Combinatorial Resultants in the Algebraic Rigidity Matroid

Goran Malić
Computer Science Department, Smith College, Northampton, MA, USA

Ileana Streinu[1]
Computer Science Department, Smith College, Northampton, MA, USA

Abstract

Motivated by a rigidity-theoretic perspective on the *Localization Problem* in 2D, we develop an algorithm for computing *circuit polynomials* in the algebraic rigidity matroid CM_n associated to the Cayley-Menger ideal for n points in 2D. We introduce *combinatorial resultants*, a new operation on graphs that captures properties of the Sylvester resultant of two polynomials in the algebraic rigidity matroid. We show that every rigidity circuit has a *construction tree* from K_4 graphs based on this operation. Our algorithm performs an *algebraic elimination* guided by the construction tree, and uses classical resultants, factorization and ideal membership. To demonstrate its effectiveness, we implemented our algorithm in Mathematica: it took less than 15 seconds on an example where a Gröbner Basis calculation took 5 days and 6 hrs.

2012 ACM Subject Classification General and reference → Performance; General and reference → Experimentation; Theory of computation → Computational geometry; Mathematics of computing → Matroids and greedoids; Mathematics of computing → Mathematical software performance; Computing methodologies → Combinatorial algorithms; Computing methodologies → Algebraic algorithms

Keywords and phrases Cayley-Menger ideal, rigidity matroid, circuit polynomial, combinatorial resultant, inductive construction, Gröbner basis elimination

Digital Object Identifier 10.4230/LIPIcs.SoCG.2021.52

Related Version *Full Version*: `https://arxiv.org/abs/2103.08432` [21]

Supplementary Material Polynomials computed by our algorithm are made available in Mathematica's compressed and portable wdx format at the following GitHub repository:
Dataset: `https://github.com/circuitPolys/CayleyMenger`
archived at `swh:1:dir:43ae808290f56bde92a4139cf1b72f4f2f57adf8`

Funding Both authors acknowledge funding from the NSF CCF:1703765 grant to Ileana Streinu.

1 Introduction

This paper addresses combinatorial, algebraic and algorithmic aspects of a question motivated by the following ubiquitous problem from *distance geometry*:

Localization. A graph together with *weights* associated to its edges is given. The goal is to find *placements* of the graph in some Euclidean space, so that the edge lengths match the given weights. In this paper, we work in 2D. A system of quadratic equations can be easily set up so that the possible placements are among the (real) solutions of this system. Rigidity Theory can help predict, *a priori,* whether the set of solutions will be discrete (if the

[1] Corresponding author

given weighted graph is *rigid*) or continuous (if the graph is *flexible*). In the rigid case, the double-exponential Gröbner basis algorithm can be used, *in principle,* to eliminate all but one of the variables. Once a polynomial in a single variable is obtained, numerical methods are used to solve it. We then select one solution, substitute it in the original equations, eliminate to get a polynomial in a new variable and repeat.

Single unknown distance problem. Instead of attempting to directly compute the coordinates of all the vertices, we restrict our attention to the related problem of finding the possible values of a *single unknown distance* corresponding to a *non-edge* (a pair of vertices that are not connected by an edge). Indeed, *if* we could solve this problem for a collection of non-edge pairs that form a trilateration when added to the original collection of edges, *then* a single solution in Cartesian coordinates could be easily computed afterwards in linearly many steps of quadratic equation solving.

Rigidity circuits. We formulate the *single unknown distance* problem in terms of Cayley-coordinates (squared distances between points) rather than Cartesian xy-coordinates. Known theorems from Distance Geometry, Rigidity Theory and Matroid Theory help reduce this problem to finding a certain irreducible polynomial in the Cayley-Menger ideal, called the *circuit polynomial*. Its support is a graph called a *circuit in the rigidity matroid*, or shortly a *rigidity circuit.* Substituting given edge lengths in the circuit polynomial results in a uni-variate polynomial which can be solved for the unknown distance.

The focus of this paper is the following:

▶ **Main Problem.** Given a rigidity circuit, compute its corresponding circuit polynomial.

Related work. While both *distance geometry* and *rigidity theory* have a distinguished history for which a comprehensive overview would be too long to include here, very little is known about computing *circuit polynomials. To the best of our knowledge,* their study in *arbitrary* polynomial ideals was initiated in the PhD thesis of Rosen [23]. His Macaulay2 code [24] is useful for exploring small cases, but the Cayley-Menger ideal is beyond its reach. A recent article [25] popularizes algebraic matroids and uses for illustration the smallest circuit polynomial K_4 in the Cayley-Menger ideal. *We could not find non-trivial examples anywhere.* Indirectly related to our problem are results such as [28], where an explicit univariate polynomial of degree 8 is computed (for an unknown angle in a $K_{3,3}$ configuration given by edge lengths, from which the placement of the vertices is determined) and [26], for its usage of Cayley coordinates in the study of configuration spaces of some families of distance graphs. A closely related problem is that of computing the *number of embeddings of a minimally rigid graph* [4], which has received *a lot* of attention in recent years (e.g. [6, 1, 10, 9], to name a few). References to specific results in the literature that are relevant to the theory developed here and to our proofs are given throughout the paper.

How tractable is the problem? Circuit polynomial computations can be done, in principle, with the double-exponential time Gröbner basis algorithm with an elimination order. The largest we could do with the **GroebnerBasis** function of Mathematica 12 (running on a 2019 iMac computer with 6 cores at 3.7Ghz and 16GB RAM) was the Desargues-plus-one (Fig. 1) circuit ($658,175$ terms), which took 5 days and 6 hours. In all other cases the execution timed out or crashed. Our goal is to make such calculations *more tractable* by taking advantage of *structural information* inherent in the problem.

Our results. We describe a new *algorithm to compute a circuit polynomial with known support.* It relies on resultant-based elimination steps guided by a novel *inductive construction for rigidity circuits.* While inductive constructions have been often used in Rigidity Theory, most notably the Henneberg sequences for Laman graphs [14] and Henneberg II sequences for rigidity circuits [2], we argue that our construction is more *natural* due to its *direct algebraic interpretation.* We have implemented our method in Mathematica and applied it successfully to compute all circuit polynomials on up to 6 vertices and a few on 7 vertices, the largest of which having over two million terms. The previously mentioned example that took over 5 days to complete with GroebnerBasis, was solved by our algorithm in less than 15 seconds.

Main theorems. We first define the *combinatorial resultant* of two graphs as an abstraction of the classical resultant. Our main theoretical result is split into the combinatorial Theorem 1 and the algebraic Theorem 2, each with an algorithmic counterpart.

