# Robotics Research Technical Report 



An Algorithm for Generalized Point
Location and Its Applications
by
Bernard Chazelle Micha Sharir

Technica! Report No. 369
Robotics Report No. 153
May, 1988

## Computer Science Division



# An Algorithm for Generalized Point <br> Location and Its Applications 

## by

Bernard Chazelle
Micha Sharir

## Technical Report No. 369

Robotics Report No. 153
May, 1988

```
New York University
Dept. of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012
```

Work by the first author has been supported in part by National Science Foundation Grant CCR-8700917. Work by the second author has been supported by Office of Naval Research Grant N00014-87-K-0129, National Science Foundation Grant DCR-83-20085, and by grants from the Digital Equipment Corporation, the IBM Corporation, and the NCRD - the Israeli Council on Research and Development.

## AN ALGORITHM FOR GENERALIZED POINT LOCATION AND ITS APPLICATIONS

Bernard Chazelle<br>Dept of Computer Science<br>Princeton Unstersity<br>Princeton, AJ 08541

Micha Sharir<br>Dept. of Computer Science<br>NYU, Courant Institute<br>Neu Y'ork, N' 1001 ?<br>and<br>School of Mathematical Sciences<br>Tel Aiar l'nitersity<br>Tel Arat, Israel


#### Abstract

: We show that Collins classical quant ifier elimunation procedure contains most of the ingredients for an efficient point location algorithm in bigher-dimensional space This leads to a polynomial-size data structure which allows us to locate a point among a collection of real algebraic varieties of constant maximum degree in logarithmuc time. This result has theoretical bearings on a number of optimization problems posed in the literature. It also gives a method for solving multidimensional searching problems in polynomial space and logarithmic query time.

^[ Bernard Cbazelle wishes to acknowledge the National Science Foundation for supporting this research in part under Grant CCR-8700917. Micha Sharir is pleased to acknowledge the support of the Office of Naval Research under Grant N00014-87-K-0129, the National Science Foundation under Grant DCR-83-20085, the Digital Equipment Corporation, the IBM Corporation, and the NCRD - the Israeli Council on Research and Development. ]


## 1. Lntroduction

The central theme of multidimensional searching is the organization of a database to which queries of a chosen type can be made. The term locus approach refers to the particular strategy which regards a query as a point in higher-dimensional space: the idea is to subdivide the query space into equivalence classes and thus reduce query-answering to point location. This approach was followed by Dobkin and Lipton [DL], who devised an efficient searching algorithm for linear varieties in $E^{d}$, for any fixed $d$. Yao and Yao $\left.M 7\right]$ bave observed that the constraint manifold can usually be made linear by throwing in additional variables, if needed. Considering (as will be shown below) that the preprocessing is doubly exponential in the number of variables, bowever, the linearization method may not always be so desirable. We will show bere that Dobkin and Lipton's method can be generalized directly to bandle arbitrary algebraic varieties. The generalization is squarely based on Colluns' cylindrical algebraic decomposition (Collins [C], Arnon. Collins, and McCallum (ACM1,ACM2]). The main idea is to transform Collins' algorithm into a data structure and add various bells and whistles to support fast searching

For a precise statement of our results we need to introduce a few notions Let $\mathcal{F}=\left\{P_{1}, \ldots, P_{n}\right\}$ be a set of $n d$-variate polynomials with rational coefficients and norm-length at most $\ell$ (The normlength of a polynomial is the number of bits needed to represent the sum of the absolute values of its coefficients) We also assume that the maximum degree of these polynomials is bounded above by a constant Note that to allow all polynomials to be distinct, it is necessary to let $\ell$ be at least on the order of $\log n$ In practice one should expect $\ell$ to be at most polylogarithmic in $n$.

The generalized point localion problem concerns the fast evaluation of the predicate

$$
\left[\exists i(1 \leq i \leq n) \mid P_{1}(x)=0\right] \text {, }
$$

for any query point $x \in E^{d}$ with rational coefficients. A simple true/false answer being a little too terse, we require supplementary information. If the predicate is true then some witness isuch that $P_{i}(x)=0$ sbould be provided (note that requiring the reporting of all such indices might by itself preclude a fast response). If, on the other band, the predicate is false, then $\boldsymbol{x}$ lies in a connected region of the open oet $\mathcal{C}=\left\{x \in E^{d} \mid \Pi_{1 \leq i \leq r .} P_{1}(x) \neq 0\right\}$. In that case, the desired output is usually the value $f(x)$ of some function $f$ which is invariant over each connected region of $\mathcal{C}$. The preprocessing will compute a distinct algebraic point, a sample, in each region of $\mathcal{C}$ and evaluate $f$ at the sample points. As it turns out, storing these algebraic points may be quite costly, so the
data structure will only store the values of the function at the sample points and not the points themselves Our assumption that $I$ should be rational is made to simplify the complexity analysis The point location algorithm can in fact be easily extended to bandle arbitrary real algebraic points

The problem is a direct generalization of the well-known planar point location problem. Previous work on point location with nonlinear boundaries has been limited to the case $d=2$, culminating in the optimal algorithms of Cole [Co] and Edelsbrunner, Guibas and Stolf [EGS] for subdivisions with "monotone" curves, and that of Sarnak and Tarjan [ST] for more general subdivisions As will be shown below, a Colline decomposition provides a simple framework for solving the point location problem in full generality. We will deacribe a data atructure of size $O\left(n^{2^{d}-1}\right)$ which allows us to answer any query in $O(\log n)$ time, the time needed to build the data structure is $O\left(n^{2^{\alpha+6}}\right)$

These bounds bold in the traditional unit-cost RAM model (Aho et al. [AHU]). One will notice, for example, that the norm-length $\ell$ does not even appear in the bounds. The reason is that in the unit-cost model an! integet operation takes constant time. regardiess of the length of the integers involved Of course, this may sometimes hide the true cost of a computation if the integers become very large Traditionally, algonthms in computational geometry have tended to ignore the true cost of precise rational arithmetic, although this issue has recently started to gain importance (see [DS. HHK]) In the case of our data structure, however, this cost must be taken into account The algorithm involves iterated computations of polynomal greatest common divisors. subresultants. Sturm sequences, etc, all of which tend to inflate the size of the coefficients of the polynomials. Indeed, storing the sample points in the data structure is to be avoided. To represent their (algebraic) coordinates by means of defining polynomals and isolating intervals requires too many bits to be of much use during query operations

With this proviso, the complexity analysis of our data structure follows (Collins [C]) without the added burden of storing cylindrical algebraic samples (though such samples might be computed, used in preprocessing, and thrown away) If we start out with d-variate polynomials of constant degree and norm-length $\ell$, the preprocessing will only generate $k$-variate polynomials ( $k \leq d$ ) of constant maximum degree and norm-length $O(\ell)$. Moreover, all operations on the coefficients of the polynomials can be carried out in a number of bit operations at most cubic in their norm-length. This means that in order to obtain upper bounds on the bit complexity of the algorithm it suffices to multiply the unit-cosi bounds given above by $\ell^{3}$. This would give us a query time of $O\left(\ell^{3} \log n\right)$ and a preprocessing time of $O\left(\ell^{3} n^{2^{2+4}}\right)$. As long as a computer word can store up to $l$ bits the storage requirement is (asymptotically) the same in both models of computation. Although our underlying
assumption will be the unit-cost model, we will also mention the bit complexity of an algorithm whenever there is a discrepancy between the two models. A final word concerning the dependency of the algorithm on the degree of the polynomials. We caution that our algorihtm, like Collins'. produces auxiliary polynomials whose maximum degrees can be truly enormous. This can add a large multiplicative factor, say around $b^{44}$, albeit constant, to the complexity of the algorithms. We will ignore this dependency in the subsequent analysis.

Interestingly, our data structure matches Dobkin and Lipton's [DL] in terms of storage requirements, both being $O\left(n^{2^{d}-1}\right)$. Although the size of our data structure is polynomial in $n$, the magnitude of the exponent puts a severe limitation on its practicality. From a theoretical standpoint, however, this result has direct application to multidimensional searching: we will discuss this relationship in some detail later. The algorithm also bas somewhat unexpected ramifications. We will use it as a tool in the solution of several problems Two of them, posed by Mckenna [Mc], seek (i) the longest line segment fully contajed within a given $n$-gon and (ii) the minimum "vertucal" distance between two collections of red and blue segments in 3 -space (see section 5 for details) Another one, due to Atallah [A], asks at which time the convex hull of $n$ points moving in the plane will first enter its final, steady configuration. Using our point location algorithm, combined with a batching technique originally proposed by Yao [Y] we are able to solve these problems. as well as other related ones, in subquadratic (albeit ever so close to quadratic) time.

These are a few concrete exemplifications of a more general principle, which is one of the main consequences of this paper. Many optimization problems in computational geometry have trivial quadratic solutions. Typically these problems involve two sets $A, B$ of $n$ objects each, and ask for the pair $(a, b) \in A \times B$ which minimizes some cost function, or satisfies some predicate; think. for example, of the diameter or closest-pair problem (Preparata and Sbamos [PS]). A considerable amount of recent work in computational geometry can be regarded as attempts to beat this trivial quadratic bound by building clever data structures which reduce the number of pairs $a \in A, b \in B$ that need testing (e.g., the Voronoi diagram for the closest-pair problem). Our results imply that if the interaction of a single pair $a \in A, b \in B$ can be stated as an algebraic expression (possibly involving Boolean algebraic predicates) in the real parameters specifying $a, b$, then the data structure that we develop can be used to reduce the problem complexity to subquadratic.

