Semialgebraic Range Stabbing, Ray Shooting, and Intersection Counting in the Plane

Timothy M. Chan
University of Illinois Urbana-Champaign, Urbana, IL, USA

Pingan Cheng
Aarhus University, Denmark

Da Wei Zheng
University of Illinois Urbana-Champaign, Urbana, IL, USA

Abstract

Polynomial partitioning techniques have recently led to improved geometric data structures for a variety of fundamental problems related to semialgebraic range searching and intersection searching in 3D and higher dimensions (e.g., see [Agarwal, Aronov, Ezra, and Zahl, SoCG 2019; Ezra and Sharir, SoCG 2021; Agarwal, Aronov, Ezra, Katz, and Sharir, SoCG 2022]). They have also led to improved algorithms for *offline* versions of semialgebraic range searching in 2D, via *lens-cutting* [Sharir and Zahl (2017)]. In this paper, we show that these techniques can yield new data structures for a number of other 2D problems even for *online* queries:

1. *Semialgebraic range stabbing.* We present a data structure for n semialgebraic ranges in 2D of constant description complexity with $O(n^{3/2+\varepsilon})$ preprocessing time and space, so that we can count the number of ranges containing a query point in $O(n^{1/4+\varepsilon})$ time, for an arbitrarily small constant $\varepsilon > 0$. (The query time bound is likely close to tight for this space bound.)
2. *Ray shooting amid algebraic arcs.* We present a data structure for n algebraic arcs in 2D of constant description complexity with $O(n^{3/2+\varepsilon})$ preprocessing time and space, so that we can find the first arc hit by a query (straight-line) ray in $O(n^{1/4+\varepsilon})$ time. (The query bound is again likely close to tight for this space bound, and they improve a result by Ezra and Sharir with near $n^{3/2}$ space and near $\sqrt{n}$ query time.)
3. *Intersection counting amid algebraic arcs.* We present a data structure for n algebraic arcs in 2D of constant description complexity with $O(n^{3/2+\varepsilon})$ preprocessing time and space, so that we can count the number of intersection points with a query algebraic arc of constant description complexity in $O(n^{1/2+\varepsilon})$ time. In particular, this implies an $O(n^{3/2+\varepsilon})$-time algorithm for counting intersections between two sets of n algebraic arcs in 2D. (This generalizes a classical $O(n^{3/2+\varepsilon})$-time algorithm for circular arcs by Agarwal and Sharir from SoCG 1991.)

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Computational geometry, range searching, intersection searching, semialgebraic sets, data structures, polynomial partitioning

Digital Object Identifier 10.4230/LIPIcs.SoCG.2024.33

Related Version *Full Version*: https://arxiv.org/abs/2403.12303

Funding *Timothy M. Chan*: Work supported by NSF Grant CCF-2224271.
Pingan Cheng: Work supported by DFF (Det Frie Forskningsråd) of Danish Council for Independent Research under grant ID DFF–7014–00404 and STIBOFONDENs IT-rejsestipendier til ph.d.-studerende during the author's visit to UIUC.

1 Introduction

The polynomial partitioning technique [36, 35] has led to a series of breakthroughs of many long-standing classic problems in computational geometry e.g., range searching [11, 41, 8], range stabbing [8], intersection searching [31, 32, 7], etc, and simplification and generalization of many existing techniques and tools [37]. Comparing to rather simple geometric objects formed by halfspaces or hyperplanes that have been studied extensively in the early days of computational geometry, polynomial partitioning enables us to attain similar results for semialgebraic sets (a set obtained by union, intersection, and complement from a set of a collection of polynomial inequalities where the number of polynomials, the number of indeterminates, and the degree of polynomials are constant). Almost all of these breakthrough results are for problems in three or higher dimensions. We complement these breakthroughs with some new results for fundamental problems involving algebraic curves in the plane.

1.1 Problems studied and related results

We consider the following three problems in this paper.

Semialgebraic range stabbing. In this problem we are given a collection of semialgebraic sets of constant complexity in $\mathbb{R}^2$ as the input, and we want to preprocess them in a data structure so that we can quickly count or report the inputs intersected or "stabbed" by a query point (this is called a "range stabbing query", also known as a "point enclosure query"). Generalizing counting, we can also consider the *semigroup model*, where every semialgebraic set is given a value in a semigroup, and we wish to apply the semigroup operation on the values of all sets stabbed. Semialgebraic range stabbing and its "dual" problem, semialgebraic range searching, are among the most classical problems in computational geometry. The two problems are relatively well-understood for linear ranges after a decade of study by pioneers in the fields in late 80s and early 90s. We refer the readers to a survey of this topic [5]. The tools and results developed for the problems have also become textbook results [29].

However, when considering general polynomial inequalities, the problem is more difficult. Before the invention of polynomial partitioning [36, 35], there was a lack of suitable tools and few tight results were known [3]. It was only very recently [11, 41, 8], that via polynomial partitioning, efficient data structures for the two problems were found for data structures with small (near-linear) space, and data structures with very fast (polylogarithmic) query time. By interpolating the two extreme solutions, we obtain space-time trade-offs. However, somewhat mysteriously, even if the extreme cases are almost tight, it is unknown whether the trade-off is close to optimal. For example, even for the planar annulus stabbing, there is a clear gap between the current upper bound[1] of $S(n) = O^*(n^2/Q(n)^{3/2})$ and the lower bound of $S(n) = \Omega^*(n^{3/2}/Q(n)^{3/4})$ [2] or $S(n) = \widetilde{\Omega}(n^2/Q(n)^2)$ [1], where $S(n)$ and $Q(n)$ denote space and query time respectively.

We mention that sometimes it is possible to solve certain range searching problems involving algebraic arcs more efficiently. For example, Agarwal and Sharir [14] gave improved algorithms for counting containment pairs between points and circular disks in $\mathbb{R}^2$, which can be viewed as an off-line version of either circular range searching or range stabbing. To get this improvement, they used a key technique known as "lens cutting" to cut planar curves into pseudo-segments. This allows us to use some of the classic tools developed for linear objects which are usually more efficient than their polynomial counterparts. However, to define the dual of pseudo-line or pseudo-segment arrangements, we need to know all the input and query objects in advance; that is the main reason why previous applications are restricted to offline settings. There were attempts to apply this technique to online problems [34] but to our knowledge they have not been generally successful.

[1] In this paper, we use the notation $O^*(\cdot)$ or $\Omega^*(\cdot)$ to hide factors of n^ε where $\varepsilon > 0$ is an arbitrary small constant. We use the notation $\widetilde{O}(\cdot)$ or $\widetilde{\Omega}(\cdot)$ to hide factors polylogarithmic in n.