▶ **Theorem 1.** *Each rigidity circuit can be obtained, inductively, by applying combinatorial resultant operations starting from K_4 circuits. The construction is captured by a binary* resultant tree *whose nodes are intermediate rigidity circuits and whose leaves are K_4 graphs.*

Theorem 1 leads to a *graph algorithm* for finding a *combinatorial resultant tree* of a circuit. Each step of the construction can be carried out in polynomial time using variations on the *Pebble Game* matroidal sparsity algorithms [18] combined with Hopcroft and Tarjan's linear time 3-connectivity algorithm [15]. However, it is conceivable that the resultant tree could be exponentially large with non-repeating subtrees, and thus the entire construction could take an exponential number of steps: understanding in detail the algorithmic complexity of our method *remains a problem for further investigation.*

▶ **Theorem 2.** *Each circuit polynomial can be obtained, inductively, from K_4 circuit polynomials by applying resultant operations in a manner guided by the combinatorial resultant tree from Theorem 1. At each step, the resultant produces a polynomial that may not be irreducible. A polynomial factorization and a test of membership in the ideal may need to be applied to identify the factor which is the circuit polynomial.*

The resulting *algebraic elimination algorithm* runs in exponential time, in part because of the growth in size of the polynomials that are being produced. Several interesting theoretical *open questions* remain, whose answers may affect the precise time complexity analysis.

Computational experiments. We implemented our algorithms in Mathematica V12.1.1.0 on two computers with the following specifications: Intel i5-9300H 2.4GHz, 32 GB RAM, Windows 10 64-bit; and Intel i5-9600K 3.7GHz, 16 GB RAM, macOS Mojave 10.14.5. We also explored Macaulay2, but it was much slower than Mathematica (hours vs. seconds) in computing one of our examples. The polynomials resulting from our calculations, summarized in Table 1 at the end of the paper, are made available on a github repository [19].

Overview of the paper. In Section 2 we introduce the background concepts from matroid theory and rigidity theory. We introduce combinatorial resultants in Section 3, prove Theorem 1 and describe the algorithm for computing a combinatorial resultant tree. In Section 4 we introduce the background concepts pertaining to algebraic matroids in the Cayley-Menger ideal and elimination theory. In Section 5 we prove Theorem 2. We conclude in Section 6 with a summary of the preliminary experimental results we carried with our implementation. For complete details and a self-contained presentation, including relevant

definitions and results from Rigidity Theory, Matroid Theory, Ideals, the Cayley-Menger ideal, Resultants and Elimination theory, complete proofs and further details on our results, the reader should refer to the full version of the paper, currently available on the arxiv [21].

2 Preliminaries: circuits in the rigidity matroid

We start with the combinatorial aspects of our problem. In this section we review well-known concepts and results from combinatorial rigidity theory that are relevant for our paper.

Notation. We work with (sub)graphs given by subsets E of edges of the complete graph K_n on vertices $[n] := \{1, \cdots, n\}$. If G is a (sub)graph, then $V(G)$, resp. $E(G)$ denote its vertex, resp. edge set. The support of G is $E(G)$. The *vertex span* $V(E)$ of edges E is the set of all edge-endpoint vertices. A subgraph G is *spanning* if its edge set $E(G)$ spans $[n]$. The *neighbours* $N(v)$ of v are the vertices adjacent to v in G.

Rigid graphs. Let $G = (V, E)$ be an undirected graph and let $\ell = \{\ell_{ij} | ij \in E\}$ be a collection of numbers interpreted as *lengths* associated to its edges. Up to rigid transformations, *in how many ways can we place the vertices of G at points in the plane, so that the segments corresponding to edges have the prescribed lengths?* The answer can be a finite number or a continuum of possibilities, and it depends in principle on both G and ℓ. For all but a measure zero set of possible lengths (said to be *generic*[2]), the answer depends only on G. We say that G is *rigid* if, for generic edge lengths, the number of placements is finite, and *flexible* otherwise. G is said to be *minimally rigid* if it is rigid, but it becomes flexible when any of its edges is removed.

Laman graphs. Minimally rigid graphs are characterized by Laman's Theorem [17]: a graph $G = (V, E)$ is minimally rigid in 2D iff (a) G is $(2, 3)$-sparse (no subset of $n' \leq n = |V|$ vertices spans more than $2n' - 3$ edges) and (b) G is $(2, 3)$-tight (has a total of $2n - 3$ edges). Graphs with these two properties will be called *Laman graphs*. A *Laman-plus-one graph* is obtained by adding one edge to a Laman graph: it has a total of $2n - 2$ edges and thus it violates the $(2, 3)$-sparsity condition on V and possibly on some other proper subsets of V.

Matroids. A matroid is an abstraction capturing (in)dependence relations among collections of elements from a *ground set*, and is inspired by both *linear* dependencies (among, say, rows of a matrix) and by *algebraic* constraints imposed by algebraic equations on a collection of otherwise free ("independent") variables. The standard way to specify a matroid is via its *independent sets*, which have to satisfy certain axioms [22] (skipped here, since they are not relevant for our presentation). A *base* is a maximal independent set and a *dependent* set is one which is not independent. A *minimal dependent set* is called a *circuit*. Relevant for our purposes are the following general aspects: (a) (hereditary property) a subset of an independent set is also independent; (b) all bases have the same cardinality, called the *rank* of the matroid. Further properties will be introduced in context, as needed. In this paper we encounter two types of matroids: a *graphic matroid*, defined on a ground set given by all the edges $E_n := \{ij : 1 \leq i < j \leq n\}$ of the complete graph K_n; this is the $(2, 3)$-*sparsity*

[2] Many results in Rigidity Theory are often using the very strong assumption that the edge lengths should be *algebraically independent* in order to be *generic*; this assumption simplifies the proofs but is not necessary and would preclude any effective algorithmic application of generic graphs.

matroid or the *generic rigidity matroid* described below; and an *algebraic matroid*, defined on an isomorphic ground set of variables $X_n := \{x_{ij} : 1 \leq i < j \leq n\}$; this is the *algebraic matroid for the Cayley-Menger ideal* and will be introduced in Section 4. These matroids are isomorphic (for a comprehensive proof, see [21]).

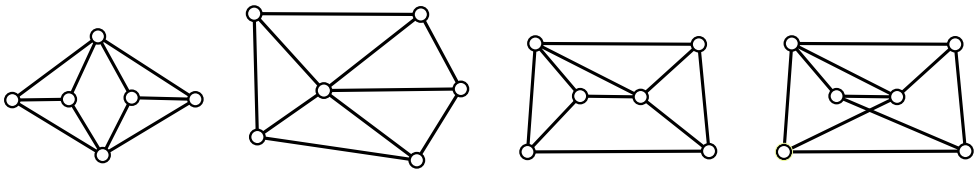

Figure 1 The four types of circuits on $n = 6$ vertices: 2D *double-banana*, 5-*wheel* W_5, *Desargues-plus-one* and $K_{3,3}$-plus-one.