In eection 2 we review the algebraic backdrop bebind the algorithms, and in section 3 we describe the point location data structure in detail. section 4 discusses the relevance of the algorithm to multidimensional searching in general. In eection 5 we tackle McKenna's problems by reducing
them to a more general optimization question of the sort just mentioned We attack Atallah's problem in section 6, and give a more general discussion of the underlying technique in section 7 Finally, we conclude this paper in section 8 with directions for future research

## 2. The Algebraic Machinery

Most of the algebraic notions involved in this work can be found exposed in great detail in (Collins [C]) and (Schwartz and Sharir [SS]). We have tried to adhere to the terminology used in these papers as much as possible. The fundamental algebraic concepts can be found in van der Waerden's classic text [ $W$ ], while for the specialized treatment of resultants and subresultants used in the paper the reader should turn to Brown and Traub [BT].

1) Collins' Decidability Tbeorem In 1948 Tarski [T] proved that every statement in elementary algebra (which is, the elementary theory of real-closed fields) is decidable. The non-elementary procedure given by Tarski was subsequently improved (computationally) in a number of diferent ways by several researchers (e.g. Seidenberg [ S ], Cohen [Coh], Collins [C], Monk and Solovay [M]. Ben-Or, Kozen and Reif (BKR]) For the purpose of the present work, we shall use Collins' decision procedure as a guiding framework Let a standard prener formula be any logical sentence of the form

$$
\left(Q_{k} I_{k}\right)\left(Q_{k+1} x_{k+1}\right) \ldots\left(Q_{d} I_{d}\right) \phi\left(I_{1} \ldots \ldots I_{d}\right),
$$

where each $Q_{1}$ is a universal or existential quantifier and $\varphi\left(x_{1} \ldots \ldots I_{d}\right)$ is a quantifier-free formula made of Boolean connectives standard comparators, and polynomials with rational coefficients in the real variables $x_{1}, \ldots, x_{d}$ A logical sentence is called an atomic formula if it is free of quantifiers and logical connectives.

Theorem 1. (Collins [C]) - Let $\Phi$ be an arbitrary standard prener formula with $d$ variables, e atomac formulas, $m$ polynomials of degree at most $b$ in any single variable, with all integral coefficients of length less than $\ell$. Whether $\Phi$ is true or false can be decided in at most $\ell^{3}(2 b)^{2^{2+1}} \mathrm{~m}^{2 d+6}$ brt operations.
II) The Cylindrical Algebraic Decomposition: This section reviews the essential components of Collins' decomposition needed for the point location algorithm. We include this discussion to make our exposition elf-contained The reader fully familiar with Collins' work may skip the next paragraphs.

A Collins decomposition of the d-dimensional Euclidean space $E^{d}$ is a refinement of the decomporition of $E^{d}$ induced by a finite collection of real algebraic varieties. (Each polynomial defining a variety is sign-invariant over each region of the decomposition) The key concept is that of a cylindrical algebraic decomposition (or cad, for short) A d-dimensional cod is a partitioning of $E^{d}$ defined inductively as follows
(i) For $d=1$, a cad is a finite set of disjoint open intervals and singletons whose union forms $E^{1}$ Each singleton contains an algebraic number: see a discussion later in this section on how to store algebraic numbers.
(ii) For $d>1$, a cad $K$ is defined in terms of a cad $K^{\prime \prime}$ of $E^{d-1}$ and a d-variate polynomial $P\left(x_{1}, \ldots, x_{d-1}, y\right)$ with rational coefficients. Let $K^{\prime}=\left\{c_{1}, \ldots, c_{\mu}\right\}$; for each $c_{1} \in K^{\prime \prime}$ there exists an integer $\nu_{1}$ such that for each $x=\left(x_{1}, \ldots, x_{d-1}\right) \in c_{1}, P(x, y)$, regarded as a polynomial in $y$, has $\nu_{1}$ real roots $f_{1.1}(x)<\cdots<f_{1, y}(x)$. each of which is a continuous function in $x$ over $c_{1}$. If $\nu_{1}=0$, set $c_{1,1}=c_{1} \times E^{1}$. If $\nu_{1}>0$. set $c_{1,2}=\left\{\left(x, f_{1, j}(x)\right) \mid x \in c_{1}\right\}$ for $1 \leq j \leq \nu_{1}$, and set $c_{1,2,+1}=\left\{(x, y) \mid x \in c_{1}\right.$ and $\left.f_{1, j}(x)<y<f_{1, j+1}(x)\right\}$ for $1 \leq j<\nu_{1}$. Also. put $c_{1,1}=\left\{(x, y) \mid x \in c_{1}\right.$ and $\left.y<f_{1,1}(x)\right\}$ and $c_{1,2 \nu,+1}=\left\{(x, y) \mid x \in c_{1}\right.$ and $\left.f_{1, \nu_{1}}(x)<y\right\}$ Finally $K$ is defined as the set of cells $\left\{c_{1,1} \ldots, c_{1,2 \nu_{1}+1}, \ldots, c_{\mu, 1}, \ldots, c_{\mu} \nu_{\nu}+1\right\}$

Following (Schwartz and Sharir [SS]) we call $P$ the base polynomial of the cad. Informally, the cells of $K$ can be formed by considering the cyinders based at each $c \in K^{\prime \prime}$ and chopping them off with the real hypersurface $P\left(x_{1} \ldots, x_{d}\right)=0$ Since $K$ is defined in terms of a unique cad of lesser dimension. by induction. it defines an induced cad for each $E^{k}(1 \leq k<d)$. Incidentally, one should note that each cell of $K$ is "eell-behaved," in the sense that it is topologically equivalent 10 a relatively open ball of dimension at most $d$.

For our purposes the base polynomial $P$ will always be of the form $\prod_{1 \leq 1 \leq n} P_{1}$, where $\mathcal{F}=$ $\left\{P_{1}, \ldots, P_{n}\right\}$ is a collection of $d$-variate polynomials with rational coefficients. The key feature of a cad is that for each $c \in K$ and each $P_{1} \in \mathcal{F}$, the value of $P_{1}(x)$ is either zero over the entire cell $c$, or it keeps the same sign over the cell: a cad which satisfies this property is said to be $\mathcal{F}$-invariant. Besides introducing the concept itself the main contribution of (Collins [C]) was to prove that any collection $\mathcal{F}$ admits of an $\mathcal{F}$-invariant cad and that it can be constructed fairly efficiently (all things considered).

To simplify the computations (as well as carry the analysis further to determine the adjacencies between the cells of a cad) Schwartz and Sharir [SS] introduce the useful concept of a well-based
decomposition $K$ is said to be well-based if, when regarded as a unvariate polynomial in $y$, the base polynomial $P(\boldsymbol{x}, \boldsymbol{y})$ is not identically zero for any given value of $\boldsymbol{x}$ in $E^{d-1}$. They show that in that case each root function $f_{1}$, (defined over $c_{1} \in K^{\prime \prime}$ ) can be extended continuously over the closure of $c_{1}$. Informally, this means that every une $\left(x_{1}, \ldots, x_{d-1}\right) \times E^{1}$ intersects the algebraic variety $P(x, y)=0$ only a finite number of times. These intersections will form the basis of the binary search underlying the point location algorithm to be presented in the next section. How can we ensure that a decomposition is well-based? Since point location is defined independently of a coordinate system, we can always modify the frame of reference to ensure this condition. As a matter of fact, Schwartz and Sharir [SS] bave suggested that a few random perturbations of the original coordinate system might be the best strategy in practice. (They also give a method for checking if a cad is well-based) For our purposes, a well-based decomposition is convenient but not necessary. Therefore another solution is, of course, not to worry about it and simply ensure that the search procedure is robust enough to handle this type of degeneracy

Following Collins' terminolog, we define an algebraic sample of $\kappa$ as a set of points with algebraic coordinates, one in each cell of $\hbar$ (recall that a real number is algebraic if it is a root of a polynomial with integer coefficients) An algebraic sample is cylindrical (abbreviated cas) if either $d=1$ or the set of $d-1$ first coordinates of each point forms a cas of $K^{\prime \prime}$. If $\left\{c_{1,1}, \ldots c_{12 v,+1}\right\}$ is the set of cells of $\kappa^{\prime}$ associated with the cell $c_{1}$ of $K^{\prime}$, the sample points in each $c_{1},\left(1 \leq j \leq 2 \nu_{1}+1\right)$ all share the same $d-1$ first coordinates
III) The Collins Construction We begin with a short review of Collins' algorithm Let $Q$ be a $d$-variate polynomial of degree $p$ with real rational coefficients. We can write $Q\left(x_{1}, \ldots, x_{d}\right)$ as a polynomial $\sum_{0 \leq 1 \leq p} Q_{1}\left(x_{1}, \ldots, x_{d-1}\right) x_{d}^{\prime}$ of a single variable $x_{d}$ with coefficients in the ring of real rational $(d-1)$-variate polynomals Let $\operatorname{deg}(Q)=p$ be the degree of $Q$ in $x_{d}$ and let $\operatorname{ldcf}(Q)=$ $Q_{F}\left(x_{1}, \ldots, x_{d-1}\right)$ denote the leading nonzero coefficient of $Q$ We define the reductum of $Q$, denoted $\operatorname{red}(Q)$, as the polynomial $\sum_{0 \leq 1 \leq p-1} Q_{i}\left(x_{1}, \ldots, x_{d-1}\right) x_{d}^{\prime}$. We also introduce $\operatorname{red}^{0}(Q)=Q$, and for each $k \geq 0, \operatorname{red}^{k+1}(Q)=\operatorname{red}\left(\operatorname{red}^{k}(Q)\right)$. Finally we let $\operatorname{der}(Q)$ denote the $x_{d}$-derivative of $Q$