Ray shooting amid algebraic arcs. We consider the problem of ray shooting where we are given a collection of algebraic arcs (of constant complexity) in $\mathbb{R}^2$ as the input, and we want to build a structure such that for any query (straight-line) ray, we can find the first arc intersecting it or assert that no such arc exists. Ray shooting is another classic problem in computational geometry with many applications in other fields such as computer graphics and robotics. Early study of ray shooting mostly centered around special cases, e.g., the input consists of line segments [4, 10], circular arcs [15], or disjoint arcs [15]. Specifically, for ray shooting queries amid line segments, it is possible to obtain a trade-off of $S(n) = O^*(n^2/Q(n)^2)$, which has been conjectured to be close to be optimal. For general algebraic curve inputs, it is possible to build an $O^*(n^2)$ space data structure with $O(\log n)$ query time in time $O^*(n^2)$ [38]. Combining the standard linear-space $O(n^{1-1/\beta})$-query time structure, we can interpolate and get a space-time trade-off curve of $S(n) = O^*(n^2/Q(n)^{\beta/(\beta-1)})$, where β is the number of parameters needed to define any polynomial in the semialgebraic sets (for bivariate polynomials of degree deg, we have $\beta \leq \binom{\mathsf{deg}+2}{2} - 1$, but in general β is often much smaller). Very recently, Ezra and Sharir [31] showed how to answer ray shooting queries for algebraic curves of constant complexity in $\mathbb{R}^2$ with $O^*(n^{3/2})$ space and $O^*(n^{1/2})$ query time, where the exponent is independent of β. Note that this gives better $O^*(n^{3/2})$-space data structures for all $\beta > 3$.

Intersection counting amid algebraic arcs. Finally, we consider intersection counting amid algebraic arcs in $\mathbb{R}^2$ – more precisely, computing the sum of the number of intersections between pairs of algebraic arcs. We show new results for both online and offline versions of the problem. For the online version where we want to build data structures to count intersections with a query object, it is known that when the query object is a line segment, a structure of space-time trade-off of $S(n) = O^*(n^2/Q(n)^{3/2})$ (resp. $S(n) = O^*(n^2/Q(n)^{\beta/(\beta-1)})$) is possible for circular arcs (resp. general algebraic arcs) [38] in the plane. To the best of our knowledge, the more general problem of algebraic arc-arc intersection counting has not been studied for offline intersection counting where we are given a collection of algebraic arcs and want to count the number of intersections points. When the input consists of circular arcs, there is an $O^*(n^{3/2})$-time algorithm for the problem [13]. For more general arcs, it is unclear if any subquadratic algorithm with exponent independent of β exists.

1.2 New results

We present improved results for these three basic problems in 2D computational geometry.

Semialgebraic range stabbing. We give a data structure with $O^*(n^{3/2})$ preprocessing time and space and $O^*(n^{1/4})$ query time for semialgebraic range stabbing in $\mathbb{R}^2$. (This holds for counting as well as the semigroup model; for reporting, we add an $O(k)$ term to the query time where k is the output size.) Interestingly, the exponents here are independent of the number β of parameters needed to define the algebraic curves (similar phenomena have recently been seen for certain problems in 3 and higher dimensions [31, 32]). The result matches known offline results (namely, a batch of queries with $n^{5/4}$ ranges on n points take $O^*(n^{3/2})$ total time [13, 8]). By interpolating with existing results, we also automatically get an improved trade-off curve for the (online) problem. In particular, when the query time is at most $n^{1/4}$, we obtain a space-time trade-off of $S(n)Q(n)^2 = O^*(n^2)$. See Figure 1 for an illustration of the trade-off curve. Note that it almost matches the curve for simplex range stabbing, and is thus likely almost optimal, in this regime. Prior to our result, online data structures matching this trade-off are known only when the query time is very small (polylogarithmic), by constructing the entire $O(n^2)$-sized arrangement of the ranges.

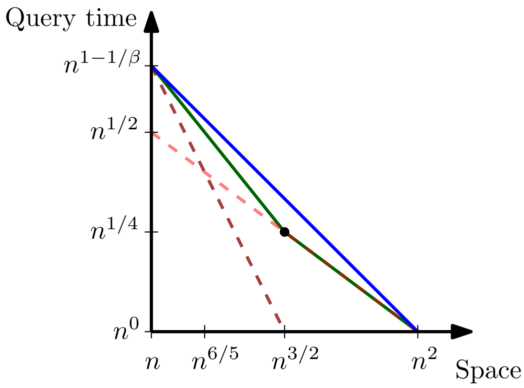

Figure 1 The blue line shows the prior known trade-off curve for semialgebraic range stabbing, and the green curve shows the improved trade-off curve we obtain. The dotted red lines show the lower bounds of Afshani [1] for simplex stabbing and Afshani and Cheng [2] for semialgebraic range stabbing, both of which apply.

Ray shooting amid algebraic curves. We present a data structure with $O^*(n^{3/2})$ preprocessing time and space that is able to answer ray shooting queries in time $O^*(n^{1/4})$, improving the $O^*(n^{1/2})$ query time of the previous best structure by Ezra and Sharir [31].[2] This again allows us to show a space-time trade-off that matches the one for ray shooting amid line segments when the query time satisfies $Q(n) = O^*(n^{1/4})$. Again, prior to our result, the trade-offs for the two problems only roughly match for polylogarithmic query time structures.

Intersection counting amid algebraic arcs. For the online version where we need to preprocess a collection of algebraic arcs so that we can count the intersection among them and a query algebraic arc, we give a structure with $O^*(n^{3/2})$ preprocessing time and space and $O^*(n^{1/2})$ query time. Prior to our work, such structure is only known for when the query is a line segment instead of an algebraic arc. A straightfoward application of our online result immediately gives an $O^*(n^{3/2})$ time algorithm for the offline problem of counting the intersections of n algebraic arcs. This generalizes a known result for circular arcs [13], as well as the line segment-arc intersection detection result by Ezra and Sharir [32].

An interesting combinatorial consequence of our algorithm is that intersection graphs of algebraic arcs in $\mathbb{R}^2$ admit *biclique covers* [6, 33] of size $O^*(n^{3/2})$; it is again surprising that the exponent here is independent of β. Biclique covers have many applications to algorithmic problems about geometric intersection graphs.

Concurrent work. In an independent work, Agarwal, Ezra, and Sharir [9] showed that offline semigroup range searching with m semialgebraic ranges with β degrees of freedom and n points in $\mathbb{R}^2$ can be solved in time $O^*(m^{\frac{2\beta}{5\beta-4}} n^{\frac{5\beta-6}{5\beta-4}} + m^{2/3}n^{2/3} + m + n)$. Furthermore, they show how to compute a biclique partition of the incidence graph between the semialgebraic sets and the points. We remark that the trade-offs we get for *online* semialgebraic range stabbing directly imply both of their results.

---

[2] To be fair, Ezra and Sharir's paper mainly focused on 3D versions of the ray shooting problem, and their 2D data structure was just one ingredient needed. However, they did consider their 2D result to be "of independent interest".

2 Semialgebraic Range Stabbing

Let Γ be a set of n semialgebraic ranges in $\mathbb{R}^2$ where the boundary of each range consists of $O(1)$ algebraic arcs of degree at most $\mathsf{deg} = O(1)$. In this section, we present data structures to count the ranges stabbed by a query point.

2.1 Preliminaries

We begin by reviewing known techniques for handling stabbing problems. One approach is by using $(1/r)$-cuttings [28, 24, 25].

