(B, M_b, B, G, 3, 2) \wedge sratio(A, C, A, M_b, 2, 1)$ is as follows:

- The points A, B, G are given (as *free* points);
- $M_b = sratioF(B, B, G, 3, 2)$;
- $C = sratioF(A, A, M_b, 2, 1)$.

Proof. Within the proof phase, we have to prove correctness for each construction plan $Plan_j(X; Y')$. We have to prove:

$$\forall X.Y'.(\Phi_a(X) \wedge ? \wedge Plan_j(X; Y') \Rightarrow \Psi(X; Y)) \quad (13)$$

where ? is some condition still to be discovered.

Automated theorem provers for geometry typically handle procedural representations of a geometric construction and can deal with conjectures of the above form. We first try to prove the conjecture:

$$\forall X.Y'.(\Phi_a(X) \wedge Plan_j(X; Y') \Rightarrow \Psi(X; Y)) \quad (14)$$

If the conjecture is proved by the prover, the prover may return also some non-degeneracy (NDG) conditions $NDG(X; Y')$ – conditions under which the conjecture holds (note that in general case these conditions are not necessarily the weakest conditions under which the conjecture holds) so only a weaker statement is proved:

$$\forall X, Y'. (\Phi_a(X) \wedge NDG(X; Y') \wedge Plan_j(X; Y') \Rightarrow \Psi(X; Y)) \quad (15)$$

Example 3. We want to prove:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (? \wedge M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \\ & \Rightarrow centroid(G, A, B, C) \wedge \neg collinear(A, B, C)) \end{aligned}$$

where ? denotes some set of conditions expressed only in terms of points A , B and G , still to be discovered.

Let us first focus on the $centroid(G, A, B, C)$ part. Let us suppose that our theorem provers support $sratioF$, but does not support $centroid$ and that we have the following definition of centroid:

$$\begin{aligned} & \forall A, B, C, M_a, M_b. \\ & (M_a = sratioF(B, B, C, 1, 2) \wedge M_b = sratioF(C, C, A, 1, 2) \wedge \\ & G = intersec(AM_a, BM_b) \Rightarrow centroid(G, A, B, C)) \end{aligned}$$

We can pass the following conjecture to an automatic (e.g., algebraic) prover:

$$\begin{aligned} & \forall A, B, G, M_b, C, M'_a, M'_b, G'. \\ & (M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \wedge \\ & M'_a = sratioF(B, B, C, 1, 2) \wedge M'_b = sratioF(C, C, A, 1, 2) \wedge \\ & G' = intersec(AM'_a, BM'_b) \Rightarrow G = G') \end{aligned}$$

For instance, the above theorem is proved by the prover based on Wu's method, implemented within OpenGeoProver [20]. The prover proves the above conjecture, but returns NDG conditions “line AM'_a is not parallel with line BM'_b , and points A and M'_a are not identical” ($\neg parallel(AM'_a, BM'_b) \wedge A \neq M'_a$), so we actually proved:

$$\begin{aligned} & \forall A, B, G, M_b, M'_a, M'_b, C. \\ & \neg parallel(AM'_a, BM'_b) \wedge A \neq M'_a \wedge \\ & M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \wedge \\ & M'_a = sratioF(B, B, C, 1, 2) \wedge M'_b = sratioF(C, C, A, 1, 2) \\ & \Rightarrow centroid(G, A, B, C) \end{aligned}$$

We also need to prove that points A , B and C are not collinear (under some conditions). It is easily proved (by the ArgoCLP prover [28]) that:

$$\forall A, B, C, G, M_b.$$

$$(collinear(A, B, C) \wedge M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \Rightarrow collinear(A, B, G))$$

and its contraposition gives:

$$\forall A, B, C, G, M_b.$$

$$(\neg collinear(A, B, G) \wedge M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \Rightarrow \neg collinear(A, B, C))$$

The formula $collinear(A, B, G)$ was discovered by trying finite number of predicates over the given points A , B , and G .

From the previous theorems, we have:

$$\begin{aligned} &\forall A, B, G, M_b, M'_a, M'_b, C. \\ &(\neg parallel(AM'_a, BM'_b) \wedge A \neq M'_a \wedge \neg collinear(A, B, G) \wedge \\ &M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \wedge \\ &M'_a = sratioF(B, B, C, 1, 2) \wedge M'_b = sratioF(C, C, A, 1, 2) \quad (16) \\ &\Rightarrow centroid(G, A, B, C) \wedge \neg collinear(A, B, C)) \end{aligned}$$

Discussion. Recall that, in general, we want to state sufficient and necessary conditions for a solution to exist (and to count the number of solutions). Ideally, within discussion we should prove a statement of the form (3). For simplicity, we will assume that $Plans(X; Y')$ has only one disjunct (i.e., only one construction plan), but the following consideration can be easily generalized for cases with more than one disjunct.

From the analysis we have:

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X; Y) \wedge Def(X; Y' \setminus Y) \Rightarrow Plans(X; Y')) \quad (17)$$

but also:

$$\forall X. (\exists Y'. (\Phi_a(X) \wedge \Psi(X; Y) \wedge Def(X; Y' \setminus Y)) \Rightarrow \exists Y'. Plans(X; Y')) \quad (18)$$

From the above formula and from (10) we get:

$$\forall X. (\exists Y. (\Phi_a(X) \wedge \Psi(X; Y)) \Rightarrow \exists Y'. Plans(X; Y')) \quad (19)$$

and

$$\forall X. (\Phi_a(X) \Rightarrow (\exists Y. \Psi(X; Y) \Rightarrow \exists Y'. Plans(X; Y'))) \quad (20)$$

On the other hand, from the proof we have:

$$\forall X, Y'. (\Phi_a(X) \wedge NDG(X; Y') \wedge Plans(X; Y') \Rightarrow \Psi(X; Y)) \quad (21)$$

and hence:

$$\forall X.(\exists Y'.(\Phi_a(X) \wedge NDG(X; Y') \wedge Plans(X; Y')) \Rightarrow \exists Y.\Psi(X; Y)) \quad (22)$$

and also:

$$\forall X.(\Phi_a(X) \Rightarrow (\exists Y'.(NDG(X; Y') \wedge Plans(X; Y')) \Rightarrow \exists Y.\Psi(X; Y))) \quad (23)$$

Therefore, we have necessary (20) and sufficient (23) conditions for $\exists Y.\Psi(X; Y)$ (under the preconditions $\Phi_a(X)$). However, they are not equal, so we still don't have a complete characterization of solvability. We can try to discover⁵ a formula $\Phi_d(X)$ such that

$$\forall X.(\Phi_a(X) \Rightarrow (\Phi_d(X) \Rightarrow \exists Y'(NDG(X; Y') \wedge Plans(X; Y')))) \quad (24)$$

If it holds that

$$\forall X.(\Phi_a(X) \Rightarrow (\exists Y.\Psi(X; Y) \Rightarrow \Phi_d(X))) \quad (25)$$

then $\Phi_a(X) \wedge \Phi_d(X)$ is the required formula $\Phi(X)$, and we finally have the theorem (3).

Note that in some cases we can discharge some of conjuncts $ndg(X; Y')$ of $NDG(X; Y')$. For instance, one conjunct may imply an other one, so the latter can be omitted. Also, if:

$$\forall X, Y'.(Plans(X; Y') \Rightarrow ndg(X; Y')) \quad (26)$$

then we can eliminate $ndg(X; Y')$ from $NDG(X; Y')$ in (21). In some cases, this way we can eliminate all of $NDG(X)$, and then $\Phi(X)$ can be equal \top .

In some cases, Φ_d involves also some Y' , but here we consider only a simple case. In addition, as history teaches us, in some cases this cannot be done using only means of synthetic geometry. (For some unsolvable problems, synthetic approach can be used using reduction, as discussed in Sect. 2.3).

The above gives a characterization of solvability. Concerning the number of solutions, in solvable case the number of solutions is the product of n numbers of possible choices for each of y_i (see Analysis).

Example 4. From the formula (11) from the analysis we have:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg collinear(A, B, C) \wedge centroid(G, A, B, C) \wedge midpoint(M_b, A, C)) \\ & \Rightarrow M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1)) \end{aligned}$$

and therefore it follows:

$$\begin{aligned} & \forall A, B, G. \\ & \exists M_b, C.(\neg collinear(A, B, C) \wedge centroid(G, A, B, C) \wedge midpoint(M_b, A, C)) \\ & \Rightarrow \exists M_b, C.(M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \\ & \wedge \neg collinear(A, B, C)) \end{aligned}$$

⁵ One can try a finite number of predicates over X .

and, thanks to (12):

$$\begin{aligned}
& \forall A, B, G. \\
& \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \\
& \Rightarrow \exists M_b, C. (M_b = \text{sratio}F(B, B, G, 3, 2) \wedge C = \text{sratio}F(A, A, M_b, 2, 1) \\
& \wedge \neg \text{collinear}(A, B, C)) \quad (27)
\end{aligned}$$

One can easily prove the lemma:

$$\begin{aligned}
& \forall A, B, G, M_b, C. \\
& (M_b = \text{sratio}F(B, B, G, 3, 2) \wedge C = \text{sratio}F(A, A, M_b, 2, 1) \\
& \wedge \neg \text{collinear}(A, B, C)) \\
& \Rightarrow \neg \text{collinear}(A, B, G) \quad (28)
\end{aligned}$$

For the choice $\Phi_d(A, B, G) = \neg \text{collinear}(A, B, G)$, using formulae (27) and (28) we can prove the following formula:

$$\begin{aligned}
& \forall A, B, G. \\
& \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \Rightarrow \neg \text{collinear}(A, B, G) \quad (29)
\end{aligned}$$

On the other hand, from the formula (16) from the proof, using the following lemma:

$$\begin{aligned}
& \forall A, B, C, G, M'_a, M'_b. \\
& (\neg \text{collinear}(A, B, C) \wedge M'_a = \text{sratio}F(B, B, C, 1, 2) \wedge M'_b = \text{sratio}F(C, C, A, 1, 2) \\
& \Rightarrow \neg \text{parallel}(AM'_a, BM'_b) \wedge A \neq M'_a)
\end{aligned}$$

and the lemma:

$$\begin{aligned}
& \forall A, B, G, M_b, C. \\
& (\neg \text{collinear}(A, B, G) \wedge M_b = \text{sratio}F(B, B, G, 3, 2) \wedge C = \text{sratio}F(A, A, M_b, 2, 1) \\
& \Rightarrow \neg \text{collinear}(A, B, C))
\end{aligned}$$

we obtain:

$$\begin{aligned}
& \forall A, B, G, M_b, C. \\
& (M_b = \text{sratio}F(B, B, G, 3, 2) \wedge C = \text{sratio}F(A, A, M_b, 2, 1) \wedge \neg \text{collinear}(A, B, G) \\
& \Rightarrow \neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C))
\end{aligned}$$

and also:

$$\begin{aligned}
& \forall A, B, G. \\
& \exists M_b, C. (M_b = \text{sratio}F(B, B, G, 3, 2) \wedge C = \text{sratio}F(A, A, M_b, 2, 1) \wedge \\
& \neg \text{collinear}(A, B, G)) \\
& \Rightarrow \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C))
\end{aligned}$$

From the above theorem, and the theorem:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & \neg \text{collinear}(A, B, G) \\ & \Rightarrow \exists M_b, C. (M_b = \text{sratio}F(B, B, G, 3, 2) \wedge C = \text{sratio}F(A, A, M_b, 2, 1)) \end{aligned}$$

we obtain:

$$\begin{aligned} & \forall A, B, G. \\ & \neg \text{collinear}(A, B, G) \Rightarrow \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \quad (30) \end{aligned}$$

which is the second direction of the statement we want to prove.

Therefore, from (29) and (30) we have proved:

$$\begin{aligned} & \forall A, B, G. \\ & \neg \text{collinear}(A, B, G) \Leftrightarrow \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \end{aligned}$$

All proofs from the above example have been fully formalized within the proof assistant Isabelle. The full proof document, consisting of all needed geometry statements used as lemmas (corresponding to simple theorems in Euclidean geometry) and all proofs, has around 1200 lines.⁶ We expect that significant portions of such developments would be shared among correctness proofs for different construction problems.

2.3 Unconstructible Cases and Reduction

Using geometrical means, it can be proved that a figure is RC-unconstructible (i) if the specification is inconsistent (so there is no required figure in Euclidean plane, no matter how it can be produced); (ii) if the problem can be reduced to another problem (typically, some *canonical* RC-unconstructible problem), known to be unsolvable using ruler and compass. Here is a simple example of the latter approach, based on one Archimedes's construction.

Example 5. Given three non-collinear points A , B and O , construct points X and Y such that (see Fig. 2):

- $OX \cong OB$,
- $XY \cong OB$,
- points B , X and Y are collinear, and
- points A , O and Y are collinear.

Using elementary geometry it can be proved that the angle $\alpha = \angle AOB$ is three times angle $\beta = \angle AYB$. Thus, if this problem is RC-solvable, so is the trisection of an angle. But it is well known that in general one cannot divide an angle in three using only straightedge and compass.

⁶ All proofs can be found here: http://www.matf.bg.ac.rs/~vesnap/adg2014_wernick6.thy.

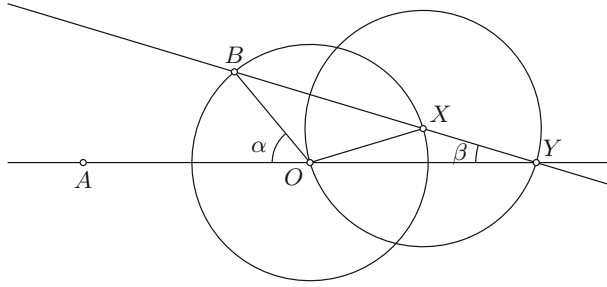


Fig. 2. Example of a RC-unconstructible problem

2.4 Mechanization

Mechanization of the solving process described in Sect. 2.2 is the subject of our current work. Our ultimate goal is producing machine verifiable (within the proof assistant Isabelle [22]) solutions for construction problems from Wernick's list. This complex task requires synergy of a tool for solving construction problems, algebraic automated theorem provers, synthetic automated theorem provers, and proof assistants, aided by some human's guidance (and dynamic geometry tools for visualization). In this section we report on the current state.

Analysis. The analysis is performed by our ArgoTriCS tool for solving construction problems [21]. Based on a small number of definitions, lemmas and construction steps, it can solve almost all solvable problems from Wernick's list (66 out of 72). In addition, it can detect if the problem is redundant or locus dependent, and in these cases a point belonging to the geometric loci of points is chosen arbitrarily and construction continues. Also, the problem is tested for symmetry to some of the previous problems according to the available definitions and lemmas. A solution trace from ArgoTriCS (which also contains a subset of definitions, lemmas and construction steps needed) is translated into a sequence of theorems (4) and the theorem (5). These conjectures (along with relevant axioms/lemmas) are fed to our coherent-logic based automated theorem prover ArgoCLP that is capable of producing machine verifiable proofs [28]. At the moment, ArgoTriCS can automatically produce input files for ArgoCLP (consisting of the set of relevant axioms, the set of relevant lemmas and the theorem to be proved) only for a subset of all considered construction problems and this is the subject of the current work. Since ArgoCLP does not support functional symbols, the formula (6) and subsequent formulae have to be proved manually in Isabelle, and this is also the subject of current work. Also, it should be proved that each of used construction steps is expressible using ruler and compass but at the moment we use them all as primitive steps.

Construction. The formal description of the construction in GCLC language [15] is automatically exported from ArgoTriCS to our dynamic geometry tool

GCLC where it can be visualized and stored in a number of formats, including \LaTeX , EPS, SVG. This part of our framework is completed.

Proof. ArgoTriCS automatically exports proof specification which can be passed to two different automated theorem provers – OpenGeoProver [20] and the provers integrated into GCLC tool. These tools return as an output a proof object, and the set of NDG conditions. For a huge part of the construction problems from Wernick’s list the central theorem (of the form (14)) is successfully proved by one of these provers.

Discussion. NDG conditions obtained from the provers may involve some auxiliary objects. Statements needed for translating these conditions into ones that involve only given objects are proved using ArgoCLP. Simple consequences that do not belong to coherent logic are proved within Isabelle manually (as they cannot be proved by ArgoCLP). For attempts at discovering a sufficient and necessary conditions for solution to exist ($\neg\text{colin}(A, B, G)$ in the running example), we test a finite number of predicates over the set of given points. Also, we check if some conjunct of *NDG* implies some other one. Finally, the final proof is glued together within Isabelle by simple steps (still to be automated, currently they are performed manually).

Overall, in our formalized solutions of construction problems, there are two important gaps. One of them is the link to external algebraic provers. Conjectures proved within the proof phase are proved using algebraic provers, but there is no trusted link between them and Isabelle. Therefore, the conjectures proved by external algebraic provers are used as axioms. Currently, there are some limited formalizations of algebraic provers for geometry within proof assistants [12, 13], but not for Isabelle. For theorems proved by ArgoCLP we have formal Isabelle proofs. The second gap is between our proofs and the typical geometrical axioms (e.g., Tarski’s or Hilbert’s axioms). In our proofs we use high-level geometry lemmas as axioms.⁷ They can be proved from basic axioms (e.g., Tarski’s or Hilbert’s axiom) but it is extremely complex task and beyond the scope of this paper. Only recent advances provide (in Coq) formally proved high-level lemmas from the basic axioms [2, 3].

3 Algebraic Approach

Mathematical progress in algebra in the beginnings of the nineteenth century enabled solving of many geometric construction problems that were open since the ancient Greeks. When considering construction problems, two aspects have to be distinguished: constructibility and construction. In both cases algebraic methods can have significant role, but algebraic tools are famous for their success

⁷ The ArgoTriCS tool, along with the list of lemmas used, is available on: <http://argo.matf.bg.ac.rs/?content=downloads>.

in proving RC-unconstructibility. Actually, it is theoretically possible to extract a geometric RC-construction from a proof of RC-constructibility but it is often impractical mainly for two reasons. First, the equations to solve can have a high degree (4, 8, 16, ...) and formal methods provide huge formulae expressing the solutions and, second, even with equations with degree 2, the direct translation from algebra to geometry leads to constructions with hundreds of elementary operations. Moreover, it is often pedagogically useless because it would only mimic algebraic computations (See Fig. 3).

3.1 Classical Results

In the introduction, we recall the definition of the constructibility of points, lines and circle from a set of points B also called base points which correspond more or less to the notion of free points in dynamic geometry. We define now RC-constructibility of numbers: a number is said to be RC-constructible from the set B if it is a coordinate of a RC-constructible point. Set B has to contain at least two points, usually, point O with coordinates $(0,0)$ and point I with coordinates $(1,0)$. Without additional information, it is said that a point or a number is RC-constructible when $B = \{O, I\}$. In terms of algebra, that means that we consider polynomials with coefficients in \mathbb{Q} . For instance, any rational number is RC-constructible. But, often, parameters are added to B . This is the case with the classical problem of the impossibility of angle trisection: given $\cos(\alpha)$ for some number α , $\cos(\alpha/3)$ is RC-unconstructible in general.

A fundamental example is given by the classical operations – addition, subtraction, multiplication, division and square radical extraction – which can all be translated in terms of RC-construction. The converse is true: a number is RC-constructible from the points of set B if and only if it is expressible with the five operations operating on the coordinates of points in B . This fact is closely related to field theory which, in turn, gave a theoretical decision procedure for RC-constructibility problems [18].

Field theory allows to link numbers and polynomials: an algebraic number over a field, usually \mathbb{Q} , is a solution of a polynomial equation. A fundamental result is that to any algebraic number over K α is associated a monic irreducible polynomial $P \in K[X]$, called the minimal polynomial of P and $K(\alpha) \sim K[X]/P$. The degree of P is the degree of the extension $[K(\alpha) : K]$ which is also called the degree of α (over K). Then, the main tool for proving RC-unconstructibility lies in Wantzel's result:

Theorem 1 (Wantzel 1837). *Each RC-constructible number is algebraic over \mathbb{Q} and its degree is equal to 2^k for some $k \in \mathbb{N}$.*

This theorem can be used to prove that $\sqrt[3]{2}$ is not RC-constructible since polynomial $X^3 - 2$ is irreducible over \mathbb{Q} . But also to prove that problem 90 of Wernick's list is not RC-constructible as, for some choice for the coordinates, it is equivalent to solving the irreducible polynomial equation (obtained by using resultants and factorization within Maxima):

$$2x_A^5 + 45x_A^4 + 372x_A^3 + 1368x_A^2 + 2160x_A + 972 = 0$$

Note that the reciprocal of Wantzel's theorem is false: for instance the roots of the irreducible polynomial $X^4 + 2X - 2$ are not RC-constructible. There are several theorems of algebra which fully determine RC-constructibility:

Theorem 2 (Galois). *Let α be an algebraic number over \mathbb{Q} or an extension of \mathbb{Q} and P be its minimal polynomial; α is RC-constructible if and only if the degree of splitting field of P is a power of 2.*

This theorem has a more practical formulation:

Theorem 3. *A number α is RC-constructible if and only if there is some algebraic number $r_1, \dots, r_n = \alpha$ such that $[\mathbb{Q}(r_1) : \mathbb{Q}] = 2$, $[\mathbb{Q}(r_{i+1}) : \mathbb{Q}(r_i)] = 2$ for every $i = 1, \dots, n - 1$.*

The method proposed by Gao and Chou exploit this latter one [10].

As far as we know, there are two automatic implemented methods for deciding RC-constructibility (but, in the mathematic literature there are several papers dealing with resolution by radicals of polynomial equations, see for instance [17]). The first one comes from a book of H. Lebesgue about geometric construction problems [18] and it has been implemented by G. Chen in 1992 [6]. The second one is described in papers presented in the second ADG workshop and published in Journal of CAD [10].⁸ For the sake of completeness, we show in the next section how algebraic method can be used to prove RC-unconstructibility of a problem.

We present two examples coming from famous Wernick's list [29], and we solve them by a classical method: thanks to a Computer Algebra System (CAS), Maple⁹ in this occasion, we obtain one or more triangular systems, if possible irreducible. There are various methods to triangularize a polynomial system: one can either using successive resultants, or computing Gröbner bases with a lexicographic order, or computing the Ritt's characteristic sets. Here, we use the Maple package *RegularChains* [19] and particularly the *Triangularize* function. Then, we study the polynomial equations as polynomials with a single variable using Wantzel's theorem or Gao and Chou's method.

3.2 Unconstructible Case

Problem 122 from Wernick's list. In this problem, points G , T_a and T_b are given, and the task is to construct a triangle $T = (A, B, C)$ such that points G , T_a and T_b are respectively the centroid, the foot of the inner-bisector from A and the inner-bisector from B of T . To our knowledge, the status (constructible/unconstructible) or this problem is still unknown (one of 15 unsolved Wernick's problems).

Without loss of generality, we choose a reference system in order to fix the coordinates: let T_b have coordinates $(0, 0)$ and let T_a have coordinates $(4, 0)$. On

⁸ The technical report can be found here: <http://www.mmrc.iss.ac.cn/pub/mm15.pdf/gao.pdf>.

⁹ <http://www.maplesoft.com/>.

one hand, if we want to prove RC-constructibility, or even to produce a construction, the coordinates of G must be parameters. On the other hand, if we only want to check RC-unconstructibility, we can choose arbitrary coordinates for G . Here, we choose the coordinates $(2, 1)$ for G . If we are unlucky, it could happen that even if the problem is not RC-constructible in general, in this particular case it is. Let us see what happen here.

First, we classically translate the geometric problem in an algebraic formulation consisting in a polynomial system S . There is, of course, some issues like, for instance, the fact that we represent both internal and external bisectors when using algebraic formulation, but we do not discuss these points here (see [27]). Then, we have to triangularize S : to this end, we use regular chains method that is implemented in Maple. For the triangulation of the polynomial system corresponding to the statement, we choose the ordering $x_C, y_C, x_B, y_B, x_A, y_A$ for the variables. We find two irreducible systems which corresponds to non-degenerate cases, say S_1 and S_2 : this means that under the non-degeneracy conditions $S \Leftrightarrow S_1 \vee S_2$ it is enough to show that none of these systems corresponds to a RC-constructible problem (if one of these systems was RC-constructible, then we could construct some solutions of the problem).

In the first one, we have the irreducible polynomial equation:

$$4y_A^4 - 12y_A^3 - 51y_A^2 + 192y_A - 144 = 0$$

and for the second one:

$$7295401y_A^6 - 30894038y_A^5 + 107596129y_A^4 - 127795968y_A^3 - 3722832y_A^2 + 24966144y_A + 4064256 = 0$$

The two polynomials are irreducible and Wantzel theorem ensures that the second one is not RC-solvable. But there is more work to do for the first equation as it is of degree 2^2 . Using the formula of Gao and Chou [10], we have to see if the polynomial:

$$8g^3 4h_2g^2 + (2h_1h_38h_0)gh_0h_3^2 + 4h_0h_2h_1^2 = 0$$

has rational solutions, where $x^4 + h_3x^3 + h_2x^2 + h_1x + h_0$ is the minimal monic irreducible polynomial we want to test: here, we have $h_3 = -12/4$, $h_2 = -51/4$, $h_1 = 192/4$ and $h_0 = 144/4$ and we get the polynomial:

$$8g^3 + 51g^2 - 144$$

Then, using the `factor` command, we prove that this polynomial is irreducible and thus that the problem is RC-unconstructible. Notice that this method can also be used to find a construction when the problem is RC-constructible, but usually the construction is impractical and not in the spirit of classical RC-constructions.

3.3 Constructible Case

Problem 116 from Wernick's list. In this problem, points G , H_a and H are given and the task is to construct a triangle $T = (A, B, C)$ such that points G , H_a

and H are respectively the centroid, the feet of the altitude from point A and the orthocenter of triangle T .

It is easy to construct the line BC , M_a and A and O using the fact that G is the center of the homothety with ratio $-1/2$ transforming O into H . We leave the construction to the reader.

Let us consider the algebraic version. Let the given points have the following coordinates: $H(0, 0)$, $H_a(1, 0)$, $G(a, b)$, where a and b are some real numbers. We have then to solve the system:

$$\begin{cases} x_A(x_B - x_C) + y_A(y_B - y_C) = 0 \\ x_B(x_C - x_A) + y_B(y_C - y_A) = 0 \\ (x_A - 1)(x_C - 1) + y_A y_C = 0 \\ (1 - x_B)(y_C - y_B) - y_B(x_B - x_C) = 0 \\ 3a - x_A - x_B - x_C = 0 \\ 3b - y_A - y_B - y_C = 0 \end{cases}$$

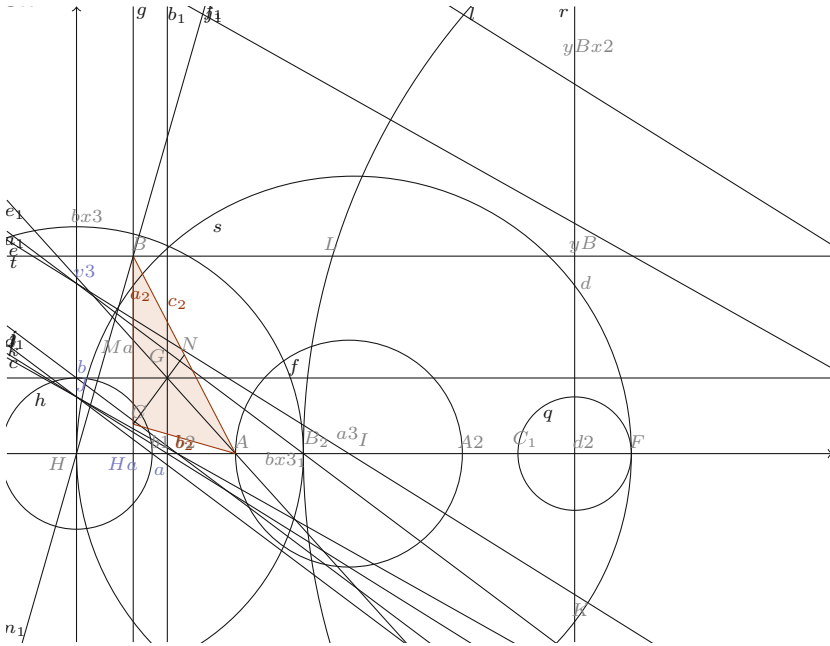


Fig. 3. Performing algebraic computations with geometry (details where the attentive reader can see, for instance, the extraction of a square root $d_2 \rightarrow d$)

After triangularization, we have only one non-degenerate system:

$$\begin{cases} x_C - 1 = 0 \\ y_C + y_B - 3b = 0 \\ -1 + x_B = 0 \\ y_B^2 - 3b \cdot y_B + 3a - 3 = 0 \\ 2 - 3a + x_A = 0 \\ y_A = 0 \end{cases}$$

which yields two, one or zero solutions depending on the discriminant of the fourth equations. The two solutions correspond to each other by permuting B and C . This problem is then obviously constructible since we are able to perform all the operations with straightedge and compass. Such a construction is depicted in Fig. 3 made with GeoGebra, where parameters a and b correspond to free points constrained to be on the x -axis and y -axis respectively. But that solution is usually not the solution that the teacher wanted. However, it is not difficult to algebraically verify that the solutions provided by the triangular system are solutions to the problem. Of course, you have to be confident in your CAS.

4 Conclusions and Future Work

We presented a geometrical and an algebraic perspective on solving construction problems using ruler and compass. We showed that many steps of this process can be automated and supported by proofs which can be formalized within proof assistants.

For our future work, we are planning to complete, as much as possible, automation of solving for problems from Wernick's corpus, but also for other classes of construction problems. We are planning to integrate this automated process into dynamic geometry systems, having in mind applications in education. We are also planning to implement proving unconstructibility by reduction.

References

1. Aldefeld, B.: Variations of geometries based on a geometric-reasoning method. *Comput. Aided Des.* **20**(3), 117–126 (1988)
2. Boutry, P., Narboux, J., Schreck, P., Braun, G.: Using small scale automation to improve both accessibility and readability of formal proofs in geometry. In: *Proceedings of the 10th International Workshop on Automated Deduction in Geometry (ADG 2014)*, CISUC TR2014/02, Universidade de Coimbra (2014)
3. Braun, G., Narboux, J.: From Tarski to Hilbert. In: *Ida, T., Fleuriot, J. (eds.) ADG 2012*. LNCS, vol. 7993, pp. 89–109. Springer, Heidelberg (2013)
4. Buoma, W., Fudos, I., Hoffman, C., Cai, J., Paige, R.: A geometric constraint solver. *CAD* **27**, 487–501 (1995)
5. Buthion, M.: Un programme qui résoud formellement des problèmes de constructions géométriques. *RAIRO Informatique* **3**(4), 353–387 (1979)
6. Chen, G.: *Les constructions à la règle et au compas par une méthode algébrique*. Technical report Rapport de DEA, Université Louis Pasteur (1992)

7. Djorić, M., Janičić, P.: Constructions, instructions, interactions. *Teach. Mathe. Appl.* **23**(2), 69–88 (2004)
8. Dufourd, J.-F., Mathis, P., Schreck, P.: Geometric construction by assembling solved subfigures. *Artif. Intell. J.* **99**(1), 73–119 (1998)
9. Essert-Villard, C., Schreck, P., Dufourd, J.-F.: Sketch-based pruning of a solution space within a formal geometric constraint solver. *Artif. Intell.* **124**(1), 139–159 (2000)
10. Gao, X.-S., Chou, S.-C.: Solving geometric constraint systems. ii. A symbolic approach and decision of rc-constructibility. *Comput. Aided Des.* **30**(2), 115–122 (1998)
11. Gelernter, H.: Realization of a geometry theorem proving machine. In: *Proceedings of the International Conference Information Processing, Paris*, pp. 273–282, 15–20 June 1959
12. Génevaux, J.-D., Narboux, J., Schreck, P.: Formalization of Wu’s simple method in Coq. In: Jouannaud, J.-P., Shao, Z. (eds.) *CPP 2011. LNCS*, vol. 7086, pp. 71–86. Springer, Heidelberg (2011)
13. Grégoire, B., Pottier, L., Théry, L.: Proof certificates for algebra and their application to automatic geometry theorem proving. In: Sturm, T., Zengler, C. (eds.) *ADG 2008. LNCS*, vol. 6301, pp. 42–59. Springer, Heidelberg (2011)
14. Gulwani, S., Korthikanti, V.A., Tiwari, A.: Synthesizing geometry constructions. In: *Programming Language Design and Implementation, PLDI 2011*, pp. 50–61. ACM (2011)
15. Janičić, P.: Geometry constructions language. *J. Autom. Reasoning* **44**(1–2), 3–24 (2010)
16. Jermann, C., Trombetti, G., Neveu, B., Mathis, P.: Decomposition of geometric constraint systems: a survey. *Int. J. Comput. Geom. Appl.* **16**(5–6), 379–414 (2006). CNRS MathSTIC
17. Landau, S., Miller, G.L.: Solvability by radicals is in polynomial time. *J. Comput. Syst. Sci.* **30**(2), 179–208 (1985)
18. Lebesgue, H.: *Leçons sur les constructions géométriques*. Gauthier-Villars, Paris (1950) (in French), re-edition by Editions Jacques Gabay, France
19. Lemaire, F., Moreno-Maza, M., Xie, Y.: The RegularChains library in Maple 10. In: Kotsireas, I.S. (ed.) *Proceedings of Maple Summer Conference 2005*, Waterloo, Canada, pp. 355–368 (2005)
20. Marić, F., Petrović, I., Petrović, D., Janičić, P.: Formalization and implementation of algebraic methods in geometry. In: Quesma, P., Back, R.-J. (eds.) *Proceedings of First Workshop on CTP Components for Educational Software*, Wrocław, Poland, 31 July 2011. *Electronic Proceedings in Theoretical Computer Science*, vol. 79, pp. 63–81. Open Publishing Association (2012)
21. Marinković, V., Janičić, P.: Towards understanding triangle construction problems. In: Campbell, J.A., Jeuring, J., Carette, J., Dos Reis, G., Sojka, P., Wenzel, M., Sorge, V. (eds.) *CICM 2012. LNCS*, vol. 7362, pp. 127–142. Springer, Heidelberg (2012)
22. Nipkow, T., Paulson, L.C., Wenzel, M. (eds.): *Isabelle/HOL. LNCS*, vol. 2283, p. 3. Springer, Heidelberg (2002)
23. Owen, J.: Algebraic solution for geometry from dimensional constraints. In: *Proceedings of the 1th ACM Symposium of Solid Modeling and CAD/CAM Applications*, pp. 397–407. ACM Press (1991)
24. Scandura, J.M., Durnin, J.H., Wulfeck II, W.H.: Higher order rule characterization of heuristics for compass and straight edge constructions in geometry. *Artif. Intell.* **5**(2), 149–183 (1974)

25. Schreck, P.: Robustness in CAD Geometric Constructions. In: IV 2001, pp. 111–116 (2001)
26. Schreck, P.: Modélisation et implantation d'un système à base de connaissances pour les constructions géométriques. *Revue d'Intelligence Artificielle* **8**(3), 223–247 (1994)
27. Schreck, P., Mathis, P.: Rc-constructibility of problems in Wernick's and Connelly's lists. In: Proceedings of the 10th International Workshop on Automated Deduction in Geometry (ADG 2014), CISUC TR2014/02, Universidade de Coimbra (2014)
28. Stojanović, S., Pavlović, V., Janičić, P.: A coherent logic based geometry theorem prover capable of producing formal and readable proofs. In: Schreck, P., Narboux, J., Richter-Gebert, J. (eds.) ADG 2010. LNCS, vol. 6877, pp. 201–220. Springer, Heidelberg (2011)
29. Wernick, W.: Triangle constructions with three located points. *Math. Mag.* **55**(4), 227–230 (1982)
30. Pambuccian, V.: Axiomatizing geometric constructions. *J. Appl. Logic* **6**(1), 24–46 (2008)
31. Beeson, M.: Logic of ruler and compass constructions. In: Cooper, S.B., Dawar, A., Löwe, B. (eds.) CiE 2012. LNCS, vol. 7318, pp. 46–55. Springer, Heidelberg (2012)

Integrated Circumradius and Area Formulae for Cyclic Pentagons and Hexagons

Shuichi Moritsugu^(✉)

University of Tsukuba, Tsukuba, Ibaraki 305-8550, Japan
moritsug@slis.tsukuba.ac.jp

Abstract. This paper describes computations of the relations between the circumradius R and area S of cyclic polygons given by the lengths of the sides. The classic results of Heron and Brahmagupta clearly show that the product of R and S is expressed by the lengths of the sides for triangles and cyclic quadrilaterals. However, the formulae of circumradius and area for cyclic pentagons and hexagons have been studied separately, and the relation between them has seldom been discussed. In this study, based on the results derived by Robbins (1994), Pech (2006), and the author (2011), we succeeded in computing *integrated formulae* of the circumradius and the area for cyclic pentagons and hexagons. They are found to be a polynomial equation in $(4SR)^2$ with degree 7 for pentagons, and the product of two polynomials each with degree 7 for hexagons. We confirmed that these three polynomials with degree 7 are uniformly expressed using the notion of *crossing parity*, the structure of which is analogous to those of the area formulae and circumradius formulae for $n = 5, 6$. Moreover, we derived a polynomial equation in $(4SR)$ itself with degree 7 for cyclic pentagons, and showed that this type of formula exists only for n -gons, where n is an odd number.