Let $A(x)=\sum_{0 \leq i \leq a} \alpha_{i} x^{i}$ and $B(x)=\sum_{0 \leq 1 \leq b} \beta_{i} z^{i}$ be two polynomials in the real variable $x$ with $\operatorname{deg}(A)=a$ and $\operatorname{deg}(B)=b$. The Sylvester metriz of $A$ and $B$ is the $(a+b) \times(a+b)$ matrix $M$ obtained by placing the coefficients of the polynomials $x^{b-1} A(x), \ldots, x A(x), A(x), x^{a-1} B(x), \ldots, x B(x), B(x)$ in consecutive rows of $M$, with the coefficients of $z^{i}$ appearing in column $a+b-i$ :

$$
\left(\begin{array}{ccccccc}
\alpha_{a} & \alpha_{a-1} & \ldots & \alpha_{0} & & & \\
& \alpha_{a} & \alpha_{a-1} & \ldots & \alpha_{0} & & \\
& & \ddots & \ddots & \ddots & \ddots & \\
& & & \alpha_{o} & \alpha_{a-1} & \ldots & \alpha_{0} \\
\beta_{b} & \beta_{b-1} & \ldots & \beta_{0} & & & \\
& \beta_{b} & \beta_{b-1} & \ldots & \beta_{0} & & \\
& & \ddots & \ddots & \ddots & \ddots & \\
& & & \beta_{b} & \beta_{b-1} & \ldots & \beta_{0}
\end{array}\right)
$$

The determinant of $M$ is the resuttant of $A$ and $B$. For $0 \leq j \leq \min (a, b), M$, is the matrix obtained by deleting the last $j$ rows of $A$ coefficients, the last $j$ rows of $B$ coefficients, and all the last $2 j$ columns. We can then define $p c^{j}(A, B)$ (the $j^{\text {th }}$ principal subresuttant coefficient of $A$ and $B$ ) as the determinant of $M$, (see (Collins [C]) for details). The unique factorization theorem for polynomials implies that $A$ and $B$ have exactly $j$ common roots (i.e., $j$ is the degree of their greatest common divisor) if and only if $j$ is the least index $k$ for which $\operatorname{pse}^{k}(A, B) \neq 0$. This result is at the basis of the recursive construction of a cad

Let $\mathcal{F}=\left\{P_{1}, \ldots, P_{n}\right\}$ be a set of $n$ teal $d$-variate polynomials $(d \geq 2)$ with rational coefficients and norm-length $\leq \ell$. As expected, the construction of an $\mathcal{F}$-invariant $K$ proceeds recursively. It suffices to specify what arguments should be passed to the decomposition algorithm at the first recursive call. To do so, we define the projection of $\mathcal{F}$, denoted $\mathcal{G}$, as the union of $\mathcal{G}_{2}, \mathcal{G}_{3}$. and $\mathcal{G}_{4}$. with
(i) $\mathcal{G}_{1}=\left\{\operatorname{red}^{k}(P) \mid P \in \mathcal{F}\right.$ and $k \geq 0$ and $\left.\operatorname{deg}\left(\operatorname{red}^{k}(P)\right) \geq 1\right\}$,
(ii) $\mathcal{G}_{2}=\left\{\operatorname{ldcf}(P) \mid P \in \mathcal{G}_{1}\right\}$,
(iii) $\mathcal{G}_{3}=\left\{\operatorname{psc}^{k}(P, \operatorname{der}(P)) \mid P \in \mathcal{G}_{1}\right.$ and $\left.0 \leq k<\operatorname{deg}(\operatorname{der}(P))\right\}$,
(iii) $\mathcal{G}_{4}=\left\{\operatorname{psc}^{k}(P, Q) \mid P, Q \in \mathcal{G}_{1}\right.$ and $\left.0 \leq k<\min (\operatorname{deg}(P), \operatorname{deg}(Q))\right\}$

The notion of a projection generalizes Doblin and Lipton's idea [DL] of pairing up hyperplanes: the pairing takes place in $\mathcal{G}_{4}$, while $\mathcal{G}_{2}$ and $\mathcal{G}_{3}$ account for the possible losses of rank in each variety. The following result is proven in (Collins [C]): let $K^{\prime \prime}$ be a $\mathcal{G}$-invariant cad of $E^{d-1}$ and let $\boldsymbol{c}_{\mathrm{i}}$ be any cell of $K^{\prime}$; the total number of distinct real roots of the polynomials in $\boldsymbol{x}_{\boldsymbol{d}}$. $P_{1}\left(x_{1}, \ldots, x_{d}\right), \ldots, P_{n}\left(x_{1}, \ldots, x_{d}\right)$, remains constant as $\left(x_{1}, \ldots, x_{d-1}\right)$ varies in $c_{1}$. These roots form a well-ordered set of continuous functions over $c_{1} \cdot f_{1,1}\left(x_{1}, \ldots, x_{d-1}\right), \ldots, f_{1, v}\left(x_{1}, \ldots, x_{d-1}\right)$; in particular, no two such roots ever coincide over $c_{1}$. As a result, for each $c_{i} \in K^{\prime \prime}$, the partition of $c_{1} \times E^{\prime}$ induced by the bypersurfaces $x_{d}=f_{i, 1}(x), \ldots, x_{d}=f_{i, v}(x)\left(x \in E^{d-1}\right)$ defines an $\mathcal{F}$-invariant cad of $E^{d}$.

This provides a recursive scheme for computing an $\mathcal{F}$-invariant cad $k$ of $E^{d}$ The algorithm takes $\mathcal{F}$ and $d$ as input and recurses by caling itself with $\mathcal{G}$, the projection of $\mathcal{F}$, and $d-1$ as arguments The output of Collins' construction includes (i) quantifier-free formulas defining each cell of $K$ and (ii) a cas of $K$ of the form $\left\{\beta_{1}, \ldots, \beta_{\nu}\right\}$, where for each $i=1, \ldots, \nu$, each coordinate of $\beta_{1} \in E^{d}$ is represented by a quantifier-free formula For our application the definitions of the cells are not really needed: instead, we need a correspondence between sample points and their associated polynomials in $\mathcal{F}$. If $d>1$ then $K$ has a base cad $K^{\prime \prime}=\left\{c_{1}, \ldots, c_{\mu}\right\}$ which is $\mathcal{G}$-invariant Let $\left\{\beta_{1}^{\prime}, \ldots, \beta_{\mu}^{\prime}\right\}$ be the cas of $\mathcal{K}^{\prime \prime}$, computed recursively. For each $i=1, \ldots, \mu$, let $\left\{\beta_{1,1}, \ldots, \beta_{1,2 \nu,+1}\right\}$ be the points of the cas of $\kappa$, ordered in ascending $x_{d}$-order, whose first $d-1$ coordinates form the point $\beta_{1}^{\prime}$. Each point $\beta_{1,2},\left(1 \leq j \leq \nu_{1}\right)$ lies on at least one algebrac variety of the form $P_{l}(x)=0$ Let $l_{1}, j$ be any such value of $l$ and let $\beta_{1,2}=\left(a_{1}, \ldots, a_{d}\right)$; we define $m_{1, j}$, as the number of distinct real roots of $Q(y)$ that are strictly smaller than $a_{d}$, where $Q(y)=P_{i_{1}, f}\left(a_{1}, \ldots, a_{d-1}, y\right)$ is regarded as a polynomial in $y$ As part of the output, we require the sequence $\left\{\left(l_{1,1}, m_{1,1}\right) \ldots,\left(l_{1},,_{1}, m_{1, ~, ~, ~}\right)\right\}$ for each $i=1 \ldots, \mu$ This sequence will be necessary later on in order to carry out the binary searches underlying the point location algonthm

The next step is to show how to dernve these sequences from $\left\{3_{1}, \ldots, \beta_{1} v_{1}+1\right\}(1 \leq 1 \leq \mu)$. Recall that the latter sequences are provided directly by the Collins construction Let $O(x)$ be the quantifier-free defining formula for $\beta_{12},\left(1 \leq j \leq \nu_{1}\right)$ Trivial! we can test the predicate

$$
\left[\Xi x \in E^{d} \mid O(x) \text { and } P_{l}(x)=0\right]
$$

for each $l=1 \ldots$, and pick as $l_{1,2}$, say, the first value of $l$ found to satisfy the predicate To obtain $m_{1}$, it suffices to express with a prenex formula the proposition, denoted $F_{k}$, that $z$ is a root of $Q$ and $Q(y)=P_{i, 0}\left(a_{1}, \ldots, a_{d-1}, y\right)$ has exactly $k$ distinct roots strictly smaller than $=$ In the spirit of (Arnon [Ar]) we express $F_{k}$ with the formula

$$
\begin{aligned}
R_{a_{1}, \ldots, a_{d-1}, k}^{\prime}(z)= & \left(\exists y_{1}, \ldots, y_{k}\right)(\forall z) \mid \\
& {\left[\left(Q^{2}(z)+Q^{2}\left(y_{1}\right)+\cdots+Q^{2}\left(y_{k}\right)=0\right) \text { and }\left(y_{1}<\cdots<y_{k}<z\right)\right. \text { and }} \\
& \left.\left(Q(x) \neq 0 \text { or } z \leq x \text { or } \prod_{1 \leq 1 \leq k}\left(y_{1}-z\right)=0\right)\right] .
\end{aligned}
$$

The value of $m_{i j}$ is then given by the unique index $k$ for which $R_{a_{1}, \ldots, a_{d-1}, k}^{\prime, \prime}(z)$ is true, with $\beta_{1,2}=$ ( $\beta_{1}^{\prime}, z$ ).
V) Complexity Analysis We assume that only rational symbolic calculations are used during the course of the computation The following complexity results are derived from (Collins [C]). Let $b$ be the maximum degree of any polynomial in $\mathcal{F}$ in any variable. Recall all the polynomials in $\mathcal{F}$ bave norm-length at most $\ell$. We assume that $d$ and $b$ (but not necessarily $\ell$ ) are constants. The $\mathcal{F}$-invariant cad produced by the Collins construction consists of $O\left((2 b)^{3^{\alpha+1}} n^{2^{d}-1}\right)=O\left(n^{2^{d}-1}\right)$ cells. (Collins' paper [C] actually states a slightly larger bound, but the one above easily follows from bis derivations.) The total number of polynomials defined in the various projections introduced in the decomposition is bounded above by $O\left((2 b)^{3^{4}} n^{2^{4-1}}\right)=O\left(n^{2^{4-1}}\right)$ and the maximum degree of each polynomial in any variable is at most $\frac{1}{2}(2 b)^{2^{d-1}}=O(1)$. The norm-length of each polynomial is at most $(2 b)^{24} \ell=O(\ell)$.