▶ **Lemma 1** ($(1/r)$-Cutting Lemma). *Given n x-monotone algebraic arcs of constant degree in $\mathbb{R}^2$ and a parameter $r \le n$, there exists a decomposition of the plane into $O(r^2)$ disjoint pseudo-trapezoid cells such that each cell is crossed by at most n/r arcs.*

The cells, the list of arcs crossing each cell, and the number of arcs completely below each cell, can all be computed in $O(nr)$ time.

Another method is based on the simplicial partition theorem:

▶ **Theorem 2** (Matoušek's Partition Theorem [40]). *Let P be a set of n points in $\mathbb{R}^d$. Then for any $r \le n$, we can partition P into r disjoint simplicial cells such that each cell contains $O(n/r)$ points and any hyperplane crosses at most $O(r^{1-1/d})$ cells. This partition can be computed in $O(n)$ when r is a constant.*

To get a linear-space data structure, one approach is to lift the input curves to a halfspace in dimension $L = \binom{\mathsf{deg}+2}{2} - 1$, and then apply the partition theorem recursively with constant r in the dual to get a data structure with $O^*(n)$ space and preprocessing time and $O^*(n^{1-1/L})$ query time (the extra $O^*(1)$ factors can be lowered or removed [21]). The query time bound can be improved to $O^*(n^{1-1/\beta})$ (recall that β is the number of parameters needed to specify the curves), by using an analog of the partition theorem for semialgebraic ranges for $\beta \le 4$ [3, 38], or the polynomial partitioning method for general constant β [11, 41].

Still better results are possible if we have more guarantees about the behavior of the arcs of Γ. The key case we will consider is when the arcs form a set of *pseudo-lines* (x-monotone curves, from $x = -\infty$ to $x = \infty$, that pairwise intersect at most once), or *pseudo-segments* (x-monotone arcs that pairwise intersect at most once).

It turns out that the general case can be reduced to the pseudo-line or pseudo-segment case by a technique known as *lens cutting*, first proposed by Tamaki and Tokuyama [44] and further developed by others [12, 20, 39]. We use the following theorem of Sharir and Zahl [43] for cutting algebraic curves into pseudo-segments. The algorithmic version is due to Agarwal, Aronov, Ezra, and Zahl [8].

▶ **Theorem 3** (Lens cutting for algebraic curves). *Given a collection Γ of n curves generated from constant-degree bivariate polynomials where no pair of polynomials shares a common factor, we can cut Γ into a collection of $O^*(n^{3/2})$ subarcs such that each pair of arcs intersect at most once.[3] Furthermore, this can be computed in $O^*(n^{3/2})$ time.*

[3] If a pair of arcs γ_1 and γ_2 intersect more than once, the part of the two arcs between two consecutive intersections is sometimes referred to as a *lens*. Hence, the problem of cutting curves into pseudo-segments is also called *lens cutting*.

Sharir and Zahl's theorem is striking in that it gives the first subquadratic bound for general algebraic arcs (previous results were for pseudo-parabolas [39] or graphs of univariate polynomials [20]), and at the same time, achieves an exponent (3/2) completely independent of the degree of the arcs! By lens cutting, we can thus turn our attention to solving the stabbing problem for ranges defined by pseudo-lines or pseudo-segments.

2.2 Counting pseudo-lines below a query point

We present our main new data structure for pseudo-lines below:

▶ **Theorem 4.** *Given a set Γ of n pseudo-linesin $\mathbb{R}^2$, there is a data structure for counting the number of pseudo-lines below a query point with $O^*(n)$ preprocessing time and space and $O^*(\sqrt{n})$ query time.*

One approach to proving this theorem is via "spanning trees with low crossing number" [45, 46]. Chazelle and Welzl [27] actually showed that such spanning trees can yield range searching data structures in a general bounded-VC-dimension setting; our problem fits their framework, and so we can immediately obtain a data structure with $O(\text{poly}(n))$ preprocessing time, $\widetilde{O}(n)$ space, and $\widetilde{O}(\sqrt{n})$ query time for our problem. We won't discuss this any further as it will be subsumed by our new approach which has much better preprocessing time and also has the advantage of supporting multi-level data structures (needed in our applications later).

Instead, our approach is based on dualizing Matoušek's partition theorem. Recall that a standard way to solve the problem of counting lines (not pseudo-lines) below a query point is to apply point/line duality to reduce the problem to counting points above a query line (i.e., halfplane range searching), which can then be solved using Matoušek's partition tree. Agarwal and Sharir [14] showed that there exists a similar duality between points and pseudo-lines. However, this duality transform is only applicable when we know all the query points in advance – we can't dualize a new query point without potentially changing the entire transform. Nonetheless, we have found a way to overcome this issue.

We say that a point p *crosses* S if there is at least one pseudo-line in S above p, and at least one pseudo-line in S below p. It turns out the right way to reformulate Matoušek's partition theorem in the dual is the following, whose proof requires several delicate steps:

▶ **Theorem 5.** *Given a set Γ of n pseudo-lines in $\mathbb{R}^2$ and a parameter $r \le n$, there exists a partition of Γ into r disjoint subsets $\Gamma_1, \dots, \Gamma_r$ each of size $\Theta(n/r)$, such that any point crosses at most $O(\sqrt{r})$ of these subsets. Furthermore, this partition can be computed in $O(nr^{O(1)})$ time.*

Proof. We start with a version of Matoušek's partition theorem, which follows directly from his original proof [40] (see also the generalization in [3, Lemma 5.2]):

(I) Given a set P of n points in $\mathbb{R}^2$, a set Q of t "test" pseudo-lines, and a parameter $r \le n$, there exists a partition of P into r disjoint subsets $P_1, \dots, P_r$ each of size $\Theta(n/r)$, together with r (pseudo-trapezoidal) cells $\Delta_1, \dots, \Delta_r$ with $P_i \subset \Delta_i$, such that any pseudo-line in Q intersects at most $O(\sqrt{r} + \log t)$ of the cells.

Next, we state a version that does not involve the cells Δ_i (this will be crucial, as it would be difficult to dualize Δ_i). Say that a pseudo-line γ *crosses* a point set P if γ is above at least one point of P and below at least one point of P.

(II) Given a set P of n points in $\mathbb{R}^2$, a set Q of t "test" pseudo-lines, and a parameter $r \le n$, there exists a partition of P into r disjoint subsets $P_1, \dots, P_r$ each of size $\Theta(n/r)$, such that any pseudo-line in Q crosses at most $O(\sqrt{r} + \log t)$ of the subsets.

Observe that (II) follows from (I), since $P_i \subset \Delta_i$ implies that any pseudo-line crossing P_i must intersect Δ_i.

Now, we apply the point/pseudo-line duality transform by Agarwal and Sharir [14], which turns (II) into the following statement:

(III) Given a set Γ of n pseudo-lines, a set M of t "test" points, and a parameter $r \le n$, there exists a partition of Γ into r disjoint subsets $\Gamma_1, \ldots, \Gamma_r$ each of size $\Theta(n/r)$, such that any point in M crosses at most $O(\sqrt{r} + \log t)$ of the subsets.

The construction time for the partition in (I), and thus (II), is naively bounded by $O(n(rt)^{O(1)})$ from Matoušek's work [40]. Unfortunately, the construction time for (III) is larger, since Agarwal and Sharir's duality transform requires $O((nt)^{O(1)})$ time to compute [14] (they obtained faster algorithms only under certain restricted settings).