Keywords: Cyclic polygons · Circumradius formula · Area formula

1 Introduction

In this study, we consider a classic problem in Euclidean geometry for cyclic polygons; that is, n -gons inscribed in a circle, given by the lengths of sides a_1, a_2, \dots, a_n . In particular, we focus on the relation between the circumradius R and the area S of cyclic pentagons and hexagons.

Firstly, for a triangle with side lengths a_1, a_2 , and a_3 , the classic formula derived by Heron gives its circumradius and area as follows:

$$\begin{cases} R = \frac{a_1 a_2 a_3}{\sqrt{(a_1+a_2+a_3)(-a_1+a_2+a_3)(a_1-a_2+a_3)(a_1+a_2-a_3)}} \\ S = \frac{\sqrt{(a_1+a_2+a_3)(-a_1+a_2+a_3)(a_1-a_2+a_3)(a_1+a_2-a_3)}}{4}. \end{cases} \quad (1)$$

This work was supported by a Grant-in-Aid for Scientific Research (25330006) from the Japan Society for the Promotion of Science (JSPS).

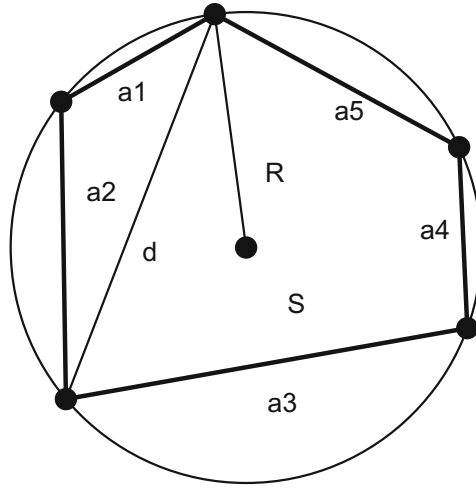


Fig. 1. A cyclic pentagon with area S

It is straightforward to combine these equations, and we obtain the relation

$$4SR = a_1a_2a_3. \tag{2}$$

Secondly, Brahmagupta’s formula gives the circumradius and area of a cyclic quadrilateral as

$$\begin{cases} R = \sqrt{\frac{(a_1a_2+a_3a_4)(a_1a_3+a_2a_4)(a_1a_4+a_2a_3)}{(-a_1+a_2+a_3+a_4)(a_1-a_2+a_3+a_4)(a_1+a_2-a_3+a_4)(a_1+a_2+a_3-a_4)}} \\ S = \frac{\sqrt{(-a_1+a_2+a_3+a_4)(a_1-a_2+a_3+a_4)(a_1+a_2-a_3+a_4)(a_1+a_2+a_3-a_4)}}{4}. \end{cases} \tag{3}$$

It is again straightforward to integrate Eq. (3) into

$$(4SR)^2 = (a_1a_2 + a_3a_4)(a_1a_3 + a_2a_4)(a_1a_4 + a_2a_3). \tag{4}$$

We should note that Eq. (3) represents the case of convex quadrilaterals, whereas the other case of non-convex, crossing figures is given by

$$\begin{cases} R = \sqrt{\frac{-(a_1a_2-a_3a_4)(a_1a_3-a_2a_4)(a_1a_4-a_2a_3)}{(a_1+a_2+a_3+a_4)(a_1+a_2-a_3-a_4)(a_1-a_2-a_3+a_4)(-a_1+a_2-a_3+a_4)}} \\ S = \frac{\sqrt{(a_1+a_2+a_3+a_4)(a_1+a_2-a_3-a_4)(a_1-a_2-a_3+a_4)(-a_1+a_2-a_3+a_4)}}{4}. \end{cases} \tag{5}$$

Hence, the latter case is expressed by the relation

$$(4SR)^2 = -(a_1a_2 - a_3a_4)(a_1a_3 - a_2a_4)(a_1a_4 - a_2a_3). \tag{6}$$

If we let $Z = (4SR)^2$, the above results for triangles and cyclic quadrilaterals are summarized as follows:

$$\begin{cases} Z - a_1^2 a_2^2 a_3^2 = 0, \\ (Z - (a_1 a_2 + a_3 a_4)(a_1 a_3 + a_2 a_4)(a_1 a_4 + a_2 a_3)) \\ \quad \times (Z + (a_1 a_2 - a_3 a_4)(a_1 a_3 - a_2 a_4)(a_1 a_4 - a_2 a_3)) = 0. \end{cases} \quad (7)$$

The goal of the present study was to find *integrated formulae* for cyclic pentagons and hexagons analogous to Eq. (7). Since Robbins [6] discovered the area formula for cyclic pentagons in 1994, the following facts for cyclic n -gons have been confirmed by several authors [1–3, 5]:

- The pentagon area formula is a polynomial in $(4S)^2$ with degree 7.
- The hexagon area formula is the product of two polynomials in $(4S)^2$, each with degree 7.
- The pentagon circumradius formula is a polynomial in R^2 with degree 7.
- The hexagon circumradius formula is the product of two polynomials in R^2 , each with degree 7.

Therefore, we can speculate that the relations between S and R for cyclic pentagons and hexagons are also expressed by the polynomials in $Z = (4SR)^2$ with degree 7 and their product, analogously to Eq. (7). As a result of this study, we succeeded in computing such formulae explicitly as speculated. Only the result for pentagons has been published up to now as a short paper [4], while the present paper describes the detailed formulae for cyclic pentagons and hexagons, elucidating their relations.

It might sound strange that the relation between the area and circumradius has seldom been discussed, and Pech [5] noted that “it is still missing” for cyclic pentagons. We have found that Svrtan et al. [7] show a likely formula with degree 7, but their result seems to contain typographical or other errors and their proof is too abbreviated to follow.

In contrast, we show two ways of computation for cyclic pentagons and confirm the correctness of both results in Sects. 2 and 3 of this paper. Hence, we believe that our result gives the correction and extension to that of Svrtan et al. [7]. In Sect. 4, we show a numerical example of the usage of the integrated formula to distinguish a convex cyclic pentagon from other, non-convex cases.

In Sect. 5, the integrated formula for cyclic hexagons is discussed. The computation is far from efficient, but it seems that the results have not been published before. Hence, we believe that we have succeeded in specifying the structure of integrated formulae for cyclic n -gons ($n = 5, 6$) in detail.

2 Brute Force Algorithm

2.1 Expression by Elementary Symmetric Functions

Since the coefficients in the area and circumradius formulae are symmetric with those of a_i^2 , such expressions as Eq. (7) can be reduced if the coefficients are expressed by elementary symmetric functions.

The conversion is processed by the following algorithm. First, we consider the polynomial ideal with elementary symmetric functions of n -th order:

$$I = \left\{ s_1^{(n)} - (a_1^2 + \dots + a_n^2), \dots, s_n^{(n)} - (a_1^2 \dots a_n^2) \right\}. \quad (8)$$

Hereafter, we abbreviate $s_i^{(n)}$ simply as s_i , if n is obvious in the context. Computing its Gröbner basis with a group ordering (“lexdeg” in the Maple computer algebra system), we obtain

$$G := \text{Basis}(I, \{a_1, \dots, a_n\} \succ \{s_1, \dots, s_n\}). \quad (9)$$

Next, computing $p := \text{NormalForm}(f, G)$ for a symmetric function f , we get the expression p by elementary symmetric functions. For example, converting Eq. (7), we obtain the formulae for $n = 3, 4$ described below.

Theorem 1. *The defining polynomials of $Z = (4SR)^2$ for triangles and cyclic quadrilaterals are respectively given as*

$$\begin{cases} Z - s_3 = 0, \\ Z^2 - 2s_3Z + (s_3^2 - s_1^2s_4) = (Z - s_3 + s_1\sqrt{s_4})(Z - s_3 - s_1\sqrt{s_4}) = 0. \end{cases} \quad (10)$$

We should note that a good insight into the structure of the formulae is provided by the introduction of an auxiliary expression $\sqrt{s_n} = a_1 \dots a_n$, as well as the notion of *crossing parity* ε [2, 6], where ε is 0 for a triangle, 1 for a convex quadrilateral, and -1 for a non-convex quadrilateral. Under these notations, the area formula in $x = (4S)^2$ for $n = 3, 4$ is written as

$$x - (-s_1^2 + 4s_2 + \varepsilon \cdot 8\sqrt{s_4}) = 0. \quad (11)$$

Similarly, the radius formula in $y = R^2$ for $n = 3, 4$ is written as

$$(-s_1^2 + 4s_2 + \varepsilon \cdot 8\sqrt{s_4})y - (s_3 + \varepsilon \cdot s_1\sqrt{s_4}) = 0. \quad (12)$$

Eventually, we have a simpler expression also for $Z = xy$.

Corollary 1. *The integrated formula (10) in $Z = (4SR)^2$ is rewritten as*

$$Z - (s_3 + \varepsilon \cdot s_1\sqrt{s_4}) = 0. \quad (13)$$

Since we have $s_3^{(3)} = s_3^{(4)}|_{a_4=0}$ and so on, the notations are intentionally combined for the cases $\varepsilon = 0, \pm 1$.

We should note that, in our formulation, the area of the triangle between $\vec{OA} = [x_1, y_1]$ and $\vec{OB} = [x_2, y_2]$ is defined as the determinant

$$S = \frac{1}{2} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix}, \quad (14)$$

whose sign depends on the direction of the angle between these two vectors. Hence, discarding the sign of area S of polygons, the formula for triangles given by Eq. (2) is rewritten as shown below.

Corollary 2. *For any triangle, we have another integrated formula:*

$$|z| - \sqrt{s_3} = 0 \quad (z = 4SR = \pm\sqrt{Z}, \quad \sqrt{s_3} = a_1 a_2 a_3). \quad (15)$$

This expression is equivalent to the first expression in Eq. (10). However, the case $n = 4$ does not have such a polynomial equation in z , and the existence of this type of formula seems limited to cases where n is an odd number.

2.2 Integrated Formula for Cyclic Pentagons

We assume that, according to [3,5,6], we have already computed the circumradius formula with 2,922 terms for cyclic pentagons:

$$\begin{aligned} \Phi_R(y) &= B_7 y^7 + B_6 y^6 + B_5 y^5 + B_4 y^4 + B_3 y^3 + B_2 y^2 + B_1 y + B_0 \\ &= 0 \quad (y = R^2, \quad B_i \in \mathbf{Z}[a_1^2, \dots, a_5^2]). \end{aligned} \quad (16)$$

Using the elementary symmetric functions $s_1 = a_1^2 + \dots + a_5^2, \dots, s_5 = a_1^2 \dots a_5^2$, we rewrite Eq. (16) into a simpler form:

$$\tilde{\Phi}_R(y) = \tilde{B}_7 y^7 + \tilde{B}_6 y^6 + \dots + \tilde{B}_1 y + \tilde{B}_0 = 0 \quad (81 \text{ terms}), \quad (17)$$

where $\tilde{B}_i \in \mathbf{Z}[s_1, \dots, s_5]$.

On the other hand, the area formula derived by Robbins [2,6] is originally given using elementary symmetric functions:

$$\begin{aligned} \tilde{\Phi}_S(x) &= x^7 + \tilde{C}_6 x^6 + \dots + \tilde{C}_1 x + \tilde{C}_0 = 0 \quad (153 \text{ terms}), \\ &\quad (x = (4S)^2, \quad \tilde{C}_i \in \mathbf{Z}[s_1, \dots, s_5]). \end{aligned} \quad (18)$$

If each coefficient \tilde{C}_i is expanded, this polynomial equation has 6,672 terms over the coefficient $C_i \in \mathbf{Z}[a_1^2, \dots, a_5^2]$ [5].

In these equations, the leading coefficients and the constant terms have the following structures:

$$\begin{cases} \tilde{B}_7 = ((s_1^2 - 4s_2)^2 - 64s_4)^2 - 2048s_5(s_1^3 - 4s_1s_2 + 8s_3), \\ \tilde{B}_0 = s_5^3, \\ \tilde{C}_0 = \tilde{B}_7 \cdot (s_1^3 - 4s_1s_2 + 8s_3)^2. \end{cases} \quad (19)$$

We let the roots of $\tilde{\Phi}_R(y)$ be y_1, \dots, y_7 , and the roots of $\tilde{\Phi}_S(x)$ be x_1, \dots, x_7 . Then the polynomial whose roots are $Z_i = x_i y_i$ should have the form described below. Since we have

$$\begin{aligned} \prod_{i=1}^7 Z_i &= (x_1 y_1) \dots (x_7 y_7) = (y_1 \dots y_7)(x_1 \dots x_7) \\ &= -(\tilde{B}_0/\tilde{B}_7) \cdot (-\tilde{C}_0) = s_5^3(s_1^3 - 4s_1s_2 + 8s_3)^2, \end{aligned} \quad (20)$$

the quantities Z_1, \dots, Z_7 should be the roots of

$$1 \cdot Z^7 + u_6 Z^6 + \dots + u_1 Z - s_5^3(s_1^3 - 4s_1s_2 + 8s_3)^2 = 0, \quad (21)$$

where u_6, \dots, u_1 are as yet unknown.

In order to combine $\tilde{\Phi}_R(y)$ and $\tilde{\Phi}_S(x)$ in Eqs. (17) and (18), first we substitute $y = Z/x$ into the radius formula, and obtain

$$\tilde{\Phi}'_R(x, Z) = x^7 \tilde{\Phi}_R(Z/x) = \tilde{B}_7 Z^7 + \tilde{B}_6 Z^6 x + \dots + \tilde{B}_1 Z x^6 + \tilde{B}_0 x^7. \quad (22)$$

Next, eliminating x by the resultant of $\tilde{\Phi}'_R(x, Z)$ and $\tilde{\Phi}_S(x)$, we obtain its primitive part as

$$\begin{aligned} \tilde{\Psi}(Z) &= \text{Res}_x(\tilde{\Phi}'_R(x, Z), \tilde{\Phi}_S(x)) \\ &= \tilde{A}_{49} Z^{49} + \dots + \tilde{A}_0 \quad \left(\tilde{A}_i \in \mathbf{Z}[s_1, \dots, s_5] \right) \quad (2,093,279 \text{ terms}). \end{aligned} \quad (23)$$

This computation required about 15 min of CPU time in the following environment: Maple 14 on Win64, Xeon (2.93 GHz) $\times 2$, 192 GB RAM.

Finally, we factorize this polynomial with degree 49. Together with the other factor of degree 42, we obtain the polynomial in Z with degree 7. This factorization required almost 80 hours of CPU time.

Theorem 2. *The defining polynomial in $Z = (4SR)^2$ for cyclic pentagons has the following form:*

$$\begin{aligned} \psi_5(Z) &= Z^7 - 4s_3 Z^6 + (-28s_1 s_5 - 2s_1^2 s_4 + 6s_3^2) Z^5 \\ &\quad + ((-s_1^4 - 10s_1^2 s_2 - 8s_2^2 + 52s_1 s_3 - 32s_4) s_5 + 4s_3 (s_1^2 s_4 - s_2^2)) Z^4 \\ &\quad + (4(37s_1^2 - 48s_2) s_5^2 + (-2s_1^4 s_3 + 12s_1^3 s_4 + 64s_3 s_4 + 4s_1^2 s_2 s_3 + 16s_2^2 s_3 \\ &\quad \quad - 20s_1 s_3^2 - 64s_1 s_2 s_4) s_5 + (s_1^2 s_4 - s_3^2)^2) Z^3 \\ &\quad + (-576s_5^3 + (64s_1 s_4 - 80s_1 s_2^2 + 28s_1^3 s_2 - 8s_1^2 s_3 + 128s_2 s_3 - 2s_1^5) s_2^2 \\ &\quad \quad + (2s_1^4 s_2 s_4 - 12s_1^3 s_3 s_4 - 8s_1^2 s_2^2 s_4 - s_1^4 s_3^2 - 4s_1 s_3^3 - 32s_1^2 s_4^2 - 32s_2^3 s_4 \\ &\quad \quad + 64s_1 s_2 s_3 s_4 - 8s_2^2 s_3^2 + 6s_1^2 s_2 s_3^2) s_5) Z^2 \\ &\quad + (-48(s_1^3 - 4s_1 s_2 + 8s_3) s_5^3 \\ &\quad \quad (-8s_1^2 s_2^2 + 256s_4^2 + s_1^4 s_2^2 - 64s_1 s_3 s_4 - 128s_2^2 s_4 + 64s_2 s_3^2 + 16s_4^4 \\ &\quad \quad + 96s_1^2 s_2 s_4 - 12s_1^2 s_3^2 - 48s_1 s_2^2 s_3 - 2s_1^5 s_3 - 16s_1^4 s_4 + 20s_1^3 s_2 s_3) s_5^2) Z \\ &\quad - s_5^3 (s_1^3 - 4s_1 s_2 + 8s_3)^2 = 0 \quad (63 \text{ terms}). \end{aligned} \quad (24)$$

If we consider the equilateral case, by putting $\forall a_i := 1$, Eq. (24) is reduced to

$$(Z^2 - 35Z + 25)(Z - 1)^5 = 0, \quad (25)$$

each factor of which respectively corresponds to the cases of a regular pentagon/pentagram and a (five degenerated) regular triangle. Therefore, we believe that Eq. (24) is the *integrated formula* for cyclic pentagons, as the extension of Eq. (13) for $n = 3, 4$.

Remark 1. The notion of the formula in $Z = (4SR)^2$ was already proposed by Svrtan et al. [7], and their Eq.(35) is supposed to correspond to Eq. (24) above.

Unfortunately, their result does not coincide with ours, and theirs is not factored when we let $\forall a_i := 1$. Therefore, their result seems to contain typographical or other errors, which might have been caused in the process of conversion into the expression by elementary symmetric functions.

3 Stepwise Algorithm

In this algorithm, we construct the formulae from scratch, without using $\tilde{\Phi}_R(y)$ and $\tilde{\Phi}_S(x)$. We start by dividing a cyclic pentagon with side lengths $\{a_1, \dots, a_5\}$ by a diagonal of length d , into a triangle of sides $\{a_1, a_2, d\}$ and a quadrilateral of sides $\{a_3, a_4, a_5, d\}$, as shown in Fig. 1. Next, using the common circumradius R , we consider the sum of the areas of the triangle and the quadrilateral.

Firstly, the circumradius formula of Heron gives the defining polynomial in $y = R^2$ as follows:

$$H_3(a_1, a_2, d; y) := (a_1 + a_2 + d)(-a_1 + a_2 + d)(a_1 - a_2 + d)(a_1 + a_2 - d)y + a_1^2 a_2^2 d^2. \tag{26}$$

Similarly, the formula of Brahmagupta gives the following polynomials for the convex and non-convex cases, respectively:

$$\left\{ \begin{array}{l} H_4^{(+)}(a_3, a_4, a_5, d; y) = \\ \quad (-a_3 + a_4 + a_5 + d)(a_3 - a_4 + a_5 + d)(a_3 + a_4 - a_5 + d)(a_3 + a_4 + a_5 - d)y \\ \quad \quad \quad - (a_3 a_4 + a_5 d)(a_3 a_5 + a_4 d)(a_3 d + a_4 a_5), \\ H_4^{(-)}(a_3, a_4, a_5, d; y) = \\ \quad (a_3 + a_4 + a_5 + d)(a_3 - a_4 - a_5 + d)(a_3 - a_4 + a_5 - d)(a_3 + a_4 - a_5 - d)y \\ \quad \quad \quad - (a_3 a_4 - a_5 d)(a_3 a_5 - a_4 d)(a_3 d - a_4 a_5). \end{array} \right. \tag{27}$$

Since the circumradius R is common to this triangle and quadrilateral, we eliminate y using the resultant and obtain the defining polynomials in the diagonal d with degree 7:

$$\left\{ \begin{array}{l} F^{(+)}(d) := \text{Res}_y(H_4^{(+)}(a_3, a_4, a_5, d; y), H_3(a_1, a_2, d; y)) \\ \quad = a_3 a_4 a_5 d^7 + (a_3^2 a_4^2 + a_3^2 a_5^2 + a_4^2 a_5^2 - a_1^2 a_2^2) d^6 + \dots, \\ F^{(-)}(d) := \text{Res}_y(H_4^{(-)}(a_3, a_4, a_5, d; y), H_3(a_1, a_2, d; y)) \\ \quad = a_3 a_4 a_5 d^7 - (a_3^2 a_4^2 + a_3^2 a_5^2 + a_4^2 a_5^2 - a_1^2 a_2^2) d^6 + \dots. \end{array} \right. \tag{28}$$

We should note that we have $F^{(-)}(d) = -F^{(+)}(-d)$, which means that the roots of both polynomials are equivalent up to their signs.

Secondly, we let S_3, S_4 , and S_5 be the area of the triangle, cyclic quadrilateral, and cyclic pentagon, respectively, given above. Since we have $S_5 = S_3 + S_4$, we get $4S_4 R = 4S_5 R - 4S_3 R$, where R is the common circumradius. For the triangle, we have $4S_3 R = a_1 a_2 d$ from Eq. (2). If we let $z = 4S_5 R$ and substitute $4S_4 R = z - a_1 a_2 d$ into Eqs. (4) and (6), we obtain the following polynomial

equations in z and d , for the cases of convex and non-convex quadrilaterals, respectively:

$$\begin{cases} f^{(+)}(z, d) := (z - a_1 a_2 d)^2 - (a_3 a_4 + a_5 d)(a_3 a_5 + a_4 d)(a_3 d + a_4 a_5) = 0, \\ f^{(-)}(z, d) := (z - a_1 a_2 d)^2 + (a_3 a_4 - a_5 d)(a_3 a_5 - a_4 d)(a_3 d - a_4 a_5) = 0. \end{cases} \tag{29}$$

Finally, we eliminate the diagonal d from $F^{(+)}(d)$ and $f^{(+)}(z, d)$ by computing the resultant:

$$\begin{aligned} P^{(+)}(z) &:= \text{Res}_d(F^{(+)}(d), f^{(+)}(z, d)) \\ &= (z^7 + \dots) (a_3 a_4 a_5 z^7 + \dots). \end{aligned} \tag{30}$$

This computation required about 2.5 minutes of CPU time in total, which is a dramatic reduction from that required for the brute force algorithm described in the preceding section. Since the latter factor in Eq. (30) is asymmetric with those of a_i , we adopt the former factor as the defining polynomial in $z = 4SR$ for cyclic pentagons. Converting it into the requisite expression by elementary symmetric functions, with $\sqrt{s_5} = a_1 a_2 a_3 a_4 a_5$, we obtain the final result:

$$\begin{aligned} \varphi_5^{(+)}(z) &= z^7 - 2s_3 z^5 - (s_1^2 + 4s_2)\sqrt{s_5} z^4 + (s_3^2 - s_1^2 s_4 - 14s_1 s_5) z^3 \\ &\quad - (s_1^2 s_3 + 8s_1 s_4 - 4s_2 s_3 + 24s_5)\sqrt{s_5} z^2 \\ &\quad - (s_1^2 s_2 - 4s_2^2 + 2s_1 s_3 + 16s_4) s_5 z \\ &\quad - (s_1^3 - 4s_1 s_2 + 8s_3) s_5 \sqrt{s_5} = 0 \quad (18 \text{ terms}). \end{aligned} \tag{31}$$

If we consider the equilateral case, by putting $\forall a_i := 1$, Eq. (31) is reduced to

$$(z^2 - 5z - 5)(z + 1)^5 = 0, \tag{32}$$

each factor of which respectively corresponds to the cases of a regular pentagon/pentagram and a (five degenerated) regular triangle, similarly to Eq. (25).

For the case of a non-convex quadrilateral, we compute the resultant of another pair of $F^{(-)}(d)$ and $f^{(-)}(z, d)$, and obtain a similar result $\varphi^{(-)}(z)$, where we have $\varphi^{(-)}(z) = -\varphi^{(+)}(-z)$. If we put $\forall a_i := 1$, the equation $\varphi^{(-)}(z) = 0$ is reduced to

$$(z^2 + 5z - 5)(z - 1)^5 = 0, \tag{33}$$

which means that we can regard the roots of $\varphi^{(+)}(z) = 0$ and $\varphi^{(-)}(z) = 0$ as equivalent up to the signs of the area S_5 . Therefore, combining the two polynomials $\varphi^{(+)}(z)$ and $\varphi^{(-)}(z)$, we obtain another formula in $z = 4SR$, as described below.

Theorem 3. *The defining polynomial in $z = 4SR$ for cyclic pentagons is given by*

$$\begin{aligned} \varphi_5(z) &= |z|^7 - 2s_3 |z|^5 - (s_1^2 + 4s_2)\sqrt{s_5} |z|^4 + (s_3^2 - s_1^2 s_4 - 14s_1 s_5) |z|^3 \\ &\quad - (s_1^2 s_3 + 8s_1 s_4 - 4s_2 s_3 + 24s_5)\sqrt{s_5} |z|^2 \\ &\quad - (s_1^2 s_2 - 4s_2^2 + 2s_1 s_3 + 16s_4) s_5 |z| \\ &\quad - (s_1^3 - 4s_1 s_2 + 8s_3) s_5 \sqrt{s_5} = 0 \quad (18 \text{ terms}). \end{aligned} \tag{34}$$

This equation corresponds to the extension of Eq. (15) for $n = 3$.

It is straightforward to rewrite Eq. (34) as a polynomial equation in $Z = z^2 = (4SR)^2$. We separate the equation $\varphi_5(z) = 0$ into the terms with even degrees and odd degrees:

$$|z| (z^6 - 2s_3z^4 + \dots) = (s_1^2 + 4s_2)\sqrt{s_5}z^4 + \dots + (s_1^3 - 4s_1s_2 + 8s_3)s_5\sqrt{s_5}. \quad (35)$$

Squaring both sides and substituting $z^2 = Z$, we obtain the same result as Eq. (24), which is a polynomial in $Z = (4SR)^2$ with degree 7. Hence, we believe that the correctness of the two algorithms proposed here is confirmed by these results.

Remark 2. The paper by Svrtan et al. [7] is also based on a similar approach, but it does not specify the final step of the computation of the resultant, which should correspond to Eq. (30), nor does it refer to the existence of the defining polynomial in $z = 4SR$. Therefore, we consider that our computation of $\psi_5(Z)$ makes corrections to theirs, and our result $\varphi_5(z)$ is an original discovery, which has been unknown so far.

4 A Numerical Example

We consider the problem where the lengths of the sides are given as $a_1 = 5$, $a_2 = 6$, $a_3 = 7$, $a_4 = 8$, $a_5 = 9$. The formula for the area ($x = 16S^2$) and the circumradius ($y = R^2$) lead to the following equations:

$$\begin{cases} x^7 - 145377x^6 + \dots - 1360512306447018480615234375 = 0, \\ 5810802381759375y^7 - \dots - 1194842734208247398400000 = 0. \end{cases} \quad (36)$$

These equations are easily solved using a conventional numerical computation library, and we obtain five real roots of each equation. However, the correspondence between the area and circumradius is not clear, and we cannot even determine which root is a case of a convex or non-convex pentagon.

On the other hand, applying the set $\{5, 6, 7, 8, 9\}$ to the integrated formula (34), we obtain the equation in z , which also yields five real roots:

$$z^7 - 2356490z^5 - \dots - 1672586387136000000 = 0. \quad (37)$$

Since we have $|z| = \sqrt{xy} = 4|S|R$, we check all 125(= 5^3) combinations of roots $\{x_i, y_j, z_k\}$, and pick those that satisfy the condition

$$\left| |S_i| \cdot R_j - \frac{|z_k|}{4} \right| < 10^{-6}. \quad (38)$$

As a result, the following pairs are selected:

$$\begin{aligned} [|S_i|, R_j] = & [10.47633365, 6.035515309], \\ & [16.78535280, 4.505907128], \\ & [23.09053708, 4.602116876], \\ & [30.69973405, 4.802909240], \\ & [82.47639518, 6.019756631], \end{aligned} \quad (39)$$

in which the last pair indicates the case where the pentagon is convex.

5 Formulae for Cyclic Hexagons

The degrees of defining polynomials in terms of area and circumradius are given by the theorem of Robbins [6]. Let

$$k_m := \frac{2m + 1}{2} \binom{2m}{m} - 2^{2m-1} = \sum_{j=0}^{m-1} (m - j) \binom{2m + 1}{j}; \tag{40}$$

that is, let $k_i := 1, 7, 38, 187, 874, \dots (i = 1, 2, 3, 4, \dots)$. Then, we have

- The degrees in $x (= 16S^2)$ and $y (= R^2)$ for $(2m + 1)$ -gon are k_m .
- The degrees in x and y for $(2m + 2)$ -gon are $2k_m$, where polynomials are factored into the product of two polynomials each with degree k_m .

It is conjectured that the integrated formula has the same degree k_m or $2k_m$ as $Z = (4SR)^2$ in $\mathbf{Z} [a_1^2, \dots, a_n^2] [Z]$. Our next goal is the case of cyclic hexagons; that is, $m = 2, k_2 = 7$, and $2k_2 = 14$. On the analogy of the formulae for quadrilaterals in Eq. (10), the integrated formula in $Z = (4SR)^2$ for cyclic hexagons should have the following structure:

$$\begin{aligned} Z^{14} + \dots & \qquad \qquad \qquad (\mathbf{Z} [s_1, \dots, s_5, s_6] [Z]) \\ = (Z^7 + \dots)(Z^7 + \dots) & \qquad \qquad \qquad (\mathbf{Z} [s_1, \dots, s_5, \sqrt{s_6}] [Z]), \end{aligned} \tag{41}$$

where $s_1 = a_1^2 + \dots + a_6^2, \dots, \sqrt{s_6} = a_1 a_2 a_3 a_4 a_5 a_6$. The latter two factors with degree 7 are transformed into each other by replacing $\sqrt{s_6}$ with $-\sqrt{s_6}$.

