Fast Isotopy Computation for T-Curves

Zoe Geiselmann[1 [0009−0009−9244−4752]], Michael Joswig[1 [0000−0002−4974−9659]], Lars Kastner[1 [0000−0001−9224−7761]], Konrad Mundinger[2 [0009−0007−8059−9090]], Sebastian Pokutta[1,2 [0000−0001−7365−3000]], Christoph Spiegel[2 [0000−0002−6545−202X]], Marcel Wack[1 [0009−0005−0360−8569]], and Max Zimmer[2 [0009−0007−8683−1030]]

[1] Technische Universität Berlin, Chair of Discrete Mathematics/Geometry
{geiselmann,joswig,kastner,wack}@math.tu-berlin.de
[2] Zuse Institut Berlin, AI in Society, Science, and Technology
{mundinger,pokutta,spiegel,zimmer}@zib.de

Abstract. A T-curve of degree d is given by a regular unimodular triangulation of $d \cdot \Delta_2$ together with a sign distribution on its lattice points. By Viro's Patchworking Theorem, this determines the ambient isotopy type (a.k.a. real scheme) of a smooth real plane projective algebraic curve of the same degree. We present a near-quadratic time algorithm for extracting that isotopy type from the triangulation and the signs. Through a GPU-accelerated implementation, this allows one to compute billions of real schemes per second, enabling exhaustive enumeration at scale. This algorithm was essential for our recent construction of all 121 real schemes of degree seven by T-curves.

Keywords: Hilbert's 16th problem · patchworking · real scheme

1 Introduction

Hilbert's 16th problem asks for a topological classification of real plane projective algebraic curves [9]. A common interpretation of "topological classification" is classification up to ambient isotopy; the resulting classes are called *real schemes*. That classification is settled through degree seven, with 56 real schemes for degree six [6] and 121 for degree seven [17]; degree eight remains open [14]. Determining the real scheme of a curve given by an explicit polynomial is computationally expensive: general methods like cylindrical algebraic decomposition [2] or specific planar techniques [15, 13, 4] rely on polynomial system solving, which is costly. For instance, in determining real schemes of degree-six curves in `Mathematica`, Kaihnsa et al. [12] found that large coefficients arising from classical constructions made the computation prohibitively slow.

Combinatorial patchworking, introduced by Viro [17, 18], provides a discrete setting in which this computation becomes more tractable. A regular unimodular triangulation of the dilated simplex $d \cdot \Delta_2$ together with a sign distribution on its lattice points determines a real algebraic curve (a *T-curve*) whose topology depends only on this combinatorial data. The trade-off is expressiveness: not every real scheme is realizable as a T-curve [10], although through degree seven all nonempty ones are [5]. Viro also describes a procedure for recovering the topology of a patchworked curve [18, Algorithms 1.3.A/C/E]: a sequence of geometric surgery operations — cutting, gluing, and contracting polygons — that yields a topological model of the curve embedded in $\mathbb{RP}^2$, from which the real scheme can be read off by inspection. This procedure operates on continuous objects and specifies no discrete data structures, pseudocode, or complexity bounds.

Existing computational tools address aspects of patchworking topology. The Combinatorial Patchworking Tool [8] visualizes patchworked plane curves and counts their connected components. `Viro.sage` [19] and `polymake` [11] compute homological invariants of patchworked hypersurfaces in arbitrary dimension,[3] which for plane curves reduce to connected component counts

[3] `Viro.sage` computes integral homology via Smith normal form on an explicit simplicial complex; `polymake` computes $\mathbb{Z}/2$-Betti numbers via rank computation over $\mathbb{F}_2$ on a cellular chain complex derived directly from the combinatorial data.

and carry no nesting information. On a related but distinct invariant (the *complex scheme*), De Loera and Wicklin [3] describe a combinatorial algorithm for the dividing type of a T-curve (Type I vs. Type II, credited to Itenberg–Viro), which represents the closest prior algorithmic work on patchwork-derived invariants. To the best of our knowledge, no prior published work provides a competitive algorithm that extracts the full real scheme from the combinatorial data.