Consider now the cas of the decomposition. Each algebraic point is represented by its coordinates. Collins uses two different representations of real algebraic numbers. One is the traditional root isolation method the number $a$ is the unique real root of an integral polynomial falling in some interval $J$, whose endpoints are rationals of the form $a / 2^{\star}$. In the other representation, a real algebraic number $\beta$ will appear as an element of the algebraic number field $Q(a)$ (ie., the smallest subfield of $\mathcal{R}$ that contains both $Q$ and $a$ ). In this case, we represent $\beta$ by a rational polynomial $B(x)$ with $\beta=B(\alpha)$. The degree of each polynomial used in the definition of the cas's is dominated by $(2 b)^{2 \alpha-1}=O(1)$ and their norm-length is at most $\ell(2 b)^{2^{2 \alpha+3} n^{2 \alpha+1}}=O\left(\ell n^{2^{\alpha+1}}\right)$ Implementing the Collins construction proper requires $O\left(t^{3}(2 b)^{2 a+4} n^{2 d+e}\right)=O\left(\ell^{3} n^{2^{\alpha+0}}\right)$ bit operations. Using Theorem 1 and the previous upper bounds, it is easy to see that this running time asymptotically dominates the overbead of computing the sequences of the form $\left\{\left(l_{1,1}, m_{1}, 1\right), \ldots,\left(l_{1}, \nu_{1}, m_{1, \nu}\right)\right\}$. In the unit-cost model, this gives us a total running time of $O\left(n^{2^{\alpha+0}}\right)$.

## 3. The Generalized Point Location Algorithm

Most of the ingredients entering the composition of the algorithm have already been introduced. The data structure $\mathcal{D}(\mathcal{F})$ is defined recursively as follows: it includes
(i) $\mathcal{D}(\mathcal{G})$, where $\mathcal{G}$ is the projection of $\mathcal{F}$;
(ii) a cas of $K$;
(iii) a set of $\nu$ one-word memory cells $C_{1} \ldots, C_{v}$ (which we conveniently associate with the cells of $K)$.

Let $C_{1}^{\beta} \ldots . C_{\mu}^{g}$ be the memory cells associated with $\mathcal{D}(\mathcal{G})$ (in one-to-one correspondence with the cells of $\left.K^{\prime \prime}=\left\{c_{1}, \ldots, c_{\mu}\right\}\right)$. Each cell $C_{i}^{\ell}(1 \leq i \leq \mu)$ stores a pointer to the sequence $\left\{\begin{array}{llll}1 & 1 & . & l_{1}\end{array}\right\}$ previously defined Recall that the cell $C_{1}^{\prime}$ is associated with $2 \nu_{1}+1$ cells of $h$ (each projecung exactly onto $\left.c_{1}\right)$. Let $W_{1}=\left\{C_{1,1}, \ldots, C_{1,2 \nu,+1}\right\}$ be the corresponding memory cells in ascending $z_{d}$. order. Consider the sequence $S_{1}=\left\{l_{1,1} \ldots \ldots l_{1,2}\right\}$ as an ordered set of keys. The possible outcomes of a binary search in this set form a sequence of $2 \nu_{3}+1$ keys and open intervals, which we put in one-to-one correspondence with $\mathrm{H}_{\text {: }}$. The data structure is now complete, so we can describe the algorithm.

The input is a family of polynomials $\mathcal{F}$, assumed to be preprocessed as previously described The generalized point location problem defined earlier can be reduced to the following given a quer: point $x=\left(x_{1}, \ldots, x_{d}\right) \in E^{d}$ with rational coordinates, compute the index isuch that $C_{1}$ corresponds to the unique cell of $\mathcal{H}$ that contans $I$ If $I$ is a zero of one of several polynomials of $\mathcal{F}$, the index of one of them will be directly availatle from $C_{1}$ If, on the other hand, $C_{1}$ lies in one of the connected regions of $\mathcal{C}=\left\{x \in E^{d} \mid \prod_{1 \leq 1 \leq n} P_{i}(x) \neq 0\right\}$, access to the sample points provided by the cas of $f$ would provide the desired answer But, as we said earlier, the data structure does not store the sample points. Recall that we are interested in evaluating a particular function $f$ which is invariant over the regions of $\mathcal{C}$. Before throwing the sample points. we will precompute and store the values of $f$ at these points Note that from a theorem of Milnor [M1] the number of connected regions in $\mathcal{C}$ is singly exponential in $d$ Therefore the evaluation of $f$ over the sample points is bound to praduce the same values repeatedly

If $d=1$ the algorithm is a trivial bonary search, so let us assume that $d>1$. Recursively: we assume that we have available the index of the cell $C_{1}^{g}$ that contains $\left(x_{1}, \ldots, x_{d-1}\right)$. Perform a binary search in $S$, with respect to $x_{d}$, and report the element of $W_{i}$ corresponding to the result of the search We can implement the generic comparisan against $l_{1,}$, as the two-fold question

1. Does $P_{l_{1, j}}(x)=0$ and is $I$ the $\left(m_{1, j}+1\right)$ st real root of $P_{l_{1, j}}$ ?
2. Is $x_{d}$ strictly larger or smaller than the $\left(m_{1},+1\right)$ st real root of $P_{1_{1}, ~}\left(x_{1}, \ldots, x_{d-1}, y\right)$, regarded here as a polynomial in $y^{?}$

Question (1) is easy to answer since it involves a simple polynomial evaluation. The second question can be answered by computing the predicate

$$
\left[(\forall y) \mid\left(y>x_{d}\right) \text { or } \neg\left(R_{x_{1}, \ldots, \varepsilon_{d-1}, m_{1, j}+1}^{d, y}(y)\right)\right] .
$$

Of course, since the polynomial $A(y)=P_{1},\left(x_{1}, \ldots, x_{d-1}, y\right)$ has constant degree and rational coefficients it might be just as simple to enumerate all its real roots (as algebraic numbers) and compare $x_{d}$ against them. Following (Schwartz and Sharir [SS]) we begin by writing $A(y)$ as $B(y) / c$, where $B$ is a polynomial with integral coefficients. Next we find an interval $I$ that contains all the real roots, e.g., $[-1-m, 1+m]$, where $m$ is the largest ratio between any coefficient of $B$ and its leading coefficient. Then we proceed by binary search over $l$, using Sturm sequences to find out how many real roots lie in a given interval. When the process terminates, each root will be enclosed by a separating interval with rational endpoints. Answering question 2 will then be trivial. A lower bound by Mabler [Ma] on the minimum separation between two distinct roots of a polynomial guarantees that the process will stop after $O\left(b^{3} \log b\right)=O(1)$ iterations (in the unit-cost model). Collins and Loos [CL] describe another technique for root isolation of bit complexity $O\left(b^{10}+b^{7} \varepsilon^{3}\right)$.

As mentioned earlier, each polynomial occurring in any projection has degree $O(1)$ and normlength $O(f)$, so the analysis above applies to all the binary searches performed in the location of $x$. Since the total number of these polynomials is in $O\left(n^{2-1}\right)$, the overall query time amounts to $O\left(\ell^{3} \log n\right)$ in the bit model and $O(\log n)$ in the unit-cost model. (lncidentally, note that these bounds involve constant factors doubly exponential in $d$ ) The preprocessing time is $O\left(n^{2^{d+0}}\right)$ in the unit-cost model and $O\left(\ell^{3} n^{2^{\alpha+0}}\right)$ in the bit model.

Theorem 2. Let $\mathcal{F}=\left\{P_{1}, \ldots, P_{n}\right\}$ be a family of $n$ d-variate polynomals with rational coefficients
and constant maximum degree. The generalized point location problem on $\mathcal{F}$ can be solved in $O(\log n)$ query time, using a data structure of size $O\left(n^{2^{d}-1}\right)$; the preprocessing time is $O\left(n^{2^{d+0}}\right)$. This assumes that operations on any integers of length proportional to the norm-length of the polynomials of $\mathcal{F}$ can be done in constant time.

## 4. Point Location and Multidimensional Searching

Multidimensional searching refers to the general task of querying a database to retrieve information of a particular nature. This can be defined formally by introducing a finite set $V$, a query space $Q$, and a response domain $R$. We also need a predicate function $p: Q \times V \rightarrow\{0,1\}$ and an evaluation function $c: 2^{\boldsymbol{v}} \rightarrow R$. A query is an arbitrary element $q \in Q$ and its output is the value of $c(\{z \in V \mid p(q, z)\})$. A classical example is orthogonal renge searching: $V$ is a set of points in $E^{d}, Q$ is the set of all d-dimensional isothetic byperrectangles, $p(q, x)$ is true if and only if the point $x$ lies in the byperrectangle $q$, and $c$ returns the cardinality of the input set in the counting version of the problem; in the reporting version, $c$ is the identity function.

Elements of both l' and $Q$ are expressed as rational points in Euclidean space, and the value of the predicate $p(q, x)$ is determined by the signs of certan polynomials $P_{s}, Q_{r}$, evaluated at $q$. The family $\mathcal{F}=\left\{P_{s}, Q_{f} \ldots \mid x \in \mathfrak{l}\right\}$ is assumed to consist of $d$-variate polynomals of constant maximum degree with rational coefficients The reduction of multidımensional searching to point location is now obvious This is called the locus approach subdivide $E^{d}$ via the varietier defined by $\mathcal{F}$ and assign to each resulting cell the corresponding (constant) value of the function $c$.