5.1 Brute Force Algorithm

The circumradius formula is factored over the coefficients of a_i [3]:

$$\begin{aligned} & \Phi_R(a_1, \dots, a_6; y) \\ & := B_{14}y^{14} + \dots + B_1y + B_0 \quad (497,417 \text{ terms}) \quad (B_i \in \mathbf{Z}[a_1^2, \dots, a_6^2]) \tag{42} \\ & = \Phi_R^{(+)}(a_i; y) \cdot \Phi_R^{(-)}(a_i; y) \quad (\text{each has degree 7 and } 19,449 \text{ terms}). \end{aligned}$$

This is rewritten into the following expression by the elementary symmetric functions:

$$\tilde{\Phi}_R^{(+)}(s_1, \dots, s_5, \sqrt{s_6}; y) = \tilde{B}_7y^7 + \tilde{B}_6y^6 + \dots + \tilde{B}_1y + \tilde{B}_0 \quad (224 \text{ terms}). \tag{43}$$

We discard the non-convex cases, because they are easily obtained by

$$\tilde{\Phi}_R^{(-)}(s_1, \dots, s_5, \sqrt{s_6}; y) = \tilde{\Phi}_R^{(+)}(s_1, \dots, s_5, -\sqrt{s_6}; y). \tag{44}$$

On the other hand, the area formula is straightforwardly given by Robbins [6]. Convex cases are represented by elementary symmetric functions as

$$\tilde{\Phi}_S^{(+)}(s_1, \dots, s_5, \sqrt{s_6}; x) = x^7 + \tilde{C}_6x^6 + \dots + \tilde{C}_1x + \tilde{C}_0 \quad (282 \text{ terms}), \tag{45}$$

and non-convex cases are given by replacing $\sqrt{s_6}$ with $-\sqrt{s_6}$.

Computing the resultant from $\tilde{\Phi}_S^{(+)}(x)$ and $\tilde{\Phi}_R^{(+)}(y)$ through $Z = xy$, we have obtained a polynomial of degree 49 in Z with 52,490,772 terms (almost 3.0 GB), which corresponds to Eq. (23). This polynomial should be factored again into two polynomials of degrees 42 and 7, respectively, in Z , but it is impractical to actually execute this operation.

5.2 Stepwise Algorithm

Firstly, we should note that in this approach, computations proceed in the coefficient $\mathbf{Z}[a_1, \dots, a_6]$, and we cannot take advantage of a simpler expression using elementary symmetric functions.

By trial and error, we found that the given cyclic hexagon with side lengths $\{a_1, \dots, a_6\}$ should be divided by a diagonal of length d , into two quadrilaterals of sides $\{a_1, a_2, a_3, d\}$ and $\{a_4, a_5, a_6, d\}$. This approach gives a resultant with degree 14 in $Z = (4SR)^2$. On the contrary, if we divide the cyclic hexagon into a pentagon $\{a_1, a_2, a_3, a_4, d\}$ and a triangle $\{a_5, a_6, d\}$, the resultant should be that with degree 49 in Z , which seems hardly feasible.

Similarly to Eq. (27), we have Brahmagupta's formulae $H_4^{(+)}(a_1, a_2, a_3, d; y)$, $H_4^{(-)}(a_1, a_2, a_3, d; y)$, $H_4^{(+)}(a_4, a_5, a_6, d; y)$, and $H_4^{(-)}(a_4, a_5, a_6, d; y)$ for quadrilaterals with sides $\{a_1, a_2, a_3, d\}$ and $\{a_4, a_5, a_6, d\}$ and the common radius $y = R^2$.

Next, we eliminate y using the resultant and obtain two types of defining polynomials in the diagonal d with degree 7:

$$\left\{ \begin{array}{l} F^{(+)}(d) := \text{Res}_y(H_4^{(+)}(a_1, a_2, a_3, d; y), H_4^{(+)}(a_4, a_5, a_6, d; y)) \\ \quad = (a_1 a_2 a_3 - a_4 a_5 a_6) d^7 + \dots, \\ F^{(-)}(d) := \text{Res}_y(H_4^{(+)}(a_1, a_2, a_3, d; y), H_4^{(-)}(a_4, a_5, a_6, d; y)) \\ \quad = (a_1 a_2 a_3 + a_4 a_5 a_6) d^7 + \dots. \end{array} \right. \quad (46)$$

We should note that other combinations of polynomial $H_4^{(\pm)}$ eventually result in one of the above.

Let the relation of the areas of the hexagon and two quadrilaterals be $S_6 = S_{41} + S_{42}$. Using the common circumradius R , we let $Z_6 = (4S_6 R)^2$, $Z_{41} = (4S_{41} R)^2$, and $Z_{42} = (4S_{42} R)^2$. Here, we have convex and non-convex cases for each quadrilateral according to Eqs. (4) and (6):

$$\left\{ \begin{array}{l} Z_{41}^{(+)} = (a_1 a_2 + a_3 d)(a_2 a_3 + a_1 d)(a_1 a_3 + a_2 d), \\ Z_{41}^{(-)} = (a_1 a_2 - a_3 d)(a_2 a_3 - a_1 d)(a_1 a_3 - a_2 d), \\ Z_{42}^{(+)} = (a_4 a_5 + a_6 d)(a_5 a_6 + a_4 d)(a_4 a_6 + a_5 d), \\ Z_{42}^{(-)} = (a_4 a_5 - a_6 d)(a_5 a_6 - a_4 d)(a_4 a_6 - a_5 d). \end{array} \right. \quad (47)$$

The relations among Z_6 , Z_{41} , and Z_{42} are given by the following computation:

$$\begin{aligned} Z_6 &= 16S_6^2 R^2 = 16(S_{41} + S_{42})^2 R^2 \\ &= 16S_{41}^2 R^2 + 16S_{42}^2 R^2 + 32S_{41}S_{42}R^2 \\ &= Z_{41} + Z_{42} + 2(4S_{41}R)(4S_{42}R). \end{aligned} \quad (48)$$

Moving the terms in Z_{41} and Z_{42} , and squaring both sides, we obtain

$$(Z_6 - (Z_{41} + Z_{42}))^2 = 4(4S_{41}R)^2(4S_{42}R)^2 = 4Z_{41}Z_{42}, \tag{49}$$

which gives the defining polynomial of Z_6 as

$$Z_6^2 - 2(Z_{41} + Z_{42})Z_6 + (Z_{41} - Z_{42})^2 = 0. \tag{50}$$

Even though we have four patterns of combinations between $Z_{41}^{(\pm)}$ and $Z_{42}^{(\pm)}$, we have found that only the following two combinations give independent definitions of Z_6 :

$$\begin{cases} G^{(+)}(Z_6, d) := Z_6^2 - 2(Z_{41}^{(+)} + Z_{42}^{(+)}Z_6 + (Z_{41}^{(+)} - Z_{42}^{(+)}))^2 \\ \qquad \qquad \qquad = (a_1a_2a_3 - a_4a_5a_6)^2d^6 + \dots, \\ G^{(-)}(Z_6, d) := Z_6^2 - 2(Z_{41}^{(+)} + Z_{42}^{(-)}Z_6 + (Z_{41}^{(+)} - Z_{42}^{(-)}))^2 \\ \qquad \qquad \qquad = (a_1a_2a_3 + a_4a_5a_6)^2d^6 + \dots. \end{cases} \tag{51}$$

Finally, we eliminate the diagonal d from $F^{(+)}(d)$ and $G^{(+)}(Z_6, d)$ by computing the resultant, and obtain the polynomial equation of Z_6 as

$$\begin{aligned} P^{(+)}(Z_6) &:= \text{Res}_d(F^{(+)}(d), G^{(+)}(Z_6, d)) \\ &= (Z_6^7 + \dots) ((a_1a_2a_3 - a_4a_5a_6)^3 Z_6^7 + \dots). \end{aligned} \tag{52}$$

The polynomial $P^{(+)}(Z_6)$ itself has 4, 276, 908 terms, and this factorization step required about 11 days of CPU time (8 days of elapsed time) in our computational environment. Discarding the second factor, which is not symmetric, we convert the first factor (44,926 terms) into the requisite expression by elementary symmetric functions and obtain the results described below. Hereafter, we rewrite Z_6 as Z to unify the notation.

Theorem 4. *One of the defining polynomials of $Z = (4SR)^2$ for cyclic hexagons has the following form:*

$$\begin{aligned} \psi_6^{(+)}(Z) &= Z^7 - (4s_3 + 28\sqrt{s_6})Z^6 + (\dots)Z^5 + \dots + (\dots)Z \\ &\quad - (s_1^3 - 4s_1s_2 + 8s_3 - 16\sqrt{s_6})^2 \\ &\quad \times (s_5^3 - 4\sqrt{s_6}^5 + (s_1^3 - 4s_1s_2 + 4s_3)\sqrt{s_6}^4 \\ &\quad \quad + (-s_1^2s_4 + 2s_1s_5 + 4s_2s_4 - s_3^2)\sqrt{s_6}^3 + (s_1s_3s_5 - 4s_4s_5)\sqrt{s_6}^2 \\ &\quad \quad - s_2s_5^2\sqrt{s_6}) \qquad \qquad \qquad (327 \text{ terms}). \end{aligned} \tag{53}$$

Corollary 3. *If we replace $\sqrt{s_6}$ by $-\sqrt{s_6}$ in $\psi_6^{(+)}(Z)$, we obtain the other polynomial $\psi_6^{(-)}(Z)$, which should be deduced from $\text{Res}_d(F^{(-)}(d), G^{(-)}(Z, d))$.*

Corollary 4. *If we substitute $\sqrt{s_6}$ by 0 in $\psi_6^{(+)}(Z)$ and $\psi_6^{(-)}(Z)$, we obtain the pentagon formula $\psi_5(Z)$ in Eq. (24). Therefore, these three polynomials are represented uniformly through the crossing parity ε .*

If we consider the equilateral case, by putting $\forall a_i := 1$, equation $\psi_6^{(+)}(Z) = 0$ is reduced to

$$Z^6(Z - 108) = 0, \quad (54)$$

each factor of which respectively corresponds to the degenerated case with $S = 0$ (6-fold) and the case of a regular hexagon. If we put $\forall a_i := 1$ in the equation $\psi_6^{(-)}(Z) = 0$, we obtain

$$(Z - 4)(Z - 8)^6 = 0, \quad (55)$$

each factor of which respectively corresponds to a regular triangle and a regular square (6-fold). Therefore, we conclude that these polynomials $\varphi_5(z)$, $\psi_5(Z)$, $\psi_6^{(+)}(Z)$, and $\psi_6^{(-)}(Z)$ are the integrated formulae for cyclic pentagons and hexagons.

6 Concluding Remarks

We have conducted a detailed investigation into the relations among several geometric quantities of cyclic pentagons and hexagons, and succeeded in computing *integrated formulae* of the area and circumradius. Theorems 2 and 4 give polynomial equations in $Z = (4SR)^2$ for $n = 5, 6$, with the structure

$$Z^7 - (4s_3 - 28\varepsilon\sqrt{s_6})Z^6 + \dots = 0, \quad (56)$$

which is an extension of Eq. (13) for $n = 3, 4$. A polynomial in $(4SR)^2$ for pentagons such as Eq. (24) was previously described by Svrtan et al. [7], but their result seems to contain errors somehow. Hence, we consider that our result for cyclic pentagons represents a correction to theirs. In particular, we believe that the hexagon formula shown above has not been shown elsewhere.

Furthermore, we have also derived a pentagon formula in $z = 4SR$, as Theorem 3 (Eq. (34)):

$$|z|^7 - 2s_3|z|^5 - (s_1^2 + 4s_2)\sqrt{s_5}|z|^4 + \dots = 0,$$

which is a straightforward extension of case $n = 3$, Eq. (15). To the best of our knowledge, there exist no other reports that discuss the defining polynomial in $z = 4SR$ for cyclic n -gons with odd number n .

Even though we have obtained the specific formulae above, which have been unknown so far, the algorithm is rather naïve and requires too much CPU time for computing the resultant and factorization. Elimination by resultants inevitably yields extraneous factors, and removing them is often a heavy task. If we can analyze the geometric meaning of Eq. (56), we may consider a constructive and efficient algorithm such as that reported by Maley et al. [2] for area formulae. This should be clarified in a future study.

If we extend our results to the cases $n = 7, 8$, polynomial equations with degree 38 in $x = (4S)^2$, $y = R^2$, and $Z = xy$ will appear. For the case of heptagons ($n = 7$), a polynomial equation in $z = 4SR$ with degree 38 will also

exist. The present status of computations is as follows. The area formulae with elementary symmetric functions are computed according to Maley et al. [2]:

$$\begin{aligned} \tilde{\Phi}_S^{(7)}(x) &= x^{38} + \tilde{C}_{37}x^{37} + \cdots + \tilde{C}_1x + \tilde{C}_0 \quad (955,641 \text{ terms}), \\ &= 0 \quad \left(x = (4S)^2, \quad \tilde{C}_i \in \mathbf{Z}[s_1, \dots, s_7]\right), \\ \tilde{\Phi}_S^{(8^+)}(x) &= x^{38} + \tilde{D}_{37}x^{37} + \cdots + \tilde{D}_1x + \tilde{D}_0 \quad (3,248,266 \text{ terms}), \\ &= 0 \quad \left(x = (4S)^2, \quad \tilde{D}_i \in \mathbf{Z}[s_1, \dots, s_7, \sqrt{s_8}]\right). \end{aligned} \tag{57}$$

If we replace $\sqrt{s_8}$ by $-\sqrt{s_8}$ in $\tilde{\Phi}_S^{(8^+)}(x)$, we obtain $\tilde{\Phi}_S^{(8^-)}(x)$. We note also that substituting $\sqrt{s_8}$ by 0 in $\tilde{\Phi}_S^{(8^+)}(x)$ and $\tilde{\Phi}_S^{(8^-)}(x)$ gives $\tilde{\Phi}_S^{(7)}(x)$.

The circumradius formula for cyclic heptagons has been computed by the author [3] with the coefficients in a_i^2 :

$$\begin{aligned} \Phi_R^{(7)}(y) &= B_{38}y^{38} + \cdots + B_1y + B_0 \quad (337,550,051 \text{ terms}), \\ &= 0 \quad \left(y = R^2, \quad B_i \in \mathbf{Z}[a_1^2, \dots, a_7^2]\right). \end{aligned} \tag{58}$$

Since each coefficient is too large, it seems almost impossible to convert it into the requisite expression by elementary symmetric functions. The size of the circumradius formulae for cyclic octagons $\Phi_R^{(8^+)}(y)$ and $\Phi_R^{(8^-)}(y)$ should be larger than $\Phi_R^{(7)}(y)$, and they seem much harder to compute.

Therefore, direct computation of the integrated formula in $Z = (4SR)^2$ for $n = 7, 8$ seems impossible for the moment, because of the exploding size of these polynomials. Hence, the existence of polynomial equations in Z and z with degree 38 for cyclic heptagons and octagons is still a conjecture.

References

1. Fedorchuk, M., Pak, I.: Rigidity and polynomial invariants of convex polytopes. *Duke Math. J.* **129**(2), 371–404 (2005)
2. Maley, F.M., Robbins, D.P., Roskies, J.: On the areas of cyclic and semicyclic polygons. *Adv. Appl. Math.* **34**(4), 669–689 (2005)
3. Moritsugu, S.: Computing explicit formulae for the radius of cyclic hexagons and heptagons. *Bull. Japan Soc. Symbolic and Algebraic Comput.* **18**(1), 3–9 (2011)
4. Moritsugu, S.: Integrating circumradius and area formulae for cyclic pentagons. In: Hong, H., Yap, C. (eds.) *ICMS 2014*. LNCS, vol. 8592, pp. 214–221. Springer, Heidelberg (2014)
5. Pech, P.: Computations of the area and radius of cyclic polygons given by the lengths of sides. In: Hong, H., Wang, D. (eds.) *ADG 2004*. LNCS (LNAI), vol. 3763, pp. 44–58. Springer, Heidelberg (2006)
6. Robbins, D.P.: Areas of polygons inscribed in a circle. *Discrete Comput. Geom.* **12**(1), 223–236 (1994)
7. Svrtan, D., Veljan, D., Volenec, V.: Geometry of Pentagons: from Gauss to Robbins. *arXiv:math.MG/0403503 v1* (2004)

Extension of Simson–Wallace Theorem on Skew Quadrilaterals and Further Properties

Pavel Pech^(✉)

Faculty of Education, University of South Bohemia,
České Budějovice, Czech Republic
pech@pf.jcu.cz

Abstract. The paper deals with the extension of the well-known Simson–Wallace theorem on skew quadrilaterals in E^3 . We investigate locus of a point P whose orthogonal projections K, L, M, N onto the sides of a skew quadrilateral form a tetrahedron of a constant volume s . It is shown that the locus is a cubic surface G .

Further, some special cases of the locus for $s = 0$ are described, where the cubic surface is decomposed into a plane and a one-sheet hyperboloid or into three planes. The conjecture is stated that these cases are the only cases of reducibility of G .

Keywords: Simson–Wallace locus · Skew quadrilaterals · Elimination · Reducibility of a cubic surface

1 Introduction

The well-known Simson–Wallace theorem reads [3], Fig. 1:

Let K, L, M be orthogonal projections of a point P onto the sides of a triangle ABC . Then the locus of P such that K, L, M are collinear, is the circumcircle of ABC .

This theorem has several generalizations [4–10]. A generalization of the Simson–Wallace theorem which is by [2] ascribed to J. D. Gergonne is as follows, Fig. 2:

Let K, L, M be orthogonal projections of a point P onto the sides of a triangle ABC . Then the locus of P such that the area of the triangle KLM is constant, is the circle through P which is concentric with the circumcircle of ABC .

If we consider a tetrahedron $ABCD$ instead of a triangle ABC then we can investigate the locus of points $P \in E^3$ whose orthogonal projections onto the faces of $ABCD$ are coplanar or form a tetrahedron of a constant volume. This was studied in [6–10].

In this paper we extend the Gergonne’s generalization of SW theorem on skew quadrilaterals in Euclidean space E^3 . We search for the locus of a point P

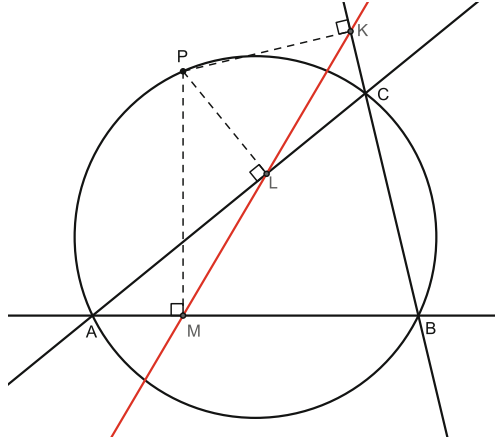


Fig. 1. Simson–Wallace theorem — points K, L, M are collinear

whose orthogonal projections K, L, M, N onto the sides on a skew quadrilateral $ABCD$ form a tetrahedron of a constant volume s . See [6, 7], where the case $s = 0$ was investigated and a necessary condition for the points K, L, M, N to be coplanar was stated. We'll find that the locus is a cubic surface G (2). It is shown that $P \in G$ is a necessary and sufficient condition for a tetrahedron $KLMN$ to be of the constant volume.

Further the reducibility of the locus for the volume $s = 0$ is studied with respect to a given skew quadrilateral. We conjecture that the only cases when the cubic surface G is reducible occur if two pairs of sides of a quadrilateral $ABCD$ (either adjacent or opposite) are of equal lengths.

Special attention is paid to the choice of a coordinate system so that we can show that the surface is really decomposable.

By searching for the locus and its properties we apply computer aided coordinate method based on Groebner bases computation and Wu–Ritt method using the software CoCoA [1] and the Epsilon library [11–13].

2 Extension on Skew Quadrilaterals

Consider a skew quadrilateral $ABCD$ in \mathbb{E}^3 and let K, L, M, N be orthogonal projections of a point P onto the sides of $ABCD$. We search for the locus of P such that the tetrahedron $KLMN$ has a constant volume. We'll prove the theorem:

Theorem 1. *Let K, L, M, N be orthogonal projections of a point P onto the sides AB, BC, CD, AD of a skew quadrilateral $ABCD$ respectively. Then the locus of P such that the tetrahedron $KLMN$ has a constant volume s is the cubic surface G (2).*

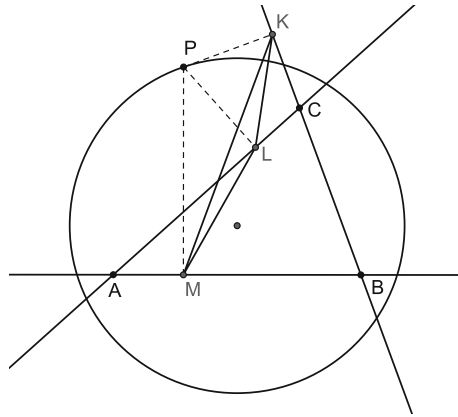


Fig. 2. Gergonne’s generalization of SW theorem — $\triangle KLM$ has a constant area

Proof. Place a skew quadrilateral $ABCD$ into a rectangular coordinate system in the following way $A = (-a, 0, 0)$, $B = (b, c, d)$, $C = (a, 0, 0)$, $D = (e, f, d)$ and $P = (p, q, r)$, Fig. 3.

Note that both diagonals AC and BD are parallel to the xy coordinate plane. This choice of the coordinate system enables to execute the decomposition of the locus into a plane and hyperboloid or into three planes without the use of radicals.

Further denote $K = (k_1, k_2, k_3)$, $L = (l_1, l_2, l_3)$, $M = (m_1, m_2, m_3)$, $N = (n_1, n_2, n_3)$ and $P = (p, q, r)$. Suppose that $ad(c - f) \neq 0$ since otherwise $ABCD$ is planar. Then:

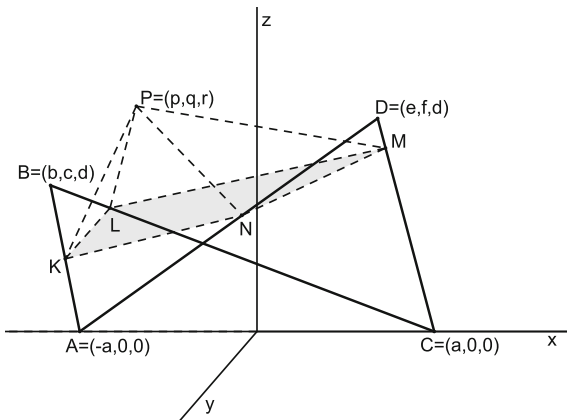


Fig. 3. SW extension on a skew quadrilateral $ABCD$ — K, L, M, N are coplanar

$$\begin{aligned}
 K \in AB &\Leftrightarrow h_1 := dk_2 - ck_3 = 0, \quad h_2 := d(k_1 + a) - (a + b)k_3 = 0, \\
 h_3 &:= c(k_1 + a) - (a + b)k_2 = 0, \\
 PK \perp AB &\Leftrightarrow h_4 := (p - k_1)(a + b) + (q - k_2)c + (r - k_3)d = 0, \\
 L \in BC &\Leftrightarrow h_5 := dl_2 - cl_3 = 0 \quad h_6 := d(l_1 - a) - (b - a)l_3 = 0, \\
 h_7 &:= c(l_1 - a) - (b - a)l_2 = 0, \\
 PL \perp BC &\Leftrightarrow h_8 := (p - l_1)(b - a) + (q - l_2)c + (r - l_3)d = 0, \\
 M \in CD &\Leftrightarrow h_9 := fm_2 - cm_3 = 0, \quad h_{10} := f(m_1 - a) - (e - a)m_3 = 0, \\
 h_{11} &:= c(m_1 - a) - (e - a)m_2 = 0, \\
 PM \perp BD &\Leftrightarrow h_{12} := (p - m_1)(e - a) + (q - m_2)c + (r - m_3)f = 0, \\
 N \in AD &\Leftrightarrow h_{13} := fn_2 - cn_3 = 0, \quad h_{14} := f(n_1 + a) - (e + a)n_3 = 0, \\
 h_{15} &:= c(n_1 + a) - (e + a)n_2 = 0, \\
 PN \perp AD &\Leftrightarrow h_{16} := (p - n_1)(e + a) + (q - n_2)c + (r - n_3)f = 0.
 \end{aligned}$$

Volume of $KLMN = s \Leftrightarrow$

$$h_{17} := \begin{vmatrix} k_1 & k_2 & k_3 & 1 \\ l_1 & l_2 & l_3 & 1 \\ m_1 & m_2 & m_3 & 1 \\ n_1 & n_2 & n_3 & 1 \end{vmatrix} - 6s = 0. \quad (1)$$

First suppose that the locus point P fulfills the conditions $h_1 = 0, h_2 = 0, \dots, h_{17} = 0$. We will find the locus equation.

Solving the system $h_1 = 0, h_2 = 0, h_3 = 0$ and $h_4 = 0$ we get

$$\begin{aligned}
 k_1 &= (p(a + b)^2 + qc(a + b) + rd(a + b) - a(c^2 + d^2))/((a + b)^2 + c^2 + d^2), \\
 k_2 &= (pc(a + b) + qc^2 + rcd + ac(a + b))/((a + b)^2 + c^2 + d^2), \\
 k_3 &= (pd(a + b) + qcd + rd^2 + ad(a + b))/((a + b)^2 + c^2 + d^2).
 \end{aligned}$$

Similarly,

$$\begin{aligned}
 l_1 &= (p(a - b)^2 - qc(a - b) - rd(a - b) + a(c^2 + d^2))/((a - b)^2 + c^2 + d^2), \\
 l_2 &= (-pc(a - b) + qc^2 + rcd + ac(a - b))/((a - b)^2 + c^2 + d^2), \\
 l_3 &= (-pd(a - b) + qcd + rd^2 + ad(a - b))/((a - b)^2 + c^2 + d^2), \\
 m_1 &= (p(a - e)^2 - qf(a - e) - rd(a - e) + a(d^2 + f^2))/((a - e)^2 + d^2 + f^2), \\
 m_2 &= (-pf(a - e) + qf^2 + rdf + af(a - e))/((a - e)^2 + d^2 + f^2), \\
 m_3 &= (-pd(a - e) + qdf + rd^2 + ad(a - e))/((a - e)^2 + d^2 + f^2), \\
 n_1 &= (p(a + e)^2 + qf(a + e) + rd(a + e) - a(d^2 + f^2))/((a + e)^2 + d^2 + f^2), \\
 n_2 &= (pf(a + e) + qf^2 + rdf + af(a + e))/((a + e)^2 + d^2 + f^2), \\
 n_3 &= (pd(a + e) + qdf + rd^2 + ad(a + e))/((a + e)^2 + d^2 + f^2).
 \end{aligned}$$

Substitution of k_1, k_2, \dots, n_3 into (1) gives the equation of a cubic surface

$$G := a^2d(f - c)H + sQ = 0, \quad (2)$$

where

$$\begin{aligned}
 H &= p^3(c^2(e^2 - a^2) - d^2(b^2 - e^2) + f^2(a^2 - b^2)) + 2p^2q(ef(c^2 + d^2) - bc(d^2 + f^2)) + \\
 &2p^2rd(e(c^2 + d^2) - b(d^2 + f^2)) + pq^2(f^2(a^2 - b^2 + d^2) - c^2(a^2 + d^2 - e^2)) + 2pqr(d(f(a^2 - \\
 &b^2 + c^2 + d^2) - c(a^2 + d^2 - e^2 + f^2)) - pr^2d^2(b^2 - c^2 - e^2 + f^2) + 2q^3cf(ce - bf) + \\
 &2q^2rd(ce(c + 2f) - bf(2c + f)) + 2qr^2d^2(e(2c + f) - b(c + 2f)) + 2r^3d^3(e - b) +
 \end{aligned}$$

$$\begin{aligned}
& p^2(e(c^2+d^2)(a^2-d^2-e^2-f^2)-b(d^2+f^2)(a^2-b^2-c^2-d^2))+pq(f(a^2-d^2-e^2-f^2)(a^2-b^2+c^2+d^2)-c(a^2+d^2-e^2+f^2)(a^2-b^2-c^2-d^2))+2prd(c^2(a^2-e^2)+d^2(b^2-e^2)-f^2(a^2-b^2))+q^2(ce-2bf)(a^2-d^2-e^2-f^2)+f(2ce-bf)(a^2-b^2-c^2-d^2))+2qrd((ce-bf+ef)(a^2-b^2-c^2-d^2)-(bc+bf-ce)(a^2-d^2-e^2-f^2))+r^2d^2((e-2b)(a^2-d^2-e^2-f^2)+(2e-b)(a^2-b^2-c^2-d^2))+pa^2(c^2(a^2-e^2)+d^2(b^2-e^2)-f^2(a^2-b^2))+q(2a^2(bc(d^2+f^2)-ef(c^2+d^2)))+(ce-bf)(a^2-d^2-e^2-f^2)(a^2-b^2-c^2-d^2))+rd((e-b)(a^2-d^2-e^2-f^2)(a^2-b^2-c^2-d^2)+2a^2(b(d^2+f^2)-e(c^2+d^2)))+a^2(b(d^2+f^2)(a^2-b^2-c^2-d^2)-e(c^2+d^2)(a^2-d^2-e^2-f^2))
\end{aligned}$$

and

$$Q = \frac{3}{2}((a-e)^2+d^2+f^2)((a+e)^2+d^2+f^2)((a+b)^2+c^2+d^2)((a-b)^2+c^2+d^2).$$

Remark 1. Note that Q in (2) is a constant which *does not* depend on p, q, r .

Remark 2. Realize that if $s = 0$ then the points K, L, M, N are coplanar and the locus is the surface $H = 0$.

Now we will prove the opposite implication. We want to show that if $P \in G$ then the volume of the tetrahedron $KLMN$ equals s .

The following proof is based on the well-known remainder formula (3) for successive pseudo-divisions of the conclusion polynomial g with respect to a triangular form f_1, f_2, \dots, f_r :

$$I_1^{s_1} I_2^{s_2} \cdots I_r^{s_r} g = Q_1 f_1 + Q_2 f_2 + \dots + Q_r f_r + R, \quad (3)$$

where R is the pseudo-remainder, and I_k are the leading coefficients of f_k in x_k [2, 14].

Denote

$$\begin{aligned}
P &:= \{h_1, h_2, \dots, h_{16}\}, \\
X &:= [f, e, d, c, b, a, r, q, p, s, k_1, k_2, k_3, l_1, l_2, l_3, m_1, m_2, m_3, n_1, n_2, n_3].
\end{aligned}$$

In Epsilon we first compute the characteristic set of $P \cup \{G\}$ with variable ordering X

```
with(charsets);
E:=charset(P union {G},X)
```

Then we express the pseudo-remainder R of h_{17} and get $R = 0$. Searching for initial polynomials by `iniset(E,X)` gives the set

$$\begin{aligned}
I_1 &= a + b, \\
I_1 &= a - b, \\
I_3 &= a + e, \\
I_4 &= a - e, \\
I_5 &= (a + b)^2 + c^2 + d^2, \\
I_6 &= (a - b)^2 + c^2 + d^2, \\
I_7 &= (a - e)^2 + d^2 + f^2, \\
I_8 &= (a + e)^2 + d^2 + f^2.
\end{aligned}$$

To prove the theorem we have to show that all initial polynomials I_1, I_2, \dots, I_8 are non-zero. Whereas the polynomials I_5, I_6, I_7, I_8 are obviously non-zero, the polynomials I_1, I_2, I_3, I_4 seem to be redundant. For instance, to get rid of $I_1 = a + b$ we add it to the set of polynomials $P \cup \{G\}$ and repeat the procedure. We get $R = 0$ with initial polynomials $\{c, f, c^2 + d^2, d^2 + f^2, d^2 + c^2 + 4b^2, d^2 + f^2 + e^2 + 2be + b^2, d^2 + f^2 - 2be + e^2 + b^2\}$.

Next we add c to the set of polynomials as above and get

$$\{d, f, d^2 + 4b^2, d^2 + f^2, d^2 + f^2 + e^2 + 2be + b^2, d^2 + f^2 - 2be + e^2 + b^2\},$$

and in the end after adding f to the set of polynomials we obtain the set

$$\{d, d^2 + 4b^2, d^2 + e^2 - 2be + b^2, d^2 + e^2 + 2be + b^2\}$$

which consists of non-zero polynomials. Now we have to add f to the set $\{G, a+b\}$ etc. with the same result. Similarly we proceed with other polynomials I_2, \dots, I_8 . The theorem is proved. \square

Remark 3. Notice the ordering of variables $f, e, d, c, b, a, r, q, p, s, \dots$ which was selected in this way to avoid more complicated subsidiary conditions.

3 Special Cases of the Locus

First let us look at some examples.

Example 1. For $a = 2, b = 0, c = 2, d = 2, e = 0, f = -2, s = 1$ we get a cubic surface Fig. 4.

$$16pqr - 16pq + 81 = 0,$$

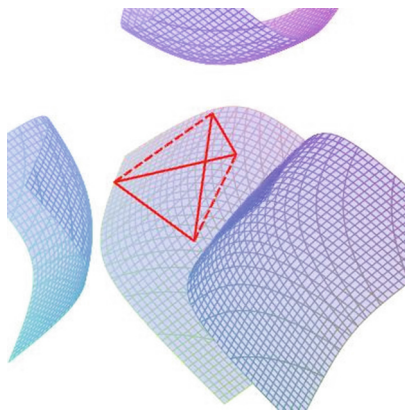


Fig. 4. Cubic surface $16pqr - 16pq + 81 = 0$ with $s = 1$

Example 2. For a skew quadrilateral $ABCD$ with $a = 1, b = 1, c = 1, d = 1, e = -1, f = -1$ and $s = 0$ we get a cubic surface

$$2p^2r + 2pqr - q^2r + r^3 - 2p^2 - 2pq + q^2 - 3r^2 + 2 = 0$$

which decomposes into a plane and a one-sheet hyperboloid, Fig. 5

$$(r - 1)(2p^2 + 2pq - q^2 + r^2 - 2r - 2) = 0.$$

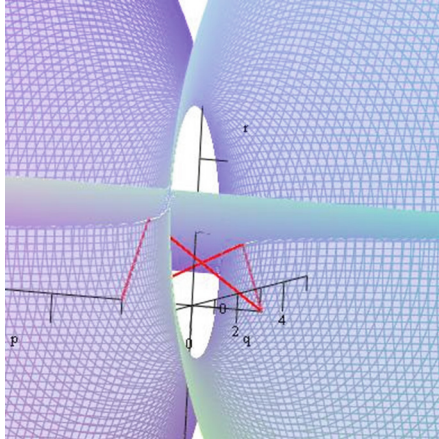


Fig. 5. Cubic surface is decomposed into a plane and one-sheet hyperboloid

Note that two pairs of opposite sides AB, CD and BC, AD are of equal lengths.