Contributions. We describe an algorithm that computes the real scheme of a combinatorial patchwork in near-quadratic time in the degree using union-find for connected components, traversal of the region adjacency graph, and rooted-tree canonicalization for the output (Section 2). This enables exhaustive enumeration of patchworks with a given triangulation, a process that was essential for the recent verification that all 121 real schemes in degree seven are realizable [5], whose computations and data are available at `github.com/dmg-lab/CombinatorialPatchworking`. A C++ implementation is publicly available at `github.com/polymake/libisotopy`. Since the algorithm requires only fixed-size, thread-local state, a GPU-accelerated implementation can classify billions of patchworks per second. Section 3 presents results obtained with these implementations for degrees up to eight.

2 Algorithm

We recall Viro's combinatorial patchwork construction, which defines a piecewise-linear (PL) curve in the real projective plane $\mathbb{RP}^2$ from purely combinatorial data, and then present an algorithm that computes the real scheme of this curve. The *lattice point set* of degree d is

$$A \;=\; d \cdot \Delta_2 \cap \mathbb{Z}^2 \;=\; \big\{(i,j) \in \mathbb{Z}^2 : i,j \geq 0,\; i+j \leq d\big\}\,, \tag{1}$$

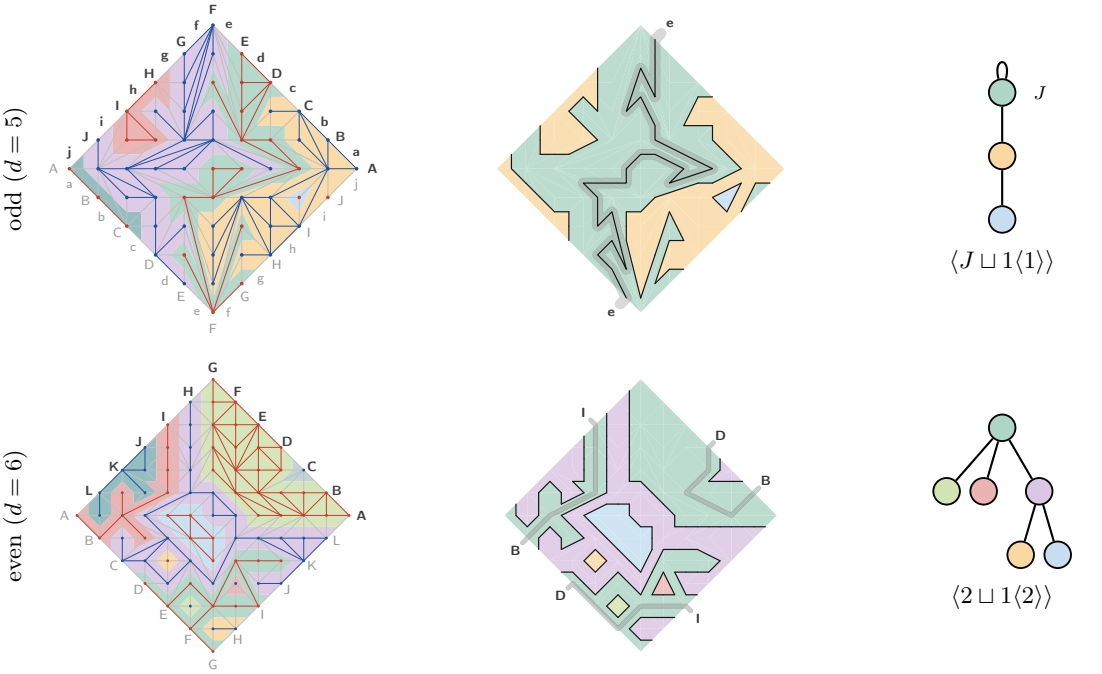
where $\Delta_2 = \operatorname{conv}\{(0,0),(1,0),(0,1)\}$ is the standard triangle; A has $\frac{1}{2}(d{+}1)(d{+}2)$ elements. A triangulation $\mathcal{T}$ of A, represented by its edge list, is *unimodular* if each triangle has Euclidean area $\frac{1}{2}$; equivalently, every point in A occurs as a vertex of $\mathcal{T}$. It is *regular* if there exists a lifting function $\omega\colon A \to \mathbb{Z}$ such that $\mathcal{T}$ is the projection of the lower convex hull of the lifted points $\{(i,j,\omega(i,j)) : (i,j) \in A\}$.