We describe a way to speed up the construction for (III). Say that two pseudo-lines γ and γ' are *equivalent* with respect to M if the subset of points of M below γ is identical to the subset of points of M below γ'. The problem of computing the equivalence classes of pseudo-lines with respect to a point set has luckily already been addressed in a paper by Chan [22, Sections 2.1–2.2] (which studied a seemingly unrelated problem: selection in totally monotone matrices). Chan observed that the number of equivalence classes is $O(t^2)$ (this follows either by using Agarwal and Sharir's duality transform to reduce to counting cells in the dual arrangement, or by direct VC dimension arguments), and he presented a simple deterministic $\widetilde{O}(n + t^3)$-time algorithm and a simple randomized $\widetilde{O}(n + t^2)$-time algorithm (by incrementally adding points of M one by one and splitting equivalence classes using dynamic data structures for lower/upper envelopes of pseudo-lines).

Afterwards, we can replace each pseudo-line with a representative member of its equivalence class. As a result, we get a multi-set Γ' of size n that has only $O(t^2)$ distinct pseudo-lines. We apply Agarwal and Sharir's duality transform to Γ' and M, which now takes only $t^{O(1)}$ time. We obtain a partition satisfying (III) for Γ', which is automatically a partition satisfying (III) for Γ by the definition of equivalence. The overall construction time is $O(n(rt)^{O(1)})$.

Finally, we construct an appropriate (small) test set M to establish our theorem. The idea is similar in spirit to Matoušek's "test set lemma" [40] (though his lemma is not directly applicable here). We first compute a $(1/(cr))$-cutting of Γ with $O(r^2)$ cells in $O(nr^{O(1)})$ time for a sufficiently large constant c; each cell is a pseudo-trapezoid, with two vertical sides and the upper/lower sides being sub-segments of the given pseudo-lines. We just define M to be the set of all vertices of these cells, with $t = |M| = O(r^2)$, and construct the partition in (III) for this test set M in $O(n(rt)^{O(1)}) = O(nr^{O(1)})$ time.

Consider an arbitrary point $q \in \mathbb{R}^2$. Let Δ be the pseudo-trapezoid cell containing q, with top-left vertex v_{TL}, bottom-left vertex v_{BL}, top-right vertex v_{TR}, and bottom-right vertex v_{BR}. Consider one subset Γ_i. Suppose that none of $v_{TL}, v_{BL}, v_{TR}, v_{BR}$ crosses Γ_i. We prove that q cannot cross Γ_i:

- Case 1: all pseudo-lines in Γ_i are between v_{TL} and v_{BL}. Then all pseudo-lines in Γ_i intersect Δ, and so $|\Gamma_i| \le n/(cr)$, which is a contradiction if we choose c large enough (compared to the hidden constant in the $\Theta(n/r)$ bound).
- Case 2: all pseudo-lines in Γ_i are above v_{TL}. If all pseudo-lines in Γ_i are also below v_{TR}, then all pseudo-lines in Γ_i intersect Δ and we again get a contradiction as in Case 1. Thus, we may assume that all pseudo-lines in Γ_i are above both v_{TL} and v_{TR}. But then all pseudo-lines in Γ_i are completely above Δ (since no pseudo-line can intersect the upper side twice), and so q cannot cross Γ_i.
- Case 3: all pseudo-lines in Γ_i are below v_{BL}. Similar to Case 2.

We conclude that the subsets Γ_i crossed by q must be crossed by one of the test points $v_{TL}, v_{BL}, v_{TR}, v_{BR}$, and so there are at most $O(4 \cdot (\sqrt{r} + \log t)) = O(\sqrt{r})$ such subsets. ◀

Proof of Theorem 4. We construct a partition for Γ by Theorem 5. For each subset Γ_i, we store its upper and lower envelopes (which, for pseudo-lines, have $O(n)$ complexity and can be constructed in $O(n \log n)$ time, e.g., by a variant of Graham's scan [29]). We recursively build the data structure for each Γ_i.

Given a query point q, we examine each subset Γ_i. If q is below the lower envelope of Γ_i (which we can check by binary search in $\widetilde{O}(1)$ time), we ignore Γ_i. If q is above the upper envelope of Γ_i, we add $|\Gamma_i|$ to the current count. Otherwise, q crosses Γ_i, and we recursively query Γ_i.

Let $P(n)$ and $Q(n)$ be the preprocessing time and query time of the data structure (space is bounded by the preprocessing time). They satisfy the following recurrence relations:

$$\begin{aligned} P(n) &= O(r) \cdot P(n/r) + \widetilde{O}(r^{O(1)} n) \\ Q(n) &= O(\sqrt{r}) \cdot Q(n/r) + \widetilde{O}(r). \end{aligned}$$

Setting r to be a large enough constant, we obtain $P(n) = O^*(n)$ and $Q(n) = O^*(\sqrt{n})$. ◀

By using a segment tree [29], we can easily extend Theorem 4 to handle pseudo-segments:

▶ **Corollary 6.** *Given a set Γ of n pseudo-segments in $\mathbb{R}^2$, there is a data structure for counting the number of pseudo-segments below a query point with $O^*(n)$ preprocessing time and space and $O^*(\sqrt{n})$ query time.*

2.3 Semialgebraic range stabbing counting

▶ **Definition 7.** *For an integer $j \geq 0$ we say that Γ is a set of $(j, \text{ALGEBRAIC})$-ranges if each range is being bounded above/below by j different x-monotone algebraic curves and at most two vertical sides. We say that Γ is a set of $(j, \text{PSEUDOSEG})$-ranges if furthermore, these j curves are pseudo-segments.*

For any set of n semialgebraic ranges, we can decompose each range vertically with $O(1)$ cuts so that we get a set of $O(n)$ many $(2, \text{ALGEBRAIC})$-ranges. For counting, it suffices to look at a set of $(1, \text{ALGEBRAIC})$-ranges with only lower bounding arcs, since we can use subtractionto express a range bounded from above and below by the difference of two ranges bounded from below.

We reduce $(1, \text{ALGEBRAIC})$-range stabbing to $(1, \text{PSEUDOSEG})$-range stabbing by lens cutting. Naively replacing n by $O^*(n^{3/2})$ would yield terrible space and query bounds. Past applications of lens cutting [12, 16, 20] first derived intersection-sensitive results for pseudo-segments, and noticed that the lens-cutting operation does not increase the number of intersections. Below, we describe a direct reduction bypassing intersection-sensitive bounds:

▶ **Theorem 8.** *There is a data structure for range stabbing counting on n semialgebraic ranges of constant complexity in $\mathbb{R}^2$ with $O^*(n^{3/2})$ preprocessing time and space and $O^*(n^{1/4})$ query time.*

Proof. Let Γ be a set of n lower arcs (extended with upward vertical rays at their endpoints) of the input ranges. Compute a set of μ cut points that turn their lower arcs into pseudo-segments. We have $\mu = O^*(n^{3/2})$ by Theorem 3. Compute a $(1/r)$-cutting Ξ of Γ with $O(r^2)$ cells by Lemma 1. Add extra vertical cuts to ensure that each cell contains at most μ/r^2 cut points; the number of cells remains $O(r^2)$. For each cell $\Delta \in \Xi$, let Γ_Δ be the arcs in Γ intersecting Δ (we know $|\Gamma_\Delta| \leq n/r$); build the data structure in Corollary 6 for the $O(n/r+\mu/r^2)$ pseudo-segments along the arcs in Γ_Δ inside Δ. Let c_Δ be the number of arcs in Γ completely below Δ. The preprocessing time/space is $O^*(nr+r^2 \cdot (n/r+\mu/r^2)) = O^*(nr+\mu)$.