Circuits in the Rigidity Matroid. Laman graphs form the *bases* of the so-called (generic) *2D rigidity matroid*. The *independent sets* are the $(2, 3)$-sparse graphs; the *dependent* sets violate the $(2, 3)$-sparsity condition on at least one subset of vertices. A *(rigidity) circuit* is a dependent graph with minimum edge support: removing any of its edges leads to a Laman graph on its spanned vertex set. A *spanning rigidity circuit* has V as its vertex set (Fig. 1). A *Laman-plus-one* graph contains a unique circuit. A spanning rigidity circuit $C = (V, E)$ is a special case of a Laman-plus-one graph: it has a total of $2n - 2$ edges and it satisfies the $(2, 3)$-sparsity condition on all proper subsets of $n' \leq n - 1$ vertices.

Operations on circuits. If G_1 and G_2 are two graphs, we use a consistent notation for their number of vertices and edges $n_i = |V(G_i)|$, $m_i = |E(G_i)|$, $i = 1, 2$, and for their union and intersection of vertices and edges, as in $V_\cup = V(G_1) \cup V(G_2)$, $V_\cap = V(G_1) \cap V(G_2)$, $n_\cup = |V_\cup|$, $n_\cap = |V_\cap|$ and similarly for edges, with $m_\cup = |E_\cup|$ and $m_\cap = |E_\cap|$. Let C_1 and C_2 be two circuits with exactly one common edge uv. Their 2-*sum* $C := C_1 \otimes C_2$ is the graph $C = (V_\cup, E_\cup \setminus \{uv\})$. The inverse operation of splitting C into C_1 and C_2 is called a 2-split [3] (Fig. 2). It is easy to see that the 2-sum of two circuits is a circuit, and that any 2-split of a circuit gives a pair of circuits ([2], Lemmas 4.1, 4.2 or use sparsity).

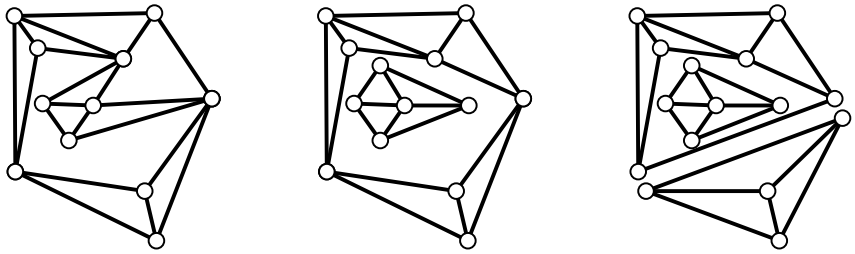

Figure 2 Splitting twice a 2-connected circuit (left) to get three 3-connected circuits (right).

[3] The 2-sum operation assumes that the two vertices of the summands coincide, and the 2-split operation produces two graphs, shown as disjoint in Fig. 2, for clarity.

Connectivity. A circuit is always a 2-connected graph (this follows easily from sparsity), and for simplicity we refer to one that is *not* 3-*connected* as a 2-*connected circuit*. The Tutte decomposition [27] of a graph into 3-connected components, applied on a circuit, induces 2-splits and produces smaller circuits: any 2-connected circuit can be constructed from smaller 3-connected circuits via 2-sums (Fig. 2).

Inductive constructions for 3-connected circuits. A *Henneberg II* extension (also called an *edge splitting* operation) is defined for an edge uv and a non-incident vertex w, as follows: the edge uv is removed, a new vertex a and three new edges au, av, aw are added. Berg and Jordan [2] have shown that, if G is a 3-connected circuit, then a Henneberg II extension on G is also a 3-connected circuit. The *inverse Henneberg II* operation on a circuit removes one vertex of degree 3 and adds a new edge among its three neighbors in such a way that the result is also a circuit. Berg and Jordan have shown that every 3-connected circuit admits an inverse Henneberg II operation which also maintains 3-connectivity. As a consequence, a 3-connected circuit has an *inductive construction*, i.e. it can be obtained from a single K_4 by Henneberg II extensions that maintain 3-connectivity. Their proof is based on the existence of two non-adjacent vertices with 3-connected inverse Henneberg II circuits. We will make use in Section 3 of the following weaker result, which does not require the maintenance of 3-connectivity in the inverse Henneberg II operation.

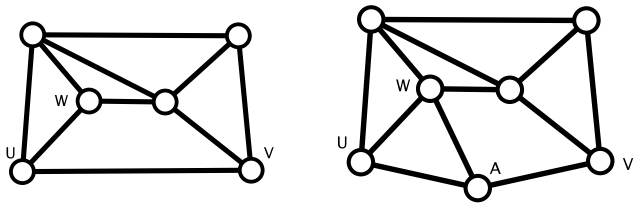

Figure 3 A Henneberg II extension of the Desargues-plus-one circuit.

▶ **Lemma 3** (Theorem 3.8 in [2]). *A 3-connected circuit with at least 5 vertices has either (a) four vertices that admit an inverse Henneberg II to a circuit, or (b) three pairwise non-adjacent vertices that admit an inverse Henneberg II to a circuit (not necessarily 3-connected).*

3 Combinatorial resultants

We define now the *combinatorial resultant* operation on two graphs, prove Theorem 1 and describe its algorithmic implications.

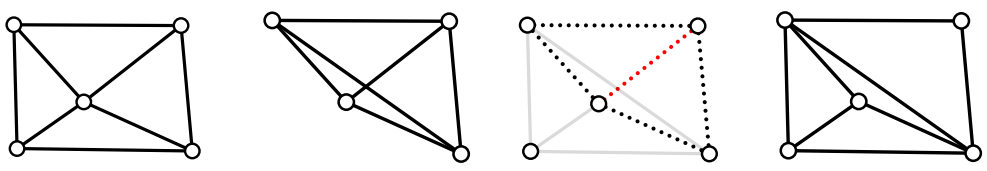

Figure 4 Circuit-invalid combinatorial resultant of two properly intersecting circuits. Left to right: a 4-wheel W_4, a complete K_4 graph, their common Laman graph (dotted, with red elimination edge) and the combinatorial resultant, which is a non-circuit Laman-plus-one graph.