Given a triangulation $\mathcal{T}$ and a *sign distribution* $\sigma\colon A \to \mathbb{F}_2$, the patchworking construction [18, Algorithms 1.3.A/C/E] proceeds as follows. Reflecting $\mathcal{T}$ into four quadrants yields a triangulation $\mathcal{T}^\diamond$ of the *diamond* $A^\diamond = \{(i,j) \in \mathbb{Z}^2 : |i| + |j| \leq d\}$, and σ extends from A to $A^\diamond$ by the rule

$$\begin{aligned} \sigma(i,-j) &\equiv \sigma(i,j) + j \pmod 2\,,\\ \sigma(-i,j) &\equiv \sigma(i,j) + i \pmod 2\,,\\ \sigma(-i,-j) &\equiv \sigma(i,j) + i + j \pmod 2 \end{aligned} \tag{2}$$

for $(i,j) \in A$; this rule is forced by the parity structure of monomials $x^i y^j$ under coordinate reflections. Identifying antipodal boundary points, $(i,d{-}i) \sim -(i,d{-}i)$ and $(i,i{-}d) \sim -(i,i{-}d)$ for $0 \leq i \leq d$, yields a cell decomposition $\mathcal{S}$ of $\mathbb{RP}^2$. The *patchworked curve* $\mathcal{C}(\mathcal{T},\sigma)$ is obtained by connecting the midpoints of edges that separate opposite signs; this produces a PL curve in the first barycentric subdivision of $\mathcal{S}$. Combinatorially, $\mathcal{C}(\mathcal{T},\sigma)$ is the subgraph of the dual graph of $\mathcal{S}$ formed by edges whose endpoints have distinct signs. A real algebraic curve *ambient isotopic* (deformable by a continuous motion of $\mathbb{RP}^2$) to $\mathcal{C}(\mathcal{T},\sigma)$ is called a *T-curve*. Viro established that such a curve exists whenever $\mathcal{T}$ is regular. We additionally require $\mathcal{T}$ to be unimodular, which ensures that the resulting curve is smooth.

Theorem 1 (Viro [17, 18]; cf. [5, Theorem 1]). *Let $\mathcal{T}$ be a regular and unimodular triangulation of A with lifting function $\omega\colon A \to \mathbb{Z}$, and let $\sigma\colon A \to \mathbb{F}_2$. Then there exists $t_0 > 0$ such that for all $t \in (0,t_0]$ the curve $V_{\mathbb{R}}(f) \subset \mathbb{RP}^2$, where $f = \sum_{(i,j)\in A} (-1)^{\sigma(i,j)}\, t^{\omega(i,j)}\, x^i y^j z^{d-i-j}$, is ambient isotopic to $\mathcal{C}(\mathcal{T},\sigma)$.*



(a) Components (b) Regions (c) Tree

Fig. 1. Algorithm A applied to a degree 5 (top) and degree 6 (bottom) example. (a) Connected components of same-sign vertices in $\mathcal{T}^\diamond$, with antipodal boundary pairs labeled. (b) Regions after boundary identification; patchworked curve $\mathcal{C}(\mathcal{T}, \sigma)$ drawn solid. The non-orientable root region is indicated by a semi-transparent band: for odd d it follows the pseudo-line; for even d ($d = 6$) it traces a Möbius strip through three components connected via boundary identifications. (c) Rooted nesting tree and real scheme; a self-loop marks the pseudo-line region J in odd degree.

Our goal is to compute the *real scheme* of $\mathcal{C}(\mathcal{T}, \sigma)$ from the combinatorial data $(\mathcal{T}, \sigma)$. The patchworked curve is a disjoint union of simple closed curves in $\mathbb{RP}^2$. Each of these is either an *oval* (if it separates $\mathbb{RP}^2$) or a *pseudoline* (if it does not). For even d, every curve component is an oval; for odd d, there is exactly one pseudoline [5, Proposition 3]. By Harnack's theorem [7], the total number of connected components is at most $M = \frac{1}{2}(d-1)(d-2) + 1$.

We call the connected components of $\mathbb{RP}^2 \setminus \mathcal{C}(\mathcal{T}, \sigma)$ *regions*. Since the ovals are pairwise disjoint and each separates $\mathbb{RP}^2$ into an interior and an exterior, their nesting defines a tree structure on the regions. This is expressed by the *region adjacency graph*, which has one vertex per region and one edge per connected component of the curve, connecting the two regions on either side (a self-loop when both sides coincide, as for the pseudoline). The interior of each oval is a disk, and the exterior region is a Möbius strip, which serves as the root of this tree. For even d, this is the exterior of all ovals; for odd d, it is the region whose closure contains the pseudoline. The *real scheme* captures the ambient isotopy class of $\mathcal{C}(\mathcal{T}, \sigma)$ in $\mathbb{RP}^2$ through the isomorphism class of this rooted tree, encoded in *Rohlin–Viro notation*: $\langle 0\rangle$ denotes the empty scheme (even d), $\langle k\rangle$ denotes k unnested ovals, $\langle X \sqcup Y\rangle$ the disjoint union of two subschemes X and Y, $\langle k\langle X\rangle\rangle$ denotes k ovals all of whose interiors have scheme X, and J marks the unique pseudoline for odd d.