To answer a query for a point q, we find the cell Δ containing q in $\widetilde{O}(1)$ time by point location [29], query the data structure for the pseudo-segments inside Δ, and add c_Δ to the current count. The query time $O^*(1 + \sqrt{n/r + \mu/r^2})$. Setting $r = \lceil \mu/n \rceil$ gives preprocessing time/space $O^*(n + \mu)$ and query time $O^*(n/\sqrt{\mu})$. ◀

By standard techniques, we can use this to obtain improvement on the entire trade-off curve between space and query time. (See the full paper.) For semialgebraic stabbing reporting, we can no longer use subtraction and need to consider $(2, \textsc{algebraic})$-ranges. Instead we can use multi-level data structures. This procedure is not straightforward, as we do not have a smooth tradeoff curve. Details are given in the full paper.

3 Ray Shooting Amid Curves

As an application of our range stabbing result, we describe an algorithm for ray shooting amid curves. Let Γ be a collection of n algebraic arcs of degree at most $\mathsf{deg} = O(1)$. By breaking each arc into a constant number of subarcs, we may assume each arc is x-monotone and either convex or concave. W.l.o.g., we assume all arcs are convex.

By parametric search [10, 42] or randomized search [19], it suffices to focus on the problem of detecting if a query line segment intersects any of the input arcs. We will present a data structure that can more generally count the number of intersections between a query line segment $\overline{pq}$ and the input arcs. With subtraction, it suffices to count the number of intersections between the arcs and two query rightward rays (a ray emanating from p and a ray emanating from q). To do this, we begin by building a segment tree structure [29, 26] on the input arcs. It suffices to focus on solving the intersection counting problem in these two cases in a slab σ of the segment tree: (i) "long-short intersections" when the query rays span the entire slab but the input arcs may not, and (ii) "short-long intersections" when the input arcs span the entire slab but the query rays may not. All other intersections may be handled by recursion.

Due to space limitation, we defer case (i) to the full paper and focus on just case (ii), which is technically more challenging. Here, the input arcs span the entire x-range $[x_1, x_2]$ of the slab σ. We may assume that each input arc γ has endpoints u and v with $u_x = x_1$ and $x_v = x_2$, and by clipping, the query ray becomes a query line segment $\overline{pq}$ with $q_x = x_2$. Let ℓ be the line extension of $\overline{pq}$.

It is not difficult to count the number of arcs γ that intersect the query segment $\overline{pq}$ *exactly once*; see the full paper for details. The more difficult subproblem is how to count the number of arcs γ that intersect the query segment $\overline{pq}$ *exactly twice*. We observe that this happens iff the following four conditions are simultaneously true (this is similar to an observation by Ezra and Sharir [32]):

- (i) $q_y < v_y$, and (ii) p is below γ, and (iii) ℓ intersects $\kappa(\gamma)$, and (iv) the slope of γ at p_x is less than the slope of ℓ. (See Figure 2.)

Condition (i) is easy to handle. This leaves three conditions remaining that we claim all correspond to semialgebraic stabbing problems. Condition (ii) corresponds to a stabbing problem in the *original space*. Condition (iii) corresponds to a stabbing problem in the *dual space*, as it is equivalent to the condition that the dual point ℓ^* lies in the dual region $\kappa^*(\gamma)$ (see the full paper). The final condition (iv) corresponds to a stabbing problem in the *tangent space*. If we let $y = f(x)$ be the equation defining the arc γ for an algebraic function f, and let $y = mx + b$ be the equation defining the line ℓ, this problem in the tangent space is equivalent to the condition that the point (p_x, m) lies above the curve: $\partial\gamma := \{\left(x, \frac{df}{dx}(x)\right) : x_1 \le x \le x_2\}$. This is an algebraic curve of degree at most $O(\mathsf{deg}^2)$ [18].

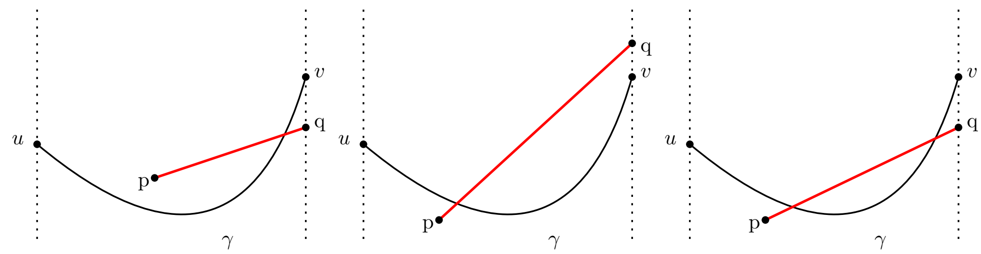

Figure 2 Three different ways a ray ρ can intersect an arc γ. The left depicts Case A, the middle depicts Case B, and the right depicts Case C.

Thus the whole problem is similar to a $(3, \text{ALGEBRAIC})$-range stabbing problem, but as all cut points can be mapped back to the original curve, we can still handle these queries despite not directly having multi-level data structures. See the full paper.

▶ **Theorem 9.** *Given n algebraic arcs of constant complexity in $\mathbb{R}^2$, there is a data structure with $O^*(n^{3/2})$ preprocessing time and space that can count intersections with a query line segment in $O^*(n^{1/4})$ time. Consequently, there is a data structure for ray shooting amid n algebraic arcs of constant complexity in $\mathbb{R}^2$ with the same bounds.*

4 Intersection Counting Amid Algebraic Arcs

Let Γ be a set of n algebraic arcs in $\mathbb{R}^2$ of degree at most $\mathsf{deg} = O(1)$. In this section we present algorithms and data structures for counting the number of intersections between the arcs of Γ. By the "number of intersections", we always mean the number of intersection points (possibly with multiplicities if we have degeneracies/tangencies), and not the number of intersecting pairs.W.l.o.g., we assume that the arcs are x-monotone.

4.1 A first approach

To better appreciate our final algorithm, we first sketch a slower but still subquadratic algorithm for the offline problem of counting intersections among n algebraic arcs in $\mathbb{R}^2$, by using the lens cutting routine of Theorem 3 as a black box as we did in earlier sections. Let r be a parameter to be chosen later.