Combinatorial resultant. Let G_1 and G_2 be two distinct graphs with non-empty intersection $E_\cap \neq \emptyset$ and let $e \in E_\cap$ be a common edge. The *combinatorial resultant* of G_1 and G_2 on the *elimination edge* e is the graph $\mathrm{CRes}(G_1, G_2, e)$ with vertex set $V_\cup$ and edge set $E_\cup \setminus \{e\}$.

The 2-sum is the special case when the two graphs have exactly one edge in common. Circuits are closed under the 2-sum operation, but they are not closed under general combinatorial resultants (Fig. 4). We are interested in combinatorial resultants that produce circuits from circuits. The following Lemma gives a necessary condition.

▶ **Lemma 4.** *The combinatorial resultant of two circuits has $m = 2n-2$ edges iff the common subgraph $G_\cap$ of the two circuits is Laman.*

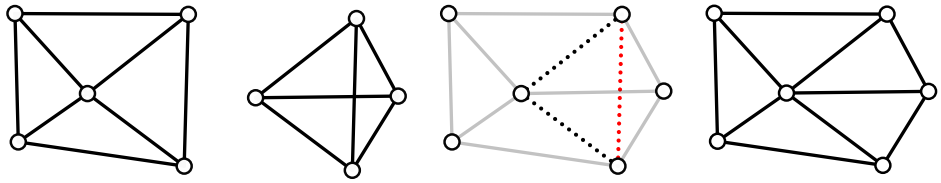

Figure 5 Circuit-valid combinatorial resultant of two properly intersecting circuits. Left to right: a 4-wheel W_4, a complete K_4 graph, their common Laman graph (dotted, with red elimination edge) and their combinatorial resultant, the 5-wheel W_5 circuit.

Proof. Let C be the combinatorial resultant with n vertices and m edges of two circuits C_1 and C_2 with n_i vertices and m_i edges, $i = 1, 2$. By inclusion-exclusion $n = n_1 + n_2 - n_\cap$ and $m = m_1+m_2-m_\cap-1$. Substituting here the values for $m_1 = 2n_1-2$ and $m_2 = 2n_2-2$, we get $m = 2n_1-2+2n_2-2-m_\cap-1 = 2(n_1+n_2-n_\cap)-2+2n_\cap-3-m_\cap = (2n-2)+(2n_\cap-3)-m_\cap$. We have $m = 2n-2$ iff $m_\cap = 2n_\cap - 3$. Since both C_1 and C_2 are circuits, it is not possible that one edge set be included in the other: circuits are minimally dependent sets of edges and thus cannot contain other circuits. As a proper subset of both $E_1 = E(C_1)$ and $E_2 = E(C_2)$, $E_\cap$ satisfies the hereditary $(2, 3)$-sparsity property. If furthermore $G_\cap$ has exactly $2n_\cap - 3$ edges, then it is Laman. ◀

Circuit-valid combinatorial resultant sequences. Two circuits are said to be *properly intersecting* if their common subgraph is Laman. Being properly intersecting is a necessary but not sufficient condition for the combinatorial resultant of two circuits to produce a circuit (Fig. 4). A combinatorial resultant operation applied to two properly intersecting circuits is said to be *circuit-valid* if it results in a spanning circuit (Fig. 5).

▶ **Open Problem 1.** *Find necessary and sufficient conditions for the combinatorial resultant of two circuits to be a circuit.*

In Section 2 we have seen that a 2-connected circuit can be obtained from 3-connected circuits via 2-sums. To complete the proof of Theorem 1 we show now:

▶ **Proposition 5.** *Let $C = (V, E)$ be a 3-connected circuit spanning $n + 1 \geq 5$ vertices. Then, in polynomial time, we can find two circuits A and B such that A has n vertices, B has at most n vertices and C is a circuit-valid combinatorial resultant of A and B. Algorithm 1 summarizes the procedure.*

Algorithm 1 Inverse Combinatorial Resultant.

Input: 3-connected circuit C
Output: circuits A, B and edge e such that $C = \mathrm{CRes}(A, B, e)$

```
for each vertex a of degree 3 do
    if inverse Henneberg II is possible on a
    and there is a non-adjacent degree 3 vertex b then
        Get circuit A and edge e by inverse Henneberg II in C on a
        Let D = C without b (and its edges) and with new edge e
        Compute unique circuit B in D
        return circuits A, B and edge e
    end if
end for
```

Proof. We apply a weaker version of Lemma 3 to find two non-adjacent vertices a and b of degree 3 such that a circuit A can be produced via an inverse Henneberg II operation on vertex a in C (see Fig. 6). Let the neighbors $N(a) = \{u, v, w\}$ of vertex a be labeled such that $e = uw$ is a non-edge in C and becomes an edge in the new circuit $A = (V \setminus \{a\}, E \setminus \{au, av, aw\} \cup \{uv\})$.

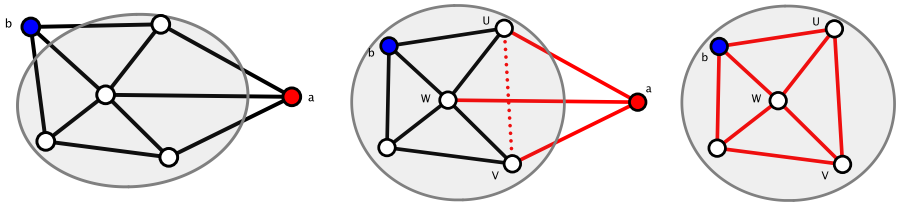

Figure 6 The 3-connected circuit C spanning $n + 1$ vertices with two non-adjacent vertices a (red) and b (blue) of degree 3. Their neighbors $N(a)$ and $N(b)$ may not be disjoint. An inverse Henneberg II at a removes the red edges at a and adds dotted red edge $e = uv$. Circuit A (red).

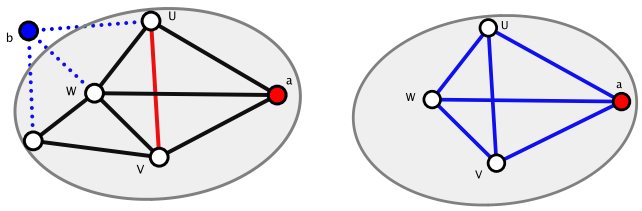

Figure 7 Remove from C the edges from b (blue dotted) and add red edge e. Circuit B (blue).

To define circuit B, we first let L be the subgraph of C induced by $V \setminus \{b\}$. Simple sparsity considerations show that L is a Laman graph. The graph D obtained from L by adding the edge $e = uv$, as in Fig. 7 (left), is a Laman-plus-one graph containing the three edges incident to a (which are not in A) and the edge e (which is in A). Thus D contains a unique circuit B (Fig. 7 left) with edge $e \in B$ (see e.g. [22, Proposition 1.1.6]). It remains to prove that B contains a and its three incident edges. If B does not contain a, then it is a proper subgraph of A: this contradicts the minimality of A as a circuit. Therefore a is a vertex in B, and because a vertex in a circuit can not have degree less than 3, it follows that B must contain all three of its incident edges.