Recall that $\mathcal{C}(\mathcal{T}, \sigma)$ arises as a subgraph in the first barycentric subdivision of $\mathcal{S}$. Its connected components are easily identified. To derive the real scheme, it remains to determine which of them are separating and to distinguish the interior from the exterior of each oval; one way to accomplish this is by $(\mathrm{mod}\, 2)$ homology computations. A naïve implementation has an estimated complexity of $O(d^5)$: there are $O(d^2)$ connected components; each requires one homology computation, e.g., via mod 2 Gauß elimination in $O(d^3)$ time.

Algorithm A Real scheme of a combinatorial patchwork.

Require: Edge list $E(\mathcal{T})$, sign distribution $\sigma\colon A \to \mathbb{F}_2$, degree d
Ensure: Canonical real scheme of $\mathcal{C}(\mathcal{T}, \sigma)$ in Viro notation

Signed reflected triangulation.
1: Construct $A^\diamond$ and $E(\mathcal{T}^\diamond)$ by reflecting into four quadrants
2: Extend σ to $A^\diamond$ via (2)

Components and regions.
3: Compute connected components of same-sign vertices in $\mathcal{T}^\diamond$
4: Coarsen into regions by identifying antipodal boundary pairs

Root and real scheme.
5: Build region adjacency graph from crossed edges of $\mathcal{T}^\diamond$
6: **if** d is even **then**
7: $\quad r \leftarrow$ unique region with an odd cycle under antipodal adjacency
8: **else**
9: $\quad r \leftarrow$ unique region containing two crossed-edge-adjacent components
10: **return** canonical form of the region adjacency graph rooted at r

Our method for extracting the real scheme from $(\mathcal{T}, \sigma)$ works on $\mathcal{T}^\diamond$ directly. We call an edge of $\mathcal{T}^\diamond$ *crossed* if its endpoints have distinct signs, and define a *component* of $(\mathcal{T}, \sigma)$ as a connected component of the subgraph induced by the non-crossed edges; each component is monochromatic by construction. The crossed edges define a first notion of adjacency on the components. To pass from $\mathbb{R}^2$ to $\mathbb{RP}^2$, we identify each of the $4d$ boundary vertices of $A^\diamond$ with its antipodal partner, merging any two components that share such a pair; this defines a second, antipodal notion of adjacency. The coarsened partition of components coincides with the regions of $\mathbb{RP}^2 \setminus \mathcal{C}(\mathcal{T}, \sigma)$ defined above.[4] The non-orientable root region is detected as follows: for odd d, it is the unique region containing two components that are crossed-edge adjacent; for even d, it is the unique region whose components form an odd cycle under antipodal adjacency. Algorithm A summarizes the procedure; see Figure 1 for two examples.

Theorem 2. *Algorithm A computes the real scheme of $\mathcal{C}(\mathcal{T}, \sigma)$ in $O(d^2 \cdot \alpha(d^2))$ time, where α denotes the inverse Ackermann function.*

Proof. We argue in two parts: that Algorithm A returns the correct rooted region adjacency graph (hence the real scheme), and that it runs within the stated bound.

Correctness. Removing the crossed edges from $\mathcal{T}^\diamond$ partitions its vertices into same-sign connected components; after antipodal boundary identification these induce the regions of $\mathbb{RP}^2 \setminus \mathcal{C}(\mathcal{T}, \sigma)$ [5, §2.4]. Each crossed edge connecting vertices in two distinct regions corresponds to an oval separating those regions (or the pseudoline for odd d), so the graph built in Line 5 is the region adjacency graph defined above. For root detection: when d is odd, the pseudoline is nonseparating, so both sides belong to the same region, producing the self-loop that identifies the root; when d is even, every curve component is an oval, and the non-orientable root region is detected as the unique region whose components form an odd cycle under antipodal identification: each identification reverses orientation, so an odd number of reversals makes the region non-orientable. Since the region adjacency graph is a tree (with a self-loop only at the root for odd d), traversal from the root and canonicalization recover the real scheme. Neither regularity nor unimodularity of $\mathcal{T}$ is needed for this computation, only for the algebraic existence guarantee of Theorem 1.