1. Compute a $(1/r)$-cutting Ξ of Γ into $O(r^2)$ disjoint cells each intersecting n/r arcs.
2. Compute a set P of $\mu = O^*(n^{3/2})$ points to chop the arcs of Γ into pseudo-segments by Theorem 3, in $O^*(n^{3/2})$ time. Refine the cutting Ξ by adding vertical line segments so that each cell of the cutting contains at most μ/r^2 points of P; the number of cells remain $O(r^2)$. The number of pseudo-segments in each cell is bounded by the number of arcs intersecting each cell and the number of points of P in the cell, which is $m = O(n/r + \mu/r^2)$.
3. In each cell, use an $O^*(m^{4/3})$-time algorithm to count the number of intersections between pseudo-segments (algorithms are known [40, 23] for counting intersections between line segments in near $m^{4/3}$ time, and they can be adapted to pseudo-segments as well, but we will not elaborate as we will present a better algorithm shortly).

The total run time is thus $O^*(r^2 \cdot (n/r + \mu/r^2)^{4/3})$. Choosing $r = \lceil \mu/n \rceil$ yields a time bound of $O^*(n + n^{2/3}\mu^{2/3}) = O^*(n^{5/3})$.

We will next show how to improve the running time for algebraic arcs to $O^*(n^{3/2})$, by opening the black box of Theorem 3 and directly modifying the algorithm for cutting algebraic arcs into pseudo-segments. Remarkably, this algorithm naturally extends to a data structure for intersection counting.

4.2 Review of lens cutting

As a preliminary, we sketch the proof of Sharir and Zahl [43] for Theorem 3. The key idea is to transform the 2D lens cutting problem into a 3D problem about eliminating depth cycles, which was already solved before by Aronov and Sharir [17] using polynomial partitioning techniques. Let Γ be a set of n algebraic x-monotone plane arcs each of degree at most $\mathsf{deg} = O(1)$. For any arc $\gamma \in \Gamma$ of the form $\{(x, f(x)) : x_1 \leq x \leq x_2\}$ for some algebraic function f and $x_1, x_2 \in \mathbb{R}$, we *lift* γ to a new arc in 3D:

$$\widehat{\gamma} = \{\Big(x, f(x), \tfrac{df}{dx}(x)\Big) : x_1 \leq x \leq x_2\}.$$

In other words, the xy-projection of $\widehat{\gamma}$ is γ, and the z-coordinate corresponds to the slope of the curve. The arc $\widehat{\gamma}$ is algebraic, with degree at most deg^2 (see Lemma 2.5 of [43] or Proposition 1 of [30] for precise details). Let $\widehat{\Gamma}$ denote the set of these arcs in $\mathbb{R}^3$.

To eliminate depth cycles in $\mathbb{R}^3$, Aronov and Sharir [17] proceeded by computing a *polynomial partition* of $\widehat{\Gamma}$, i.e., a polynomial $P(x, y, z)$ whose zero set $Z(P) := \{(x, y, z) \in \mathbb{R}^3 : P(x, y, z) = 0\}$ separates $\mathbb{R}^3$ into cells such that not too many arcs of $\widehat{\Gamma}$ intersect each cell. This was proved to exist by Guth [35] and the construction was made algorithmic by Agarwal, Aronov, Ezra, and Zahl [8]. The theorem applies to general varieties in any dimension, but we will present it specialized to curves in $\mathbb{R}^3$.

▶ **Theorem 10** (Polynomial partitioning of curves in $\mathbb{R}^3$)**.** *Let Γ be a collection of n algebraic arcs in $\mathbb{R}^3$ each of which has degree at most $\mathsf{deg} = O(1)$. Then for any $D \geq 1$ there is a non-zero polynomial P of degree at most D such that $\mathbb{R}^3 \setminus Z(P)$ contains $O(D^3)$ cells and each cell crosses at most $O(n/D^2)$ arcs of Γ.*

The polynomial P and the semi-algebraic representation of every cell $\mathbb{R}^3 \setminus Z(P)$ can be constructed in $O(2^{\text{poly}(D)})$ randomized expected time. Furthermore this representation of the cells has size $O(\text{poly}(D))$, and given any algebraic arc, we can output the cells of $\mathbb{R}^3 \setminus Z(P)$ that it crosses (or that it lies completely within $Z(P)$) in $O(\text{poly}(D))$ time. In particular, we can compute the set of arcs intersecting every cell of $\mathbb{R}^3 \setminus Z(P)$ in $O(n \,\text{poly}(D))$ time.

We proceed next by cutting each curve of $\widehat{\Gamma}$ at its intersection points with the zero set $Z(P)$ of a partitioning polynomial P of degree D. We further cut each curve at its intersection points with another surface Z_{bad} which is the vertical cylinder passing through all points with vertical tangency at $Z(P)$ (this is also a zero set of a polynomial, of degree $O(D^2)$). For any point z, let $h(z)$ denote the number of times a vertical downward ray emanating from z intersects $Z(P)$. Then points z in the same cell of $\mathbb{R}^3 \setminus (Z(P) \cup Z_{bad})$ have the same $h(z)$ value, by our definition of Z_{bad}.

The key observation is that two curves $\gamma_1, \gamma_2 \in \Gamma$ in 2D intersect twice if and only if their corresponding curves $\widehat{\gamma}_1, \widehat{\gamma}_2 \in \widehat{\Gamma}$ in 3D form a length-2 *depth cycle* in the z-direction, i.e., there are four points $(x, y, z_1), (x', y', z_1') \in \widehat{\gamma}_1$, and $(x, y, z_2), (x', y', z_2') \in \widehat{\gamma}_2$ where $z_1 > z_2$ and $z_1' < z_2'$, or vice versa. (This is because in 2D, at two consecutive intersection points between γ_1 and γ_2, the slope of γ_1 is larger than the slope of γ_2 at one point, and vice versa at the other point.)

Recall that we have cut the arcs at intersections with $Z(P)$ as well as Z_{bad}. Suppose a subarc of $\widehat{\gamma}_1$ is contained in a cell Δ of $\mathbb{R}^3 \setminus Z(P)$, and suppose a subarc of $\widehat{\gamma}_2$ is not contained in the same cell Δ. We observe that the two subarcs cannot form a length-2 depth cycle. This is because otherwise, $h(z_1) > h(z_2)$ and $h(z_1') < h(z_2')$, or vice versa, which is a contradiction.

Thus, it suffices to eliminate length-2 depth cycles for pairs of arcs contained in the same cell Δ of $\mathbb{R}^3 \setminus Z(P)$; this can be handled by recursion in each cell. Arcs contained in $Z(P)$ or Z_{bad} can be handled naively as there can only be $O(D^2)$ such arcs. The run time and number of cuts satisfy a recurrence of the form $T(n) = O(D^3) \cdot T(n/D^2) + O(n\,\text{poly}(D) + 2^{\text{poly}(D)})$. By choosing D to be a sufficiently large constant, this recurrence solves to $T(n) = O^*(n^{3/2})$, thereby proving Theorem 3.

4.3 An improved data structure for counting intersections

In this section we directly adapt the approach in Section 4.2 to design a new data structure for counting intersections between a query algebraic arc γ and a set of n input algebraic arcs Γ in $\mathbb{R}^2$.