The combinatorial resultant $\mathrm{CRes}(A, B, e)$ of the circuits A and B with e the eliminated edge satisfies the desired property that $C = \mathrm{CRes}(A, B, e)$. Algorithm 1 captures this procedure. The main steps, the Inverse Henneberg II step on a circuit at line 4 and finding the unique circuit in a Laman-plus-one graph at line 6 can be done in polynomial time using properties of the $(2, 3)$ and $(2, 2)$-sparsity pebble games from [18]. ◀

▶ **Corollary 6.** *The representation of C as the combinatorial resultant of two smaller circuits is, in general, not unique: an example is the 2D "double-banana" (Fig. 8).*

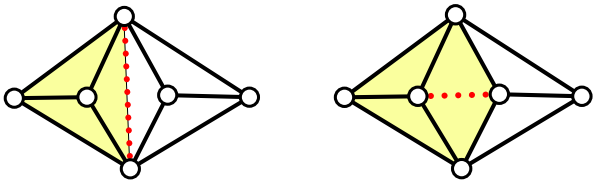

Figure 8 The 2-connected *double-banana* circuit can be obtained as combinatorial resultant from two K_4 graphs (left, 2-sum), and from two wheels on 4 vertices (right). Dashed lines indicate the eliminated edges, and in each case one of the two circuits is highlighted to distinguish K_4 from W_4.

Resultant tree. The inductive construction of a circuit using combinatorial resultant operations is captured by a *tree* structure, whose size influences the complexity of the algorithm. A *resultant tree* T_C for the circuit C on n vertices is a rooted binary tree with C as its root and such that: (a) the nodes of T_C are circuits; (b) circuits on level k have at most $n - k$ vertices; (c) the two children $\{C_a, C_b\}$ of a parent circuit C_c are such that $C_c = \mathrm{CRes}(C_a, C_b, e)$, for some common edge e, and (d) the leaves are complete graphs on 4 vertices. The depth of the tree is at most $n - 4$, and its size may be anywhere between linear to exponential in n. The best case occurs when the resultant tree is path-like, with each internal node having a K_4 leaf. The worst case *could be* a complete binary tree: each internal node at level k would combine two circuits with the same number $n - k - 1$ of vertices into a circuit with $n - k$ vertices. Sporadic examples of small balanced combinatorial resultant trees exist (e.g. for K_{33}-plus-one), but it remains an open problem to find infinite families of such examples. Even if such a family would be found, it is still conceivable that alternative, non-balanced combinatorial resultant trees could yield the same circuit. Answers to the following questions would refine the algorithm's analysis.

▶ **Open Problem 2.** *Are there infinite families of circuits with only balanced combinatorial resultant trees?*

▶ **Open Problem 3.** *Characterize the circuits produced by the worst-case size of the combinatorial resultant tree.*

4 Preliminaries: the Cayley-Menger ideal and its circuit polynomials

In preparation for the proof of Theorem 2, we turn now to the algebraic aspects of our problem and review concepts from polynomial ideals and algebraic matroids. We define *circuit polynomials* in the 2D Cayley-Menger ideal and make the connection with combinatorial rigidity circuits. Complete details and proofs appear in [21].

Notations and conventions. To keep the extended abstract focused, we avoid giving the most general definitions unless necessary. We restrict the presentation to the field of rational numbers $\mathbb{Q}$, to the set of variables $X_n = \{x_i : 1 \leq i \leq n\}$ (or $X_n = \{x_{i,j} : 1 \leq i < j \leq n\}$ in the Cayley-Menger setting) and to polynomial rings $R = \mathbb{Q}[X]$ over sets of variables $X \subset X_n$. The *support* $\operatorname{supp} f$ of a polynomial $f \in \mathbb{Q}[X_n]$ is the set of indeterminates appearing in f.

Polynomial ideals. A set of polynomials $I \subset \mathbb{Q}[X]$ is an *ideal of* $\mathbb{Q}[X]$ if it is closed under addition and multiplication by elements of $\mathbb{Q}[X]$. If $I \subset \mathbb{Q}[X]$ is an ideal and $X' \subset X$ is a subset of variables, then $I \cap \mathbb{Q}[X']$ is also an ideal, called the *elimination ideal* of I with eliminated variables $X \setminus X'$. A *generating set* for an ideal is a set $S \subset \mathbb{Q}[X]$ such that every polynomial in the ideal is an algebraic combination (addition and multiplication) of elements in S with coefficients in $\mathbb{Q}[X]$. *Hilbert Basis Theorem* (see [7]) guarantees that every ideal in a polynomial ring has a finite generating set. Ideals generated by a single polynomial are called *principal*. An ideal I is a *prime* ideal if, whenever $fg \in I$, then either $f \in I$ or $g \in I$. A polynomial is *irreducible* (over $\mathbb{Q}[X]$) if it cannot decomposed into a product of non-constant polynomials in $\mathbb{Q}[X]$. A principal ideal of $\mathbb{Q}[X]$ is prime iff it is generated by an irreducible polynomial. However, an ideal generated by two or more irreducible polynomials is not necessarily prime. We'll make use of the following well-known result.

▶ **Proposition 7.** *If I is a prime ideal of $\mathbb{Q}[X]$ and $X' \subset X$ is non-empty, then the elimination ideal $I \cap \mathbb{Q}[X']$ is prime.*

The theory of Gröbner bases [5, 7] gives a general framework for computing generating sets of an ideal. Gröbner bases also give a general approach for computing elimination ideals: if $\mathcal{G}$ is a Gröbner basis for I with respect to an *elimination order* (see Exercises 5 and 6 in §1 of Chapter 3 in [7]), e.g. the lexicographic order $x_{i_1} > x_{i_2} > \cdots > x_{i_n}$, then the elimination ideal $I \cap \mathbb{Q}[x_{i_{k+1}}, \ldots, x_{i_n}]$ which eliminates the first k indeterminates from I in the specified order has $\mathcal{G} \cap \mathbb{Q}[x_{i_{k+1}}, \ldots, x_{i_n}]$ as its Gröbner basis.