Complexity. The reflected triangulation has $O(d^2)$ vertices and edges. Union-find with path compression and union by rank [16] (Lines 3–4) runs in $O(d^2 \cdot \alpha(d^2))$ time; Line 4 adds $O(d)$ union operations for boundary pairs. Building the region adjacency graph (Line 5) scans $O(d^2)$

[4] Note that antipodal boundary vertices have opposite signs if and only if d is odd.

Table 1. Search space parameters by degree. $|E(\mathcal{T}^\diamond)|$ is the number of edges in the reflected graph; M is Harnack's bound. Regular unimodular triangulations up to $\mathfrak{S}_3$-symmetry (symmetric representatives, i.e., those fixed by $(x, y) \mapsto (y, x)$, in parentheses); a dash marks counts we did not enumerate, the total being infeasibly large. Sign distributions up to the $(\mathbb{Z}/2)^3$-action, giving $2^{|A|-3}$ per triangulation.

d	$\lvert A\rvert$	$\lvert E(\mathcal{T}^\diamond)\rvert$	M	triangulations (symmetric)	sign distributions
2	6	28	1	2 (2)	8
3	10	60	2	18 (7)	128
4	15	104	4	1 278 (74)	4 096
5	21	160	7	561 885 (1 194)	262K
6	28	228	11	1 198 202 590 (62 960)	34M
7	36	308	16	– (4 729 700)	8.6B
8	45	400	22	– (1 199 795 773)	4.4T

edges; root identification (Lines 6–9) takes $O(d)$ time. The rooted region adjacency graph has at most $M + 1$ nodes for even d and at most M for odd d (where M is given by Harnack's bound), and is canonicalized in linear time, e.g., via the algorithm in [1]. The total is $O(d^2 \cdot \alpha(d^2))$, dominated by union-find. □

3 Exhaustive enumeration

Algorithm A classifies a single pair $(\mathcal{T}, \sigma)$. In order to explore the real schemes which are realizable as T-curves for a given degree d it is useful to enumerate triangulations and sign distributions systematically. Under a natural equivalence relation, the number of sign distributions is reduced by a factor of eight, and triangulation orbits generally contain six elements, three for symmetric triangulations [5, Section 3].[5] Table 1 summarizes the resulting counts. This explains our predominant search mode: check all sign distributions for one or a few triangulations. Here, orchestration overhead is minimal and throughput is near the compute floor; as a side benefit, this yields the distribution of the number of ovals across all sign distribution classes (Figure 2). For degrees where checking all triangulations is not feasible, we employed a second mode: check all symmetric triangulations for one or a few sign distributions, at the cost of generating triangulations and allocating memory for each.

For $d \leq 5$, exhaustively checking all triangulation orbits against all sign distributions is fast. For degree six, the full process over all triangulation orbits is just feasible at roughly 10 000 GPU-hours but unlikely to produce valuable insights. Restricting to equivalence classes containing a symmetric triangulation makes this feasible. Through these searches, all nonempty real schemes are realized as T-curves for $d \leq 5$ (a single triangulation suffices), and two triangulations suffice to cover all 55 nonempty real schemes of degree six [5, Theorem 20].

For degree seven, exhaustive sign distribution sweeps over four hand-designed triangulations realize all 121 nonempty real schemes [5, Theorem 24]. One of these triangulations already produces 115 of the 121 real schemes.

For degree eight, prior work has focused on real schemes with the maximum number of ovals given by Harnack's bound (22 ovals for $d = 8$). At most 89 are realizable [14], of which 83 have algebraic constructions, leaving six open. Beyond the maximum case, no systematic classification of degree eight real schemes exists. Our approach combined two strategies to build an initial pool of triangulations and realized types: simulated annealing searches over triangulations, optimizing for a target (p, n)-count or for the number of distinct types realized, and hand-designed triangulations guided by a curated pool of sign distributions. We then iterated: exhaustively check all sign distributions for the selected triangulations and extract one witness

[5] Fixing a triangulation and sweeping over sign distribution representatives yields all types realizable by that orbit of triangulations. The converse — fixing a sign distribution and sweeping over triangulation representatives — does not, since the triangulation equivalence also acts on the signs.