Example 3. For a skew quadrilateral $ABCD$ with $a = 1, b = 0, c = 1, d = 1, e = 0, f = -1$ and $s = 0$ we get a cubic surface

$$pq\left(r - \frac{1}{2}\right) = 0$$

which decomposes into three planes, Fig. 6. Note that all four sides of the quadrilateral $ABCD$ are of equal length.

Now we investigate properties which are related to the examples above. Suppose that $s = 0$. First we'll be concerned with the case when a cubic surface decomposes into a plane and a hyperboloid. We'll prove the theorem:

Theorem 2. *Let $ABCD$ be a skew quadrilateral with two pairs of adjacent sides of equal lengths m and $n, m \neq n$. Then the locus H (2) decomposes into a plane which bisects $ABCD$ and a one-sheet hyperboloid.*

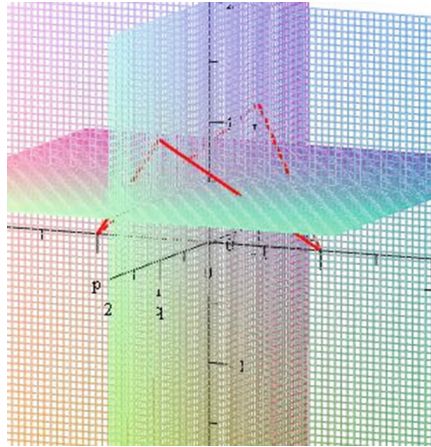


Fig. 6. Cubic surface is decomposed into three mutually orthogonal planes

Proof. Let $|AB| = |BC|$ and $|CD| = |AD|$. Then for the vertices $A = (-a, 0, 0)$, $B = (b, c, d)$, $C = (a, 0, 0)$ and $D = (e, f, d)$ the relations

$$b = 0 \quad \text{and} \quad e = 0$$

hold. Substitute these two values into $H = 0$ in (2)

Use $R := \mathbb{Q}[a, b, c, d, e, f, p, q, r]$;
 Subst(P , $[[b, 0], [e, 0]]$);

We get

$$p \cdot S = 0, \tag{4}$$

where

$$S := p^2 a^2(c + f) + q^2(c + f)(a^2 + d^2) - r^2 d^2(c + f) + 2qrd(a^2 + d^2 - cf) + q(a^4 - a^2(c + f)^2 - (d^2 + f^2)(c^2 + d^2)) - 2a^2rd(c + f) - a^4(c + f).$$

We see that (4) is the equation of a cubic surface which decomposes into a plane $p = 0$ which bisects $ABCD$, and a quadric $S = 0$. Let us explore the properties of the quadric $S = 0$. The discriminant Δ of S equals

$$\Delta = \frac{1}{4} a^2(c + f)^2 d^2 (a^2 + c^2 + d^2)^2 (a^2 + d^2 + f^2)^2$$

and the chief minor δ

$$\delta = -a^2(c + f) d^2 (a^2 + c^2 + d^2) (a^2 + d^2 + f^2).$$

Suppose that $c + f \neq 0$. Then S is a regular central quadric with three non zero eigenvalues λ_1, λ_2 and λ_3 which do not have the same sign. Actually

$$\lambda_1 = a^2(c + f)$$

and λ_2 and λ_3 are solutions of

$$\lambda^2 - \lambda a^2(c + f) - d^2(a^2 + c^2 + d^2)(a^2 + d^2 + f^2) = 0.$$

By the formulas of Viète $\lambda_2 \cdot \lambda_3 < 0$, hence λ_2 and λ_3 have opposite signs. Taking into account that λ_1 and Δ/δ have opposite signs, the quadric S must be a one-sheet hyperboloid. The theorem is proved. \square

In the Theorem 2 we considered $c + f \neq 0$. Suppose now that $c + f = 0$ and $b = 0, e = 0$. Then

$$|AB| = |BC| = |CD| = |AD|,$$

i.e., the skew quadrilateral $ABCD$ is equilateral and vice versa. For equilateral skew quadrilaterals the following theorem holds:

Theorem 3. *If a skew quadrilateral $ABCD$ is equilateral then the locus H (2) decomposes into three mutually orthogonal planes.*

Proof. Substitution $b = 0, e = 0$ and $f = -c$ into $H = 0$ in (2) gives

$$pq(a^2 - c^2 - d^2 + 2dr) = 0. \tag{5}$$

The cubic $H = 0$ decomposes by (5) into three planes $p = 0, q = 0$ and $a^2 - c^2 - d^2 + 2dr = 0$ which are mutually orthogonal. \square

Remark 4. This theorem can also be proved classically using the following consideration. A point P of the plane which bisects $ABCD$ (e.g. the plane through BD which is orthogonal to AC) obeys the condition that its projections K, L, M, N to the sides of $ABCD$ are coplanar since the segments KL and MN are parallel to AC and thus lie in a plane. The plane through AC which is orthogonal to BD has the same property. As the resulting surface is a cubic then the third plane remains.

The following theorem describes the case when two pairs of *opposite* sides of a skew quadrilateral are of equal lengths.

Theorem 4. *Let $ABCD$ be a skew quadrilateral with two pairs of opposite sides of equal lengths m and $n, m \neq n$. Then the locus H (2) decomposes into a plane and a one-sheet hyperboloid.*

Suppose that two pairs of opposite sides of $ABCD$ are of equal lengths, i.e. $|AB| = |CD|$ and $|BC| = |DA|$. Then

$$b + e = 0 \quad \text{and} \quad c^2 - f^2 = 0.$$

It is obvious that $c - f = 0$ leads to a planar $ABCD$ and we can rule it out. Substitution of $e = -b$ and $f = -c$ into $H = 0$ (2) gives

$$(a^2 - b^2 - c^2 - d^2 + 2dr) \cdot T = 0, \tag{6}$$

where

$$T = p^2b(c^2 + d^2) - bc^2q^2 + bd^2r^2 + pqc(a^2 - b^2 + c^2 + d^2) + rbd(a^2 - b^2 - c^2 - d^2) - a^2b(c^2 + d^2).$$

Thus the cubic surface H decomposes by (6) into a plane $a^2 - b^2 - c^2 - d^2 + 2dr = 0$ and a quadric $T = 0$. The determinant Δ of T equals

$$\Delta = \frac{1}{16}b^2c^2d^2((a + b)^2 + c^2 + d^2)^2((a - b)^2 + c^2 + d^2)^2$$

and the chief minor δ

$$\delta = -\frac{1}{4}bc^2d^2((a - b)^2 + c^2 + d^2)((a + b)^2 + c^2 + d^2).$$

If $b \neq 0, c \neq 0, d \neq 0$ then T is a regular central quadric with three non zero eigenvalues λ_1, λ_2 and λ_3 which do not have the same sign. Actually

$$\lambda_1 = bd^2$$

and λ_2 and λ_3 are solutions of

$$4\lambda^2 - 4bd^2\lambda - c^2((a - b)^2 + c^2 + d^2)((a + b)^2 + c^2 + d^2) = 0.$$

By the formulas of Viète $\lambda_2 \cdot \lambda_3 < 0$ hence $sgn\lambda_2 \neq sgn\lambda_3$. Taking into account that λ_1 and Δ/δ have opposite signs, the quadric T must be a one-sheet hyperboloid. The theorem is proved. \square

Remark 5. The coordinate system which was used in the proof of the Theorems 2, 3, 4 is very specific. In this coordinate system the diagonal AC is in the x -axis and the second diagonal BD is parallel to coordinate plane xy . This enables to execute the decomposition of a cubic into a plane and a hyperboloid or into three planes. In general such decomposition is quite difficult because of the appearance of radicals.

Remark 6. It seems that the cases above are the only cases when the cubic surface H decomposes either into a plane and one-sheet hyperboloid or into three planes.

On the basis of this remark we can state a conjecture:

Conjecture 1. The Simson–Wallace locus which is a cubic surface H (2) is decomposable iff two pairs of sides of a skew quadrilateral $ABCD$ are of equal lengths.

Concluding Remarks

For the construction of the generalization above it is not essential that the four edges form a skew quadrilateral. One could treat the case of four skew given lines

a, b, c, d and ask for the pedal points K, L, M, N of a point P to be coplanar. It would be interesting to know if it makes a difference whether the given lines are generators of a regulus or not. An open question is whether the cases from the Theorems 2, 3 and 4 are all the cases when we get a reducible cubic or not? Finally, the extension of the Simson–Wallace theorem on skew closed $(n+1)$ -gons in \mathbb{E}^n appears to be feasible.

Acknowledgements. The author wish to thank the referees for their valuable and helpful suggestions.

References

1. Capani, A., Niesi, G., Robbiano, L.: CoCoA, a System for Doing Computations in Commutative Algebra. <http://cocoa.dima.unige.it>
2. Chou, S.C.: Mechanical Geometry Theorem Proving. D. Reidel Publishing Company, Dordrecht (1987)
3. Coxeter, H.S.M., Greitzer, S.L.: Geometry revisited. Toronto New York (1967)
4. Giering, O.: Affine and projective generalization of Wallace lines. *J. Geom. Graph.* **1**, 119–133 (1997)
5. Guzmán, M.: An extension of the Wallace-Simson theorem: projecting in arbitrary directions. *Amer. Math. Monthly* **106**, 574–580 (1999)
6. Pech, P.: On Simson-Wallace theorem and its generalizations. *J. Geom. Graph.* **9**, 141–153 (2005)
7. Pech, P.: Selected Topics in Geometry with Classical vs Computer Proving. World Scientific, Singapore New Jersey (2007)
8. Pech, P.: On a 3D extension of the Simson-Wallace theorem. *J. Geom. Graph.* **18**, 205–215 (2014)
9. Riesinger, R.: On Wallace Loci from the projective point of view. *J. Geom. Graph.* **8**, 201–213 (2004)
10. Roanes-Macías, E., Roanes-Lozano, E.: Automatic determination of geometric Loci. 3D-extension of Simson-Steiner theorem. In: Campbell, J., Roanes-Lozano, E. (eds.) AISC 2000. LNCS (LNAI), vol. 1930, pp. 157–173. Springer, Heidelberg (2001)
11. Wang, D.: Epsilon: a library of software tools for polynomial elimination. In: Cohen, A., Gao, X.S., Takayama, N. (eds.) *Mathematical Software*, pp. 379–389. World Scientific, Singapore (2002). <http://www-calfor.lip6.fr/~wang/epsilon/>
12. Wang, D.: *Elimination Methods. Texts and Monographs in Symbolic Computation.* Springer, Wien New York (2001)
13. Wang, D.: *Elimination Practice: Software Tools and Applications.* Imperial College Press, London (2004)
14. Wu, W.-T.: On the decision problem and the mechanization of theorem-proving in elementary geometry. *Scientia Sinica* **21**, 159–172 (1978)

Current Status of the I2GATP Common Format

Pedro Quaresma¹(✉) and Nuno Baeta²

¹ CISUC/Department of Mathematics, University of Coimbra,
Coimbra, Portugal
`pedro@mat.uc.pt`

² ISEC, Polytechnic Institute of Coimbra,
Coimbra, Portugal
`nmsb@isec.pt`

Abstract. The I2GATP format is an extension of the I2G (Intergeo) common format aimed to support conjectures and proofs produced by geometric automatic theorem provers. The goal in building such a format is to provide a communication channel between different tools from the field of geometry, allowing linking such tools, as well as allowing the use of geometric knowledge kept in different repositories.

In this article we report the current status of the I2GATP format and its accompanying components: the XSD files with the specification of the format; the C++ library to create the container with all the information regarding a geometric problem or to break it into its components; the filters to convert from/to geometric tools formats to/from I2GATP; the integration with repositories of geometric knowledge.

1 Introduction

The I2GATP format is an extension of the I2G (Intergeo) common format [14] aimed to support conjectures and proofs produced by geometric automatic theorem provers (GATP). As such, it is to be used by tools from the field of geometry, allowing its linking: in geometric knowledge repositories like TGTP [11] and GeoThms [4] as a common format; in a learning environment for geometry like the Web Geometry Laboratory (WGL) [13, 15, 16] connecting geometric tools, allowing the formal validation of the geometric constructions and even the introduction of formal proofs in a learning environment.

In this article we report the current status of the I2GATP format, i.e., the XSD¹ files, the programs to build the container and break it into its components; the filters between the format and the different geometric tools; integration within repositories of geometric knowledge, learning environments for geometry and other “clients” of geometric knowledge.

The XSD files contain the specification of the format: `information.xsd` with the meta-information about a given geometric problem; `intergeo.xsd` no more than the XSD for the I2G format; `conjecture.xsd` with the specification of the conjectures and `proofInfo.xsd` with the meta-information about the proof(s).

¹ An XML Schema (XSD) file describes the structure of an XML document.

All the XML files containing the information about a geometric problem and also other auxiliary files, are packaged in the I2GATP container, an extension of the I2G container. The packaging and also the opposite operation of breaking-off the container into its components is a part of the auxiliary library built to support the I2GATP format.

To allow the linking of geometric tools, filters from/to the I2GATP need to be implemented. We begin by considering the filters needed to convert from/to languages of the GATPs contained in the TGTP repository to/from the I2GATP format.

Having made the previous steps, one last step is the integration of all this in repositories of geometric knowledge such as TGTP [11] and GeoThms [4]. It will allow the use of the common format to store all the information regarding one given geometric problem and then break it into components, e.g., to feed a conjecture to a given GATP. In the opposite direction, we can build the I2GATP container for any problem contained in the TGTP repository.

From the previous report about this project [12], the XSD and the container specifications have been improved; the open source library was built and provides now some of the filters needed alongside with the programs to manage the container file; and the integration with the TGTP system is underway.

Related Work. The geometry description language (GDL) propose by Chen [1] aims to convert a natural (mathematical) language description of geometric problems, as found in the literature, to an equivalente in a formal language that can be automatically processed and convert to system-native representations. The interconnections between this work and ours has to be explored.

Paper Overview. In Sect. 2 the overall structure and the status of the format are described. In Sect. 3 questions about the container are described. In Sect. 4 the implementation of the library and its integration with repositories of geometric problems and other systems are described. Finally in Sect. 5 some final conclusions are drawn and future work is discussed.

2 The I2GATP Format

The Intergeo (I2G) file format is a specification based on the markup language XML designed to describe constructions created with a DGS. It is one of the main results of the intergeo project, an eContentplus European project dedicated to the sharing of interactive geometry constructions across boundaries. For more information about the project, visit the site <http://i2geo.net> and look into the documentation available there, as well as to [6, 7].

An intergeo file takes the form of a compress file package. The main file is `intergeo.xml`, which provides a textual description of the construction in three parts, the elements part describing a (static) initial instance of the configuration, the constraints part where the geometric relationships are expressed and the display part where the details regarding the rendering of the construction are placed. For more details on the file format see [14].

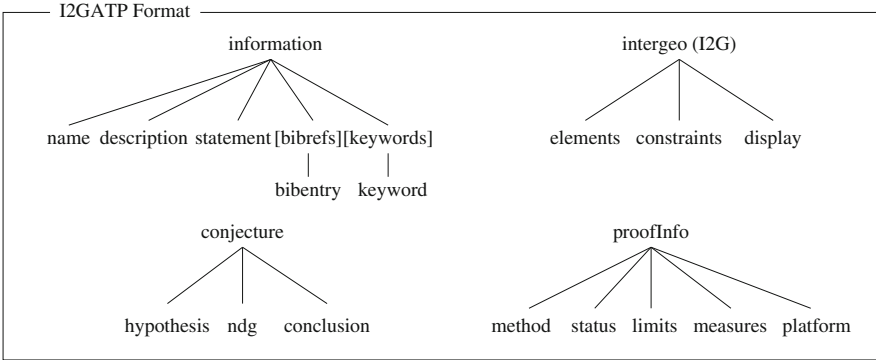


Fig. 1. Structure of the I2GATP File Format

The I2GATP Format is thought as an extension of the I2G common format aimed to support conjectures and proofs produced by GATPs and to be used in repositories of geometric knowledge. Using the information already contained in the TGTP database as a template, the I2GATP format is a combination of four XSD files (see Fig. 1): `information.xsd`; `intergeo.xsd`; `conjecture.xsd` and `proofInfo.xsd`.

Information. The XSD file `information.xsd` (see Listing 1.1) contains all the information regarding the conjecture. This XSD format mirrors the information contained in the TGTP database.

For the `statement` tag the MathML XSD format is used. For the `bibrefs` tag, a list of bibliographic references, the file `bibtexml.xsd`² from the `BIBTEXML` project is used. Conversion tools from `LATEX` and `BIBTEX` to the corresponding XML file are to be used, for example, `tex4ht`³ and `BIBTEXML` project converters respectively.

Listing 1.1. Fragment of `information.xsd`

```

<!-- information -->
<xs:element name="information">
  <xs:complexType>
    <xs:all>
      <xs:element ref="conjecture_id"/>
      <xs:element ref="conjecture_name" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="submission_date" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="level" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="description" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="statement" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="bibrefs" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="keywords" minOccurs="0" maxOccurs="1"/>
    </xs:all>
  </xs:complexType>
</xs:element>

```

² <http://bibtexml.sourceforge.net/>.
³ <http://tug.org/applications/tex4ht/>.

Construction. This is the `intergeo.xsd` file format from the Intergeo I2G format.

As we will explain below (see Sect. 4) one of the I2GATP project's goals is to provide filters from/to the DGS/GATP (at least the ones presented in TGTP) languages to/from this format.⁴

Conjecture. The XSD file `conjecture.xsd` is a work-in-progress (see Listing 1.2). For now, it is a GCLC Area Method related format file, i.e. it is the translation of `conjecture` tag of the DTD file `geocons.dtd` from the XML suite incorporated in that GATP to the XSD format [2–4, 10].

Listing 1.2. Fragment of `conjecture.xsd`

```

<!-- conjecture -->
<xs:element name="conjecture">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="prove" minOccurs="0" />
    </xs:sequence>
    <xs:attribute ref="conjName" use="required" />
  </xs:complexType>
</xs:element>

<xs:element name='prove'>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref='xs:equality' />
    </xs:sequence>
    <xs:attribute name='proof_limit' type='string' use='required' />
    <xs:attribute name='proof_level' type='string' use='required' />
  </xs:complexType>
</xs:element>

<xs:element name='equality'>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref='xs:expression' />
      <xs:element ref='xs:expression' />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name='expression'>
  <xs:complexType>
    <xs:choice>
      <xs:element ref='xs:number' />
      <xs:element ref='xs:constant' />
      <xs:element ref='xs:sum' />
      <xs:element ref='xs:mult' />
      <xs:element ref='xs:fraction' />
      <xs:element ref='xs:segment_ratio' />
      <xs:element ref='xs:signed_area3' />
      <xs:element ref='xs:pythagoras_difference3' />
      <xs:element ref='xs:identical_points' />
      <xs:element ref='xs:collinear' />
    </xs:choice>
  </xs:complexType>
</xs:element>

```

⁴ Given the fact that we are more concern in the specification of a geometric construction and less in its rendering, the focus of the filters is in the construction's specification.

ProofInfo The XSD file `proofInfo.xsd` (see Listing 1.3) contains the meta-information regarding the proof generated by a given GATP on a given computing platform.

For the proofs themselves we are not considering a unique format. Given the fact that the GATPs using the area method or the full-angle method, the ones using coherent logic, or the ones using algebraic methods, all have very different proof formats, we do not see as possible to have some sort of confluence of those formats into a common format. Having that in mind the proof, if present, is included in the I2GATP container file, in the GATP own proof output format (see Sect. 3).