First we consider an easier special case, where we are guaranteed that the query arc γ intersects each curve of Γ at most once. This special case will be useful later. We prove the following lemma in the full paper, by standard reductions to semialgebraic range searching and range stabbing (the bounds below may not be tight, but will be good enough):

▶ **Lemma 11.** *Given a set Γ of n algebraic arcs of constant complexity in $\mathbb{R}^2$, there is a data structure with $O^*(n^{3/2})$ preprocessing time and space that can count intersections with a query algebraic arc γ in $O^*(\sqrt{n})$ time if the query arc is guaranteed to intersect each arc of Γ at most once.*

We begin by considering $\widehat{\Gamma}$, the lifted version of each arc in $\mathbb{R}^3$, and taking a polynomial partition P of the curves of $\widehat{\Gamma}$ with degree D, which we will choose to be a sufficiently large constant. Let Γ_{bad} denote the set of bad arcs that, when lifted to $\mathbb{R}^3$, are contained in $Z(P)$ or Z_{bad} as defined in Section 4.2; there are at most $O(D^2)$ bad arcs. For each cell $\Delta \in \mathbb{R}^3 \setminus Z(P)$, let $\widehat{\Gamma}(\Delta)$ denote the set of all maximal subarcs of all $\widehat{\gamma} \in \widehat{\Gamma} \setminus \widehat{\Gamma}_{bad}$ that are contained in Δ. Furthermore, let $\widehat{\Gamma}'(\Delta)$ denote the set of subarcs of the arcs in $\widehat{\Gamma}(\Delta)$ after cutting each arc at its intersections with Z_{bad}. Let $\Gamma(\Delta)$ denote the xy-projections of the arcs of $\widehat{\Gamma}(\Delta)$, and define $\Gamma'(\Delta)$ similarly. We recursively build our data structure for each $\Gamma(\Delta)$. In addition, we preprocess each $\Gamma'(\Delta)$ in the data structure $\mathcal{D}(\Delta)$ from Lemma 11.

Given a query arc γ_q, we first cut $\widehat{\gamma}_q$ into subarcs at intersections with $Z(P) \cup Z_{bad}$; there are $O(D^2)$ such subarcs. We cut γ_q at the corresponding points. Our query algorithm is as follows:

1. For each cell $\Delta \in \mathbb{R}^3 \setminus Z(P)$ crossed by γ_q: we count the intersections of γ_q with $\Gamma(\Delta)$ by recursion. There are $O(D)$ recursive calls, since γ_q can cross $Z(P)$ at most $O(D)$ times.
2. For each cell $\Delta \in \mathbb{R}^3 \setminus Z(P)$ not crossed by γ_q, and for each subarc γ'_q of γ_q: we know that $\widehat{\gamma}'_q$ is not contained in Δ. As we have observed in Section 4.2, in this case, $\widehat{\gamma}'_q$ cannot form length-2 depth cycles with the subarcs in $\widehat{\Gamma}'(\Delta)$, and thus γ'_q cannot form lenses with the subarcs in $\Gamma'(\Delta)$, i.e., $\widehat{\gamma}'_q$ can intersect each arc of $\Gamma'(\Delta)$ at most once. Using the data structure $\mathcal{D}(\Delta)$ from Lemma 11, we can count the intersections of γ'_q with $\Gamma'(\Delta)$.
3. Finally, we naively count the intersections of γ_q with Γ_{bad}. This takes $O(D^2)$ time.

This way, every intersection point along γ_q is counted exactly once.

The query time satisfies the following recurrence:

$$Q(n) \;=\; O(D) \cdot Q(n/D^2) + O^*(D^{O(1)}\sqrt{n}),$$

which solves to $Q(n) = O^*(\sqrt{n})$ by choosing an arbitrarily large constant D. The preprocessing time (and thus space) satisfies the following recurrence:

$$P(n) \;=\; O(D^3) \cdot P(n/D^2) + O^*(D^{O(1)} n^{3/2} + 2^{\text{poly}(D)}),$$

which solves to $P(n) = O^*(n^{3/2})$.

▶ **Theorem 12.** *Given n algebraic arcs of constant complexity in $\mathbb{R}^2$, there is a data structure with $O^*(n^{3/2})$ preprocessing time and space that can count intersections with a query algebraic arc γ in $O^*(\sqrt{n})$ time. In particular, we can count the number of intersection points between two sets of n algebraic arcs of constant complexity in $\mathbb{R}^2$ in $O^*(n^{3/2})$ time.*

References

1 Peyman Afshani. Improved pointer machine and I/O lower bounds for simplex range reporting and related problems. *Internat. J. Comput. Geom. Appl.*, 23(4-5):233–251, 2013. Preliminary version in SoCG 2012. doi:10.1142/S0218195913600054.

2 Peyman Afshani and Pingan Cheng. Lower bounds for semialgebraic range searching and stabbing problems. *J. ACM*, 70(2):16:1–16:26, 2023. Preliminary version in SoCG 2021. doi:10.1145/3578574.

3 P. K. Agarwal and J. Matoušek. On range searching with semialgebraic sets. *Discrete Comput. Geom.*, 11(4):393–418, 1994. doi:10.1007/BF02574015.

4 Pankaj K. Agarwal. Ray shooting and other applications of spanning trees with low stabbing number. *SIAM J. Comput.*, 21(3):540–570, 1992. Preliminary version in SoCG 1989. doi:10.1137/0221035.

5 Pankaj K. Agarwal. Range searching. In J. E. Goodman, J. O'Rourke, and C. Toth, editors, *Handbook of Discrete and Computational Geometry*. CRC Press, 2016.

6 Pankaj K. Agarwal, Noga Alon, Boris Aronov, and Subhash Suri. Can visibility graphs be represented compactly? *Discret. Comput. Geom.*, 12:347–365, 1994. Preliminary version in SoCG 1993. doi:10.1007/BF02574385.

7 Pankaj K. Agarwal, Boris Aronov, Esther Ezra, Matthew J. Katz, and Micha Sharir. Intersection queries for flat semi-algebraic objects in three dimensions and related problems. In *Proc. 38th International Symposium on Computational Geometry (SoCG)*, pages 4:1–4:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/lipics.socg.2022.4.

8 Pankaj K. Agarwal, Boris Aronov, Esther Ezra, and Joshua Zahl. Efficient algorithm for generalized polynomial partitioning and its applications. *SIAM J. Comput.*, 50(2):760–787, 2021. Preliminary version in SoCG 2019. doi:10.1137/19M1268550.

9 Pankaj K. Agarwal, Esther Ezra, and Micha Sharir. Semi-algebraic off-line range searching and biclique partitions in the plane. In *Proc. 40th International Symposium on Computational Geometry (SoCG)*, 2024. To appear.

10 Pankaj K. Agarwal and Jiří Matoušek. Ray shooting and parametric search. *SIAM J. Comput.*, 22(4):794–806, 1993. doi:10.1137/0222051.

11 Pankaj K. Agarwal, Jiří Matoušek, and Micha Sharir. On range searching with semialgebraic sets. II. *SIAM J. Comput.*, 42(6):2039–2062, 2013. doi:10.1137/120890855.

12 Pankaj K. Agarwal, Eran Nevo, János Pach, Rom Pinchasi, Micha Sharir, and Shakhar Smorodinsky. Lenses in arrangements of pseudo-circles and their applications. *J. ACM*, 51(2):139–186, 2004. doi:10.1145/972639.972641.

13 Pankaj K. Agarwal, Marco Pellegrini, and Micha Sharir. Counting circular arc intersections. *SIAM J. Comput.*, 22(4):778–793, 1993. Preliminary version in SoCG 1991. doi:10.1137/0222050.