Two implementations of the locus approach suggest themselves. One is to apply the point location algorithm of the previous section. The other spproach, suggested by Yao and Yao MY"], is to linearize the polynomials by throwing in additional variables. For example, the variety in $E^{3}$.

$$
x y^{2} z^{3}+2 z^{2} y^{3}-3 z^{4}+z^{2} y+z=0
$$

can be replaced by the hyperplane in $E^{5}$

$$
z_{1}+2 z_{2}-3 z_{3}+z_{4}+z_{5}=0
$$

Searching for the location of the point ( $x . y, z$ ) among a real algebrave manifold is thus reduced to the point location of $\left(x y^{2} z^{3}, ~ 2 x^{2} y^{3}, 3 z^{4}, x^{2} y, z\right) \in E^{3}$ among a union of linear varieties, which can be solved using Dobkin and Lipton's method [DL]. The obvious disadvantage of the latter method is that the number of variables may jump from, say, $d$, to $(b+1)^{d}$, where $b$ is the maximum degree in a single variable of any polynomal of $\mathcal{F}$. This can have dire consequences, as the preprocessing cost will be doubly exponential in $d$ in one case and doubly exponential in $(b+1)^{d}$ in the other

Let us illustrate our point location approach on a specific example. Let $V$ be a set of $n$ points in $E^{d}$ and $Q$ be the set of $d$-variate polynomials of degree at most $b$ with rational coefficients. Given a query polynomial $q \in Q$, count the number of points $x \in V$ sucb that $q(x)>0$. In our framework the query $q$ can be regarded as a rational point in $E^{c}$, where $c$ is the dimension of the vector space $Q$. It is well-known that $c=\binom{d+b}{b}$. (The dimension $c$ is equal to the number of ways one can assign exponents to $x_{1}, x_{2}, \ldots, x_{d}$ adding up to at most $b$. Think of a sequence of $d$ bars and $b$ crosses intermixed. Such a sequence provides a unique assignment of exponents. A bar corresponds to some $x_{1}$ and the number of consecutive crosses to its leff indicates its power. Furthermore, the bars can be placed anywhere in the sequence.) The family $\mathcal{F}$ consists of $n$ linear forms of $c$ variables. A query is answered in time $O(\log n)$ at the cost or $O\left(n^{2^{x}-1}\right)$ space.

Consider now the case where $V$ is a set of $n$ points in $E^{d}$ and a query is a pair ( $q, r$ ) consisting of a point $q \in E^{d}$ and a positive rational number $r$. The response to the query is the number of points in $V$ lying within a distance $r$ of $q$. The family $\mathcal{F}$ consists of $n(d+1)$-variate polynomials $p\left(x_{1}, \ldots, x_{d+1}\right)$ of the form $\left(x_{1}-a_{1}\right)^{2}+\cdots+\left(x_{d}-a_{d}\right)^{2}-x_{d+1}^{2}$. A query is answered in time $O(\log n)$ at the cost of $O\left(n^{2+1}-1\right)$ space.

## 5. Biggest Stick, Line Shifting, and Other Related Problems

In this section we concern ourselves with the following class of problems: given two collections $A, B$ of $n$ objects each and a real-valued function $F$ defined on $A \times B$, compute the minimum of $F$ over $A \times B$. If the function $F$ can be evaluated anywhere in constant time, problems of this type always bave trivial $O\left(n^{2}\right)$ solutions Note that many common problems fall in this category. e.g, Hopcrofi's problem (given a collection of hnes and points in the plane, determine whether any line pass through any point), diameter problem in $E^{3}$ (given a three-dimensional polytope, what is the largest interdistance between any two vertices), etc.

We will give a method for solving these problems in subquadratic time. The technique is very general, and will always work as long as a fixed number of rational parameters are needed to represent objects in $A$ or $B$, and the expression $F(a, b)$ can be specified by a straight-line program of constant length involving algebraic functions (in the parameters specifying $a$ and $b$ ) of bounded degree. Rather than describing the method in full generality we will illustrate it by looking at two problems posed in the literature. In a different context the next section will also provide an example of the same basic technique.

Here is a problem posed by Mckenna [Mc]: given two collections $F$ and $G$ of nonvertical segments in $E^{3}$, such that each segment in $F$ (resp. $G$ ) is parallel to the $x z$-plane (resp. the $y z-$ plane) and each segment in $F$ lies above every segment in $G$, find the largest distance $d$ by which $F$ can be shifted downwards until it hits $G$. It is easy to rephrase this problem in the framework outlined above. As it turas out, it is not much more difficult to solve the more general line shifling problem obtained by removing any restriction on the orientation of the segments.

As a starter, we consider the restricted line shifting problem, where both $F$ and $G$ are collections of infinite lines. Each line of $F$ is of the form ( $y=y_{1}, z=a_{1} z+b_{1}$ ), for $i=1, \ldots, m$, and each
line of $G$ is of the form $(z=z, z=c, y+d$, ) fo: $\jmath=1, \ldots, n$ In this particular case we want to compute

$$
\min _{12}\left(a_{1} x_{2}+b_{1}-c_{2} y_{1}-d_{j}\right)
$$

Put $u_{1}=\left(a_{1}, b_{1}, y_{1}, 1\right)$, for $i=1, \ldots, m$ and $v_{2}=\left(x_{1}, 1,-c_{1},-d_{j}\right)$, for $j=1, \ldots, n$ We must now compute $\min _{1}, u_{1} \cdot v_{j}$, which gives us a new problem.

Given two sets of vectors $U^{\prime}=\left\{u_{1} \ldots, u_{m}\right\}$ and $V^{\prime}=\left\{v_{1}, \ldots, v_{n}\right\}$ in $E^{d}$, find

$$
\min \left(u_{1} \quad i, \mid 1 \leq i \leq m, 1 \leq j \leq n\right)
$$

Clearly the minimum must be attained at points $u_{1}$ and $v$, lying on the convex hulls of $U$ and $V$, respectively. So, without loss of generality, suppose that all points $u_{1}, v$, lie on the corresponding bulls. (Note that for the set of vectors arising in the line shifting problem, computation of the convex bulls can be done in time $O(m \log m)$ and $O(n \log n)$, respectively. because each of these sets hes in a 3 -dimensional cross-section of $E^{4}$ ) Next, without loss of generality, assume that $m \leq n$ For each vector $v$, the munimum of $u_{1}, v$ is attained at that point (or points) $u_{1} \in l^{\prime}$ at which $l^{\prime}$ is supported by a hyperplane whose inward drawn normal is $z$. Therefore, we need to preprocess $C^{\prime}$ into a data structure that supports quentes of the above form This is reasonably easy to do in $E^{2}$ and $E^{3}$. In $E^{3}$. for example, we use the standard Gaussian sphere representation, also known as the normal diagram of $l^{\circ}$. That is we define a map on the sphere $S^{2}$ with $O(n)$ regions, so that for each region $R_{1}$ there corresponds a vector $u_{1} \in l$ such that all planes with inward drawn normals in $R_{1}$ support $l$ at $u_{1}$ Next, we construct a data structure which supports $O(\log m)$ point location queries in this sphencal map Such a data structure can be obtained in $O(m \log m$ ) preprocessing We can now determine $\min _{1}\left(u_{1} \cdot v\right)$ in logarithmic time by simply locating $v$ in the map In a total time of $O((m+n) \log \min (m, n))$ we can thus find the required minimum.

Note that this approach is also applicable to the restricted line shifting problem, because the underlying set is essentially 3-dimensional We can thus preprocess $U$ as above; given any vector $v$ we simply remove its fourth coordinate and find the plane supporting $U$ whose inward normal is precisely the truncated $v$. This gives us an $O((m+n) \log \min (m, n))$ time solution to Mckenna's restricted line sbifting problem

Next, we turn to the general line shifting problem Let $A$ and $B$ be two collections, each consisting of $n$ nonvertical segments in $E^{3}$. The problem is to find the smallest positive vertical distance $F(a, b)$ between any pair of segments $a \in A$ and $b \in B$. The function $F(a, b)$ is defined as
follows: if the projections of $a$ and $b$ onto the $x y$-plane intersect at some point $x$, then $F(a, b)$ is equal to $z_{a}-z_{b}$, where $z_{a}$ (resp $z_{b}$ ) is the $z$-coordinate of the point of $a$ (resp. b) projecting into $z$. otherwise $F(a, b)=+\infty$.

Fix two segments $a \in A, x \in B$. Each of them can be specified by six parameters, eg., the coordinates of its two endpoints. Let $a_{1}, a_{2}$ be the endpoints of $a$ and $x_{1}, x_{2}$ be the endpoints of $x$. Let $a_{1}^{;}, a_{2}^{*}, x_{1}^{0}, x_{2}^{;}$be the projections of these four points onto the $x y$-plane. We first find the $t w o$ rational parameters $\alpha, \beta$, satisfying

$$
a_{1}^{*}+\alpha\left(a_{2}^{*}-a_{1}^{*}\right)=x_{1}^{*}+\beta\left(x_{2}^{*}-x_{1}^{*}\right) .
$$

For the projections of $a$ and $x$ to intersect, it is necessary and sufficient that $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$. Clearly both $a$ and $\beta$ are rational functions of $a_{1}, a_{2}^{0}, x_{1}^{-}$, and $x_{2}^{*}$. Once $a$ and $\beta$ bave been found (and lie between 0 and 1) the desired $F(a, \Sigma)$ is equal to

$$
Z(a, \boldsymbol{x})=a_{1}^{2}+\alpha\left(a_{2}^{2}-a_{1}^{2}\right)-x_{1}^{2}-\beta\left(x_{1}^{2}-x_{2}^{2}\right),
$$

where $a_{1}^{z}, a_{2}^{z}, z_{1}^{2}$, and $x_{2}^{2}$ are the $z$-coordinates of the four corresponding points
In other words, regarding each segment $x \in B$ as a point in $E^{6}$, for each $a \in A$, we can express $F(a, x)$ as follows

$$
F(a, x)= \begin{cases}Z(a, x), & \text { if } 0 \leq a(a, x) \leq 1 \text { and } 0 \leq \beta(a, x) \leq 1 \\ +\alpha, & \text { otherwise. }\end{cases}
$$

where $\alpha(a, x), \beta(a, x)$, and $Z(a, x)$ are all rational functions of $a$ and $\boldsymbol{x}$. Now consider the collection $\mathcal{F}$ of $O\left(n^{2}\right)$ rational functions consisting of $\alpha(a, x), 1-\alpha(a, x), \beta(a, x), 1-\beta(a, x)$, and $Z(a, x)-Z(b, x)$. for $a, b \in A$. If we now apply the point location algorithm to the collection $\mathcal{F}$ we will be able to compute $\min _{a \in A} F(a, x)$ by simply locating $x$ in its proper Collins cell.