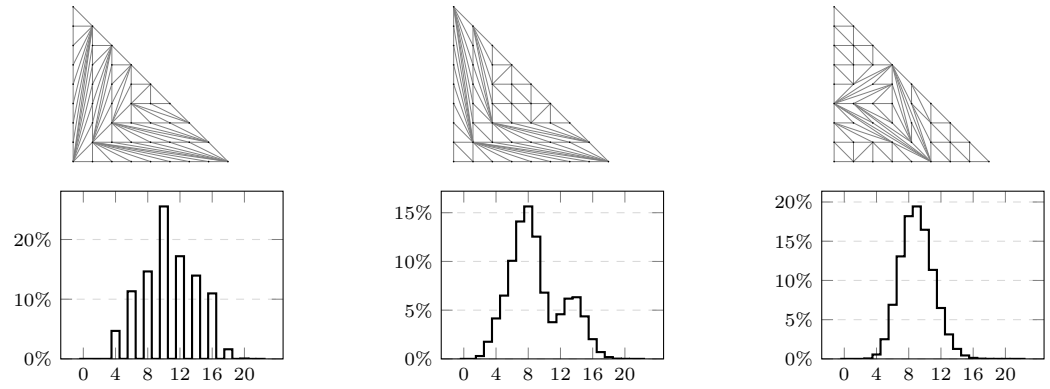


Fig. 2. Three degree-eight triangulations of $A = 8 \cdot \Delta_2 \cap \mathbb{Z}^2$ (top, first quadrant shown) and the distribution of the number of ovals over all 2^{42} sign distributions (bottom). The first is the bow tie triangulation [5, Section 4.3], which realizes only even oval counts. The three triangulations realize 123, 359, and 353 real schemes, respectively.

sign distribution per newly realized type, then sweep all symmetric triangulations against the accumulated witnesses. Repeat until convergence. This iterative process realized exactly 2359 distinct real schemes with representatives at every oval count from 1 to 22, of which only 30 are maximum. Among maximum real schemes, the six left open by Orevkov remain unresolved, and 53 others have no known T-curve realization.

Since Algorithm A classifies each pair $(\mathcal{T}, \sigma)$ independently, exhaustive enumeration is embarrassingly parallel, and since all data structures are degree-bounded and fit in thread-local memory, it is naturally suited to GPU execution. We implemented both a minimal, single-patchwork-focused C++ library (`libisotopy`) and a companion Rust implementation focused on throughput for large exhaustive searches across one or more GPUs, with both CUDA and Metal backends. At scale, orchestration overhead — memory transfer, histogram aggregation, kernel dispatch — and data collection become the computationally challenging aspects rather than the classification itself. At the compute floor (all overhead amortized), a single NVIDIA A100 classifies roughly 10^8 pairs per second for degree eight, with the exact rate depending on the triangulation. An exhaustive sweep over all 2^{42} sign distributions for one degree-eight triangulation thus takes on the order of hours on a single GPU; multi-GPU execution scales linearly.

4 Conclusion

We presented an algorithm for computing the real scheme of a combinatorial patchwork in near-quadratic time. The key ingredients are a union-find computation on the reflected signed triangulation, a bipartiteness check on the antipodal component graph for root identification, and the eightfold symmetry of the sign distribution space that makes exhaustive sweeps tractable. Two implementations are provided: `libisotopy`, a publicly available C++ library, and a multi-GPU Rust implementation with CUDA and Metal backends.

Two directions stand out for future work. First, our algorithm computes the real scheme but not the finer *complex scheme*, which additionally records dividing type and complex orientations of ovals; algorithms were described by De Loera and Wicklin [3] and Kaihnsa et al. [12], but neither invariant seems to have a general-purpose implementation. Second, a central open question is whether degree eight is where the limitations of T-curves become apparent. Our results so far point in this direction: we realize only 30 of 89 maximum real schemes as T-curves, and the gap between the 2359 real schemes found and the (unknown) total may be substantial — a qualitative departure from degrees through seven, where every nonempty real scheme is a T-curve. Our exploration of the patchwork search space is far from exhaustive, however, and

future searches may cover more. Even if T-curves fall short, constructive approaches beyond classical patchworking may provide realizations for some or all of the remaining cases.