Listing 1.3. Fragment of `proofInfo.xsd`

```

<!--
  proofInfo - all the information about the proof, the gatp used
  (name & method) is mandatory all the other information is optional.
-->
<xs:element name="proof_info">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="gatp"/>
      <xs:element ref="status" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="limits" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="measures" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="platform" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

3 The I2GATP Container

The I2GATP container is an extension of the I2G container. In addition to the information in the I2G container, all the information regarding the geometric conjecture and all the proofs attempts are kept in the I2GATP container (see Table 1).

The public methods for the construction/deconstruction of the container are already implemented in the I2GATP library (see Sect. 4). The public methods allow deconstructing the container in its components and, in the opposite direction, building the container, having its components. The public method `addProofInfo` allow adding a new proof to an existing container (see Fig. 2).

4 Implementation

The I2GATP library is an open source project,⁵ implemented in C++, to support the I2GATP common format. The C++ classes are: `I2GATP`, to support the management of the I2GATP container; `FilterDGS/GATPtoI2GATP` each implementing a filter for a given `DGS/GATP`; `TGTPtoI2GATP` containing the

⁵ <http://itogatplibrary.sourceforge.net>.

Table 1. The I2GATP container

information/	mandatory
information/information.xml	optional
construction/	mandatory
(...)	
conjecture/	mandatory
conjecture/conjecture.xml	optional
proofs/	mandatory
proofs/proof<GATP><Version><Method>/	optional
proofs/proof<GATP><Version><Method>/proofInfo.xml	optional
proofs/proof<GATP><Version><Method>/proof.xml	optional
proofs/proof<GATP><Version><Method>/(...)	optional
metadata/	optional
(...)	
resources/	optional
resources/<image_files>	optional
resources/dgs<DGS><Version>/	optional
resources/dgs<DGS><Version>/<dgscode>	optional
resources/gatp<GATP><Version><Method>/	optional
resources/gatp<GATP><Version><Method>/<gatpcode>	optional
resources/(...)	optional
private/	optional
(...)	

methods to deal with the integration of the I2GATP format in the TGTP repository.

The I2GATP class aggregate all the filter classes. The TGTPtoI2GATP inherits from the I2GATP class, expanding this last one with the methods necessary to link TGTP and the I2GATP format (see Fig. 2).

The I2GATP library depends on the following libraries: `libzip`,⁶ a library to deal with `zip` archives, placing a list of files in a `zip` and in the opposite direction to open the `zip` archive; `boost.system` and `boost.filesystem`, libraries⁷ to deal with files and directories. Given the fact that TGTP uses a MySQL database management system, the TGTPtoI2GATP class depends from the `mysqlcppconn` library⁸ to make the connection with the MySQL database.

All the code is assembled in a dynamic library `libi2gatp.so`. Apart from the library, the program `callMakeI2GATP` is also created. It is used by the TGTP repository to build the I2GATP container for a given geometric conjecture and,

⁶ <http://www.nih.at/libzip/>.

⁷ <http://www.boost.org/doc/libs/1.57.0/libs/filesystem/doc/index.htm>.

⁸ <http://dev.mysql.com/downloads/connector/cpp/>.

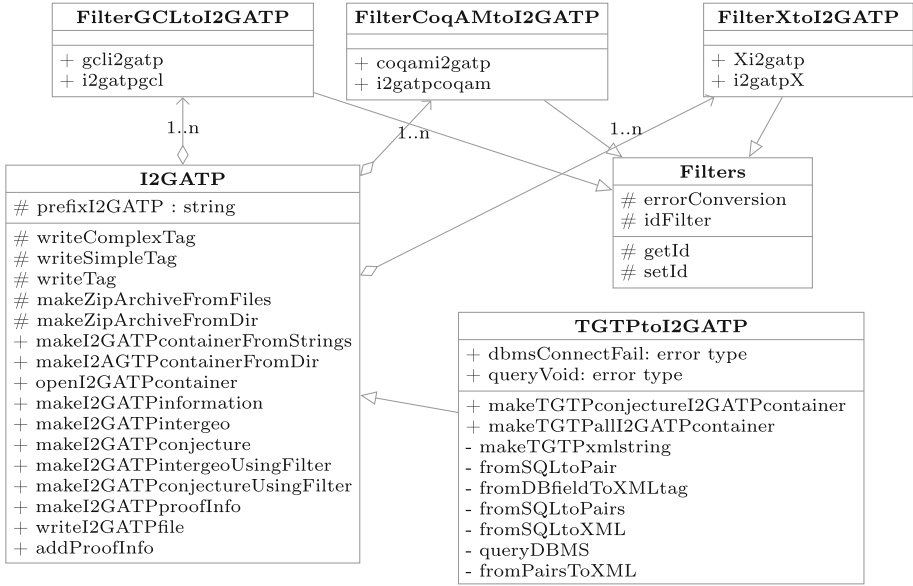


Fig. 2. UML Diagram for the I2GATP Library

in this way, to provide the TGTP’s user all the info about that conjecture, packed in a unique file.

4.1 I2GATP Filters

As a first step we aim to support the I2GATP format in TGTP [11], GeoThms [4] and WGL [13, 15, 16]. For that purpose we need filters from/to GCLC,⁹ CoqAM¹⁰ [2, 3, 8] and also GeoGebra.¹¹ The class I2GATP (see Fig. 2) will use the filter’s classes, e.g., FilterGCLctoI2GATP, in order to convert from/to the DGS/GATP code to/from the I2GATP format. As said previously, we are more concern with the specification of the construction than with its rendering. Because of that the filters are built aiming to fully support the constructions’ specification, disregarding, eventually, some of the rendering features.

The FilterGCLctoI2GATP and FilterCoqAMtoI2GATP are essentially wrappers for the filters built using lexical analyser and parser tools, e.g., flex¹² and bison.¹³ The integration of other filters, eventually built with other tools, should follow the same guidelines, a function with one input string, with the DGS/GATP’s code, and three output strings, corresponding to intergeo.xml,

⁹ <http://poincare.matf.bg.ac.rs/~janicic/gcl/>.
¹⁰ GeoProof: <http://home.gna.org/geoproof/>.
¹¹ <http://www.geogebra.org/>.
¹² <http://flex.sourceforge.net/>.
¹³ <http://www.gnu.org/software/bison/>.

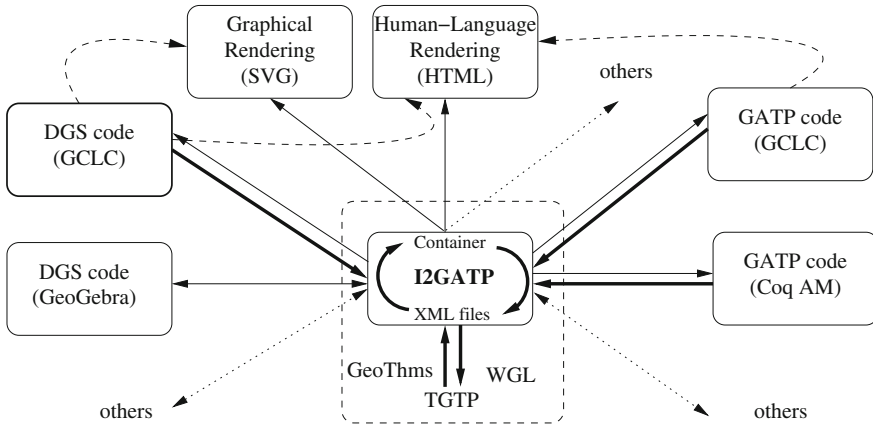


Fig. 3. Conversions From/To I2GATP To/From Geometric Tools

`conjecture.xml` and the report of the filter’s execution (*log* string). The class `Filters` defines a set of error types common to all the filters and an identification number to identify, if needed, each of the filters.

4.2 Integration Issues

The integration of this library into other programs is already being done (see Fig. 3). The class `TGTPtoI2GATP` provides the public methods (see Fig. 2) to build a container for a given problem or to do it for all the problems in TGTP.

The full integration will happen when the TGTP database will keep all the information using the I2GATP format. Then, using the filters and the methods already described, it will be possible to link the geometric information with the DGSs and GATPs.

The integration with GeoThms will be similar and will be done as soon as the integration with TGTP is completed.

A different set of goals are present when we consider the integration with a system like the Web Geometry Laboratory. This is a collaborative and adaptive blended-learning Web platform [13, 15, 16]. It is intended to allow verifying the geometric constructions deductively [5] and, because of that, the I2GATP format will be useful. In this case only the connection between the DGSs and the GATPs will be needed.

In Fig. 3 we can see the current status (“fat” arrows) and the planned work. The dashed arrows refer to a previous XML suite [10] which is integrated in GCLC.

5 Conclusions and Future Work

A format like I2GATP, allowing the connection between geometric tools, is important to open the geometric knowledge contained in repositories to all the

community. It is important to be able to connect different tools in geometric environments, namely learning environments.

The main tasks ahead are:

- the (re)definition of the conjecture specification. We are planning to define a common format for this. Stojanović et al. propose in “*A Vernacular for Coherent Logic*” [17], a simple yet expressive proof representation from which proofs for different proof assistants can easily be generated. This format is designed for conjectures in Coherent Logic and for forward chaining proofs (often used in common mathematical practice). Von Plato writes in *The axioms of constructive geometry* [9], that the general form of a geometric problem can be stated as: for a given data x of type A , find y such that condition $C(x, y)$ is fulfilled, with the type of y dependent of x , denoted $B(x)$. We are considering rewriting the XSD file in such a way that we can state the geometric problems in a more general form, following the ideas in [9, 17], but allowing specific XML tags, for different theories introduced by the different GATPs. Also the work of Chen [1] has to be considered in terms of exploiting his geometric description language, and the transformation mechanisms provided, to allow complex changes in representations, e.g., as said in [1], to exchange from a predicate form to a constructive form.
- the definition of an application programming interface (API), fully documented, to support an easy integration of the I2GATP format in different platforms. This should be complemented by an Web interface to exemplify how to extend/apply the I2GATP library. We will use our experience with the TGTP integration to achieve this goal;
- the construction of filters to/from other geometric tools. Having set up the I2GATP library as a *sourceforge*¹⁴ open source project we hope that other developers of geometric software can contribute writing their own filters and incorporating them in the library.

Having already done some work, much more is still in need to be done.

References

1. Chen, X.: Representation and automated transformation of geometric statements. *J. Syst. Sci. Complex.* **27**(2), 382–412 (2014)
2. Janičić, P.: GCLC — a tool for constructive euclidean geometry and more than that. In: Iglesias, A., Takayama, N. (eds.) ICMS 2006. LNCS, vol. 4151, pp. 58–73. Springer, Heidelberg (2006)
3. Janičić, P., Narboux, J., Quresma, P.: The area method: a recapitulation. *J. Autom. Reasoning* **48**(4), 489–532 (2012)

¹⁴ <http://sourceforge.net/about>.

4. Janičić, P., Quaresma, P.: System description: GCLCprover + Geothms. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 145–150. Springer, Heidelberg (2006)
5. Janičić, P., Quaresma, P.: Automatic verification of regular constructions in dynamic geometry systems. In: Botana, F., Recio, T. (eds.) ADG 2006. LNCS (LNAI), vol. 4869, pp. 39–51. Springer, Heidelberg (2007)
6. Kortenkamp, U., Blessing, A.M., Dohrmann, C., Kreis, Y., Libbrecht, P., Mercat, C.: Interoperable interactive geometry for europe-first technological and educational results and future challenges of the intergeo project. In: CERME, vol. 6 (2006)
7. Kortenkamp, U., Dohrmann, C., Kreis, Y., Dording, C., Libbrecht, P., Mercat, C.: Using the intergeo platform for teaching and research. In: Proceedings of the 9th International Conference on Technology in Mathematics Teaching (ICTMT-9) (2009)
8. Narboux, J.: Formalization of the area method. Coq user contribution (2009). http://dpt-info.u-strasbg.fr/narboux/area_method.html
9. von Plato, J.: The axioms of constructive geometry. *Ann. Pure Appl. Logic* **76**, 169–200 (1995)
10. Quaresma, P., Janičić, P., Tomašević, J., Vujošević-Janičić, M., Tošić, D.: XML-bases format for descriptions of geometric constructions and proofs. In: *Communicating Mathematics in The Digital Era*, pp. 183–197. A. K. Peters, Ltd., Wellesley (2008)
11. Quaresma, P.: Thousands of geometric problems for geometric theorem provers (TGTP). In: Schreck, P., Narboux, J., Richter-Gebert, J. (eds.) ADG 2010. LNCS, vol. 6877, pp. 169–181. Springer, Heidelberg (2011)
12. Quaresma, P.: An XML-format for conjectures in geometry. In: Davenport, J., Jeuring, J., Lange, C., Libbrecht, P. (eds.) *CEUR Workshop Proceedings of 24th OpenMath Workshop, 7th Workshop on Mathematical User Interfaces (MathUI), and Work in Progress*, vol. 921, pp. 54–65, Aachen (2012)
13. Quaresma, P., Santos, V., Bouallegue, S.: The web geometry laboratory project. In: Carette, J., Aspinall, D., Lange, C., Sojka, P., Windsteiger, W. (eds.) *CICM 2013*. LNCS, vol. 7961, pp. 364–368. Springer, Heidelberg (2013)
14. Santiago, E., Hendriks, M., Kreis, Y., Kortenkamp, U., Marquès, D.: i2g Common File Format Final Version. Technical report. D3.10, The Intergeo Consortium (2010). <http://i2geo.net/xwiki/bin/view/I2GFormat/>
15. Santos, V., Quaresma, P.: Integrating DGSs and GATPs in an adaptative and collaborative blended-learning Web-environment. In: *First Workshop on CTP Components for Educational Software (THedu 2011)*. EPTCS, vol. 79, pp. 111–123 (2012)
16. Santos, V., Quaresma, P.: Collaborative aspects of the WGL project. *Mathematics and Technology LLC. Electron. J. Math. Technol.* **7**(6) (2013). <https://php.radford.edu/~ejmt/ContentIndex.php#v7n6>
17. Stojanović, S., Narboux, J., Bezem, M., Janičić, P.: A vernacular for coherent logic. In: Watt, S.M., Davenport, J.H., Sexton, A.P., Sojka, P., Urban, J. (eds.) *CICM 2014*. LNCS, vol. 8543, pp. 388–403. Springer, Heidelberg (2014)

On Flattenability of Graphs

Meera Sitharam and Joel Willoughby^(✉)

University of Florida, Gainesville, Florida
jw5@cise.ufl.edu

Abstract. We consider a generalization of the concept of d -flattenability of graphs - introduced for the l_2 norm by Belk and Connelly - to general l_p norms, with integer P , $1 \leq p < \infty$, though many of our results work for l_∞ as well. The following results are shown for graphs G , using notions of genericity, rigidity, and generic d -dimensional rigidity matroid introduced by Kitson for frameworks in general l_p norms, as well as the cones of vectors of pairwise l_p^p distances of a finite point configuration in d -dimensional, l_p space: (i) d -flattenability of a graph G is equivalent to the convexity of d -dimensional, inherent Cayley configurations spaces for G , a concept introduced by the first author; (ii) d -flattenability and convexity of Cayley configuration spaces over specified non-edges of a d -dimensional framework are not generic properties of frameworks (in arbitrary dimension); (iii) d -flattenability of G is equivalent to all of G 's generic frameworks being d -flattenable; (iv) existence of one generic d -flattenable framework for G is equivalent to the independence of the edges of G , a generic property of frameworks; (v) the rank of G equals the dimension of the projection of the d -dimensional stratum of the l_p^p distance cone. We give stronger results for specific norms for $d = 2$: we show that (vi) 2-flattenable graphs for the l_1 -norm (and l_∞ -norm) are a larger class than 2-flattenable graphs for Euclidean l_2 -norm case and finally (vii) prove further results towards characterizing 2-flattenability in the l_1 -norm. A number of conjectures and open problems are posed.

1 Introduction, Preliminaries, Contributions

A *realization or framework of a graph* $G = (V, E)$ under norm $\|\cdot\|$ is an assignment $r : V \rightarrow \mathbb{R}^m$ of points in the corresponding normed vector space \mathbb{R}^m . A *linkage* (G, δ_G) is a graph $G = (V, E)$ together with an assignment $\delta_G : E \rightarrow \mathbb{R}$ of positive real assignments of lengths to the edges of G . A *realization of a linkage* (G, δ^G) in d dimensional $\|\cdot\|$ -normed space is an assignment $r : V \rightarrow \mathbb{R}^d$, such that $\forall (v, w) \in E, \|r(v) - r(w)\| = \delta_{vw}^G$. A realization under norm $\|\cdot\|$ is a realization in d dimensional $\|\cdot\|$ -normed space, for some dimension d . In this paper, we are concerned with standard l_p norms. By *general l_p norms*, we mean norms with integer p , $1 \leq p < \infty$. However, many results of this paper hold for l_∞ as well. While under the l_2 norm a realization r of intrinsic dimension d - i.e., whose points lie on a d -dimensional subspace of some higher d' -dimensional space - is

M. Sitharam—This research was supported in part by the grant NSF CCF-1117695

linearly isometric to a d -dimensional realization, this is not the case for other l_p norms unless the subspace is axis parallel, i.e., a coordinate subspace or a translated (affine) subspace. Hence the dimension of such a realization r under general norms is considered to be d' rather than d . A graph G is d -*flattenable* if for every realization r of G under norm $\|\cdot\|$, the linkage (G, δ^G) where $\delta_{vw}^G := \|r(v) - r(w)\|$. This is an illustration of a 2-flattenable graph that does not refer to realizations of intrinsic dimension 2 in some higher dimensional space. It also has a realization in the d -dimensional $\|\cdot\|$ -normed space. This definition does not imply that there is a continuous path of realizations starting from a realization of (G, δ_G) in some higher dimension to the realization in d -dimensions, nor does it refer to realizations of intrinsic dimension d in some higher dimensional space. For a clarification of the latter, see the example in the Proof of Theorem 11 where we give a realization of a graph on a 2-dimensional subspace of \mathbb{R}^3 , this does not imply that the graph is 2-flattenable. This particular graph turns out to be 2-flattenable as every l_1 realizable linkage of it can be realized in \mathbb{R}^2 .

This concept was first introduced in [9] for the Euclidean or l_2 norm. However they called it “ d -realizability,” which can be confused with the realizability of a given linkage in d -dimensions. This is one of the reasons we introduced the term: *flattenability*.

The term flattening has also been used by Matousek [18] in the context of non-isometric embeddings (with low distortion via Johnson-Lindenstrauss lemma in l_2 [19], impossibility of low distortion in l_1 [10], etc.). Our paper admits arbitrary distortions of non-edge lengths, but forces edge lengths to remain undistorted.

A *minor* of G is any graph G' that can be obtained from G from a series of edge-contractions or edge-deletions. If a property of G remains consistent under the operation of taking minors, that property is *minor-closed*. A useful result due to [25] is that if a property is minor-closed, then there is a finite set of *forbidden minor* \mathcal{F} such that if G has any element of \mathcal{F} as a minor, then G does not have that property.

Immediately by definition, d -flattenability is a minor-closed property under any norm. A full characterization for 3-flattenable graphs was given for the Euclidean or l_2 norm by [9].

This paper gives basic results illustrating how d -flattenability for general norms is a natural link between combinatorial rigidity and configuration spaces of frameworks on the one hand, and coordinate shadows (projections) of the faces of the cone – consisting of vectors of pairwise l_p^p -distances of n -point configurations (see Fig. 1) – on the other hand (see Fig. 2). We define the l_∞ cone to be the limit of the l_p^p cones as $p \rightarrow \infty$. This definition permits some of our results to hold for the l_∞ norm as well. Thus, via d -flattenability, graph minors and topological embeddings, as well as combinatorial rigidity tools can now be used to understand the structure of these cone faces that play a crucial role in convex and semidefinite programming, spectral graph theory and metric space embedding [7]. The latter techniques are used widely in approximation of optimal solutions to NP-hard combinatorial problems and in complexity theory, where

in particular, non-Euclidean norms such as l_1 and l_∞ play a crucial role [20, 32]. Thus d -flattenability is a nexus connecting diverse techniques and applications.

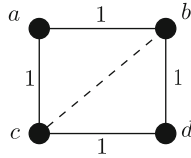


Fig. 1. Example of a linkage. The corresponding *pairwise distance vector* for this graph is given by: $\delta = (1, 1, 1, 1, \|a-d\|, \|b-c\|)^T$. The Cayley configuration space on the non-edge ad can take any distance in the range $[0, 2]$. The ordering of pairs as coordinate positions in the vector is arbitrary, but fixed by convention.

In the remainder of this section we give preliminary definitions, state the paper’s contributions and organization, and provide a brief listing of related work on the above topics in Sect. 1.1.

In [27] one of the authors introduced an alternative perspective on the configuration or realization space for a given linkage (G, δ^G) , defining the d -dimensional Cayley configuration space over some set of non-edges, F , of G under the l_2^d norm. This Cayley configuration space is denoted $\Phi_{F, l_2}^d(G, \delta^G)$, and is the set of vectors δ^F of Euclidean lengths attained by the non-edges F over all the realizations of the linkage (G, δ^G) . This same space is also sometimes referred to as the *Cayley configuration space of any realization or framework* (G, r) whose edge lengths are δ^G . The definition readily extends to arbitrary norms. In [27], it was shown that for the l_2 norm, d -flattenability of a graph G implies G has a d -dimensional, *inherent convex Cayley* configuration space, i.e., for all partitions of $G = H \cup F$, and all length vectors δ^H for the edges of H , $\Phi_{F, l_2}^d(H, \delta^H)$ is a convex set (see Fig. 3). This property was then used towards highly efficient atlasing of molecular configuration spaces [28], compared and hybridized with standard monte carlo methods in [22, 23], with multiple applications demonstrated in [28, 34]. Our first result in Sect. 2 shows the converse of the above result and generalizes both directions to general l_p norms, leading to our first main result:

- For l_p norms, G is d -flattenable if and only if G has a d -dimensional, *inherent convex Cayley* configuration space. As a direct corollary, it follows that both properties are minor-closed for general l_p norms.

For the next set of results given in Sect. 3, we refer the reader to combinatorial rigidity preliminaries in [17], defined for the Euclidean or l_2 normed space. The d -dimensional rigidity matrix of a graph $G = (V, E)$, denoted $R(G)$, is a matrix of indeterminates $r_1(v), r_2(v), \dots, r_d(v)$ for $v \in V$. These represent the coordinate position $r(v) \in \mathbb{R}^d$ of the point corresponding to a vertex $v \in V$ in an arbitrary realization or framework r of G . The matrix has one row for each edge each

vertex $v \in V$. The row corresponding to $e = (u, v) \in E$ represents the *bar* from $r(u)$ to $r(v)$ and has d non-zero entries $r(u) - r(v)$ (resp. $r(v) - r(u)$), in the d columns corresponding to u (resp. v). An instantiation of $R(G)$ to a particular framework is called the rigidity matrix of that framework. A *regular* or *generic* framework (G, r) (with respect to infinitesimal rigidity), is one whose corresponding instantiation of $R(G)$ has maximal rank over all instantiations.

A subset of edges of a graph G is said to be *independent* if the corresponding set of rows of $R(G)$ are generically independent. The maximal independent set yields the rank of G in the d -dimensional rigidity matroid (independent sets of edges of the complete graph). The graph (resp. generic framework) is (resp. infinitesimally) *rigid* if the number of generically independent rows or the rank of $R(G)$ is maximal, i.e., $d|V| - \binom{d+1}{2}$, where $\binom{d+1}{2}$ is the number of Euclidean isometries in \mathbb{R}^d [17].

For frameworks in polyhedral norms (including the l_p norms), Kitson [21] has defined properties such as *well-positioned*, *regular* analogous to the above, which have been used to show (infinitesimal) rigidity to be a generic property of frameworks.

We refer the reader to Kitson's paper for a precise definition. Intuitively, a well-positioned d -dimensional framework under norm $\|\cdot\|$ is one in whose d -dimensional neighborhood in $\|\cdot\|$ -normed space the pairwise distances between points can be expressed in polynomial form.

- For general l_p frameworks in arbitrary dimension, d -flattenability of a graph G is equivalent to all generic frameworks of G being d -flattenable.
- However, already for the Euclidean or l_2 case, d -flattenability is not a generic property of frameworks (in arbitrary dimension), and neither is the convexity of Cayley configuration spaces over specified non-edges of a d -dimensional framework. The latter uses minimal, 1-dof Henneberg-I frameworks for $d = 2$ constructed in [29, 30].
- The *existence* of a generic d -flattenable framework (in arbitrary dimension) is equivalent to independence of the rows of the generic d -dimensional rigidity matrix of its graph - we use the genericity concepts developed by Kitson [21] for l_p norms.

The next result, also in Sect. 3 concerns the cone Φ_{n, l_p} consisting of vectors δ_r of pairwise l_p^p -distances of n -point configurations r . (A proof that this set is a cone can be found in [5], which also applies to infinite dimensional settings).

The d -dimensional *stratum* of this cone consists of pairwise distance vectors of d -dimensional point configurations and is denoted Φ_{n, l_p}^d . The projection or shadow of this cone (resp. stratum) on a subset of coordinates i.e., pairs corresponding to the edges of a graph G is denoted Φ_{G, l_p} (resp. Φ_{G, l_p}^d). This projection is the set of realizable edge-length vectors δ^G of linkages (G, δ^G) in l_p^p (resp. in d -dimensions) (See Fig. 2).

Notice that Φ_{n, l_p} is the same as Φ_{K_n, l_p} , where K_n is the complete graph on n vertices. The l_p -*flattening dimension* of a graph G (resp. class C of graphs) is the minimum dimension d for which G (resp. all graphs in C) are flattenable

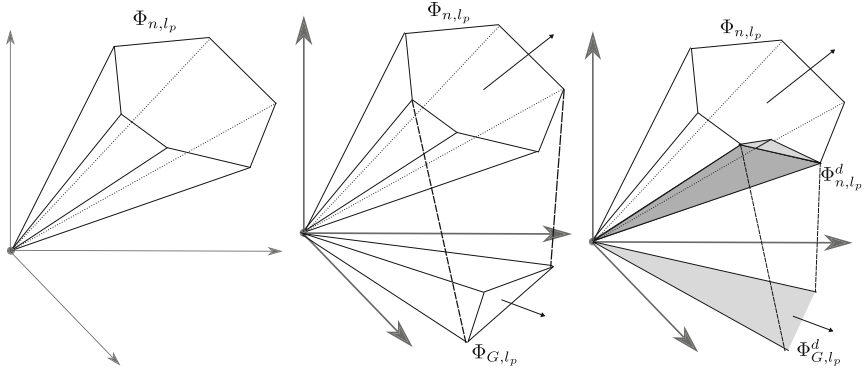


Fig. 2. Visualizing operations common to our proofs. On the left we have the cone of realizable distance vectors under l_p . It is shown here as a polytope, but in general that is not the case; these are not rigorous figures – their purpose is intuitive visualization. The cone lives in $\binom{n}{2}$ -dimensional space where each dimension is a pairwise distance among n points. In the middle is a projection onto the edges of some graph. This will yield a lower dimensional object (unless G is complete). On the right, a d -dimensional stratum is highlighted and lines show the projection onto coordinates representing edges of a graph. In general this stratum is not just a single face. Note that this projection is equal to the projection as the whole cone (middle) iff G is d -flattenable.

in l_p . Let n_p be the flattening dimension of K_n . It is not hard to show [12] that in fact $n_p \leq \mathbb{R}^{\binom{n}{2}}$ (using this finite dimensionality, a slight simplification of Ball’s proof of convexity of Φ_{n,l_p} is presented for completeness in Sect. 2). For the Euclidean or l_2 case, a further result of Barvinok [8] shows that the flattening dimension of any graph $G = (V, E)$ (although he did not use this terminology), is at most $O(\sqrt{|E|})$. Notice additionally that $\Phi_{F,l_p}^d(G, \delta^G)$, namely the d -dimensional Cayley configuration space of a linkage (G, δ^G) in l_p is the coordinate shadow of the (G, δ^G) -fiber of $\Phi_{G \cup F,l_p}^d$, i.e. all linkages $(G \cup F, \delta_{G \cup F})$ that have δ_G assigned to the edges of G , on the coordinate set F (see Fig. 3). In this paper, we show the following:

- Consider the coordinate shadow (or projection) of any neighborhood in the stratum Φ_{n,l_p}^d onto the edges of an n -vertex graph G . The dimension of this coordinate shadow equals the rank of G (size of maximal independent set) in the generic d -dimensional rigidity matroid [21] in l_p .

In Sect. 4, we give stronger results for specific norms for $d = 2$:

- The class of 2-flattenable graphs for the l_1 -norm (and l_∞ -norm) strictly contains the class of 2-flattenable graphs for the Euclidean l_2 -norm case, (the latter being the partial 2-tree graphs that avoid the K_4 minor). In particular, K_4 is 2-flattenable in l_1 . Graphs with *Banana* graphs as minors, however, are not 2-flattenable. We also consider other graphs such as the *4-wheel* and the

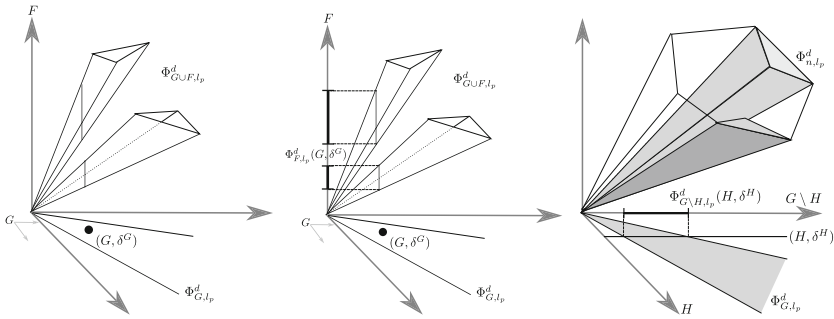


Fig. 3. This is an example of $\Phi_{G \cup F, l_p}^d$ that is not convex. The linkage (G, δ^G) and its fiber in $\Phi_{G \cup F, l_p}^d$ are shown on the left. Note that the fiber is not convex. In the middle, this fiber is then projected onto the remaining edges of $G \cup F$ to form $\Phi_{F, l_p}^d(G, \delta^G)$. Note that it is not convex either. On the right, Φ_{n, l_p}^d is projected onto the edges of some d -flattenable G (note that this is the same as projection of Φ_{n, l_p}). The inherent Cayley configuration space corresponding to some subgraph $G \setminus H$ of G is then shown projected onto the edges of $G \setminus H$. This projection is convex.

doublet and $K_{3,3}$ towards obtaining a forbidden-minor characterization of 2-flattenability in the l_1 -norm.

Finally, in Sect. 5, a number of conjectures and open problems are posed.

1.1 Related Results

The structure of the cone Φ_{n, l_p} , its strata and faces are well-studied. The fact that the object is a cone even in the infinite dimensional case is a useful observation by Ball [5]. For l_2 this is called the *Euclidean distance matrix (EDM) cone* [11, 13, 31], which is a simple, linear transformation of the cone of positive semi-definite matrices, a fact first observed by Schoenberg [26]. Consequently understanding its structure is important in semidefinite programming relaxations and the so-called sums of squares method with numerous applications [6, 16, 24]. Connections between combinatorial rigidity and the structure of the EDM have been investigated extensively by Alfakih [1, 2]. The reader is additionally referred to [12] for a comprehensive survey of key results about the EDM cone, including observations about the face structure and dimensional strata of the EDM cone. The l_1 -cone is often called the *cut cone*, whose extreme rays correspond to 1-dimensional realizations and characteristic vectors of cuts in a complete graph. The cone has been studied by [4, 12, 33] and plays an important role in metric space embeddings used in the study of (non-)approximability in polynomial time, of NP-hard optimization problems, including ramifications of the unique games conjecture [18, 20]. Kitson’s recent work [21] has shown that many of the results in combinatorial rigidity for the Euclidean or l_2 norm case have parallels in the case of general polyhedral norms, including the l_p norms.

2 l_p : Flattenability and Inherent Convex Cayley Configuration Space

In this section, we improve in the work done in [27] in relating d -flattenability to convex inherent Cayley configuration spaces of a given graph. The results of this section hold for $p = \infty$ as well, appealing to the definition of l_∞ cone as the limit of the l_p^p cones as $p \rightarrow \infty$. Our main result of this section follows.

Theorem 1. *For any l_p norm, a graph G is d -flattenable if and only if G admits convex inherent d -dimensional Cayley configuration spaces for each of its sub-graphs.*

This “only if” direction of this statement was shown in [27] for the l_2 norm. The argument only required the fact that the cone of squared distance vectors is convex. Hence, we can use the same proof if we can show Φ_{n,l_p} is convex. The proof of the “if” direction requires that the cone is the convex hull of l_p^p distance vectors in any dimension d .

Proposition 1. *Φ_{n,l_p} for general l_p is contained in the convex hull of the l_p^p distance vectors of the 1-dimensional n -point configurations in \mathbb{R} .*

Proof. Take some $\delta \in \Phi_{n,l_p}$. Let $r(1), \dots, r(n)$ denote some realization of the complete linkage (K_n, δ) . We refer to this as a realization of δ . So, $r(i) \in \mathbb{R}^k$ for some k . It was shown in [5] that for any l_p -norm, the flattening dimension $n_p \leq \binom{n}{2}$, so there is a realization in some finite dimension. We have

$$\delta_{ij} = \|r(i) - r(j)\|_p^p = \sum_{l=1}^k |r_l(i) - r_l(j)|^p$$

where $r_l(i)$ denotes the l th coordinate of the i th point. Then, if we construct the matrix δ^l such that $\delta_{ij}^l = \|r_l(i) - r_l(j)\|_p^p$, then δ^l is a valid l_p^p distance matrix with an n -point configuration in \mathbb{R} . This point configuration simply being $r_l(1), \dots, r_l(n)$. Also, for any $\alpha > 0$, $\alpha\delta^l$ is a valid l_p^p distance matrix with realization $\alpha^{\frac{1}{p}}r_l(1), \dots, \alpha^{\frac{1}{p}}r_l(n)$. Finally, $\delta = \sum_l \frac{1}{k} [k\delta^l]$, which is a convex combination of n -point configurations in \mathbb{R} . \square

A well-known result shows that Φ_{n,l_p} is convex.

Observation 2. *Φ_{n,l_p} for general l_p is convex*

Proof. The proof for this result (see [5]) is well known even for the infinite dimensional case. Here we give a simplified proof for finite dimensions for completeness.

Let r and s be two n -point configurations with corresponding distance vectors $\delta_r, \delta_s \in \Phi_{n,l_p}$. Assume r and s are realized in some dimension k . Let $0 \leq \lambda \leq 1$ and consider the convex combination $\delta = \lambda\delta_r + (1 - \lambda)\delta_s$. We will construct an n -point configuration in $2k$ dimensions with δ as its distance matrix. Note that $\delta_{ij} = \|r(i) - r(j)\|_p^p + \|s(i) - s(j)\|_p^p = \sum_l^k |r_l(i) - r_l(j)|^p + \sum_l^k |s_l(i) - s_l(j)|^p$.

Then a realization for t can be found by simply concatenating the coordinates of r and s and scaling them appropriately:

$$t = (\lambda^{\frac{1}{p}} r, (1 - \lambda)^{\frac{1}{p}} s)$$

It is easy to verify that t is a realization of δ . □

Proposition 1 and Observation 2 lead to the following, which is useful to us in proving Theorem 1.

Proposition 2. Φ_{n,l_p} , $1 \leq p \leq \infty$ is the convex hull of the l_p^p distance vectors of the 1-dimensional, n -point configuration vectors in \mathbb{R} .

Proof. Follows from Proposition 1 and Observation 2 and the fact that in Proposition 1, the points making up the convex hull are in Φ_{n,l_p} . □

Since from the 1-dimensional vectors, we can construct - as convex combinations - vectors realizable in any arbitrary d -dimensions, we get the following Corollary:

Corollary 1. Φ_{n,l_p} is the convex hull of the vectors in Φ_{n,l_p}^d for any d for general l_p .

The following observation is useful in characterizing d -flattenability.

Observation 3. If G is d -flattenable, then the projection of Φ_{n,l_p}^d onto the edges of G is exactly the projection Φ_{n,l_p} onto the edges of G .

Using these results, we can now prove the “if” part of Theorem 1.

Proof. [of Theorem 1] Suppose G is d -flattenable under some l_p -norm. Then, because $\Phi_{|V|,l_p}$ is convex, Φ_{G,l_p} is convex. From Observation 3, Φ_{G,l_p}^d is convex. Given a subgraph F of G , if we break G into H and F and fix the values of E corresponding to a linkage (H, δ^H) , we are taking a section of $\Phi_{H \cup F, l_p}^d$, which is again convex. This is also exactly the Cayley configuration space $\Phi_{F, l_p}^d(H, \delta^H)$. Note that this holds for any partition H and F , so G always admits a convex d -dimensional Cayley configuration space for each of its subgraphs.

For the other direction, suppose for a linkage (G, δ^G) , all d -dimensional Cayley configurations corresponding to subgraphs of G , $\Phi_{F, l_p}^d(G \setminus F, \delta^{G \setminus F})$ are convex. Certainly this holds for the empty subgraph as well. We note that $\Phi_{G, l_p}^d(\emptyset, \delta^\emptyset)$ is just Φ_{G, l_p}^d and because it is convex, Φ_{G, l_p}^d is its own convex hull. We also know that the convex hull of Φ_{G, l_p}^d is the projection of the convex hull of $\Phi_{|V|, l_p}^d$. By Proposition 2 and its Corollary, we know this to be the entire cone $\Phi_{|V|, l_p}$. Thus, $\Phi_{G, l_p}^d = \Phi_{G, l_p}$. Hence, G is d -flattenable. □