What does that presuppose? First of all. Theorem 2 tells us that the collection $\mathcal{F}$ should consist of polynomials and not rational functions. This is clearly not a problem any rational $P / Q$ can be replaced in $\mathcal{F}$ by the two polynomals $P$ and $Q$, thus at most doubling the size of the collection. Secondly, we need to have an explicit correspondence between each cell of the Collins decomposition and its associated "winning" segment of $A$ (if any). To do so, the naive solution will do: simply interpret each point of the cas as a eegment in $E^{3}$ and test it against every segment of $A$. Break ties arbitrarily. What is the complexity of this solution? The cas consists of $O\left(\left(n^{2}\right)^{2^{0}-1}\right)=O\left(n^{2^{7}-2}\right)$ algebraic points, so labeling the Collins cells with the proper eegments of $A$ will require $O\left(n^{2^{2}-1}\right)$ time. This is largely dominated by the estimate of the preprocessing time given by Theorem 2 , that is, $O\left(\left(n^{2}\right)^{212}\right)=O\left(n^{2^{23}}\right)$.

We can now use these observations to obtain a subquadratic solution to the line shifung protlem We use a batching strategy inspired by Yon's work on higher-dimensional minimum spanning trees [ [] The technique is somewhat remuniscent of the Four-Russian algonthm for Boolean matrix multiplication (Ahoet a] [AH'] ) - see also (Chazelle [Ch2]) for a generalized version of it. The idea is to partition the collection $A$ into $\left\{n^{1-1 / 2^{13}}\right\}$ subsets of roughly equal size Instead of preprocessing the entireset $A$ for point location, as decribed above, we work on each subset $A$, separately for every segment $x \in B$ we will determine in logarithmic time the minimum distance to $A_{1}$. Repeating this operation for each subset $A$, gives an overall running time of $O\left(n^{1-1 / 2^{13} n} n \log n\right)=O\left(n^{2-1 / 2^{13}} \log n\right)$

Theorem 3. The line shifting problem on $n$ line segments can be solved in $O\left(n^{1999678}\right)$ time

We shall follow a similar approach to solve another problem posed by Mckenna [Mc]: given a simple $n$-gon $P$ what is the longest line segment that can be drawn in the closure of $P^{\prime}$ In Mchenna's terminology the segment is called the biggest stick of $P$. Obvous)y, the biggest stick is not necessarily unique (think of a regular n-gon), so the term actually refers to any segment with the characteristics described above A simple yet crucial observation of McKenna is that any biggest stick must pass through two distunct vertices of $P$ Going through $n$ iterations of a linear vertex-visibilty algonthm (ElGindy and Avis (EA]), an $O\left(n^{2}\right)$ solution to the biggest stick problem follows readily Can one do better"

To begin with. we set the stage for divide-and-conquer by applying the polygon-cutung theorem (Chazelle [Ch1]). In $O(n \log n)$ time we find a diagonal c which partitions $P$ into two subpolygons. $P_{1}$ and $P_{2}$, each of size at least $n / 3$ The diagonal c is a line segment inside $P$ joining two of its vertices (Its computation can be made more efficient using the triangulation algorithm of (Tarjan and Van W'yk (TV]), but it does not have to) Next, we call the algonthm recursively to determine a biggest stick in each of $P_{1}$ and $P_{2}$. What remains to be done is to find a biggest stick crossing the diagonal $c$ and keep the biggest of all three as the output. To look at the problem in dual space will clarify some of the issues.

Consider the dual mapping which puts in oneto-one correspondence the point ( $a, b$ ) and the line $a x+b y+1=0$. If $d$ is the distance from the origin $O$ to the point $p$, the dual of $p$ is the line perpendicular to $O p$ at distance $1 / d$ from $O$ and placed on the other side of $O$. Any line $\ell$ crossing $c$ is thus mapped to a point $\mathscr{C}$ in the dual plane. This point lies in the double wedge $W^{\prime}$ (not containing the origin) formed by the dual lines of the endpoints of $c$. For each such line and
$i=1,2$, let $F_{1}(\rho)$ be the length of the connected portion of $\ell \cap P_{1}$, one of whose endpoints lie on $c$ Clearly the length of a biggest stick $\varepsilon^{\circ}$ through $c$ is given by

$$
\max \left\{F_{1}(\kappa)+F_{2}(\kappa) \mid \ell \text { crossing } c\right\}
$$

As shown in (Cbazelle and Guibas [CG]) each of the functions $F_{1}$ and $F_{2}$ can be represented as a piecerise smooth function such that the projection of its smootb portions form a straight-edge convex subdivision of $W^{\prime}$. The domain of the functions can be clearly extended to the whole plane (setting $F_{i}$ to 0 outside of $W$ ), wbich gives us two convex subdivisions $M$ and $N$ of the plane. These subdivisions encode the set of boundary points of $P$ which are visible from the diagonal $c$. More precisely, an edge of $M$ (resp. $N$ ) encodes the visibility of a vertex of $P_{1}$ (resp. $P_{2}$ ) from $c$ : a vertex of $M$ (resp. $N$ ) is dual to a line passing through $c$ and two vertices of $P_{1}$ (resp. $P_{2}$ ) with no other contact with $P$ between these three intersections. It is shown in [CG] that both $M$ and $N$ have $O(n)$ vertices and can be computed in $O(n \log n)$ time. Mckenna has observed that $F_{1}\left(\ell^{*}\right)+F_{2}\left(\ell^{*}\right)$ is maximized either at a vertex of $M$ or $N$ or at an intersection between an edge of $M$ and an edge of $N$. Finding the biggest stick $\varepsilon^{*}$ can now be regarded as a special case of the following more general problem.

Given two bivariate functions $F(x, y)$ and $G(x, y)$ and two associated convex plana: subdivisions $M$ and $N$, such that $F(\operatorname{resp} G)$ is smooth (or continuous. or convex) over each region of $M$ (resp $N$ ), and such that $F+G$ attains its maximum either at vertices of $M$ of $N$ at intersections of edges of $M$ with edges of $N$, is it possible to determine the maximum of $F+G$ in $o(m n)$ time, where $m$ (resp. $n$ ) denotes the number of vertices of $M$ (resp. $N$ )? By preprocessing each subdivision for point location we can evaluate $F$ and $G$ at any point in $O(\log n+\log m)$ time. This allows us to evaluate $F+G$ at all the vertices of $M$ and $N$ in $O(n \log n+m \log m)$ time. There now remains the more difficult task of testing edges against each other.

It should be clear that our previous batching tecbnique can be applied in much the same way. A few differences are worth noticing, bowever. Let us discuss the problem of computing $\max _{(x, y) \in e} F(x, y)+G(x, y)$ in logaritbmic time, given some edge e of, say, $N$. First, we must recall the geometric meaning of the subdivision $N$. With each edge $e$ of $N$ is associated a vertex $v$ of $P_{2}$ as well as a line segment $\ell$ on the boundary of $P_{2}$ (Figure 1). For this reason, e can be represented as a point $z \in E^{6}$ (two coordinates for $v$ and four for the segment $l$ ). Note that each point of $e=e(z)$ is mapped dually to a line passing through botb the diagonal $c$ and the segment $\ell$. The edge $e(z)$ is "obtajned" by pivoting the line in question around $v$ and scanning all of $\ell$. Given two
edges $a \in M$ and $e(z) \in N$, we define $H(a, z)$ as being 0 if $a$ and $e(z)$ do not intersect and the value of $F(x, y)+G(z, y)$, where $(x, y)=a \cap e(z)$, otherwise. Given an edge $e(z)$ of $N$ we compute $\max _{a \in M} H(a, z)$ by performing a point location in the appropriate cad in $E^{6}$. Which functions should be included in the underlying collection $\mathcal{F}$ ? Let $(p(a, z), q(a, z))$ be the intersection of $a$ and $e(z)$. Since the endpoints of the edge e(z) can be expressed as rational functions of $:$ (ie., of its coordinates), so can $p(a, z)$ and $q(a, z)$. Note that the intersection of $a$ and $e(z)$ can be enforced by the sign of four twoby-two determinants, each also a rational function of $z$. These constraints ensure that the endpoints of $a$ and $e(z)$ he on opposite sides of the intersection of the lines containing the segments.