Resultants. The resultant can be introduced in several equivalent ways [11]. Here we use its definition as the determinant of the Sylvester matrix. Given two polynomials f and g in one variable x, of degrees r, resp. s, the Sylvester matrix is an $(r+s) \times (r+s)$ matrix whose entries are a special arrangement of the coefficients of f and g; the precise formulation is given in the full paper [21]. If the coefficients of f, g are themselves polynomials in a ring $R = \mathbb{Q}[X]$, i.e. $f, g \in R[x]$, then the resultant $\operatorname{Res}(f, g, x) \in \mathbb{Q}[X]$ is a polynomial in the coefficients' variables but not in x. In short, the resultant *eliminates* the variable x. A proof of the following proposition can be found in [7, pp. 167].

▶ **Proposition 8.** *Let I be an ideal of $R[x]$ and $f, g \in I$. Then $\operatorname{Res}(f, g, x)$ is in the elimination ideal $I \cap R$.*

Algebraic matroid of a prime ideal. Intuitively, a collection of variables is *independent* if it is not constrained by any polynomial in the ideal, and *dependent* otherwise. Formally, let I be a *prime ideal* of the polynomial ring $\mathbb{Q}[X_n]$. We define the *algebraic matroid* $\mathcal{A}(I)$ of I on the ground set X_n by its independent sets: subsets of variables that are *not* supported by any polynomial in the ideal. Its *dependent sets* are supports of polynomials in the ideal (see [25] for some small examples).

Circuits and circuit polynomials. A *circuit* is a minimal set of variables supported by a polynomial in I, called a *circuit polynomial*. The following theorem, encompassing a result of Lovasz and Dress [8], implies that *circuit polynomials generate elimination ideals supported on circuits.*

▶ **Theorem 9.** *Let I be a prime ideal in $\mathbb{Q}[X]$ and $C \subset X$ a circuit of the algebraic matroid $\mathcal{A}(I)$. The ideal $I \cap \mathbb{Q}[C]$ is principal and generated by an irreducible circuit polynomial p_C, which is unique up to multiplication by a constant.*

The Cayley-Menger ideal and its algebraic matroid. We turn now to the Cayley-Menger setting specific to our paper, where we use variables $X_n = \{x_{i,j} : 1 \leq i < j \leq n\}$ for unknown squared distances between pairs of points. The *distance matrix* of n labeled points is the matrix of squared distances between pairs of points. The *Cayley matrix* is the distance matrix bordered by a new row and column of 1's, with zeros on the diagonal:

$$\begin{pmatrix} 0 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 0 & x_{1,2} & x_{1,3} & \cdots & x_{1,n} \\ 1 & x_{1,2} & 0 & x_{2,3} & \cdots & x_{2,n} \\ 1 & x_{1,3} & x_{2,3} & 0 & \cdots & x_{3,n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1,n} & x_{2,n} & x_{3,n} & \cdots & 0 \end{pmatrix}$$

Cayley's Theorem says that, if the distances come from a point set in the Euclidean space $\mathbb{R}^d$, then the rank of this matrix must be at most $d+2$. Thus all the $(d+3) \times (d+3)$ minors of the Cayley-Menger matrix should be zero. These minors induce polynomials in $\mathbb{Q}[X_n]$ which generate the (n,d)-Cayley-Menger ideal. They are called the *standard generators*, are *homogeneous polynomials* with integer coefficients and are *irreducible* over $\mathbb{Q}$. The (n,d)-Cayley-Menger ideal is a *prime ideal* of dimension $dn - \binom{d+1}{2}$ [3, 12, 13, 16] and codimension $\binom{n}{2} - dn + \binom{d+1}{2}$. We work with the *2D Cayley-Menger ideal* CM_n, generated by the 5×5 minors of the Cayley matrix. Its algebraic matroid is denoted by $\mathcal{A}(\mathrm{CM}_n)$.

The following result (see [21] for a complete proof), allows us to identify the ground set of the Cayley-Menger algebraic matroid X_n with the edges of the complete graph K_n, and the circuits in the algebraic matroid (supports of circuit polynomials in CM_n), as sets of variables, with graph-theoretical rigidity circuits on n vertices.

▶ **Theorem 10.** *The Cayley-Menger algebraic matroid is isomorphic to the rigidity matroid.*

From now on, we will use the isomorphism to *move freely between subsets of variables $X \subset X_n$ and their corresponding edge sets*, and between algebraic circuits and rigidity circuits. Given a (rigidity) circuit C, we denote by p_C its corresponding *circuit polynomial.*

Resultants in the Cayley-Menger ideal. Let f, g be two polynomials in the Cayley-Menger ideal with x_{ij} one of their common variables. We treat them as polynomials in x_{ij}, therefore the coefficients are themselves polynomials in the remaining variables. The fact that the entries in the Sylvester matrix are polynomials supported exactly on the variables corresponding to the *combinatorial resultant* of the supports of f and g on elimination variable (edge) ij is what motivated the definition given in Section 3.

The following lemma, whose proof follows immediately from Proposition 8, provides the connection with combinatorial resultants.

▶ **Lemma 11.** *Let I be an ideal in $\mathbb{Q}[X_n]$, $f, g \in I$ with supports $S_f = \operatorname{supp} f$ and $S_g = \operatorname{supp} g$, and let x_{ij} be a common variable in $S_f \cap S_g$. Let $S = \operatorname{CRes}(S_f, S_g, ij) \subset X_n$ be the combinatorial resultant of the supports. Then $\operatorname{Res}(f, g, x_{ij}) \in I \cap \mathbb{Q}[S]$.*

Homogeneous properties. The standard generators of the Cayley-Menger ideal, in particular those that correspond to K_4 graphs, are obviously homogeneous. The following proposition (to be used in Algorithm 2) shows that the polynomials obtained via resultants are also homogeneous, and allows us to infer their homogeneous degrees.

▶ **Proposition 12.** *The resultant $\operatorname{Res}(f, g, x)$ of homogeneous polynomials f and g of homogeneous degree m and n is a homogeneous polynomial of degree:*

$$m \deg_x g + n \deg_x f - \deg_x f \cdot \deg_x g.$$

5 Computing resultants of circuit polynomials

We are now ready to prove our main result, Theorem 2. Algorithm 2 below describes how to obtain the circuit polynomial p_C from the circuits p_A and p_B, when $C = \operatorname{CRes}(A, B, e)$.

Algorithm 2 Circuit polynomial: resultant step.

Input: Circuits A, B and edge e such that $C = \operatorname{CRes}(A, B, e)$. Circuit polynomials p_A and p_B and elimination variable x_e.
Output: Circuit polynomial p_C for C.

```
1: Compute the resultant p = Res(p_A,p_B,x_e).
2: if p is irreducible then
3:     p_C = p return p_C
4: else
5:     factors = factorize p over Q
6:     factors = discard factors with support not equal to C
7:     if exactly one remaining factor (possibly with multiplicity) then
8:         p_C = the unique factor supported on C
9:     else
10:        apply a test of membership in the CM ideal on the remaining factors
11:        p_C = unique factor for which ideal membership test succeeded
12:        return p_C
13:    end if
14: end if
```

The correctness of Steps 1–4 follows from the following Lemma.