14 Pankaj K. Agarwal and Micha Sharir. Pseudo-line arrangements: duality, algorithms, and applications. *SIAM J. Comput.*, 34(3):526–552, 2005. doi:10.1137/S0097539703433900.

15 Pankaj K. Agarwal, Marc J. van Kreveld, and Mark H. Overmars. Intersection queries in curved objects. *J. Algorithms*, 15(2):229–266, 1993. Preliminary version in SoCG 1991. doi:10.1006/JAGM.1993.1040.

16 Boris Aronov and Micha Sharir. Cutting circles into pseudo-segments and improved bounds for incidences and complexity of many faces. *Discret. Comput. Geom.*, 28(4):475–490, 2002. doi:10.1007/S00454-001-0084-1.

17 Boris Aronov and Micha Sharir. Almost tight bounds for eliminating depth cycles in three dimensions. *Discret. Comput. Geom.*, 59(3):725–741, 2018. doi:10.1007/S00454-017-9920-9.

18 Egbert Brieskorn and Horst Knörrer. *Plane Algebraic Curves.* Springer Science & Business Media, 2012. Translated by J. Stillwell.

19 Timothy M. Chan. Geometric applications of a randomized optimization technique. *Discret. Comput. Geom.*, 22(4):547–567, 1999. Preliminary version in SoCG 1998. doi:10.1007/PL00009478.

20 Timothy M. Chan. On levels in arrangements of curves. *Discret. Comput. Geom.*, 29(3):375–393, 2003. doi:10.1007/S00454-002-2840-2.

21 Timothy M. Chan. Optimal partition trees. *Discrete Comput. Geom.*, 47(4):661–690, 2012. Preliminary version in SoCG 2010. doi:10.1007/s00454-012-9410-z.

22 Timothy M. Chan. Near-optimal randomized algorithms for selection in totally monotone matrices. In *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1483–1495, 2021. doi:10.1137/1.9781611976465.89.

23 Timothy M. Chan and Da Wei Zheng. Hopcroft's problem, log-star shaving, 2D fractional cascading, and decision trees. In *Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 190–210, 2022. doi:10.1137/1.9781611977073.10.

24 B. Chazelle and J. Friedman. A deterministic view of random sampling and its use in geometry. *Combinatorica*, 10(3):229–249, 1990. doi:10.1007/BF02122778.

25 Bernard Chazelle. Cutting hyperplanes for divide-and-conquer. *Discret. Comput. Geom.*, 9:145–158, 1993. doi:10.1007/BF02189314.

26 Bernard Chazelle, Herbert Edelsbrunner, Leonidas J. Guibas, and Micha Sharir. Algorithms for bichromatic line-segment problems and polyhedral terrains. *Algorithmica*, 11(2):116–132, 1994. doi:10.1007/BF01182771.

27 Bernard Chazelle and Emo Welzl. Quasi-optimal range searching in space of finite VC-dimension. *Discret. Comput. Geom.*, 4:467–489, 1989. doi:10.1007/BF02187743.

28 Kenneth L. Clarkson and Peter W. Shor. Application of random sampling in computational geometry, II. *Discret. Comput. Geom.*, 4:387–421, 1989. Preliminary version in SoCG 1988. doi:10.1007/BF02187740.

29 Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational Geometry: Algorithms and Applications.* Springer, 3rd edition, 2008.

30 Jordan S. Ellenberg, Jozsef Solymosi, and Joshua Zahl. New bounds on curve tangencies and orthogonalities. *Discrete Analysis*, November 2016. doi:10.19086/da.990.

31 Esther Ezra and Micha Sharir. Intersection searching amid tetrahedra in 4-space and efficient continuous collision detection. In *Proc. 30th Annual European Symposium on Algorithms (ESA)*, pages 51:1–51:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/lipics.esa.2022.51.

32 Esther Ezra and Micha Sharir. On ray shooting for triangles in 3-space and related problems. *SIAM J. Comput.*, 51(4):1065–1095, 2022. Preliminary version in SoCG 2021. doi:10.1137/21M1408245.

33 Tomás Feder and Rajeev Motwani. Clique partitions, graph compression and speeding-up algorithms. *J. Comput. Syst. Sci.*, 51(2):261–272, 1995. doi:10.1006/JCSS.1995.1065.

34 Ido Y. Grabinsky. Deferred data structures for online disk range searching. Master's thesis, Tel-Aviv University, February 2009. URL: http://www.cs.tau.ac.il/thesis/thesis/thesis_ido.pdf.

35 Larry Guth. Polynomial partitioning for a set of varieties. *Math. Proc. Cambridge Philos. Soc.*, 159(3):459–469, 2015. doi:10.1017/S0305004115000468.

36 Larry Guth and Nets Hawk Katz. On the Erdős distinct distances problem in the plane. *Ann. of Math. (2)*, 181(1):155–190, 2015. doi:10.4007/annals.2015.181.1.2.

37 Haim Kaplan, Jirí Matousek, and Micha Sharir. Simple proofs of classical theorems in discrete geometry via the Guth-Katz polynomial partitioning technique. *Discret. Comput. Geom.*, 48(3):499–517, 2012. doi:10.1007/S00454-012-9443-3.

38 Vladlen Koltun. Segment intersection searching problems in general settings. *Discret. Comput. Geom.*, 30(1):25–44, 2003. Preliminary version in SoCG 2001. `doi:10.1007/S00454-003-2924-7`.

39 Adam Marcus and Gábor Tardos. Intersection reverse sequences and geometric applications. *J. Comb. Theory, Ser. A*, 113(4):675–691, 2006. `doi:10.1016/J.JCTA.2005.07.002`.

40 Jirí Matoušek. Efficient partition trees. *Discret. Comput. Geom.*, 8:315–334, 1992. Preliminary version in SoCG 1991. `doi:10.1007/BF02293051`.

41 Jiří Matoušek and Zuzana Patáková. Multilevel polynomial partitions and simplified range searching. *Discrete Comput. Geom.*, 54(1):22–41, 2015. `doi:10.1007/s00454-015-9701-2`.

42 Nimrod Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983. `doi:10.1145/2157.322410`.

43 Micha Sharir and Joshua Zahl. Cutting algebraic curves into pseudo-segments and applications. *J. Combin. Theory Ser. A*, 150:1–35, 2017. `doi:10.1016/j.jcta.2017.02.006`.

44 Hisao Tamaki and Takeshi Tokuyama. How to cut pseudoparabolas into segments. *Discret. Comput. Geom.*, 19(2):265–290, 1998. Preliminary version in SoCG 1995. `doi:10.1007/PL00009345`.

45 Emo Welzl. Partition trees for triangle counting and other range searching problems. In *Proc. 4th Annual Symposium on Computational Geometry (SoCG)*, pages 23–33, 1988. `doi:10.1145/73393.73397`.

46 Emo Welzl. On spanning trees with low crossing numbers. In *Data Structures and Efficient Algorithms*, volume 594 of *Lecture Notes in Comput. Sci.*, pages 233–249. Springer, Berlin, 1992. `doi:10.1007/3-540-55488-2_30`.