Let $L$ be line dual to the point $(p(a, z), q(a, z))$ and let $A$ (resp. $B$ ) be the intersections of $L$ with the portion of the boundary of $P_{1}\left(\operatorname{resp} \quad P_{2}\right)$ associated with the edge a (resp. $\ell$ ). If $H(a, z)$ is not zero then it is equal to the length of the segment $|A B|$. The idea is to include all functions $H(a, z)-H(b, z)$ in $\mathcal{F}$, for all edges $a, b \in M$ To keep all functions in $\mathcal{F}$ rational we can include $(H(a, z))^{2}-(H(b, z))^{2}$, instead We also add $10 \mathcal{F}$ the determinants that test for intersections between $\epsilon(z)$ and edges of $M$

Let us mention in passing a general method for dealing with sets $\mathcal{F}$ that include arbitrary algebraic functions (and not just polynomals) Suppose that we have a function $f\left(x_{1}, \ldots, x_{k}\right)$ which is expressed as a root of a univariate polynomial $P(=)$ whore coefficients are poly nomials in $x_{1}, \ldots, I_{k}$ Any zero $\left(x_{1}, \ldots, x_{k}\right)$ of $f$ is also a zero of the polynomial $P(0)$, so we can replace $f$ by $P(0)$ in $\mathcal{F}$. As long as the queries are not zeros of $P(0)$ themselves the Collins decomposition obtained after the replacement will be refined enough for our purposes For example, let $f(x, y, z)=\sqrt{x}-2 \sqrt{y}+3 \sqrt{z}$ Using repeated squaring, we denve the identit:

$$
\left(\left(f^{2}+x-4 y-9 z\right)^{2}-4 x f^{2}-144 y z\right)^{2}=2304 x y z f^{2} .
$$

Therefore we can use the substitute function

$$
x^{2}+16 y^{2}+81 z^{2}-8 x y-72 y z-18 x z .
$$

Returning now to the biggest stick problem, we are about ready to conclude. Once again the cost of precomputing the function on the algebraic points of the cas is dominated by the construction of the data structure. As in the line sbifting problem, $\mathcal{F}$ contains $O\left(n^{2}\right)$ functions, therefore the preprocessing requires $O\left(n^{2^{13}}\right)$ time. The batching trick leads to the following result.

Theorem 4. Computing the biggest slick of a simple $n$-gon can be done in $O\left(n^{1999576}\right)$ tume.

## 6. A Problem on Points in Motion

Atallah [ A ] has posed the following problem Suppose that $n$ points are moving in the plane, with the trajectory of each point described by a polynomial function of the time. What is the first instant at which their convex hull will enter a steady-state, i.e., a combinatorially stable configuration? We shall assume that each polynomial has degree bounded above by a constant. Since the output can be a fairly arbitrary algebraic number, we will bave to content ourselves with a description of that number involving a defining polynomial along with an isolating interval. The naive algorithm consists of computing the steady convex bull in $O(n \log n)$ time [A], and then determining the first time each point will achreve its steady positioning with respect to each edge on the hull Let $e$ be an edge of the hull for each moving point $p$ it suffices to compute the last instant (if any) at which $p$ lies on the line supporting the edge $e$. Let $t(e)$ be the maximum value obtained by this process. The desired answer is the maximum value of $t(\epsilon)$ over all edges $e$ of the convex hull Can this quadratic algorithm be improved? We will show that it can indeed-well, at least theoretically. We will use our generalized point location algorithm to produce an $O\left(n^{2-c}\right)$ time algorithm, for some small positive constant $\epsilon$

Let $l=\left\{p_{1}, \ldots, p_{n}\right\}$ be a set of $n>2$ moving points in the Euclidean plane. We assume the existence of $2 n$ univariate polynomials $x_{1}, y_{1}, \ldots, x_{n}, y_{n}$ of degree $d$ with rational coefficients, such that for each $i(1 \leq i \leq n), x_{1}(t)$ and $y_{1}(t)$ are respectively the $x$ and $y$ coordinates of $p_{1}$ at time $t \geq 0$. Let $p_{1_{1}}, \ldots, p_{14}$ be the points on the boundary of the convex bull of $V$ at time $t$. given in clockwise order, with $i_{1}<\min \left(i_{2}, \ldots, i_{k}\right)$. (If somehow $p_{1}, \ldots, p_{1}$, coincide for some $j, j^{\prime}$ then their indices should appear in the order $i,<\ldots<i, \ldots$.) Let $H(t)$ be the uniquely defined sequence ( $i_{1}, \ldots, i_{k}$ ). Obviously, $H(t)$ converges as $t$ grows to infinity [A]. We define the threshold of $H(+\infty)$ as the smallest value of $t \geq 0$ such that $\left[\left(\forall t^{\prime} \geq t\right) \mid H(t)=H\left(t^{\prime}\right)\right]$.

Let $x_{1}(t)=\sum_{0 \leq j \leq d} a_{1, j} t^{j}$ and $y_{1}(t)=\sum_{0 \leq j \leq d} b_{1, j} t^{j}$, for $i=1, \ldots, n$. Without loss of generality, assume that $H(+\infty)$ is the sequence $(1, \ldots, m)$, for $m \leq n$, and that all $n$ points $\left(a_{i, 0}, \ldots, a_{i, d}, b_{i}, 0, \ldots, b_{i, d}\right)$ of $Q^{2 d+2}$ are pairwise distinct. In $O(n \log n)$ time compute $H(+\infty)[\mathrm{A}]$ and check all pairs ( $p_{i}, p_{1+1}$ ) ( $1 \leq i \leq m$ ) in order to determine the largest $t_{0} \geq 0$ such that, for some $i$, we have $x_{i}\left(t_{0}\right)=x_{1+1}\left(t_{0}\right)$ and $y_{1}\left(t_{0}\right)=y_{1+1}\left(t_{0}\right)$; note that we may bave $t_{0}=-\infty$. Here
(as in the following), undex arithmetic is taken mod $m$. Similarly, we ensure the convexity of the polygon $\left\{p_{1} \ldots, p_{m}\right\}$ by considering the function

$$
f_{1}(q)=\left(y_{1}(t)-y_{1+1}(t)\right) q_{2}+\left(x_{1+1}(t)-x_{1}(t)\right) q_{v}+x_{1}(t) y_{1+1}(t)-y_{1}(t) x_{1+1}(t)
$$

The point $q=\left(q_{x}, q_{y}\right) \in E^{2}$ lies to the right (resp on or to the left) of the oriented line $\left(p_{1}, \overline{p_{1} p_{1}+1}\right)$ iff $f_{i}(q)<0$ (resp. $f_{1}(q)=0$ or $\left.f_{1}(q)>0\right)$. For each $p_{1}(1 \leq i \leq m)$ compute the largest real root of $f_{i}+i\left(p_{1}\right)$ as a polynomial in $t$; discard every case where the polynomial is identically zero. Let $t_{1}$ be the largest value thus obtained (or $-\infty$ if there is none), and let $t_{2}=\max \left(0, t_{0}, t_{1}\right)$. Once $H(+\infty)$ is available, $t_{2}$ can be easily computed in $O(n)$ time. All that remains to be done is to compute the first instants at which each $p_{j}(m<j \leq n)$ lies inside $H(+\infty)$ for good To do so, we allow ourselves some preprocessing Let $q(t)=\left(q_{x}(t), q_{y}(t)\right)$ be a time-varying point in $E^{2}$. with $q_{x}(t)=\sum_{0 \leq j \leq d} q_{j}^{\tau} t^{y}$ and $q_{y}(t)=\sum_{0 \leq, \leq d} q_{j}^{y} t^{z}$. The point $\lambda=\left(q_{0}^{x}, \ldots, q_{d}^{x}, q_{0}^{y} \ldots, q_{d}^{y}\right)$ belongs to $E^{2 d+2}$ and $1 s$ independent of $n$ Let $\operatorname{sign} A=-1$ (resp $=0,1$ ) if $A<0(\operatorname{resp} A=0,>0)$. We define

$$
t(x)=\min \left\{t \in \Re \mid t \geq t_{2} \text { and }(\forall i, 1 \leq i \leq m)\left(\forall t^{\prime}>t\right) \operatorname{sign} f_{1}(q(t))=\operatorname{sign} f_{1}\left(q\left(t^{\prime}\right)\right)\right\}
$$

Clearly. $t(x)$ can be computed in $O(m)$ time Next, we describe a fast algorithm for computing $t(x)$ based on point location

Let $\mathcal{F}=\left\{O_{1}(x, t), \ldots, \theta_{m}(x, t)\right\}$, where $O_{1}(x, t)$ denotes the $(2 d+3)$-variate polynomial of degree $2 d+1$.

$$
f_{1}\left(\sum_{0 \leq 1 \leq d} q_{j}^{x} t^{2}, \sum_{0 \leq: \leq d} q_{j}^{y} t^{2}\right)
$$

with $t$ being the $(2 d+3)$ rd coordinate. Let $K$ be the $\mathcal{F}$-invariant cad of $E^{2 d+3}$ provided by the procedure described in section 2, and let ${K^{\prime \prime}}^{\prime}=\left\{c_{2}, \ldots, c_{\mu}\right\}$ be its base cad (i.e., the induced cad of $E^{2 d+2}$ ). Recall that for each $c_{t} \in K^{\prime \prime}$ the procedure provides us with a sequence of indices (possibly empty) $S_{1}=\left\{l_{1,1}, \ldots, l_{1, k}\right\}$ with the following meaning for any given $\chi \in c_{1}$ the line $\lambda \times E^{1}$ contains an increasing sequence of real roots for the univariate polynomials $\phi_{l_{1,1}}(\lambda, t), \ldots, \phi_{1, \ldots,}(\lambda, t)$. The interpretation of this sequence is trivial: it gives the indices of the lines passing through $p_{1} p_{1+1}$ that are intersected by the trajectory of $x$ in chronological order (from $t=-\infty$ to $t=+\infty$ ). If the sequence is empty then $\chi$ never intersects such a line. Once $K^{\prime}$ has been preprocessed for point location, computing $t(X)$ is atraightforward. Locate the cell $c_{i}$ that contains $\chi$ and check whether the sequence $S_{1}$ is empty. If yes, set $t(\chi)=t_{2}$. If the sequence is not empty, the trajectory of $\chi$
intersects the line passing through $p_{l, \ldots,} p_{l, \ldots, 1}$ at some time $t$ and does not intersect any other such line subsequently. We obtain $t$ by computing the largest real root of $\phi_{1, \ldots,}(x, t)$ as a polynomial in $t$ (which must exist). Finally we set $t(\lambda)=\max \left(t_{2}, t\right)$. From Theorem 2 we immediately conclude that in $O\left(m^{2 d+0}\right)$ time it is possible to construct a data structure so that the function $t(\lambda)$ can be evaluated at any point $\chi \in E^{2 d+2}$ in $O(\log m)$ time.