This result provides a nice link between d -flattenability and convex Cayley configuration spaces. It leads to the following tools.

Corollary 2. *Having a d -dimensional convex Cayley configuration space on all subgraphs is a minor-closed property.*

Another immediate result is that d -flattenability and convex Cayley configuration spaces have the same forbidden minor characterizations for given d under the same l_p -norm. This gives us a nice tool when trying to find forbidden minors for other l_p norms:

Observation 4. *If for some assignment of distances l to some edges E of G leads to a non-convex $\Phi_{F,l_p}^d(G,l)$, then G is not d -flattenable.*

We use this to show that the “banana” graph in 5 vertices is not 2-flattenable for l_1 (and l_∞) in Theorem 12. The banana is a K_5 graph with one edge removed.

3 l_2 : Flattenability, Genericity, Independence in Rigidity Matroid

In this section we show relationships between d -flattenability and combinatorial rigidity concepts via the cone Φ_{n,l_p} .

The definition of d -flattenability of a graph G in l_p requires every l_p framework of the graph G – in an arbitrary dimension – to be d -flattenable.

To accommodate the arbitrary dimension of the original framework, we first give a suitable definition of generic frameworks for d -flattenability.

Definition 1. *Given an l_p framework (G,r) , with n vertices, in arbitrary dimension, consider its pairwise length vector, δ_r , in the cone Φ_{n,l_p} (this was used in Sect. 2). A framework (G,r) of n vertices is generic with respect to d -flattenability if the following hold: (i) there is an open neighborhood Ω of δ_r in the (interior of the) cone Φ_{n,l_p} , (recalling that n_p is the flattening dimension of the complete graph K_n , Ω corresponds to an open neighborhood of n_p -dimensional point-configurations of r); and (ii) (G,r) is d -flattenable if and only if all the frameworks in Ω are.*

Item (i) implies that there is a “full measure” or n_p -dimensional neighborhood of (G,r) that corresponds to a neighborhood of distance vectors in the interior of the cone. Item (ii) asserts that all frameworks in this neighborhood are d -flattenable iff (G,r) is.

Theorem 5. *Every generic framework of G is d -flattenable if and only if G is d -flattenable.*

Proof. The “if” direction follows immediately from the definition of d -flattenability. For the “only if” direction, notice that a non-generic, (bounded) framework (G,r) is a limit of a sequence Q of generic, bounded frameworks $\{(G,r_i)\}_i$, with a corresponding sequence of pairwise distance vectors in Φ_{G,l_p} , and further a corresponding sequence of projections onto the edges of G , i.e., a sequence Q' of bounded linkages of G . Because each (G,r_i) is d -flattenable, each

linkage in Q' must be realizable as some generic bounded framework (G, r'_i) in d -dimensions, i.e., each linkage is the projection of the pairwise distance vector of some d -dimensional bounded framework (G, r') , i.e., a pairwise distance vector in the d -dimensional stratum of the cone Φ_{G, l_p} . The projection of the limit framework (G, r) of the sequence Q is the limit linkage of the projected sequence Q' of linkages with bounded edge lengths, whose corresponding sequence of realizations as d -dimensional bounded frameworks has a limit (G, r') . The latter limit follows from the fact that the realization map that takes a linkage to its d -dimensional bounded framework realizations (for any given bound) is a closed map. This completes the proof. \square

A similar argument can be used to show that the projection of Φ_{n, l_p}^d onto the edges of any graph G , denoted Φ_{G, l_p}^d is closed. This result was shown for l_2 in [31].

Although d -flattenability is equivalent to the presence of an inherent convex Cayley configuration space for G , (as shown in Sect. 2), we now move beyond inherent convex Cayley configuration spaces to Cayley configuration spaces over specified non-edges F . These could be convex even if G itself is not d -flattenable (simple examples can be found for $d = 2, 3$ for l_2 in [27]). A complete characterization of such G, F is shown in [27], in the case of l_2 norm for $d = 2$, conjectured for $d = 3$, and completely open for $d > 3$. In Sect. 5, we extend the conjecture for general d .

An analogous theorem to Theorem 5 can be proven for the property of a d -dimensional framework (G, r) having a convex Cayley configuration space over specified non-edge set F . However, since this framework is d -dimensional rather than of arbitrary dimension, the definition of genericity has to be modified from Definition 1.

Definition 2. Let δ_r be as in Definition 1. A framework (G, r) of n vertices in d -dimensions is generic with respect to the property of convexity of $\Phi_{F, l_p}^d(G, \delta^G)$ if (i) there is an open neighborhood Ω of δ_r in the stratum Φ_{n, l_p}^d , (this corresponds to an open neighborhood of d -dimensional point-configurations of r); and (ii) (G, r) has convex Cayley configuration space over F if and only if all the frameworks in Ω do.

Theorem 6. Every generic d -dimensional framework (G, r) has a convex Cayley configuration space over F if and only if for all δ^G , the linkage (G, δ^G) has a d -dimensional, convex Cayley configuration space over F .

Proof. The “if” direction follows immediately from the definitions. Moreover, it is sufficient to prove the “only if” direction for edge length vectors δ_G that are attained by some (potentially non-generic) d -dimensional framework (G, r) , because otherwise the d -dimensional Cayley configuration space of the linkage (G, δ^G) is empty and hence trivially convex. Now as in Theorem 5, every non-generic d -dimensional framework (G, r) with edge length vector δ^G is a limit of a sequence $\{(G, r_i)\}_i$ of generic frameworks with edge length vectors $\delta^{G, i}$.

Since convexity of the Cayley configuration space $\Phi_{F,l_p}^d(G, \delta^{G,i})$ is preserved over a open neighborhoods of (G, r_i) , it follows that the limit of the sequence of spaces $\{\Phi_{F,l_p}^d(G, \delta^{G,i})\}_i$ exists, is convex, and is the Cayley configuration space of (G, r) . Since (G, r) was chosen to have the edge length vector δ_G , this space is in fact $\Phi_{F,l_p}^d(G, \delta^G)$, the Cayley configuration space of the linkage (G, δ_G) . \square

A *property* of frameworks is said to be *generic* if the *existence* of a generic framework with the property implies that the property holds for *all* generic frameworks. Next we show that neither of the properties discussed above is a generic property of frameworks even for l_2 .

Theorem 7. *d-flattenability and convexity of Cayley configuration spaces over specific non-edges F are not a generic property of frameworks (G, r) .*

Proof. For d -flattenability: since the flattening dimension n_2 of K_n in l_2 is $n - 1$, we show the counterexample of a 5-vertex graph G for which one generic 4-dimensional framework (G, r) and its neighborhood is 2-flattenable in l_2 , while another such neighborhood is not. See Fig. 4. For convexity of Cayley configuration spaces: there are minimal, so-called Henneberg-I graphs [30] G , constructed on a *base* or initial edge f with the following property: for some 2-dimensional frameworks (and neighborhoods) (G, r) with edge length vector δ^G , the 1-dimensional Cayley configuration space $\Phi_{f,l_2}^2(G \setminus f, \delta^{G \setminus f})$ (i.e., the attainable lengths for f) is a single interval, while for other such frameworks (and neighborhoods) it is 2 intervals. Please see Appendix in [30]. \square

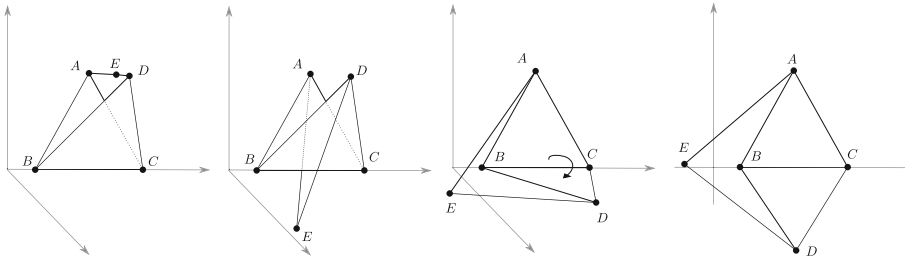


Fig. 4. 2 realizations of the same graph. In the first figure (left), we have edge lengths for (a, e) and (d, e) that do not allow G to be flattened. The second graph is realized in 3-dimensions, but by “unfolding it” as shown, we can flatten it into 2-dimensions

Next, we consider the implication of the *existence* of a generic d -flattenable framework. Specifically, we prove two theorems connecting the d -flattenability with independence in the rigidity matroid: we use the notion of rigidity matrix, and consequently regular frameworks and generic rigidity matroid developed by Kitson [21], as well as the equivalence of finite and infinitesimal rigidity using the notion of *well-positioned* frameworks, which intuitively means that the l_p balls

of size given by the corresponding edge-lengths centered at the points intersect properly (i.e., the intersection of k $(d - 1)$ -dimensional ball boundaries is of dimension $d - k$).

The “if” direction of this next theorem is a restatement of Proposition 2 in Asimov and Roth [3]. We extend their result here and show the other direction as well.

Theorem 8. *For general l_p norms, there exists a generic d -flattenable framework of G if and only if G is independent in the d -dimensional generic rigidity matroid.*

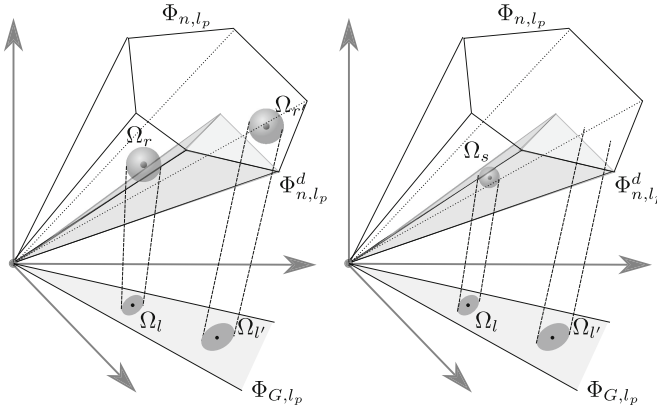


Fig. 5. On the left we have 2 neighborhoods Ω_r and $\Omega_{r'}$ of 2 distance vectors δ_r and $\delta_{r'}$ in the cone. We then project Ω_r and $\Omega_{r'}$ onto the edges of G to obtain Ω_l and $\Omega_{l'}$, which are essentially the neighborhoods of (G, δ_r^G) and $(G, \delta_{r'}^G)$. On the right, we then take the fiber of Ω_l and $\Omega_{l'}$ on Φ_{n,l_p}^d . The fiber of Ω_l is completely contained in the stratum while that of $\Omega_{l'}$ misses (does not intersect) the stratum.

Proof. For the forward direction, we note that existence of a generic d -flattenable framework (G, r) is equivalent to the statement that the pairwise distance vector δ_r has an open neighborhood Ω_r in the interior of the cone Φ_{n,l_p} , and the d -flattenings (G, s) form an open neighborhood Ω_s of pairwise distance vector δ_s in the relative interior of the stratum Φ_{n,l_p}^d . This is also equivalent to saying there is a corresponding open neighborhood of d -flattenable linkages $(G, \delta_s^G = \delta_r^G)$. Now Ω_s (resp. Ω_r) must contain an open neighborhood of pairwise distance vectors δ_s (resp. δ_r) that correspond to well-positioned and regular frameworks (G, s) , (resp. (G, r)), hence without loss of generality, we can take that neighborhood to be Ω_s (resp. Ω_r), consisting of d -dimensional, well-positioned, regular frameworks (G, s) (resp. (G, r)) that are realizations of an open neighborhood of Ω_G of linkages $(G, \delta_s^G = \delta_r^G)$. These linkages correspond to a coordinate shadow or projection of Ω_r and Ω_s onto (the edges in) G .

Now observe that the generic rigidity matrix of G is the Jacobian of the distance map from the d -dimensional point-configuration s to the edge-length vector δ_s^G at the point s . For l_2 and integral $p > 1$, this map is clearly specified by polynomials. For l_1 and l_∞ , we use the notion of well-positioned frameworks from [21]. If the frameworks of Ω_r are well-positioned, then it follows that the distance map is locally specified by linear polynomials corresponding to a relevant facet of the l_1 or l_∞ ball. Because Ω_r has dimension equal to the number of edges in G , these polynomials are algebraically independent. Hence, their Jacobian has rank equal to the number of edges in G . Therefore, the existence of well-positioned, regular realizations s to an entire neighborhood of edge-length vectors δ_s^G implies the statement that the rows of the generic rigidity matrix – that correspond to the edges of G – are independent.

The converse follows from Proposition 2 of Asimow-Roth [3] observing that at well-positioned and regular points, for all $1 \leq p \leq \infty$ the l_p distance map from point configurations to pairwise distance vectors is a smooth map. \square

The following corollary is immediate from the forward direction of the above proof.

Corollary 3. *For general l_p norms, a graph G is d -flattenable only if G is independent in the d -dimensional rigidity matroid.*

The following theorem and corollary utilize the dimension of the projection of the d -dimensional stratum on the edges of G from the above proof. Note that in the above proof, if G is an n -vertex graph, the neighborhood Ω_r has dimension n_p , i.e., the flattening dimension of K_n ; Ω_s has dimension equal to that of the stratum Φ_{n,l_p}^d , and Ω_G has dimension equal to the number of edges of G (see Fig. 6).

The first item of the following Theorem is a restatement of Theorem 8 and as such, the “only if” direction appears in [3].

Theorem 9. *For general l_p norms, a graph G is*

- I. *independent in the generic d -dimensional rigidity matroid (i.e., the rigidity matrix of a well-positioned and regular framework has independent rows), if and only if coordinate projection of the stratum Φ_{n,l_p}^d onto G has dimension equal to the number of edges of G ;*
- II. *maximal independent (minimally rigid) if and only if projection of the stratum Φ_{n,l_p}^d onto G is maximal (i.e., projection preserves dimension) and is equal to the number of edges of G ;*
- III. *rigid in d -dimensions if and only if projection of the stratum Φ_{n,l_p}^d onto G preserves its dimension;*
- IV. *not independent and not rigid in the generic d -dimensional rigidity matroid if and only if the projection of Φ_{n,l_p}^d onto G is strictly smaller than the minimum of: the dimension of the stratum and the number of edges in G .*

Proof. The proof of this theorem follows from the proof of the previous result: Theorem 8. Each case is illustrated in Fig. 6. \square

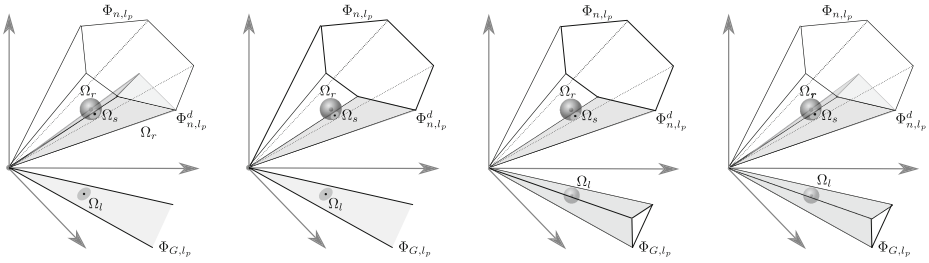


Fig. 6. These are visualizations of when frameworks are isostatic and independent. In all of these cases $\dim(\Omega_r) \geq \max\{\dim(\Omega_s, \dim(\Omega_l))\}$. We only show 2 and 3 dimensions here, but in general the dimensions will be much higher. See Fig. 5 for explanation of what each is. In the following, when we use equality or inequality, we are referring to dimension. On the left, $\Omega_s = \Omega_l < \Phi_{n,l_p}^d$ meaning δ_r is independent but not isostatic. Middle left: $\Omega_s = \Omega_l = \Phi_{n,l_p}^d$, so δ_r is maximal independent or isostatic. Middle right: $\Omega_s = \Phi_{n,l_p}^d < \Omega_l$ meaning δ_r is rigid but not independent. Right: $\Omega_s < \Omega_l$ and $\Omega_s < \Phi_{n,l_p}^d$ meaning δ_r is neither independent nor rigid.

We note that the “only if” direction of item 9 of Theorem 9 is the same result that appears as Proposition 2 (in a different form) in [3]. However, to the best of our knowledge, the “if” direction and the rest of Theorem 9 and the proof of Theorem 8 are new results. We obtain the following useful corollary.

Corollary 4. *For l_p norms, the rank of a graph G in the d -dimensional rigidity matroid is equal to the dimension of the projection Φ_{G,l_p}^d on G of the d -dimensional stratum Φ_{n,l_p}^d .*

4 l_1 : 2-Flattenability

We now turn our attention to the l_1 norm in 2-dimensions. We note that the l_1 and l_∞ norms in 2-dimensions are equivalent by simply applying a rotation to our axes (for argument, see [12]). Specifically, we would like to characterize the class of graphs that are 2-flattenable under the l_1 norm. A result from [33] shows that K_4 is 2-flattenable. We note that K_4 is the only forbidden minor for 2-flattenability under the l_2 norm. It immediately follows that the 2-flattenable l_2 graphs are a strict subset of the 2-flattenable l_1 graphs. In the remainder of this section, we narrow down the possible candidates for forbidden minors.

Observation 10. *All partial 2-trees are 2-flattenable.*

This follows from the fact that partial 2-trees are exactly graphs without a K_4 minor. We define 2-trees recursively. A triangle is a 2-tree. Given any 2-tree, attaching another triangle onto a single edge is also a 2-tree. A partial 2-tree is any subgraph of a 2-tree. Because the 2-flattenable graphs for l_2 are exactly the partial 2-trees, it follows partial 2-trees are 2-flattenable for l_1 .

In order to generalize our results, we introduce the following Theorem which involves a 2-sum operation. A 2-sum of graph G_1 and G_2 is a new graph G made by gluing an edge of G_1 to one of G_2 , i.e. we identify an edge of G_1 with an edge of G_2 .

Theorem 11. *A 2-sum of 2-flattenable graphs is 2-flattenable if and only if at most one graph has a K_4 minor.*

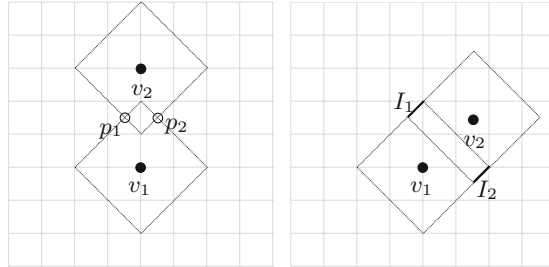


Fig. 7. On the left is a partial realization of G_2 if we assume a vertical orientation for (v_1, v_2) . On the right is the same for (v_1, v_2) at an angle of 45 degrees

Proof. Suppose G_1 and G_2 are 2-flattenable and only G_1 has a K_4 minor. Then, G_2 is a partial 2-tree. Thus the 2-sum of G_1 and G_2 can be built by taking a realization of G_1 , identifying the 2-sum, and then adding the vertices of G_2 one at a time. Let r and s be the 2 vertices we are attaching some new vertex v to. No matter the orientation of r and s , as long as the triangle formed by r, s , and v obeys the triangle inequality, the l_1 -balls surrounding r and s with distances corresponding to their distance to v will always intersect in 2-dimensions. This can be verified by placing r at the origin, moving s along the l_1 ball of r in the first quadrant and observing the balls surrounding r and s as s moves. Hence, the triangle r, s, v can be realized in 2-dimensions.

Suppose G_1 and G_2 both have a K_4 minor. We give a counter-example to show that the 2-sum is not 2-flattenable. Let G_1 be the equidistant K_4 with each edge having distance 3. Let G_2 have every edge with distance 2 except for (v_1, v_2) , which has distance 3. G_1 has only one realization modulo rearranging vertices: all points at the corners of the distance 3 l_1 -ball. The edges of G_1 are all either vertical/horizontal or at an angle of 45 degrees. We claim G_2 has no realization with (v_1, v_2) at those angles.

If we assume that (v_1, v_2) is vertical, looking at Fig. 7, we see that the remaining 2 vertices can only lie at p_1 and p_2 . The possible distances they can obtain are 0 and 1, which means G_2 cannot be completed. Looking at the 45 degree case on the right of Fig. 7, we see that the other 2 vertices can only lie in I_1 and I_2 . This leads to possible distances of $[0, 1]$ and 4. Thus G_2 still cannot be

completed. Note that the horizontal and other 45 degree orientations are just flips of these two cases.

Hence, G_2 has no realization with (v_1, v_2) at any of the angles of G_1 's edges. So, the 2-sum of G_1 and G_2 is not 2-realizable. We note that this 2-sum does have a realization in 3-dimensions: $v_1 = (0, 0, 0), v_2 = (1.5, 1.5, 0), v_3 = (0.5, 1, 0.5), v_4 = (1, 0.5, -0.5)$ give a realization for G_2 with (v_1, v_2) at a 45 degree angle, so G_1 2-sum G_2 is not 2-flattenable. \square

Note that the realization given at the end of the above proof has a K_4 lying on a 2-dimensional subspace of \mathbb{R}^3 . This is still a 3-dimensional realization as the spanned 2-dimensional subspace is not equipped with an 2-dimensional l_1 norm. However, it is equipped with an l_1 norm in 3-dimensions. The K_4 has a realization in R^2 , we only need it in 3-dimensions in the proof to 2-sum it to the other K_4 .

Another result from [5] shows that K_5 is not 2-flattenable. Hence, we search the subgraphs of K_5 and check them for 2-flattenability. This leads to the following example of a non-2-flattenable graph, which we prove using the techniques developed in this paper.

Theorem 12. *The so called “banana” graph or K_5 minus one edge is not 2-flattenable under the l_1 norm.*

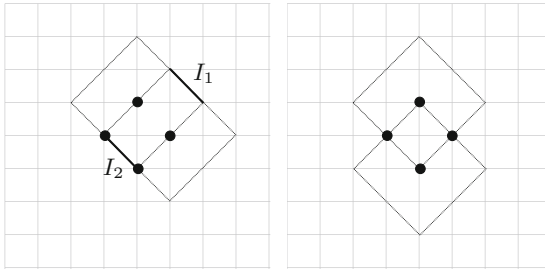


Fig. 8. On the left is an illustration of an equidistant K_4 with the possible intervals of Case 1. On the right is the same for Case 2

Proof. We will invoke Observation 4 to show this.

Consider a distance vector for the banana with unit distances for all except one edge, f . This has a realization in 3-dimensions as K_5 is 3-flattenable for the l_1 norm (see [5]). Then, we have an equidistant K_4 as a subgraph. The only realization for such a K_4 in 2-dimensions is to have all 4 points arranged as the vertices of the unit ball centered at the origin. The 2 remaining unit edges then connect a new vertex to 2 of these points. Here we have 2 cases: the 2 vertices border the same quadrant or they lie across one of the axes from each other.

Case 1: Without loss of generality, we assume the 2 vertices are the upper right of the K_4 . In Fig. 8, it can be seen that the new vertex can lie anywhere in I_1 or I_2 . If it lies in I_1 , the remaining edge of the banana can take lengths in the range $[0, 1]$. If it lies in I_2 , the only length it can be is 2.

Case 2: Without loss of generality, assume the 2 vertices are the top-most and bottom-most. Again from Fig. 8, the new vertex only has 2 positions it can be in, each leading to a length of 1 for the remaining edge.

Hence, $\Phi_{F,l_1}^2(G \setminus F, \delta^{G \setminus F}) = [0, 1] \cup \{2\}$, where G is the banana and $F = \{f\}$. This is not convex and thus by Theorem 1, the banana is not 2-flattenable. \square

Observation 13. K_5 minus 2 edges incident to a single vertex is 2-flattenable.

Proof. This follows directly from Theorem 11. \square

Observation 14. Connected graphs on 5 vertices with 7 edges are 2-flattenable

Proof. The only 2 such graphs are a subgraph of the Observation 13 and the complete 2-tree on 5 vertices. Both of these we know to be 2-flattenable. \square

The only remaining 5 vertex graph we have not looked at yet is the *wheel* graph. So far we have shown that for 5 vertices, graphs with the wheel as a minor are not 2-flattenable and graphs without are 2-flattenable. Thus, if we can show that the wheel is not 2-flattenable, then it becomes the only forbidden minor for l_1 2-flattenability. We discuss this more in Sect. 5 and conjecture that in fact the wheel is the only forbidden minor for 2-flattenability under the l_1 (and l_∞) norm.

5 Conjectures and Open Problems

5.1 Combinatorial Rigidity and Structure of Φ_{n,l_p}

In Theorem 8 and Theorem 9, we have shown that combinatorial rigidity properties of a graph in d -dimensions is tied to the dimension of the projection of some “face” of d -dimensional stratum of $\Phi_{n,p}$. These properties are not generic when viewed as properties of frameworks Ω_r in the flattening dimension of K_n , i.e., when viewed as distance vectors δ_r in the interior of the cone $\Phi_{n,p}$. However, since we know that these properties are generic in d -dimensions (via combinatorial rigidity techniques), this means it must be that that the projection of every face of the d -dimensional stratum of $\Phi_{n,p}$ onto G has the same dimension. Thus combinatorial rigidity and Cayley configuration spaces can help understand the structure of the cone. However, it would be good to have an independent proof of these properties directly via the cone geometry. More formally:

Conjecture 1. G is d -independent if and only if the projection of every face of Φ_{G,l_p}^d has dimension equal to the number of edges of G .

It would be useful to show a stronger property about the continuous mapping used in the proof of Theorem 5. Doing so would require deeper understanding of the d -flattening process itself. Formally:

Question 1. Given a realization (G, r) of a d -flattenable linkage (G, δ_G) in some high dimension, is there always a continuous path from (G, r) to (G, r') in d -dimensions for general l_p norms?

It would be useful to even show a weaker version of Question 1: Does a continuous path of high dimensional frameworks of a d -flattenable graph G always correspond to a path in d -dimensional frameworks?

In the case of the Euclidean or l_2 norm many questions remain concerning core results and applications of convex Cayley configuration spaces.

The question of convexity of Cayley configuration spaces of graphs G over specified edge sets F is fully understood, and the proof [27] uses the existence of a specific type of homeomorphism to produce forbidden minors. The property is relatively close to that of 2-flattenability which is equivalent to convexity of inherent 2-dimensional Cayley configuration spaces. In fact the class of graphs (partial 2-trees) have convex Cayley configuration spaces in any dimension (follows immediately from the close relationship to 2-flattenability). Thus, as in Sect. 5.1, we expect that fully understanding the structure of convex Cayley configuration spaces of partial 2-trees in 2-dimensions (which relies on combinatorial rigidity and forbidden minor properties) will help in understanding the structure of 2-dimensional stratum of the cone.

We believe the study of Cayley configuration spaces of partial 2-trees can simplify results related to the so-called Walker conjecture about the topology of Cartesian configuration spaces for a very simple class of partial 2-trees, namely polygonal graphs [14, 15], as well as to extend them to general, partial 2-trees. In fact, we believe that the Cayley configuration space of partial 2-trees can help to understand entire structure of Φ_{n, l_2} .

While convex Cayley configuration spaces over specified non-edges F in 2-dimensions are fully characterized, very little is known (beyond the forbidden minors for 3-flattenability) in higher dimensions. In particular, there are graphs G that are themselves not 3-flattenable, but their Cayley configuration spaces are convex over certain non-edges F . Several natural conjectures in [27] relate to the specific type of homeomorphism used to produce the forbidden minor characterizations in the 2D case. These still remain open for higher dimensions.

5.2 2-Flattenability Under l_1

In Sect. 2, we showed a number of techniques to prove (non)-2-flattenability of certain graphs under the l_1 norm. Mostly these dealt with a constructive argument like the partial 2 tree case to prove flattenability and showing non-convexity of inherent Cayley configuration space for non-flattenability.

It is still an open question as to what the forbidden minor characterization of 2-flattenability under l_1 is. Our results show that the only 5 vertex graph to classify is the wheel. Due to the fact that the wheel is a minor to all of the other non-2-flattenable graphs on 5 vertices, we raise the following conjecture.

Conjecture 2. The forbidden minor characterization for 2-flattenability under the l_1 and l_∞ norms consists of only the wheel on 5 vertices.

Showing this requires only that we show that the wheel is not 2-flattenable. If this result is proven to be negative, then it will be necessary to look at 6 vertex graphs such as $K_{3,3}$, the doublet, and $K_{2,2,2}$.

5.3 Other Metrics

We would like to extend the results of this paper to other polyhedral norms faces. Some of the major obstacles have been outlined in [21]. In particular the nonexistence of well-positioned and regular frameworks, all of whose sub-frameworks are also regular. Some work was done in this paper on this paper for the specific case of l_1 .

Extending the results of this paper to other metrics would increase its applicability in combinatorial optimization settings. Doing this will require us to first choose an appropriate notion of dimension for metric topologies, be it the doubling dimension or some other classical notion of dimension.

Acknowledgement. We thank Bob Connelly, Steven Gortler and Derek Kitson for interesting conversations related to this paper.

References

1. Alfakih, A.Y.: Graph rigidity via euclidean distance matrices. *Linear Algebra Appl.* **310**(1–3), 149–165 (2000)
2. Alfakih, A.Y.: On bar frameworks, stress matrices and semidefinite programming. *Math. Program.* **129**(1), 113–128 (2011)
3. Asimow, L., Roth, B.: The rigidity of graphs. *Trans. Amer. Math. Soc.* **245**, 279–289 (1978)
4. Avis, D., Deza, M.: The cut cone, l_1 embeddability, complexity, and multicommodity flows. *Networks* **21**(6), 595–617 (1991)
5. Ball, K.: Isometric embedding in l_p -spaces. *Eur. J. Comb.* **11**(4), 305–311 (1990)
6. Barak, B., Raghavendra, P., Steurer, D.: Rounding semidefinite programming hierarchies via global correlation. *CoRR*, abs/1104.4680 (2011)
7. Barak, B., Steurer, D.: Sum-of-squares proofs and the quest toward optimal algorithms. *CoRR*, abs/1404.5236 (2014)
8. Barvinok, A.I.: Problems of distance geometry and convex properties of quadratic maps. *Discrete Comput. Geom.* **13**(1), 189–202 (1995)
9. Belk, M.: Realizability of graphs in three dimensions. *Discrete Comput. Geom.* **37**(2), 139–162 (2007)
10. Brinkman, B., Charikar, M.: On the impossibility of dimension reduction in l_1 . *J. ACM* **52**(5), 766–788 (2005)
11. Dattorro, J.: *Convex Optimization & Euclidean Distance Geometry*. Meboo Publishing USA, Palo Alto (2011)
12. Deza, M.M., Laurent, M.: *Geometry of Cuts and Metrics*. Algorithms and Combinatorics. Springer, Heidelberg (2010)
13. Drusvyatskiy, D., Pataki, G., Wolkowicz, H.: *Coordinate shadows of semi-definite and euclidean distance matrices* (2014)
14. Farber, M., Hausmann, J.-C., Schuetz, D.: On the conjecture of kevin walker. *J. Topology Anal.* **01**(01), 65–86 (2009)

15. Farber, M., Fromm, V.: The topology of spaces of polygons. *Trans. Amer. Math. Soc.* **365**, 3097–3114 (2010)
16. Gatermann, K., Parrilo, P.A.: Symmetry groups, semidefinite programs, and sums of squares. *J. Pure Appl. Algebra* **192**(13), 95–128 (2004)
17. Graver, J., Servatius, B., Servatius, H.: *Combinatorial Rigidity*. Graduate Studies in Mathematics. American Mathematical Society, Providence (1993)
18. Indyk, P., Matousek, J.: Low-distortion embeddings of finite metric spaces. In: *Handbook of Discrete and Computational Geometry*, pp. 177–196. CRC Press (2004)
19. Johnson, W., Lindenstrauss, J.: Extensions of lipschitz mappings into a Hilbert space. In: *Conference in Modern Analysis and Probability* (New Haven, Conn., 1982). *Contemporary Mathematics*, vol. 26, pp. 189–206. American Mathematical Society (1984)
20. Khot, S.: On the unique games conjecture. In: *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Pittsburgh, PA, USA, 23–25 October 2005, p. 3 (2005)
21. Kitson, D.: *Finite and infinitesimal rigidity with polyhedral norms* (2014)
22. Ozkan, A., Flores-Canales, J.C., Sitharam, M., Kurnikova, M.: Fast and flexible geometric method for enhancing MC sampling of compact configurations for protein docking problem. *ArXiv e-prints*, August 2014
23. Ozkan, A., Sitharam, M.: Best of both worlds: uniform sampling in cartesian and cayley molecular assembly configuration space. *ArXiv e-prints*, September 2014
24. Parrilo, P.A., Sturmfels, B.: Minimizing polynomial functions. In: *Proceedings of the Dimacs Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science*, pp. 83–100. American Mathematical Society (2003)
25. Robertson, N., Seymour, P.D.: Graph minors. xx. wagner’s conjecture. *J. Comb. Theor. Ser. B Spec. Issue Dedicated Professor W.T. Tutte* **92**(2), 325–357 (2004)
26. Schoenberg, I.J.: Remarks to maurice fréchet’s article ‘sur la définition axiomatique d’une classe d’espaces distanciés vectoriellement applicable sur l’espace de hilbert’. *Ann. Math.* **36**(3), 724–732 (1935)
27. Sitharam, M., Gao, H.: Characterizing graphs with convex and connected cayley configuration spaces. *Discrete Comput. Geom.* **43**(3), 594–625 (2010)
28. Sitharam, M., Ozkan, A., Pence, J., Peters, J.: Easal: efficient atlasing, analysis and search of molecular assembly landscapes. *CoRR*, abs/1203.3811 (2012)
29. Sitharam, M., Wang, M., Gao, H.: Cayley configuration spaces of 1-dof tree-decomposable linkages, part I: structure and extreme points. *CoRR*, abs/1112.6008 (2011)
30. Sitharam, M., Wang, M., Gao, H.: Cayley configuration spaces of 1-dof tree-decomposable linkages, part II: combinatorial characterization of complexity. *CoRR*, abs/1112.6009 (2011)
31. Tarazaga, P.: Faces of the cone of euclidean distance matrices: characterizations, structure and induced geometry. *Linear Algebra Appl.* **408**, 1–13 (2005)
32. Trevisan, L.: On khot’s unique games conjecture. *Bull. AMS* **49**, 91–111 (2011)
33. Witsenhausen, H.S.: Minimum dimension embedding of finite metric spaces. *J. Comb. Theory, Ser. A* **42**(2), 184–199 (1986)
34. Wu, R., Ozkan, A., Bennett, A., Agbandje-Mckenna, M., Sitharam, M.: Robustness measure for an adeno-associated viral shell self-assembly is accurately predicted by configuration space atlasing using easal. In: *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine, BCB 2012*, pp. 690–695. ACM, New York (2012)

Discovering Geometric Theorems from Scanned and Photographed Images of Diagrams

Dan Song¹(✉), Dongming Wang^{1,2}, and Xiaoyu Chen^{1,2}

¹ LMIB - School of Mathematics and Systems Science,
Beihang University, Beijing 100191, China
songdan151100@163.com

² SKLSDE - School of Computer Science and Engineering,
Beihang University, Beijing 100191, China

Abstract. This paper extends our work on automated discovery of geometric theorems from diagrams by taking scanned and photographed images instead of images produced with dynamic geometry software. We first adopt techniques of Hough transform and randomized detection algorithms to detect geometric objects from scanned and photographed images, then use methods of image matching to recognize labels for the detected geometric objects, and finally employ numerical-computation-based methods to mine geometric relations among the objects. Experiments with a preliminary implementation of the techniques and methods demonstrate the effectiveness and efficiency of geometric information retrieval from scanned and photographed images for the purpose of discovering geometric theorems automatically.

Keywords: Shape recognition · Pattern matching · Theorem discovery · Geometric knowledge management

1 Introduction

The authors have proposed an approach for discovering geometric theorems automatically from images of diagrams in [5], where the images are produced from diagrams drawn by using dynamic geometry software (DGS). This paper extends our previous work by taking images scanned or photographed from diagrams on paper in classic textbooks.

Diagrams are widely used to illustrate given geometric theorems. A natural question is how to find those theorems which a given diagram may illustrate or imply. Answers to this question may lead to methods to search for geometric theorems from images of diagrams, to create geometric theorem bases semi-automatically, and to discover possibly new geometric theorems. We have given an answer to the above question with an efficient approach for images of DGS-produced diagrams. The basic idea is to first recognize geometric objects and their labels and mine geometric relations from the images, then formulate propositions as candidate geometric theorems (or called *conjectures*), and