▶ **Lemma 13.** *Let C be a rigidity circuit $C = \operatorname{CRes}(A, B, e)$ obtained as a combinatorial resultant of two other circuits A and B with p_A and p_B as their circuit polynomials. Then:*

1. *The resultant $\operatorname{Res}(p_A, p_B, x_e)$ is supported on C and contained in the elimination ideal $\langle p_C \rangle$ generated by the circuit polynomial p_C.*
2. *The circuit polynomial p_C of C is an irreducible factor over $\mathbb{Q}$ of $\operatorname{Res}(p_A, p_B, x_e)$.*
3. *When $\operatorname{Res}(p_A, p_B, x_e)$ is irreducible then it will be equal to p_C.*

Proof. The resultant $\operatorname{Res}(p_A, p_B, x_e)$ is a non-constant polynomial supported on C. Since $\langle p_A, p_B \rangle \subset \operatorname{CM}_n$, by Lemma 11 we have that $\operatorname{Res}(p_A, p_B, x_e)$ is contained in the elimination ideal $\operatorname{CM}_n \cap \mathbb{Q}[C] = \langle p_C \rangle$. ◀

Steps 6–9 would not be necessary if the resultant would always be irreducible. But in general p_C will only be one of the irreducible factors over $\mathbb{Q}$ of $\mathrm{Res}(p_A, p_B, x_e)$.

▶ **Lemma 14.** *The resultant of two circuit polynomials is not always a circuit polynomial.*

Proof. We prove the Lemma with an example, which can be easily generalized. Recall from Corollary 6 that in general a rigidity circuit C can be represented as the combinatorial resultant of two circuits in more than one way. If $C = \mathrm{CRes}(C_1, C_2, e) = \mathrm{CRes}(C_3, C_4, f)$ and p_{C_i} for $i \in \{1, 2, 3, 4\}$ are the corresponding circuit polynomials, then $\mathrm{Res}(p_{C_1}, p_{C_2}, x_e)$ and $\mathrm{Res}(p_{C_3}, p_{C_4}, x_f)$ will in general be distinct elements of $\langle p_C \rangle$. The 2-connected circuit in Fig. 8 has two distinct combinatorial resultant trees. Using Prop. 12 to compute the homogeneous degrees of the resultants, we obtain degrees 8 and 48. Both resultants have the same circuit as its supporting set, hence they are both in the elimination ideal of the circuit, but only the one of homogeneous degree 8 is the circuit polynomial. ◀

In general, if two resultant steps produce two polynomials on the same rigidity circuit support, the one of higher degree must have a non-trivial factor, while the other one will possibly be irreducible, but this is not guaranteed.

▶ **Corollary 15.** *Under the assumptions of Theorem 13, the resultant* $\mathrm{Res}(p_A, p_B, x_e)$ *is a circuit polynomial if and only if it is irreducible (over* $\mathbb{Q}$*).*

This leads to the following natural question.

▶ **Open Problem 4.** *Let C, A and B be rigidity circuits such that $C = \mathrm{CRes}(A, B, e)$ with p_C, p_A and p_B the corresponding circuit polynomials. Identify sufficient conditions under which* $\mathrm{Res}(p_A, p_B, x_e)$ *is* p_C*.*

Since in general p_C will only be one of the irreducible factors over $\mathbb{Q}$ of $\mathrm{Res}(p_A, p_B, x_e)$, we must proceed to Step 7 in the Algorithm and factorize the resultant p. *The need for the factorization step was already encountered in the examples we have so far computed.*

Determining the circuit polynomial from the resultant. If $\mathrm{Res}(p_A, p_B, x_e)$ is not irreducible, then exactly one of its irreducible factors (over $\mathbb{Q}$) is in CM_n, and that "good" factor is precisely the circuit polynomial p_C. A "bad" factor can be discarded in two steps: by analysing its support (lines 8–11) and by an ideal membership test (lines 13-14).

Analysing the supports of the irreducible factors. The elimination ideal $\langle p_C \rangle$ is an ideal of $\mathbb{Q}[C]$, and since $\mathrm{Res}(p_A, p_B, x_e) \in \langle p_C \rangle$, any irreducible factor (over $\mathbb{Q}$) of this resultant is supported on a subset of C that is not necessarily proper. We know that at least one of these factors must be supported on exactly C, and if there is only one such factor (possibly with multiplicity), then it must be p_C. Factors with a proper support are necessarily not in the ideal (they are *independent* in the Cayley-Menger matroid) and can be discarded. Hence, if there are no other factors supported on C, the algorithm stops on line 11 and returns P_C without any additional calculations. *So far, in all the concrete examples on which we could complete the calculations with our algorithm, the "bad" factors were supported on strict subsets of C*: whether this is always true remains an intriguing open question/conjecture. Should it be true, then there will be no need for Steps 9–14 in our algorithm.

▶ **Open Problem 5.** *Identify sufficient conditions for the resultant* $\mathrm{Res}(p_A, p_B, x_e)$ *of two circuit polynomials to have exactly one factor (up to multiplicity) supported on* $\mathrm{CRes}(A, B, e)$.

Lacking a definitive answer at this time, we describe a way for the algorithm to proceed.

Ideal membership test. Let's assume that we have an irreducible factor of $\mathrm{Res}(p_A, p_B, x_e)$ that is supported on C. We will have to test it for membership in CM_n. The *ideal membership test* [7, pp. 97] invokes a Gröbner basis calculation, but it can be with any monomial order, not necessarily an elimination order, hence it has a better chance at succeeding. Further details and possible algorithm optimization ideas (which we did not have yet to employ in the experimental part reported in the next section) are described in the full paper [21].

6 Concluding remarks

In this paper we introduced the combinatorial resultant operation, analogous to the classical resultant of polynomials, and used it to derive a new algorithm for computing circuit polynomials in the 2D Cayley-Menger ideal. Our approach highlights an inherent combinatorial structure in this ideal and leads to further theoretical and algorithmic questions. To demonstrate the effectiveness of our method we conclude by listing in Table 1 the circuit polynomials that we could compute within a reasonable amount of time. The most challenging was the $K_{3,3}$-plus-one circuit, which required an extension of the method presented here: this *extended resultant* is the topic of an upcoming paper [20].