Acknowledgments. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – "The Berlin Mathematics Research Center MATH+" (EXC-2046/1, EXC-2046/2, project ID 390685689), "Symbolic Tools in Mathematics and their Application" (TRR 195, project ID 286237555), "Mathematical Modelling, Simulation and Optimization Using the Example of Gas Networks" (SFB/TRR 154, project ID 239904186), "Mathematical Research Data Initiative (MaRDI)" (project ID 460135501) as well as by the German Federal Ministry of Research, Technology and Space (Research Campus MODAL, fund number 05M14ZAM, 05M20ZBM) and the VDI/VDE Innovation + Technik GmbH (fund number 16IS23025B).

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

[2] G. E. Collins. "Quantifier elimination for real closed fields by cylindrical algebraic decomposition". In: *Automata theory and formal languages (Kaiserslautern, 1975)*. Vol. 33. Lecture Notes in Comput. Sci. Springer, Berlin-New York, 1975, pp. 134–183.

[3] J. A. De Loera and F. J. Wicklin. "On the need of convexity in patchworking". In: *Adv. in Appl. Math.* 20.2 (1998), pp. 188–219.

[4] A. Frühbis-Krüger, M. Joswig, and L. Kastner. *Drawing real plane algebraic curves in OSCAR*. 2026. arXiv: `2603.12985`.

[5] Z. Geiselmann, M. Joswig, L. Kastner, K. Mundinger, S. Pokutta, C. Spiegel, M. Wack, and M. Zimmer. "121 Patchworked Curves of Degree Seven". In: *preprint* (2026). arXiv: `2602.06888`.

[6] D. A. Gudkov. "The topology of real projective algebraic varieties". English. In: *Russ. Math. Surv.* 29.4 (1974), pp. 1–79.

[7] A. Harnack. "Ueber die Vieltheiligkeit der ebenen algebraischen Curven". In: *Math. Ann.* 10 (1876), pp. 189–199.

[8] B. El-Hilany, J. Rau, and A. Renaudineau. *Combinatorial Patchworking Tool*. `https://math.uniandes.edu.co/~j.rau/patchworking_english/patchworking.html`. 2017.

[9] D. Hilbert. "Mathematische Probleme". In: *Nachr. Ges. Wiss. Göttingen Math.-Phys. Kl.* 1900 (1900). English translation (M. F. Winston Newson): Bull. Amer. Math. Soc. 8 (1902), 437–479, pp. 253–297.

[10] I. Itenberg. "Counter-examples to Ragsdale conjecture and T-curves". In: *Real algebraic geometry and topology (East Lansing, MI, 1993)*. Amer. Math. Soc., Providence, RI, 1995, pp. 55–72.

[11] M. Joswig and P. Vater. "Real tropical hyperfaces by patchworking in `polymake`". In: *Mathematical software – ICMS 2020*. Ed. by A. M. Bigatti, J. Carette, J. H. Davenport, M. Joswig, and T. de Wolff. Vol. 12097. Lecture Notes in Computer Science. Springer, 2020.

[12] N. Kaihnsa, M. Kummer, D. Plaumann, M. Sayyary Namin, and B. Sturmfels. "Sixty-four curves of degree six". In: *Exp. Math.* 28.2 (2019), pp. 132–150.

[13] M. Kerber and M. Sagraloff. "A worst-case bound for topology computation of algebraic curves". In: *J. Symbolic Comput.* 47.3 (2012), pp. 239–258.

[14] S. Y. Orevkov. "Classification of flexible M-curves of degree 8 up to isotopy". In: *Geom. Funct. Anal.* 12.4 (2002), pp. 723–755.

[15] R. Seidel and N. Wolpert. "On the exact computation of the topology of real algebraic curves". In: *Computational geometry (SCG'05)*. ACM, New York, 2005, pp. 107–115.

[16] R. E. Tarjan. "Efficiency of a Good But Not Linear Set Union Algorithm". In: *Journal of the ACM* 22.2 (1975), pp. 215–225.

[17] O. Y. Viro. "Gluing of plane real algebraic curves and constructions of curves of degrees 6 and 7". In: *Topology (Leningrad, 1982)*. Vol. 1060. Lecture Notes in Math. Springer, Berlin, 1984, pp. 187–200.

[18] O. Viro. "Patchworking real algebraic varieties". In: *preprint* (2006). arXiv: `math/0611382`.

[19] T. de Wolff, E. O. Kwaakwah, and C. O'Neill. *Viro.sage*. `https://cdoneill.sdsu.edu/viro/`. v0.5b, posted Sep 7, 2021. 2021.