finally confirm some of the propositions as theorems by provers based on algebraic methods. In this paper, we extend the approach to deal with scanned and photographed images of diagrams in which the implied geometric relations are less exact. In this case, the retrieval of geometric information becomes more difficult and requires some special techniques from image matching and pattern recognition.

The rest of the paper is organized as follows. In Sect. 2, our general approach for automated discovery of geometric theorems from images of diagrams is reviewed. Special techniques for retrieving geometric information from scanned and photographed images are presented in Sect. 3, followed by discussions on implementation issues with experimental results given in Sect. 4. The paper is concluded with some remarks in Sect. 5.

The work described in [5] and this paper demonstrates the feasibility of discovering geometric theorems from different types of images of diagrams and has potential applications in geometric knowledge management and education¹ [1]. For example, geometric knowledge bases can be expanded by adding theorems discovered from images of diagrams automatically or semi-automatically; formalized representation of discovered theorems may facilitate the processing and management of such theorems and their automation. Moreover, we expect that students will be able to see statements of geometric theorems in natural languages generated automatically by drawing diagrams on the screen. We are encouraged to further our study on the subject, in particular developing software tools to help generate geometric theorems from images of diagrams taken from the Internet or hand-sketched on mobile devices. It is also expected that our approach and the underlying ideas can be extended to deal with digitalized mathematical literature (e.g., recognizing images and figuring out what they illustrate).

2 Discovering of Geometric Theorems from Images of Diagrams: A General Approach

In this section, we provide a short review of the general approach for automated discovering of geometric theorems from images of diagrams described in [5]. The approach can be used to discover, from any given image of diagram in plane Euclidean geometry, one or several geometric theorems which the diagram implies. It has a high probability of success when the diagram may be used to illustrate some theorem and is drawn precisely enough. The approach works by making use of techniques of pattern recognition to retrieve geometric information, introducing strategies to mine candidate theorems, and employing

¹ In [1] it is explained how to translate mathematical problems stated in natural languages to propositions formulated in Zermelo–Fraenkel axiomatic set theory. Combination of the method of natural language processing discussed therein with our approach of theorem discovering from images of diagrams may help increase the degree of automation for mathematical problem solving in education.

automated provers to validate theorems. Three key concepts used in our approach are recalled below.

- A point P in an image of diagram is a point of *attraction* if P is an endpoint of a segment, or the starting point of a half line, or ... (see [5] for details).
- A point P of attraction is a *characteristic* point if it is used in the expressions of properties or the specifications of propositions implied in the diagram.
- A geometric relation is a *branch relation* if it can be derived from other geometric relations by simple deduction in some specific cases, for example, $\|AB = \|EF\|$ is a *branch relation* when $\|AB\| = \|CD\|$ and $\|CD\| = \|EF\|$ hold.

The general approach consists of four main steps presented briefly as follows.

1. Specifying basic objects and relations. Identify a set \mathbb{O} of basic geometric objects and a set \mathbb{R} of basic geometric relations and represent the objects, their labels, and their relations formally using a formal language (such as Geometry Description Language presented in [4]). Let an image I of a diagram for some geometric theorem involving the elements of \mathbb{O} and \mathbb{R} be given.
2. Retrieving geometric information. First use shape recognition techniques to detect geometric objects from I , then use techniques of image cutting and matching to recognize labels of the detected objects, and finally use numerical computation to mine geometric relations among the detected objects.
3. Generating candidate propositions. From the mined geometric relations, remove irrelevant ones by dropping points which are not *characteristic* and by discarding *branch relations*. Then use derived geometric relations to replace some of the remaining basic ones and generate candidate propositions from the resulting geometric relations.
4. Mining theorems. Validate candidate propositions to obtain geometric theorems by proving them using a theorem prover (such as GEOTHER [12]) based on algebraic methods and output the proved theorems.

In step 2 above, different techniques may be required when the images of diagrams are produced in different ways. In the next section, we will focus our study on special techniques for the recognition of geometric objects and their labels and for the mining of geometric relations from scanned and photographed images of diagrams.

Remark 1. By *discovering*, we mean generating geometric theorems from images of diagrams rather than finding theorems new in geometry.

3 Retrieval of Geometric Information from Scanned and Photographed Images

We restrict our study to images of diagrams which may occur in textbooks of plane Euclidean geometry. The retrieval of geometric information from such images includes the recognition of basic geometric objects and their labels and the mining of basic geometric relations. In this section, we explain how to perform these tasks and discuss how to formally represent the retrieved information.

3.1 Recognizing Basic Geometric Objects

We consider the following three kinds of basic geometric objects which are used to form most diagrams in plane Euclidean geometry.

- **Point.** A point is represented by a pair of coordinates (x, y) in the coordinate system determined by the image of diagram.
- **Line.** A straight line (with no endpoint) is represented by $\text{line}(P_1, P_2)$, where P_1 and P_2 are two distinct points on the line. Similarly, a half line is represented by $\text{halfline}(O, P)$, where O is the initial point and P is another point distinct from O on the halfline; a segment is represented by $\text{segment}(E_1, E_2)$, where E_1 and E_2 are the endpoints of the segment.
- **Circle.** A circle may be represented by $\text{circle}(O, r)$, where O is the center and $r (> 0)$ is the radius of the circle, or $\text{circle}(A, B, C)$, where A, B, C are three different points incident to the circle.

Any basic geometric object \mathcal{O} introduced above has the form $f(O_1, \dots, O_n)$, where f is the name of the object and O_1, \dots, O_n are other geometric objects which determine \mathcal{O} . The recognition of \mathcal{O} consists in determining O_1, \dots, O_n . For example, recognition of a circle means determining the coordinates of the center and the radius of the circle.

In order to use previously retrieved information, recognition tasks should be arranged in the order of circles, lines, and points. In what follows, we present the main algorithmic steps for recognizing basic geometric objects. Most of them are refined from their corresponding steps described in [5].

1. **Circle recognition.** Let \mathbb{C} be the set of the detected circles.
 - (a). Preprocessing. For any given scanned (or photographed) image of diagram I , perform resizing, image enhancement, contrast adjustment, gray-ing, binarization, and thinning² operations on I to obtain I_1 .
 - (b). Detection. Detect circles from I_1 by using the following randomized detection algorithm derived from the algorithms proposed in [2, 3].
 - i. Set $N := 0$ and $\mathbb{C} := \emptyset$, where N is a counter.
 - ii. Randomly generate two different horizontal lines l_1 and l_2 . Set $\mathcal{P}_1 := \emptyset$, $\mathcal{P}_2 := \emptyset$. For each black point $P \in I_1$, if P is incident to l_1 , then set $\mathcal{P}_1 := \mathcal{P}_1 \cup \{P\}$; if P is incident to l_2 , then set $\mathcal{P}_2 := \mathcal{P}_2 \cup \{P\}$. Set $F := 0$, where F is a flag.
 - iii. For each set $\{P_1, P_2 \in \mathcal{P}_1, P_3, P_4 \in \mathcal{P}_2 \mid P_1 \neq P_2, P_3 \neq P_4\}$ of four points, if P_4 is incident to³ the candidate circle $C := \text{circle}(O, r)$ ⁴ determined by $\{P_1, P_2, P_3\}$ and furthermore the condition $n_{\text{Count}} / (2\pi r) \geq c_{\text{Rate}}$ ⁵ is satisfied, where n_{Count} denotes the number of black points incident to C in I_1 and $c_{\text{Rate}} \in [0, 1]$ is a prespecified tolerance

² For example, using Zhang's technique of parallel thinning [11].

³ $|||OP_4|| - r| \leq \tau_{pc}$, where τ_{pc} is a prespecified tolerance.

⁴ The radius of C is measured by the number r of pixel points.

⁵ On the one hand, this condition allows a certain degree of numerical errors; on the other hand, it is applicable for both large and small radii of circles.

(e.g., $c_{\text{Rate}} = 0.5$), then set $\mathbb{C} := \mathbb{C} \cup \{C\}$, $N := 0$, $F := 1$; otherwise, proceed to the next set of four points.

- iv. If $F = 0$, then set $N := N + 1$.
- v. If $N \geq T$, where T is a predetermined maximum number of failure times (e.g., $T = 50$), then the algorithm terminates and output the set \mathbb{C} . Otherwise, go to step ii.

2. Line recognition.

- (a) Preprocessing. Perform binarization and thinning operations on I to obtain I_2 .
- (b) Detection. Apply the improved progressive probabilistic Hough transform (see [6,7]) on I_2 to acquire a set \mathbb{L} of segments.
- (c) Postprocessing. Merge those short segments which are recognized from longer ones, determine the endpoints of each segment more accurately, identify the actual line types of the obtained segments, drop nonexistent lines,⁶ and finally obtain a set \mathbb{L} of lines.

3. Collecting points of interest.

Collect as points of interest the centers of circles in \mathbb{C} , the endpoints of lines in \mathbb{L} , and all the intersection points (if such points exist) of two basic geometric objects in $\mathbb{C} \cup \mathbb{L}$.

It is necessary to assign a unique label to each of the recognized basic geometric objects, such that geometric relations among the objects can be expressed clearly. Labels for important geometric objects (such as points) are usually contained in diagrams. We shall present a method to extract information on label assignment from images of diagrams in the next subsection.

Remark 2. Step 1(a) is adopted to preprocess the image I for later steps. Here “resizing” is for improving the efficiency of image processing, “image enhancement” and “contrast adjustment” are for making the background and foreground of the image easily distinguishable, and “thinning” is for improving the accuracy of geometric object recognition. Moreover, because of inexactness of images of diagrams which we deal with, the algorithm used by us for circle recognition here is different from the one used in [5].

3.2 Recognizing Labels of Geometric Objects

In this paper, we only deal with the relatively easy case when labels have no overlap with geometric objects. In this case, which happens actually for most diagrams in textbooks, labels in an image of diagram may be recognized by checking whether each of them matches a character template according to the following steps.

- 1. Preparing character templates. Cut all the labels in the images for testing automatically and then screen out character templates from them by data analysis (i.e., analyzing SIFT features [8]) and then add the representative one for each letter to a set \mathbb{T} of binary images of letters.

⁶ Due to numerical errors of line detection, some arcs may be recognized as line segments, in particular when the radius of the circle is large.

2. Preprocessing. Redraw the basic geometric objects recognized in Subsect. 3.1 with white (background) color. Set $\mathbf{L} := []$ and $\mathbf{P} := []$.
3. Cutting out blocks with labels.
 - (a). For a black (foreground) point $P(x, y)$, initialize a small rectangular window $W(t, d, l, r)$ containing P , where t, d, l, r denotes top, down, left and right bounds respectively, with $t := y - 1, d := y + 1, l := x - 1, r := x + 1$.
 - (b). Gradually expand t, d, l, r of W towards outside until there are no black points on the four boundaries of W . Then $W(t, d, l, r)$ determines an image block B .
4. Matching character templates. For each image block B with corresponding window $W(t, d, l, r)$, find the character template T of letter L such that the distance⁷ $d_{T,B}$ is equal to $d_{T,B} := \min\{d_{T_p,B} \mid T_p \in \mathbb{T}\}$; if $d_{T,B}$ is less than a specified threshold, then append L to the list \mathbf{L} . Compute the center $((l + r)/2, (t + d)/2)$ of the cutting window and append it to the list \mathbf{P} . Note that the center of the i th label in \mathbf{L} corresponds to the i th point in \mathbf{P} .

After these steps, we assign the recognized labels to the corresponding geometric objects. For any geometric object not labeled in the image, a unique label is automatically generated by our program to refer to the object. Taking the scanned image of diagram (Fig. 1) as an example, we show the geometric objects obtained and the labels recognized or generated automatically.

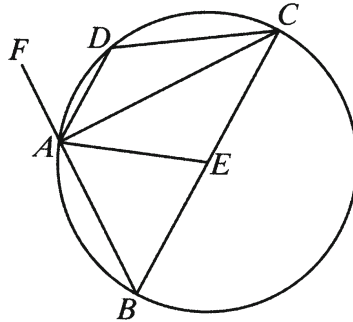


Fig. 1. A scanned image of diagram

- The set \mathbb{P} of points of interest:

$$C := (724, 1424), A := (165, 427), D := (295, 186), B := (360, 819), \\ E := (543, 480), F := (66, 228).$$

⁷ $d_{T,B}$ is computed by the SIFT algorithm [8], where $T \in \mathbb{T}$ and B is an image block obtained in step 3(b).

– The set \mathbb{L} of lines:

$$a := \text{segment}(A, C), b := \text{segment}(D, C), c := \text{segment}(B, C), \\ d := \text{segment}(F, B), e := \text{segment}(A, E), f := \text{segment}(A, D).$$

– The set \mathbb{C} of circle: $g := \text{circle}(E, 383)$.

The features of diagrams are depicted mainly via geometric relations among the involved objects. Based on the retrieved information of geometric objects, we shall present a method to mine geometric relations in the next subsection.

Remark 3. For step 4, an alternative is to use a general-purpose engine of Optical Character Recognition (OCR [9]). There are many such engines available, for example, Tesseract [14], an OCR engine sponsored by Google since 2006 and considered as one of the most accurate open-source OCR engines currently. We have also written a routine for step 4 using Tesseract. However, this routine is less efficient than the one based on the method described in step 4, while it can deal with a set of labels in different fonts for which the latter works unsatisfactorily.

3.3 Mining Basic Geometric Relations

As an example, we take the relations listed in Table 1 as basic geometric relations. These relations can be used to describe most of the features about sizes and positions of geometric objects and from them many other relations can be derived.

To mine basic geometric relations, we check whether there exists a geometric relation among some of the recognized geometric objects which is *close* to one of the basic geometric relations in Table 1. Here closeness is measured algebraically by a function $|\cdot|$ which depends on the basic geometric relation in question. If the closeness is less than a prespecified threshold value, then the basic relation is taken.

Table 1. Basic geometric relations

Representation	Meaning
$\text{incident}(A, l)$	point A lies on line l
$\text{pointOnC}(A, o)$	point A is on circle o
$\text{parallel}(l_1, l_2)$	l_1 is parallel to l_2
$\text{perpendicular}(l_1, l_2)$	l_1 is perpendicular to l_2
$\text{equal}(\text{distance}(A, B), \text{distance}(C, D))$ or $\ AB\ = \ CD\ $	the distance between A and B is equal to the distance between C and D
$\text{equal}(\text{size}(\text{angle}(A, B, C)), \text{size}(\text{angle}(D, E, F)))$ or $\angle ABC = \angle DEF$	the size of $\angle ABC$ is equal to the size of $\angle DEF$

Let \mathbb{P} be the set of points, \mathbb{L} be the set of lines, and \mathbb{C} be the set of circles recognized from an image I . The process of mining basic geometric relations from $\mathbb{P}, \mathbb{L}, \mathbb{C}$ works by considering all the relations in Table 1 as follows.

Let $\mathbb{R} := \emptyset$ and τ_1, \dots, τ_6 be small positive real numbers to be used as threshold values.

1. For each $A \in \mathbb{P}$ and each $l \in \mathbb{L}$, check whether $\|Al\| < \tau_1$, where $\|Al\|$ denotes the distance from A to l ; if so, then add `incident(A, l)` to \mathbb{R} .
2. For each $A \in \mathbb{P}$ and each $o \in \mathbb{C}$, check whether $|\|AO\| - r| < \tau_2$, where O is the center and r is the radius of o ; if so, then add `pointOnC(A, o)` to \mathbb{R} .
3. For each pair of $l_1, l_2 \in \mathbb{L}$, check whether $|\text{size}(\alpha_{l_1 l_2}) - 0| < \tau_3$, where $\alpha_{l_1 l_2}$ denotes the included angle of l_1 and l_2 ; if so, then add `parallel(l_1, l_2)` to \mathbb{R} .
4. For each pair of $l_1, l_2 \in \mathbb{L}$, check whether $|\text{size}(\alpha_{l_1 l_2}) - \frac{\pi}{2}| < \tau_4$; if so, then add `perpendicular(l_1, l_2)` to \mathbb{R} .
5. For each set of points $A, B, C, D \in \mathbb{P}$, where A and B are connected by a line, and so are C and D , check whether $|\|AB\| - \|CD\|| < \tau_5$; if so, then add `equal(distance(A, B), distance(C, D))` to \mathbb{R} .
6. For each set of points $A, B, C, D, E, F \in \mathbb{P}$, where A and B are connected by a line, as are B and C , as are D and E , and as are E and F , check whether $|\text{size}(\angle ABC) - \text{size}(\angle DEF)| < \tau_6$; if so, then add `equal(size(angle(A, B, C)), size(angle(D, E, F)))` to \mathbb{R} .
7. Return the set \mathbb{R} of basic geometric relations.

For example, following the above steps, one may obtain 12 basic geometric relations for Fig. 1:

`incident(A, d), incident(E, c), pointOnC(C, g), pointOnC(A, g),
pointOnC(D, g), pointOnC(B, g), perpendicular(a, d), parallel(c, f),
equal(distance(C, D), distance(A, B)),
equal(distance(C, E), distance(B, E)),
equal(distance(C, E), distance(A, E)),
equal(distance(B, E), distance(A, E)).`

From these relations, five candidate propositions are formulated according to step 3 in Sect. 2 (see also Sect. 3 in [5] and No. 5 in Table 3).

Proposition₁([`equal(distance(D, C), distance(A, B)), equal(distance(E, B), distance(A, E)), $E := \text{midpoint}(B, C)$, incident(D , circle(B, C, A)), parallel(segment(B, C), segment(A, D))],
[perpendicular(segment(A, C), segment(A, B))]).`

Proposition₂([`equal(distance(D, C), distance(A, B)), equal(distance(E, B), distance(A, E)), $E := \text{midpoint}(B, C)$, incident(D , circle(B, C, A)), perpendicular(segment(A, C), segment(A, B))],
[parallel(segment(B, C), segment(A, D))]).`

Proposition₃([equal(distance(D, C), distance(A, B)), equal(distance(E, B), distance(A, E)), $E := \text{midpoint}(B, C)$, parallel(segment(B, C), segment(A, D)), perpendicular(segment(A, C), segment(A, B))), [incident(D , circle(B, C, A))]).

Proposition₄([equal(distance(D, C), distance(A, B)), $E := \text{midpoint}(B, C)$, incident(D , circle(B, C, A)), parallel(segment(B, C), segment(A, D)), perpendicular(segment(A, C), segment(A, B))), [equal(distance(E, B), distance(A, E))]).

Proposition₅([equal(distance(E, B), distance(A, E)), $E := \text{midpoint}(B, C)$, incident(D , circle(B, C, A)), parallel(segment(B, C), segment(A, D)), perpendicular(segment(A, C), segment(A, B))), [equal(distance(D, C), distance(A, B))]).

Each candidate proposition is presented in the form Proposition_{*index*}([R_1, \dots, R_m], [R_c]), where *index* is a unique integer generated for the proposition, R_1, \dots, R_m form the hypothesis, and R_c is the conclusion of the proposition. The five candidate propositions listed above have all been proved to be true by using the theorem prover GEOTHER [12].

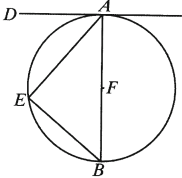
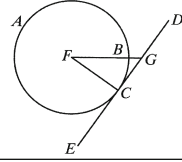
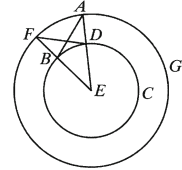
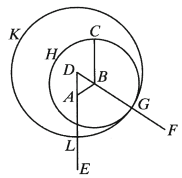
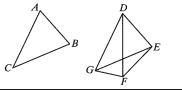
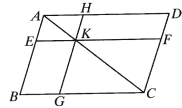
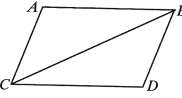
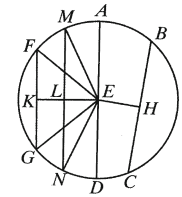
Remark 4. When geometric objects and relations among the objects are detected, it is relatively easy to draw diagrams satisfying the relations. In fact, automated generation of diagrams is a problem of geometric constraint solving which has been studied extensively. For example, the second author has shown in [10] how to generate dynamic diagrams for geometric theorems automatically.

4 Implementation and Experiments

The techniques of geometric information retrieval described in Sect. 3 have been implemented in C++ using OpenCV [13]. The formulation and validation of candidate propositions (see steps 3 and 4 in Sect. 2) have been implemented in Java using the Maple package GEOTHER [12]. Images of diagrams used for our experiments were scanned from a Chinese version of Euclid's *Elements*⁸ and photographed from the same classic using a mobile phone Sony LT26i.

⁸ Currently, carefully photographed images of diagrams can be successfully processed. The problem of processing images of diagrams carelessly photographed is still under investigation.

Table 2. Test results for scanned or photographed images

No.	Image	Time	Size	Source	Candidates	Theorems
1		0.228	506×500	Scan	2	0
2		0.299	572×500	Scan	3	1
3		0.258	517×500	Scan	11	11
4		0.300	508×500	Scan	8	1
5		0.474	917×500	Scan	8	4
6		0.428	790×500	Scan	21	16
7		0.458	897×500	Scan	4	4
8		0.639	500×544	Scan	16	11

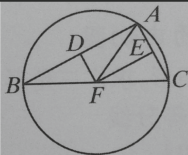
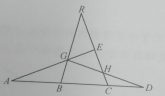
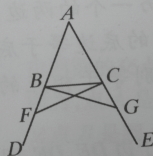
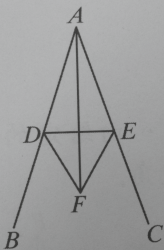
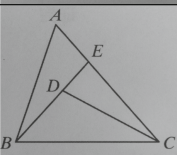
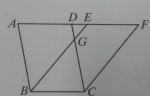
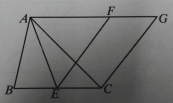
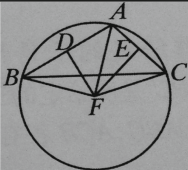
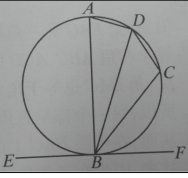
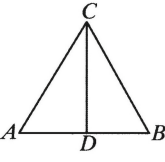
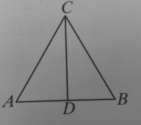
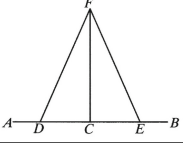
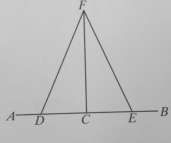
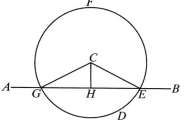
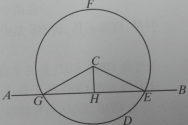
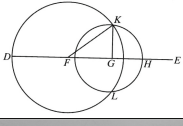
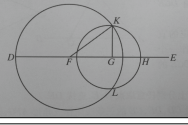
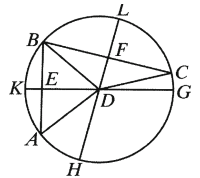
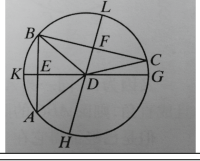
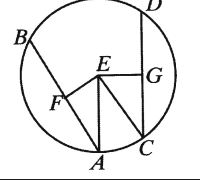
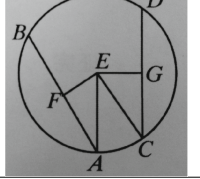
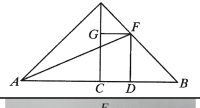
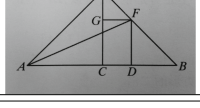
No.	Image	Time	Size	Type	Candidates	Theorems
9		0.288	534×500	Photo	18	17
10		0.360	816×500	Photo	8	1
11		0.293	506×500	Photo	11	7
12		0.373	500×737	Photo	6	5
13		0.201	502×500	Photo	4	3
14		0.275	742×500	Photo	10	5
15		0.295	752×500	Photo	1	0
16		0.390	627×500	Photo	20	6

Table 3. Test results for scanned images in comparison with photographed images

No.	Image	Time	Size	Source	Candidates	Theorems
1		0.276	736×500	Scan	3	3
2		0.227	623×500	Photo	3	3
3		0.693	1083×500	Scan	14	0
4		0.448	873×500	Photo	18	0
5		0.259	519×500	Scan	5	5
6		0.235	521×500	Photo	5	5
7		0.273	517×500	Scan	21	20
8		0.283	594×500	Photo	21	20
9		0.263	542×500	Scan	4	2

No.	Image	Time	Size	Type	Candidates	Theorems
10		0.221	571×500	Photo	6	5
11		0.234	507×500	Scan	5	5
12		0.285	625×500	Photo	1	0
13		0.298	631×500	Scan	6	5
14		0.218	596×500	Photo	6	5
15		0.308	694×500	Scan	6	5
16		0.374	746×500	Photo	6	5
17		0.288	762×500	Scan	8	8
18		0.429	769×500	Photo	6	4

No.	Image	Time	Size	Type	Candidates	Theorems
19		0.386	514×500	Scan	8	1
20		0.337	572×500	Photo	9	4
21		0.354	500×519	Scan	9	8
22		0.277	539×500	Photo	7	4
23		0.419	854×500	Scan	13	1
24		0.370	791×500	Photo	21	7

We have made some preliminary experiments with a number of selected images of diagrams, for which geometric information retrieval was carried out on a PC with 2.83 GHz CPU and 3.00 GB of memory. We present some of the experimental results in Tables 2 and 3,⁹ where “Time” is recorded in seconds for information retrieval from the image; “Size” denotes the image size in pixels; “Type” is used to indicate the means by which the image was obtained, “Scan” is for scanned image, and “Photo” is for photographed image; “Candidates” denotes the number of generated candidate propositions; and “Theorems” denotes the number of proved theorems.

⁹ The objects recognized and the theorems discovered automatically from images of diagrams are presented on the website <http://geo.cc4cm.org/data/recognizer/>.

Some of the generated candidate propositions may be wrong. For example, the two candidate propositions

Proposition₁([incident(C , circle(B, A, D)), perpendicular(segment(B, D), segment(A, D)), $B := \text{foot}(A, \text{segment}(E, F))$, $G := \text{midpoint}(B, A)$, equal(distance(G, B), distance(D, G))), [midpoint($B, \text{segment}(E, F)$)]);

Proposition₃([incident(C , circle(B, A, D)), perpendicular(segment(B, D), segment(A, D)), $G := \text{midpoint}(B, A)$, midpoint($B, \text{segment}(E, F)$), equal(distance(G, B), distance(D, G))), [$B := \text{foot}(A, \text{segment}(E, F))$])

generated for No. 9 in Table 3 are wrong, as confirmed by GEOTHER. Tables 2 and 3 contain no candidate propositions which are known theorems but cannot be proved automatically. Of course, there are such propositions, for instance, the candidate propositions formulated for the image of diagram of Morley’s theorem (see No. 10 in Table 6 of [5], page 23), which could not be proved by GEOTHER.

For our test results, the correctness rate for label recognition is about 92% statistically. Due to inaccuracy of geometric information retrieval with large threshold values, some undesired distance relations may be produced, bringing more candidate propositions (see¹⁰ Nos. 3 and 4 in Table 3). Furthermore, for a scanned and a photographed image of the same diagram (see Table 3), such large threshold values may lead to different sets of undesired distance relations, and thus lead to different numbers of candidates propositions formulated and different numbers of theorems discovered. Appropriate trade-off in the choice of threshold values for different images can help improve the accuracy of geometric information retrieval. In our implementation, the threshold values are determined by making experiments on a set of test images with fixed size, e.g., 400×400 . For each given image I of a diagram, the threshold values are adjusted automatically according to the size of I .

Some of the discovered geometric theorems which an image of diagram illustrates are trivial, while most of the theorems discovered from the image are nontrivial and can be found in standard textbooks. Whether the trivial theorems can be ruled out depends on how candidate propositions are formulated.

Remark 5. There may be a notable difference between the number of candidate propositions formulated and the number of theorems proved. This may happen in particular when threshold values are not properly initialized — in this case, undesired false geometric relations may be retrieved, leading to self-contradictory candidate propositions. For example, the reader may note that for No. 16 in Table 2 and Nos. 23 and 24 in Table 3 most of the candidate propositions are not

¹⁰ There is a theorem about the equality of the areas of the parallelograms $ABCD$ and $EBCH$ implied by the image. The theorem could not be mined because area relations are not among our chosen basic geometric relations.

proved to be theorems, because the threshold value chosen for τ_5 (see step 5 in Sect. 3.3) is too big.

5 Concluding Remarks

We have presented different techniques for geometric information retrieval, including recognition of geometric objects and their labels and mining of geometric relations from scanned and photographed images of diagrams. These techniques have been used to extend our general approach for automated discovering of geometric theorems from images, which has potential applications in geometric knowledge base creation and knowledge management.

We are still improving the correctness rate of geometric object and label recognition and the overall efficiency of our approach. We expect to further extend our work to deal with such images as photographed images of hand-drawn diagrams, which are hardly accurate. Moreover, we plan to develop a software tool to help collect geometric theorems discovered automatically from images of diagrams which are taken from the Internet by means of search.

Acknowledgements. The authors wish to thank the referees for their constructive comments which have helped improve the paper significantly. This work has been supported by the project SKLSDE-2015ZX-18.

References

1. Arai, N.H., Matsuzaki, T., Iwane, H., Anai, H.: Mathematics by machine. In: Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation (Kobe, Japan, July 23–25, 2014), pp. 1–8. ACM Press, New York
2. Chen, T.C., Chung, K.L.: A new randomized algorithm for detecting lines. *Real-Time Imaging* **7**(6), 473–481 (2001)
3. Chen, T.C., Chung, K.L.: An efficient randomized algorithm for detecting circles. *Comput. Vis. Image Underst.* **83**(2), 172–191 (2001)
4. Chen, X.: Representation and automated transformation of geometric statements. *J. Syst. Sci. Complex.* **27**(2), 382–412 (2014)
5. Chen, X., Song, D., Wang, D.: Automated generation of geometric theorems from images of diagrams. *Geometric Reasoning – Special issue of Annals of Mathematics and Artificial Intelligence*. Springer (2014). doi:[10.1007/s10472-014-9433-7](https://doi.org/10.1007/s10472-014-9433-7)
6. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. *Commun. Assoc. Comput. Mach.* **15**(1), 11–15 (1972)
7. Galambos, C., Kittler, J., Matas, J.: Gradient based progressive probabilistic Hough transform. *Vis. Image Signal Process.* **148**(3), 158–165 (2001)
8. Lowe, G.D.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision* **60**(2), 91–110 (2004)
9. Mori, S., Nishida, H., Yamada, H.: *Optical Character Recognition*. Wiley, New York (1999)
10. Wang, D.: Automated generation of diagrams with Maple and Java. In: Joswig, M., Takayama, N. (eds.) *Algebra, Geometry, and Software Systems*, pp. 277–287. Springer-Verlag, Berlin (2003)

11. Zhang, T.Y., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. *Commun. Assoc. Comput. Mach.* **27**(3), 236–239 (1984)
12. GEOTHER. <http://www-polsys.lip6.fr/wang/GEOTHER/>. Accessed on 24 April 2015
13. OpenCV. <http://opencv.org/>. Accessed on 24 April 2015
14. Tesseract-OCR. <http://code.google.com/p/tesseract-ocr/>. Accessed on 24 April 2015

Combinatorial Rigidity and Independence of Generalized Pinned Subspace-Incidence Constraint Systems

Menghan Wang^(✉) and Meera Sitharam

University of Florida, Gainesville, Florida
menghan@cise.ufl.edu

Abstract. Given a hypergraph H with m hyperedges and a set X of m pins, i.e. globally fixed subspaces in Euclidean space \mathbb{R}^d , a *pinned subspace-incidence system* is the pair (H, X) , with the constraint that each pin in X lies on the subspace spanned by the point realizations in \mathbb{R}^d of vertices of the corresponding hyperedge of H . Pinned subspace-incidence systems arise in modeling dictionary learning problems as well as biomaterials such as cell wall microfibrils. We are interested in combinatorial characterization of pinned subspace-incidence systems that are *minimally rigid*, i.e. those systems that are guaranteed to generically yield a locally unique realization. As is customary, this is accompanied by a characterization of generic independence as well as rigidity. Previously, such a combinatorial rigidity characterization is only known for a more restricted version of pinned subspace-incidence systems, with H being a uniform hypergraph and pins in X being 1-dimension subspaces. In this paper, we extend the combinatorial characterization to general pinned subspace-incidence systems, with H being a non-uniform hypergraph and pins in X being subspaces with arbitrary dimension. As there are generally many data points per subspace in a dictionary learning problem, which can only be modeled with pins of dimension larger than 1, such an extension enables application to a much larger class of dictionary learning problems.

1 Introduction

A *pinned subspace-incidence system* (H, X) is an incidence constraint system specified as a hypergraph H together with a set X of subspaces or *pins* in \mathbb{R}^d in one-to-one correspondence with the hyperedges of H . A realization of (H, X) assigns points in \mathbb{R}^d to the vertices of H , thereby subspaces to the hyperedges of H . The subspace in \mathbb{R}^d corresponding to a hyperedge of H contains the associated pin from X . We are interested in characterization of pinned subspace-incidence systems that are *minimally rigid*, i.e. those systems that are guaranteed to generically yield a locally unique realization.

In a previous paper [19], we used pinned subspace-incidence systems towards solving *fitted dictionary learning* problems, i.e. dictionary learning with specified

This research was supported in part by the grant NSF CCF-1117695.

underlying hypergraphs. *Dictionary learning* (aka sparse coding) is the problem of obtaining a set *dictionary vectors* that sparsely represent a set of given data points in \mathbb{R}^d . Geometrically, such a sparse representation can be viewed as a subspace arrangement spanned by the dictionary vectors that contains all the data points. In fitted dictionary learning, the underlying hypergraph H of the subspace arrangement is specified, and the problem becomes a pinned subspace-incidence systems with the pins corresponding to the span of data points on each subspace.

Moreover in a recent paper [3], we have used pinned subspace-incidence systems in modeling biomaterials such as cross-linking cellulose and collagen microfibrils in cell walls [4, 5, 20]. In such materials, each fibril is attached to some fixed larger organelle/membrane at one site, and cross-linked at two locations with other fibrils. Consequently, they can be modeled using a pinned line-incidence system with H being a graph, where each fibril is modeled as an edge of H with the two cross-linkings as its two vertices, and the attachment is modeled as the corresponding pin.

We gave in [19] a combinatorial characterization of minimal rigidity for a restricted version of pinned subspace-incidence system, with the underlying hypergraph H being a uniform hypergraph and pins in X being 1-dimension subspaces of \mathbb{R}^d (treated as points in $\mathbb{P}^{d-1}(\mathbb{R})$ – see Sect. 2 for details).