Table 1 Results: all circuit polynomials on $n \leq 6$ vertices and two circuit polynomials on $n = 7$ vertices. For the definition of Extended Resultant, see [20].

n	Circuit	Method	Comp. time (seconds)	No. terms	Hom. degree
4	K_4	Determinant	0.0008	22	3
5	Wheel on 4 vertices	Gröbner Resultant	0.02 0.013	843	8
6	2D double banana	Gröbner Resultant	0.164 0.029	1 752	8
6	Wheel on 5 vertices	Gröbner Resultant	10 857 7.07	273 123	20
6	Desargues-plus-one	Gröbner Resultant	454 753 14.62	658 175	20
6	$K_{3,3}$-plus-one	Extended Resultant	1 402	1 018 050	18
7	2D double banana $\oplus_{16}$ K_4^{1567}	Resultant	38.14	1 053 933	20
7	2D double banana $\oplus_{56}$ K_4^{4567}	Resultant	89.86	2 579 050	20

References

1 Evangelos Bartzos, Ioannis Z. Emiris, Jan Legerský, and Elias Tsigaridas. On the maximal number of real embeddings of minimally rigid graphs in R^2 , R^3 and S^2. *Journal of Symbolic Computation*, 102:189–208, 2021. doi:10.1016/j.jsc.2019.10.015.

2 Alex R Berg and Tibor Jordán. A proof of Connelly's conjecture on 3-connected circuits of the rigidity matroid. *Journal of Combinatorial Theory, Series B*, 88(1):77–97, 2003. doi:10.1016/S0095-8956(02)00037-0.

3 Ciprian S. Borcea. Point configurations and Cayley-Menger varieties, 2002. arXiv:math/0207110.

4 Ciprian S. Borcea and Ileana Streinu. The number of embeddings of minimally rigid graphs. *Discrete and Computational Geometry*, 31:287–303, February 2004. doi:10.1007/s00454-003-2902-0.

5 B. Buchberger. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. *Aequationes Math.*, 4:374–383, 1970. doi:10.1007/BF01844169.

6 Jose Capco, Matteo Gallet, Georg Grasegger, Christoph Koutschan, Niels Lubbes, and Josef Schicho. Computing the number of realizations of a Laman graph. *Electronic notes in Discrete Mathematics*, 61:207–213, 2017. doi:10.1016/j.endm.2017.06.040.

7 David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms. An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer, Cham, fourth edition, 2015.

8 A. Dress and L. Lovász. On some combinatorial properties of algebraic matroids. *Combinatorica*, 7(1):39–48, 1987. doi:10.1007/BF02579199.

9 I. Emiris and B. Mourrain. Computer algebra methods for studying and computing molecular conformations. *Algorithmica*, 25:372–402, 1999. doi:10.1007/PL00008283.

10 Ioannis Z. Emiris, Elias P. Tsigaridas, and Antonios Varvitsiotis. Mixed Volume and Distance Geometry Techniques for Counting Euclidean Embeddings of Rigid Graphs. In Mucherino, Antonio and Lavor, Carlile and Liberti, Leo and Maculan, Nelson, editor, *Distance Geometry. Theory, Methods, and Applications*, chapter 2, pages 23–46. Springer, New York, Heidelberg, Dordrecht, London, 2013. doi:10.1007/978-1-4614-5128-0.

11 I.M. Gelfand, M. Kapranov, and A. Zelevinsky. *Discriminants, Resultants, and Multidimensional Determinants*. Modern Birkhäuser Classics. Birkhäuser Boston, 2009. doi:10.1016/0001-8708(90)90047-Q.

12 G.Z. Giambelli. Sulle varietá rappresentate coll'annullare determinanti minori contenuti in un determinante simmetrico od emisimmetrico generico di forme. *Atti della R. Acc. Sci, di Torino*, 44:102–125, 1905/06.

13 J. Harris and L.W. Tu. On symmetric and skew-symmetric determinantal varieties. *Topology*, 23:71–84, 1984. doi:10.1016/0040-9383(84)90026-0.

14 Lebrecht Henneberg. *Die graphische Statik der starren Systeme*. B. G. Teubner, 1911.

15 John E. Hopcroft and Robert Endre Tarjan. Dividing a graph into triconnected components. *SIAM J. Comput.*, 2(3):135–158, 1973. doi:10.1137/0202012.

16 T. Józefiak, A. Lascoux, and P. Pragacz. Classes of determinantal varieties associated with symmetric and skew-symmetric matrices. *Math. USSR Izvestija*, 18:575–586, 1982. doi:10.1070/im1982v018n03abeh001400.

17 Gerard Laman. On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics*, 4:331–340, 1970. doi:10.1007/BF01534980.

18 Audrey Lee-St. John and Ileana Streinu. Pebble game algorithms and sparse graphs. *Discrete Mathematics*, 308(8):1425–1437, April 2008. doi:10.1016/j.disc.2007.07.104.

19 Goran Malić and Ileana Streinu. CayleyMenger - Circuit Polynomials in the Cayley Menger ideal, a GitHub repository. https://github.com/circuitPolys/CayleyMenger, 2020.

20 Goran Malić and Ileana Streinu. Circuit polynomial for the $K_{3,3}$-plus-one circuit, 2021. In preparation.

21 Goran Malić and Ileana Streinu. Combinatorial resultants in the algebraic rigidity matroid, 2021. ArXiv/2103.08432 [Math.CO]. arXiv:2103.08432.

22 James Oxley. *Matroid theory*, volume 21 of *Oxford Graduate Texts in Mathematics*. Oxford University Press, Oxford, second edition, 2011.

23 Zvi Rosen. *Algebraic Matroids in Applications*. PhD thesis, University of California, Berkeley, 2015. URL: https://digitalassets.lib.berkeley.edu/etd/ucb/text/Rosen_berkeley_0028E_15261.pdf.

24 Zvi Rosen. algebraic-matroids, a GitHub repository. https://github.com/zvihr/algebraic-matroids, 2017.

25 Zvi Rosen, Jessica Sidman, and Louis Theran. Algebraic matroids in action. *The American Mathematical Monthly*, 127(3):199–216, February 2020. doi:10.1080/00029890.2020.1689781.

26 Meera Sitharam and Heping Gao. Characterizing graphs with convex and connected Cayley configuration spaces. *Discrete and Computational Geometry*, 43(3):594–625, 2010. doi:10.1007/s00454-009-9160-8.

27 William T. Tutte. *Connectivity in graphs*. Toronto University Press, Toronto, 1966.

28 D. Walter and M.L. Husty. On a nine-bar linkage, its possible configurations and conditions for flexibility. In Merlet J.-P. and M. Dahan, editors, *Proceedings of IFFToMM 2007, Besançon, France*, 2007. URL: http://geometrie.uibk.ac.at/cms/datastore/husty/A681.pdf.