1.1 Contributions

In this paper, we extend the combinatorial characterization of minimal rigidity to general pinned subspace-incidence systems, where H is any non-uniform hypergraph and each pin in X is a subspace with arbitrary dimension. Such an extension enables application to a much larger class of fitted dictionary learning problems, since there are generally many data points per subspace in a dictionary learning problem, which can only be modeled with pins of dimension larger than 1.

As in our previous paper [19], we apply the classic method of White and Whiteley [24] to combinatorially characterize the rigidity of general pinned subspace-incidence systems. The primary technique is using the Laplace decomposition of the *rigidity matrix*, which corresponds to a *map-decomposition* [21] of the underlying hypergraph. The polynomial resulting from the Laplace decomposition is called the *pure condition*, which characterizes the conditions that the framework has to avoid for the combinatorial characterization to hold.

1.2 Related Works

Previous works on related types of geometric constraint frameworks include pin-collinear body-pin frameworks [14], direction networks [26], slider-pinning rigidity [22], body-cad constraint system [10], k -frames [24, 25], and affine rigidity [8]. However, we are not aware of any previous results on systems that are similar to pinned subspace-incidence systems.

2 Preliminaries

In this section, we introduce the formal definition of pinned subspace-incidence systems and basic concepts in combinatorial rigidity.

A *hypergraph* $H = (V, E)$ is a set V of vertices and a set E of hyperedges, where each hyperedge is a subset of V . The *rank* $r(H)$ of a hypergraph H is the maximum cardinality of any edge in E , i.e. $r(H) = \max_{e_k \in E} |e_k|$. A hypergraph is *s-uniform* if all edges in E have the same cardinality s . A *configuration* or *realization* of a hypergraph $H = (V, E)$ in \mathbb{R}^d is a mapping from the vertices of H to the points in \mathbb{R}^d , i.e. $p : V \rightarrow \mathbb{R}^d$. When there is no ambiguity, we simply use p_i to denote the point $p(v_i)$, and $p(e_k)$ to denote the set of points $\{p(v_i) | v_i \in e_k\}$.

An example of a rank-3 hypergraph is given in Fig. 1a.

In the following, we use $\langle S \rangle$ to denote the subspace *spanned* by a set S of points in \mathbb{R}^d .

Definition 1 (Pinned Subspace-Incidence System). A pinned subspace-incidence system in \mathbb{R}^d is a pair (H, X) , where $H = (V, E, m)$ is a weighted hypergraph of rank $r(H) < d$, and $X = \{x_1, x_2, \dots, x_{|E|}\}$ is a set of pins (subspaces of \mathbb{R}^d) in one-to-one correspondence with the hyperedges of H . Here the weight assignment is a function $m : E \rightarrow \mathbb{Z}^+$, where $m(e)$ denotes the dimension of the pin associated with the hyperedge e . Often we ignore the weight m and just refer to the hypergraph (V, E) as H .

A pinned subspace-incidence framework realizing the pinned subspace-incidence system (H, X) is a triple (H, X, p) , where p is a realization of H , such that for all pins $x_k \in X$, x_k is contained in $\langle p(e_k) \rangle$, the subspace spanned by the set of points realizing the vertices of the hyperedge e_i corresponding to x_k .

We may write $m(x_k)$ or simply m_k in substitute of $m(e_k)$, where x_k is the pin associated with the hyperedge e_k .

Since we only care about incidence relations, we projectivize the Euclidean space \mathbb{R}^d to treat the pinned subspace-incidence system in the real projective space $\mathbb{P}^{d-1}(\mathbb{R})$, and use the same notation for the pins and hypergraph realization when the meaning is clear from the context.

Note: as the pins in X are treated as globally fixed subspaces, the *trivial motion space* of a pinned subspace-incidence system (H, X) reduces to the identity.

Fig. 1b gives an example of a pinned subspace-incidence framework in the projective space $\mathbb{P}^{d-1}(\mathbb{R})$ with $d = 4$, where each pin is the subspace spanned by the set of cross-denoted points on the corresponding hyperedge.

Definition 2. A pinned subspace-incidence system (H, X) is independent if none of the algebraic constraints is in the ideal generated by others, which generically implies the existence of a realization. It is rigid if there exist at most finitely many realizations. It is minimally rigid if it is both rigid and independent. It is globally rigid if there exists at most one realization.

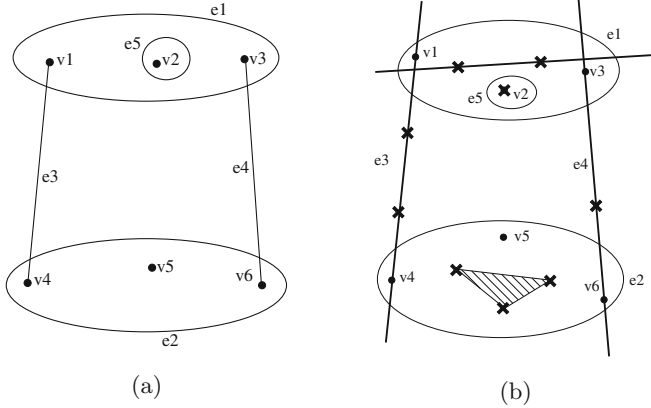


Fig. 1. (a) A rank-3 non-uniform hypergraph with 6 vertices and 5 edges, where $|e_1| = |e_2| = 3$, $|e_3| = |e_4| = 2$, $|e_5| = 1$. (b) A pinned subspace-incidence framework in $d = 4$ (projectivized in $\mathbb{P}^3(\mathbb{R})$) on the hypergraph from (a), with $m_1 = m_3 = 2$, $m_2 = 3$, $m_4 = m_5 = 1$, where the crosses on each hyperedge represent points spanning the associated pin.

3 Algebraic Representation and Linearization

In the following, we use $A[R, C]$ to denote a submatrix of a matrix A , where R and C are respectively index sets of the rows and columns contained in the submatrix. In addition, $A[R, \cdot]$ represents the submatrix containing row set R and all columns, and $A[\cdot, C]$ represents the submatrix containing column set C and all rows.

3.1 Algebraic Representation

A pin x_k associated with a hyperedge $e_k = \{v_1^k, v_2^k, \dots, v_{|e_k|}^k\}$ is constrained to be contained in the subspace $\langle p(e_k) \rangle$ spanned by the point set $\{p_1^k, p_2^k, \dots, p_{|e_k|}^k\}$. As x_k is a subspace of dimension $m_k - 1$ in $\mathbb{P}^{d-1}(\mathbb{R})$, we can pick a set of m_k points $\{x_1^k, x_2^k, \dots, x_{m_k}^k\}$ spanning x_k . Now the constraint is equivalent to requiring $\langle p(e) \rangle$ to contain each point x_l^k , for $1 \leq l \leq m_k$. We call each such point x_l^k a *atomic pin* as it acts like a pin with $m(x_l^k) = 1$.

Using homogeneous coordinates $p_i^k = [p_{i,1}^k \ p_{i,2}^k \ \dots \ p_{i,d-1}^k]$ and $x_l^k = [x_{l,1}^k \ x_{l,2}^k \ \dots \ x_{l,d-1}^k]$, we write this incidence constraint for each point x_l by letting all the $|e_k| \times |e_k|$ minors of the $|e_k| \times (d - 1)$ matrix

$$E_l^k = \begin{bmatrix} p_1^k - x_l^k \\ p_2^k - x_l^k \\ \vdots \\ p_{|e_k|}^k - x_l^k \end{bmatrix}$$

be zero. There are $\binom{d-1}{|e_k|}$ minors, giving $\binom{d-1}{|e_k|}$ equations. Note that any $d - |e_k|$ of these $\binom{d-1}{|e_k|}$ equations are independent and span the rest. So we can write the incidence constraint as $(d - |e_k|)$ independent equations:

$$\det(E_l^k[\cdot, C(t)]) = 0, \quad 1 \leq t \leq d - |e_k| \quad (1)$$

where $C(t)$ denote the following index sets of columns in E :

$$C(t) = \{1, 2, \dots, |e_k| - 1\} \cup \{|e_k| - 1 + t\}, \quad 1 \leq t \leq d - |e_k|$$

In other words, $C(t)$ contains the first $|e_k| - 1$ columns together with column $|e_k| - 1 + t$.

Now the incidence constraint for the pin x_k is represented as $m_k(d - |e_k|)$ equations for all the m_k atomic pins $\{x_1^k, x_2^k, \dots, x_{m_k}^k\}$. Consequently, the pinned subspace-incidence problem reduces to solving a system of $\sum_{k=1}^{|E|} m_k(d - |e_k|)$ equations, each of form (1). We denote this algebraic system by $(H, X)(p) = 0$.

3.2 Genericity

We are interested in characterizing *minimal rigidity* of pinned subspace-incidence systems. However, checking independence relative to the ideal generated by the variety is computationally hard and best known algorithms, such as computing Gröbner basis, are exponential in time and space [17]. However, the algebraic system can be linearized at *generic* or *regular* (non-singular) points, whereby the independence and rigidity of the algebraic system $(H, X)(p) = 0$ reduces to linear independence and maximal rank at *generic* frameworks.

In algebraic geometry, a property being generic intuitively means that the property holds on the open dense complement of an (real) algebraic variety. Formally,

Definition 3. A pinned subspace-incidence system (H, X) is generic w.r.t. a property Q if and only if there exists a neighborhood $N(X)$ such that for all systems (H, X') with $X' \in N(X)$, (H, X') satisfies Q if and only if (H, X) satisfies Q .

Similarly, a framework (H, X, p) is generic w.r.t. a property Q if and only if there exists a neighborhood $N(p)$ such that for all frameworks (H, X, q) with $q \in N(p)$, (H, X, q) satisfies Q if and only if (H, X, p) satisfies Q .

Furthermore we can define generic properties in terms of the underlying weighted hypergraph.

Definition 4. A property Q of pinned subspace-incidence systems is generic (i.e., becomes a property of the underlying weighted hypergraph alone) if for any weighted hypergraph $H = (V, E, m)$, either all generic (w.r.t. Q) systems (H, X) satisfies Q , or all generic (w.r.t. Q) systems (H, X) do not satisfy Q .

Once an appropriate notion of genericity is defined, we can treat Q as a property of a hypergraph. The primary activity of the area of combinatorial rigidity is to give purely combinatorial characterizations of such generic properties Q . In the process of drawing such combinatorial characterizations, the notion of genericity may have to be further restricted by so-called pure conditions that are necessary for the combinatorial characterization to go through (we will see this in the proof of Theorem 1).

3.3 Linearization as Rigidity Matrix

Next we follow the approach taken by traditional combinatorial rigidity theory [2,9] to show that rigidity and independence (based on nonlinear polynomials) of pinned subspace-incidence systems are generically properties of the underlying weighted hypergraph H , and can furthermore be captured by linear conditions in an infinitesimal setting. Specifically, we give a lemma showing that rigidity of a pinned subspace-incidence system is equivalent to existence of a full rank *rigidity matrix*, obtained by taking the Jacobian of the algebraic system $(H, X)(p)$ at a regular point.

A *rigidity matrix* of a framework (H, X, p) is a matrix whose kernel is the infinitesimal motions (flexes) of (H, X, p) . A framework is *infinitesimally rigid* if the space of infinitesimal motions is trivial, i.e. the rigidity matrix has full rank. To define a rigidity matrix for a pinned subspace-incidence framework (H, X, p) , we take the Jacobian of the algebraic system $(H, X)(p) = 0$ by taking partial derivatives with respect to the coordinates of p_i 's. In the Jacobian, each vertex v_i has $d - 1$ corresponding columns, and each pin x_k associated with the hyperedge $e_k = \{v_1^k, v_2^k, \dots, v_{|e_k|}^k\}$ has $m_k(d - |e_k|)$ corresponding rows, where each equation $\det(E_l^k[\cdot, C(t)]) = 0$ (1), i.e. equation t of the atomic pin x_l^k , gives the following row (the columns corresponding to vertices not in e_k are all zero):

$$\begin{aligned}
 & \left[0, \dots, 0, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{1,1}^k}, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{1,2}^k}, \dots, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{1,d-1}^k}, 0, \dots \right. \\
 & \quad \dots, 0, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{2,1}^k}, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{2,2}^k}, \dots, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{2,d-1}^k}, 0, \dots \\
 & \quad \dots \dots \dots \\
 & \quad \left. \dots, 0, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{|e_k|,1}^k}, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{|e_k|,2}^k}, \dots, \frac{\partial \det E_l^k[\cdot, C(t)]}{\partial p_{|e_k|,d-1}^k}, 0, \dots, 0 \right] \tag{2}
 \end{aligned}$$

Let V^k be the matrix whose rows are coordinates of $p_1^k, p_2^k, \dots, p_{|e_k|}^k$:

$$\begin{bmatrix} p_{1,1} & p_{1,2} & \dots & p_{1,d-1} \\ p_{2,1} & p_{2,2} & \dots & p_{2,d-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{|e_k|,1} & p_{|e_k|,2} & \dots & p_{|e_k|,d-1} \end{bmatrix}$$

We use $M(H)$ or simply M to denote the generic rigidity matrix for a weighted hypergraph H . Note that the rank of M cannot be less than the rank of $M(p)$ for any specific realization p .

Lemma 1. *Generic infinitesimal rigidity of a pinned subspace-incidence framework (H, X, p) is equivalent to generic rigidity of the system (H, X) .*

Proof. The proof of Lemma 1 follows the approach taken by traditional combinatorial rigidity [2].

First we show that if a framework (H, X, p) is regular, infinitesimal rigidity implies rigidity. Consider the polynomial system $(H, X)(p)$ of equations. The Implicit Function Theorem states that there exists a function g , such that $p = g(X)$ on some open interval, if and only if the rigidity matrix M has full rank. Therefore, if the framework is infinitesimally rigid, the solutions to the algebraic system are isolated points (otherwise g could not be explicit). Since the algebraic system contains finitely many components, there are only finitely many such solution and each solution is a 0 dimensional point. This implies that the total number of solutions is finite, which is the definition of rigidity.

To show that generic rigidity implies generic infinitesimal rigidity, we take the contrapositive: if a generic framework is not infinitesimally rigid, we show that there is a finite flex. If (H, X, p) is not infinitesimally rigid, then the rank r of the rigidity matrix M is less than $(d - 1)|V|$. Let E^* be a set of edges in H such that $|E^*| = r$ and the corresponding rows in M are all independent. In $M[E^*, \cdot]$, we can find r independent columns. Let p^* be the components of p corresponding to those r independent columns and $p^{*\perp}$ be the remaining components. The r -by- r submatrix $M[E^*, p^*]$, made up of the corresponding independent rows and columns, is invertible. Then, by the Implicit Function Theorem, in a neighborhood of p there exists a continuous and differentiable function g such that $p^* = g(p^{*\perp})$. This identifies p' , whose components are p^* and the level set of g corresponding to p^* , such that $(H, X)(p') = 0$. The level set defines the finite flexing of the framework. Therefore the system is not rigid.

4 Combinatorial Rigidity Characterization

4.1 Required Hypergraph Properties

This section introduces pure hypergraph properties and definitions that will be used in stating and proving our main theorem.

Definition 6. *A hypergraph $H = (V, E)$ is $(k, 0)$ -sparse if for any $V' \subset V$, the induced subgraph $H' = (V', E')$ satisfies $|E'| \leq k|V'|$. A hypergraph H is $(k, 0)$ -tight if H is $(k, 0)$ -sparse and $|E| = k|V|$.*

This is a special case of the (k, l) -sparsity condition that was formally studied widely in the geometric constraint solving and combinatorial rigidity literature before it was given a name in [16]. A relevant concept from graph matroids is *map-graph*, defined as follows.

Definition 7. An orientation of a hypergraph is given by identifying as the tail of each edge one of its endpoints. The out-degree of a vertex is the number of edges which identify it as the tail and connect v to $V - v$. A map-graph is a hypergraph that admits an orientation such that the out degree of every vertex is exactly one.

The following lemma from [21] follows Tutte-Nash Williams [18,23] to give a useful characterization of $(k, 0)$ -tight graphs in terms of maps.

Lemma 2. A hypergraph H is composed of k edge-disjoint map-graphs if and only if H is $(k, 0)$ -tight.

Our characterization of rigidity of a weighted hypergraph H is based on map-decomposition of a multi-hypergraph \widehat{H} obtained from H .

Definition 8. Given a weighted hypergraph $H = (V, E, m)$, the associated multi-hypergraph $\widehat{H} = (V, \widehat{E})$ is obtained by replacing each hyperedge e_k in E with a set E^k of $m_k(d - |e_k|)$ copies of multi-hyperedges.

A labeling of a multi-hypergraph \widehat{H} gives a one-to-one correspondence between E^k and the set R^k of $m_k(d - |e_k|)$ rows for the hyperedge e_k in the rigidity matrix M , where the multi-hyperedge corresponding to the row $r_{t,l}^k$ is labeled $e_{t,l}^k$.

Note: an alternative representation commonly adopted in geometric constraint solving [1,7,11] is to represent H as a bipartite graph $B(H)$, with $d - 1$ copies of each vertex in V as one of its vertex set, and $m_k(d - |e_k|)$ copies of each hyperedge in E as the other vertex set. A combinatorial rigidity characterization can be equivalently stated using either flow based conditions on $B(H)$ or hypergraph sparsity conditions on \widehat{H} [12,13,15]. However, such an equivalence of the two combinatorial properties has no bearing on the proof of equivalence of either combinatorial property to generic rigidity, an algebraic property. Showing that the combinatorial property generically implies the algebraic property is the substance of the proof of the theorem. This is generally called the ‘‘Laman direction’’ and it is in fact where the hardness of every combinatorial characterization of rigidity lies.

4.2 Characterizing Rigidity

In this section, we apply [24] to give combinatorial characterization for minimal rigidity of pinned subspace-incidence systems.

Theorem 1 (main theorem). A pinned subspace-incidence system is generically minimally rigid if and only if:

- (1) The underlying weighted hypergraph $H = (V, E, m)$ satisfies $\sum_{k=1}^{|E|} m_k(d - |e_k|) = (d - 1)|V|$, and $\sum_{e_k \in E'} m_k(d - |e_k|) \leq (d - 1)|V'|$ for every vertex induced subgraph $H' = (V', E')$. In other words, the associated multi-hypergraph $\widehat{H} = (V, \widehat{E})$ has a decomposition into $(d - 1)$ maps.

- (2) *There exists a labeling of \widehat{H} compatible with the map-decomposition (defined later) such that in each set E^k of multi-hyperedges,*
 (2a) *two multi-hyperedges e_{t_1, l_1}^k and e_{t_2, l_2}^k with $l_1 = l_2$ are not contained in the same map in the map-decomposition,*
 (2b) *two multi-hyperedges e_{t_1, l_1}^k and e_{t_2, l_2}^k with $t_1 = t_2$ do not have the same vertex as tail in the map-decomposition.*

To prove Theorem 1, we apply Laplace expansion to the determinant of the rigidity matrix M , which corresponds to decomposing the $(d - 1, 0)$ -tight multi-hypergraph \widehat{H} as a union of $d - 1$ maps. We then prove $\det(M)$ is not identically zero by showing that the minors corresponding to each map are not identically zero, as long as a certain polynomial called *pure condition* is avoided by the framework. The pure condition characterizes the non-genericity that the framework has to avoid in order for the combinatorial characterization to go through: see Example 4.

A Laplace expansion rewrites the determinant of the rigidity matrix M as a sum of products of determinants (brackets) representing each of the coordinates taken separately. In order to see the relationship between the Laplace expansion and the map-decomposition, we first group the columns of M into $d - 1$ column groups C_j according to the coordinates, where columns for the first coordinate of each vertex belong to C_1 , columns for the second coordinate of each vertex belong to C_2 , etc.

Example 2 For $d = 4$, consider a pin x with $m(x) = 2$ associated with the hyperedge $e = v_1, v_2$. The regrouped rigidity matrix has $d - 1 = 3$ column groups, where the pin x has the following 4 rows (the index k is omitted):

$$\begin{array}{l}
 t = 1, l = 1 \\
 t = 1, l = 2 \\
 t = 2, l = 1 \\
 t = 2, l = 2
 \end{array}
 \left[\begin{array}{cccc|cccc}
 & v_{1,1} & & v_{2,1} & & v_{1,2} & v_{2,2} & & v_{1,3} & v_{2,3} \\
 \dots & D_{1,1}b_1^1 & \dots & D_{1,1}b_2^1 & \dots & \dots & Db_1^1 & \dots & Db_2^1 & \dots \\
 \dots & D_{1,1}b_1^2 & \dots & D_{1,1}b_2^2 & \dots & \dots & Db_1^2 & \dots & Db_2^2 & \dots \\
 \dots & D_{2,1}b_1^1 & \dots & D_{2,1}b_2^1 & \dots & \dots & \dots & \dots & Db_1^1 & \dots & Db_2^1 & \dots \\
 \dots & D_{2,1}b_1^2 & \dots & D_{2,1}b_2^2 & \dots & \dots & \dots & \dots & Db_1^2 & \dots & Db_2^2 & \dots
 \end{array} \right]$$

We have the following observation on the pattern of the regrouped rigidity matrix.

Observation 1. *In the rigidity matrix M with columns grouped into column groups, a hyperedge e_k has $m_k(d - |e_k|)$ rows, each associated with a multi-hyperedge of e_k in \widehat{H} . In a column group j where $j \leq |e_k| - 1$, each row associated with e_k contains $|e_k|$ nonzero entries at the columns corresponding to vertices of e_k . In a column group j where $j = |e_k| - 1 + t \geq |e_k|$, a row associated with a multi-hyperedge $e_{r,l}^k$ of e_k is all zero if $r \neq t$; the remaining m_k rows contains $|e_k|$ nonzero entries at the columns corresponding to vertices of e_k .*

A labeling of \widehat{H} compatible with a given map-decomposition can be obtained as following. We start from the last column group of M and associate each column group j with a map in the map-decomposition. For each multi-hyperedge

of the map that is a copy of the hyperedge e_k , we pick a row $r_{t,j}^k$ that is not all zero in column groups j and label the multi-hyperedge as $e_{t,j}^k$. By the above observation, this is always possible if each map contains at most m_k multi-hyperedges of the same hyperedge e_k , which must be true if there exists any labeling of \widehat{H} satisfying Theorem 1(2a).

In the Laplace expansion

$$\det(M) = \sum_{\sigma} \left(\pm \prod_j \det M[R_j^{\sigma}, C_j] \right) \tag{4}$$

the sum is taken over all partitions σ of the rows into $d - 1$ subsets $R_1^{\sigma}, R_2^{\sigma}, \dots, R_j^{\sigma}, \dots, R_{d-1}^{\sigma}$, each of size $|V|$. In other words, each summation term of (4) contains $|V|$ rows R_j^{σ} from each column group C_j . Observe that for any submatrix $M[R_j^{\sigma}, C_j]$, each row has a common coefficient $D_{t,j}^k$, so

$$\det(M[R_j^{\sigma}, C_j]) = \left(\prod_{r_{t,j}^k \in R_j^{\sigma}} D_{t,j}^k \right) \det(M'[R_j^{\sigma}, C_j])$$

where each row of $M'[R_j^{\sigma}, C_j]$ is either all zero, or of the pattern

$$[0, \dots, 0, b_1^{k,l}, b_2^{k,l}, 0, \dots, b_{|e_k|}^{k,l}, 0, \dots, 0] \tag{5}$$

with non-zero entries only at the $|e_k|$ indices corresponding to $v_i^k \in e_k$.

For a fixed σ , we refer to a submatrix $M[R_j^{\sigma}, C_j]$ simply as M_j .

Example 3. Figure 2a shows a pinned subspace-incidence system in $d = 3$ with 4 vertices and 5 hyperedges, where $e_1 = \{v_1\}, e_2 = \{v_2\}, e_3 = \{v_1, v_3\}, e_4 = \{v_2, v_4\}, e_5 = \{v_3, v_4\}$, and $m_k = 1$ for all $1 \leq k \leq 5$ except that $m_5 = 2$. Figure 2b gives a map-decomposition of the multi-hypergraph \widehat{H} of (a). The labeling of multi-hyperedges is given in the regrouped rigidity matrix (6), where the shaded rows inside the column groups constitute the submatrices M_j^{σ} in the summation term of the Laplace decomposition corresponding to the map-decomposition. The system is generically minimally rigid, as the map-decomposition and labeling of \widehat{H} satisfies the conditions of Theorem 1.

	$v_{1,1}$	$v_{2,1}$	$v_{3,1}$	$v_{4,1}$	$v_{1,2}$	$v_{2,2}$	$v_{3,2}$	$v_{4,2}$
$e_{1,1}^1$	$D^1 b^1$							
$e_{2,1}^1$					$D^1 b^1$			
$e_{1,1}^2$	$D^2 b^2$							
$e_{2,1}^2$					$D^2 b^2$			
$e_{1,1}^3$	$D_1^3 b_1^3$		$D_1^3 b_2^3$		$D^3 b_1^3$	$D^3 b_2^3$		
$e_{1,1}^4$	$D_1^4 b_1^4$		$D_1^4 b_2^4$		$D^4 b_1^4$		$D^4 b_2^4$	
$e_{1,1}^5$	$D_1^{5,1} b_1^{5,1}$ $D_1^{5,1} b_2^{5,1}$				$D_2^{5,1} b_1^{5,1}$		$D_2^{5,1} b_2^{5,1}$	
$e_{1,2}^5$	$D_1^{5,2} b_1^{5,2}$ $D_1^{5,2} b_2^{5,2}$				$D_2^{5,2} b_1^{5,2}$ $D_2^{5,2} b_2^{5,2}$			

(6)

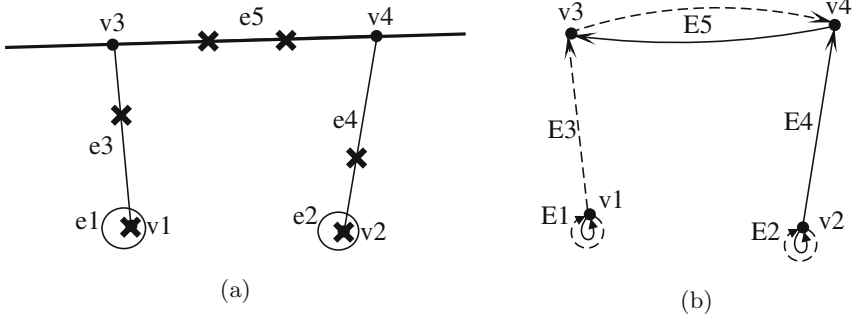


Fig. 2. (a) A minimally rigid pinned subspace-incidence system in $d = 3$. (b) A map-decomposition of the multi-hypergraph of the system in (a), where multi-hyperedges with different patterns are in different maps, and the tail vertex of each multi-hyperedge is pointed to by an arrow.

Proof (main theorem). First we show the only if direction. For a generically minimally rigid pinned subspace-incidence framework, the rigidity matrix M is generically full rank, so there exists at least one summation term σ in (4) where each submatrix M_j is generically full rank. As the submatrices don't have any overlapping rows with each other, we can perform row elimination on M to obtain a matrix N with the same rank, where all submatrices M_j are simultaneously converted to a *permuted reduced row echelon form* N_j , where each row in N_j has exactly one non-zero entry β_i^j at a unique column i . In other words, all N_j 's can be converted simultaneously to reduced row echelon form by multiplying a permutation matrix on the left of N . Now we can obtain a map-decomposition of \widehat{H} by letting each map j contain multi-hyperedges corresponding to rows of the submatrix N_j , and assigning each multi-hyperedge in map j the vertex i corresponding to the non-zero entry β_i^j in the associated row in N_j as tail. In addition, such a map-decomposition must satisfy the conditions of Theorem 1(2):

Condition (2a): assume two multi-hyperedges $e_{t_1,l}^k$ and $e_{t_2,l}^k$ are in the same map j , i.e. the rows corresponding to these two edges are included in the same submatrix M_j . If $j > s - 1$, one of these rows must be all-zero in M_j by Observation 1, contradicting the condition that M_j is full rank. If $j \leq s - 1$, both of these rows in M_j will be a multiple of the same row vector (5), contradicting the condition that M_j is full rank.

Condition (2b): note that the rows in M corresponding to multi-hyperedges e_{t,l_1}^k and e_{t,l_2}^k have the exactly the same pattern except for different values of b 's. If e_{t,l_1}^k has vertex i as tail, after the row elimination, the column containing b_i^{k,l_1} will become the only non-zero entry of column i for N_{j_1} , while the column containing b_i^{k,l_2} will become zero in all column groups, thus i cannot be assigned as tail for e_{t,l_2}^k .

Next we show the if direction, that the conditions of Theorem 1 imply infinitesimal rigidity.

Given labeled multi-hypergraph \widehat{H} with a map-decomposition satisfying the conditions of Theorem 1, we can obtain summation term σ in the Laplace decomposition (4) according to the labeling of \widehat{H} , where each submatrix M_j contain all rows corresponding to the map associated with column group j .

We first show that each submatrix M_j is generically full rank. According to the definition of a map-graph, the function $\tau : \widehat{E} \rightarrow V$ assigning a tail vertex to each multi-hyperedge is a one-to-one correspondence. We perform symbolic row elimination of the matrix M to simultaneously convert each M_j to its permuted reduced row echelon form N_j , where for each row of N_j , all entries are zero except for the entry $\beta_{t,l}^k$ corresponding to the vertex $\tau(e_{t,l}^k)$, which is a polynomial in $b_i^{k,l}$'s in the submatrix M_j . Since M_j cannot contain two rows with the same k and l by (2a), the $b_i^{k,l}$'s in different rows of a same map are independent of each other, $\beta_{t,l}^k \neq 0$ under a generic specialization of $b_i^{k,l}$. Since each row of N_j has exactly one nonzero entry and the nonzero entries from different rows are on different columns, the $|V| \times |V|$ matrix N_j is clearly full rank. Thus M_j must also be generically full rank.

We conclude that

$$\det(M) = \sum_{\sigma} \left(\pm \prod_j \left(\left(\prod_{r_{t,j}^k \in R_j^{\sigma}} D_{t,j}^k \right) \det M'[R_j^{\sigma}, C_j] \right) \right) \tag{7}$$

where the sum is taken over all σ corresponding to a map-decomposition of \widehat{H} . Generically, the summation terms of the sum (7) do not cancel with each other, since $\det(M'[R_j^{\sigma}, C_j])$ are independent of the multi-linear coefficients $\prod_{r_{t,j}^k \in R_j^{\sigma}} D_{t,j}^k$, and any two rows of M are independent by (2b). This implies that \widehat{M} is generically full rank.

The polynomial (7) gives the pure condition for genericity. In particular, when there is a subgraph (V', E') with $|V'| < d$ and $\sum_{e_k \in E'} m_k > |V'|$, the pure condition vanishes and the system won't be minimally rigid: see Example 4.

Pure Condition. The pure condition (7) obtained in the proof of Theorem 1 characterizes the badly behaved cases that break the combinatorial characterization of infinitesimal rigidity. However, the geometric meaning of the pure condition is not completely clear. One particular condition not captured by Theorem 1 but enforced by the pure condition is that there cannot exists a subgraph (V', E') of H with $|V'| < d$ such that $\sum_{e_k \in E'} m_k > |V'|$, otherwise simple counterexamples can be constructed to the characterization of the main theorem. An immediately consequence is that for any hyperedge e_k , the dimension m_k of its associated pin must be less than or equal to its cardinality $|e_k|$.

Example 4. Figure 3a shows a pinned subspace-incidence system in $d = 4$ with 4 vertices and 5 hyperedges, where m_k is 2 for $k = 5$ and is 1 otherwise. A map-decomposition of the multi-hypergraph \widehat{H} of the system is given in Fig. 3b, and we can easily find a labeling of \widehat{H} satisfying conditions in Theorem 1. However,

the system is not minimally rigid, as generically the pin x_1 will not fall on the plane spanned by pins x_4 and x_5 . Note that the sub-hypergraph (V', E') spanned by vertices v_1, v_2, v_4 violates the pure condition as $\sum_{e_k \in E'} m_k = m_1 + m_4 + m_5 = 4 > |V'| = 3$.

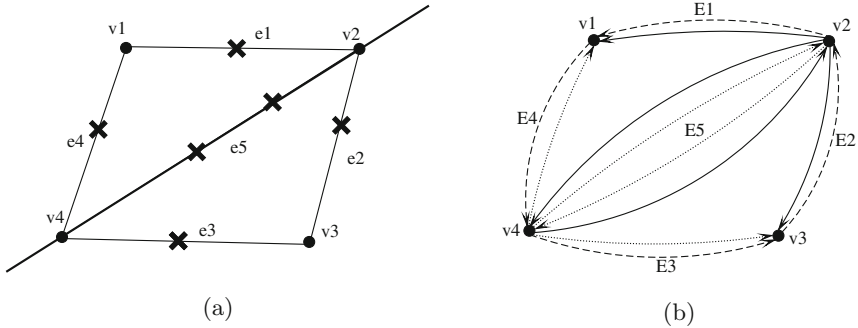


Fig. 3. (a) A pinned subspace-incidence system in $d = 4$. (b) A map-decomposition of the multi-hypergraph of the system in (a), where multi-hyperedges with different patterns are in different maps, and the tail vertex of each multi-hyperedge is pointed to by an arrow.

5 Conclusion

In this paper, we studied the rigidity of pinned subspace-incidence systems. We extend our results in [19] and obtain a combinatorial characterization of minimal rigidity for general pinned subspace-incidence systems with non-uniform underlying hypergraphs and pins being subspaces with arbitrary dimensions.

As future work, we plan to extend the underlying group of the pinned subspace-constraint system, i.e. consider two frameworks to be congruent if the point realization and pin set of one can be obtained from the other under the action of a certain group, for example the projective group. Another possible direction is to apply Cayley factorization [6] to find geometric interpretations of some of the pure conditions.

References

1. Ait-Aoudia, S., Jegou, R., Michelucci, D.: Reduction of constraint systems. In: Proceedings of Compugraphics, pp. 83–92 (1993)
2. Asimow, L., Roth, B.: The rigidity of graphs. *Trans. Am. Math. Soc.* **245**, 279–289 (1978)
3. Baker, T., Sitharam, M., Wang, M., Willoughby, J.: Designing qusecs optimally: Quasi-uniform or self-similar layered materials as recursively decomposed solutions of geometric constraint systems, accepted to *Computer Aided Geometric Design* (2015)

4. Buehler, M.J.: Nanomechanics of collagen fibrils under varying cross-link densities: atomistic and continuum studies. *J. Mech. Behav. Biomed. Mater.* **1**(1), 59–67 (2008)
5. Fall, A.B., Lindström, S.B., Sprakel, J., Wågberg, L.: A physical cross-linking process of cellulose nanofibril gels with shear-controlled fibril orientation. *Soft Matter* **9**(6), 1852–1863 (2013)
6. Farre, J., John, A.L.S., Sidman, J., Theran, L.: Special positions of body-and-cad frameworks. CoRR abs/1306.1572 (2013). <http://arxiv.org/abs/1306.1572>
7. Fudos, I., Hoffmann, C.M.: A graph-constructive approach to solving systems of geometric constraints. *ACM Trans. Graph.* **16**(2), 179–216 (1997). <http://doi.acm.org/10.1145/248210.248223>
8. Gortler, S.J., Gotsman, C., Liu, L., Thurston, D.P.: On affine rigidity. *J. Comput. Geom.* **4**(1), 160–181 (2013)
9. Graver, J.E., Servatius, B., Servatius, H.: *Combinatorial Rigidity*, vol. 2. AMS Bookstore, Providence (1993)
10. Haller, K., John, A.L.S., Sitharam, M., Streinu, I., White, N.: Body-and-cad geometric constraint systems. *Comput. Geom.* **45**(8), 385–405 (2012)
11. Hoffmann, C.M., Joan-Arinyo, R.: Symbolic constraints in constructive geometric constraint solving. *J. Symbolic Comput.* **23**(2), 287–299 (1997)
12. Hoffmann, C.M., Lomonosov, A., Sitharam, M.: Finding solvable subsets of constraint graphs. In: Smolka, G. (ed.) CP97. LNCS, vol. 1330, pp. 463–477. Springer, Heidelberg (1997)
13. Hoffmann, C.M., Lomonosov, A., Sitharam, M.: Geometric constraint decomposition. In: Brüderlin, B., Roller, D. (eds.) *Geometric Constraint Solving and Applications*, pp. 170–195. Springer, Heidelberg (1998)
14. Jackson, B., Jordán, T.: Pin-collinear body-and-pin frameworks and the molecular conjecture. *Discrete Comput. Geom.* **40**(2), 258–278 (2008)
15. Jacobs, D.J., Hendrickson, B.: An algorithm for two-dimensional rigidity percolation: the pebble game. *J. Comput. Phys.* **137**(2), 346–365 (1997)
16. Lee, A., Streinu, I., Theran, L.: Graded sparse graphs and matroids. *J. UCS* **13**(11), 1671–1679 (2007)
17. Mittmann, J.: *Gröbner bases: Computational algebraic geometry and its complexity* (2007)
18. Nash-Williams, C.S.J.: Edge-disjoint spanning trees of finite graphs. *J. London Math. Soc.* **1**(1), 445–450 (1961)
19. Sitharam, M., Tarifi, M., Wang, M.: An incidence geometry approach to dictionary learning. In: *Proceedings of CCCG*, pp. 410–430 (2014). <http://www.cccg.ca/proceedings/2014/>
20. Smith, G.: Plant cell wall structure and cell wall growth. *Tuatara* **19**, 43–50 (1971)
21. Streinu, I., Theran, L.: Sparse hypergraphs and pebble game algorithms. *Eur. J. Comb.* **30**(8), 1944–1964 (2009)
22. Streinu, I., Theran, L.: Slider-pinning rigidity: a maxwell-laman-type theorem. *Discrete Comput. Geom.* **44**(4), 812–837 (2010)
23. Tutte, W.T.: On the problem of decomposing a graph into n connected factors. *J. London Math. Soc.* **1**(1), 221–230 (1961)
24. White, N., Whiteley, W.: The algebraic geometry of motions of bar-and-body frameworks. *SIAM J. Algebraic Discrete Methods* **8**(1), 1–32 (1987)
25. White, N.L., Whiteley, W.: The algebraic geometry of stresses in frameworks. *SIAM J. Algebraic Discrete Methods* **4**(4), 481–511 (1983)
26. Whiteley, W.: Some matroids from discrete applied geometry. *Contemp. Math.* **197**, 171–312 (1996)

Author Index

- Alam, Md. Ashraful 1
- Baeta, Nuno 119
- Borcea, Ciprian S. 21
- Chen, Xiaoyu 149
- Davenport, James H. 37
- England, Matthew 37
- Janičić, Predrag 72
- Kovács, Zoltán 53
- Marinković, Vesna 72
- Moritsugu, Shuichi 94
- Pech, Pavel 108
- Quaresma, Pedro 119
- Schreck, Pascal 72
- Sitharam, Meera 129, 166
- Song, Dan 149
- Streinu, Ileana 1, 21
- Wang, Dongming 149
- Wang, Menghan 166
- Willoughby, Joel 129