We are now ready to attack Atallah's problem, using the same batching trick used in the previous section. This leads to an algorithm with a running time of $O\left(n^{2-1 / 2^{2 d+\theta}} \log n\right)$. We omit the details.

Theorem 5. In $O\left(n^{2-1 / 4^{+3}}\right)$ time it is possible to compute the threshold time of the steady-state conver hull of $n$ points moving in the plane according to polynomial functions of time of marimum degree $d$.

We close this section with a few remarks about Atallah's problem. Our technique clearly is general enough to be applied to other problems (e.g., closest/farthest pairs). An interesting question is to determine whether a more ad hoc treatment of these problems might lead to a more efficient solution. For example, a continuity argument easily shows that ensuring the local coherence of the steady-state Voronoi diagram is sufficient to compute its threshold (e.g., checking the nonzero length of its edges). It is then fairly simple to devise an $O(n \log n)$ algorithm for computing the steady-state Voronoi diagram of $n$ moving points as well as its threshold Note that the same argument can be made for convex bulls if all the points are guaranteed to he on it. One essential feature of these easy cases is that the output contains all the input Is this in general a necessary condition of efficiency?

## 7. A Discussion of the Batching Technique

In the preceding sections we bave given a few examples of techniques for obtaining (slightly) subquadratic solutions to a large variety of geometric problems which admit trivial exhaustive quadratic solutions. Many other problems yield to our technique. For example,
(i) Given a eet of $m$ red objects (algebraic curves, surface patches, etc.) and $n$ blue objects, does any red object intersect any blue object? Hopcroft's problem, mentioned above, is such a problem; detecting intersection beween a collection of red segments and a collection of blue regronts is another example.
(ii) Given $m$ rays and $n$ triangles in 3 -space, find the first triangle bit by each of the rays, or alternatively; find the number of triangles stabbed by each ray.
(iii) Given a collection of $n$ (disjoint) triangles in three dimensions, find all pairs of mutually visible vertices Here, we regard each pair of vertices as a query point $z$ in $E^{6}$ and each triangle $د$ corresponds to a Boolean predicate that expresses the fact that $z$ is not blocked by $\Delta$ Batching the triangles and using our technique we can obtain a subcubic solution (a cubic solution being trivial).

Refiecting over our results, we can see a whole spectrum of problems amenable to our techniques
(A) The simplest of them admit linear or near-linear data structures and can be solved in $O(n \log n)$ time, e.g., the closest-pair problem in the plane.
(B) Next in line, we have problems for which we bave efficient, polynomial-size data structures and for which the batching technique is very effective, e.g., computing the diameter of $n$ points in $E^{3}$, Hoperoft's problem
(C) Then we bave the problems of the type discussed in this paper. sufficiently complicated to offer no alternatives but computing Collins decompositions (in theory, that is).
(D) Finally, we bave problems for which it is not clear even how to obtain a Collins decomposition of the form used in this paper for example, consider a variant of problem (iii) above in which we want all pairs of mutually visible triangles (i e., each triangle containing at least one point which is visible from the other one) Can this problem be solved in subcubic time using our techmque? As it turns out, the best solution currently known runs in time $O\left(n^{4} \alpha(n)\right)$ (Mckenna and O'Rourke (MO])

As another example, consider the line shifting problem discussed in section 5 . We have presented two versions of the problem, an easy one (of type (A)) involving shifting infinite lines parallel to the $x z$ - or $y z$-plane and a difficult version (of type (C)) involving two arbitrary collections of line segments. What can be said about the intermediate type (B) version, in which one set $F$ consists of $m$ segments parallel to the $y z$-plane and the other set $G$ consists of $n$ segments parallel to the $x:-$ plane? Let $\alpha$ be some positive integer. Batch the segments of $F$ into a subsets of size roughly $\mathrm{m} / \alpha$. By using a standard slab decomposition of the $x y$-projections, we obtain roughly $2 \mathrm{~m} / \alpha$ horizontal slabs in the $x y$-plane (for each subset), each consisting of $O(m / \alpha)$ subsegments which can be treated as infinite lines for any query segment of $G$ whose projection falls in that slab. By maintaining the subeegments in each slab as a balanced binary tree, and by storing the four-dimensional convex hull associated with the lines containing the subsegments in each subtree (as in section 5 ), it is easy
to answer each query from $G$ in time $O\left(a \log ^{2} m\right)$. The preprocessing time is $O\left(\left(m^{2} / a\right) \log ^{2} m\right)$. Choosing $a=\lfloor m / \sqrt{n}\rfloor$ gives a total running time of $O\left(m \sqrt{n} \log ^{2} m\right)$.

A final comment concerns the use of probabilistic algorithms for the problems studied in this paper. Such a method has been developed recently in (Haussler and Welzl [HW], Clarkson [Cl1, Cl2]. Edelsbrunner, Guibas, and Sharir [EGSh]). It can be roughly described as a divide-and-conquer paradigm which uses random sampling of a small subset of the input objects to obtain a cell decomposition such that, with high probability, each cell contains (or intersects) only a small number of the given objects. If the queries are known in advance (which is the case in the problems studied in section 5) then we can partition them among the cells of the decomposition, so that each point can interact only with a small number of objects. We believe that one can develop a general framework for blending this randomized method with our Collins decomposition technique, and obtain (slightly) improved probabilistic algorithms for such problems We leave this as an open problem

## 8. Conclusions

The contribution of this work has been two-fold: generalize Dobkin and Lipton's point location algoritbm, and demonstrate some theoretical applications of the new algorithm. Further work includes the (difficult) problem of drastically reducing the high space-complexity of the generalized point location algorithm. Even the case of hyperplanes is still open. For an $O(\log n)$ query time, the $O\left(n^{2^{4}-1}\right)$ space requirement of the Dobkin-Lipton data structure was lowered to $O\left(n^{d+c}\right)$, for any $\varepsilon>0$, by Clarkson [Cl1], and recently to $O\left(n^{d}\right)$ by Chazelle and Friedman [CF]. Unless nontrivial lower bounds can be established, bowever. these bounds still leave much to be desired The idea of a cylindrical decomposition arises naturally in the elimination of quantifiers in Tarski sentences. It is less than obvious, however, that it is the right approach for point location. Milnor's bounds suggest that a single exponential might be achievable: this would be a considerable improvement over our method.

Acknowledgments: We wish to thank Johan Swenker for useful comments on this work.

## REFERENCES

[AHC] Aho, A V., Hopcroft, J.E, Ullman, J D The design and anolysis of computer algorithms, Reading, MA, Addison-Wesley, 1974
[Ar] Arnon, D.S. Algorithms for the geometry of sems-algebrac sets, Tech. Rep. 436, Computer Science Dept., University of Wisconsin, Madison, 1981.
[ACM1] Arnon, D.S., Collins, G.E., McCallum, S. Cylandrical algebraic decomposition I: the basic algorithm, SIAM J. Comput., 13 (1984), 865-877.
[ACM2 Arnon, D.S., Collins, G.E., McCallum, S. Cylindrical algebrac decomposition II. an adjacency algorithm for the plane. SIAM J. Comput., 13 (1984), 878-889.
[A] Atallah, M.J Dynamic computational geometry, Comput Math with Applications, 11 (1985), 1171-1181
[BKR] Ben-Or, M., Kozen, D. Reir. J The complezity of elementory algebra and geometry, Proc 16th Annu. ACM Symp on Theory of Computing (1984). 457-464
[BT] Brown, W., Traub. JF On Euchd's algorithm and the theory of subresultants, J ACM, 18 (1971). 505-514
[Ch1] Chazelle, B. A theorem on polygon cuting with applications, Proc of 23 rd Annu JEEE Symp on Foundat of Computer Science (1982). 339-349
[Ch2] Chazelle. B. Some technaques for geometric searching with implicit set representations. Acta Informatica. 24 (1987), 565-582
[CF] Chazelle, B., Friedman, J A deterministic vew of nandom sampling and its use in geometry. manuscript, 1988.
[CG] Chazelle, B., Guibas, L.J. Visbality and intersection problems in plane geometry, Proc. 1st Ann. ACM Sympo6. Comput. Geom. (1985), 135-146. To appear in Disc. Comput. Geom.
[C11] Clarkson, K.L. New applications of random sampling in computational geometry, Disc. Comp. Geom., 2, (1987), 195-222.
[C12] Clarkson, K. Applications of rendom sampling in computational geometry, Il, Proc. 4th Ann. ACM Sympos. Comput. Geom. (1988), to appear.
[Coh] Cohen, P.J. Decision procedures for real and p-adic fields, Comm Pure and Applied Math. 22 (1969). 131-151.
[Co] Cole, R. Searching and storing similar lasts, J Algorithms, 7 (1986), 111-119
[C] Collins, G.E., Quantifier elimination for real closed fields by cylindric algebrac decomposition. Proc. 2nd GI Conf. on Automata Theory and Formal Languages, Springer-Verlag, LNCS 35 , Berlin (1975), 134-183
[CL] Collins, G.E., Loos, R Polynomial real root isolation by differentiation, Proc. ACM Symp. on Symbolic and Algebraic Computations, Yorktown Heights, NY (1976), 15-25.
[DL] Doblin, D.P., Lipton, R.J. Multidimenssonal searching problems, SIAM J. Comput. 5 (1976), 181-186.
[DS] Dobkin. D.P., Silver. D. Recipes for Geometry and Numerical Analysıs - Part I: An Empartal Study. Proc. 4th Ann ACM Sympos Comput. Geom. (1988), to appear
[EGS] Edelsbrunner. H., Guibas. L.J., Siolfi. J. Optimal point location in a monotone subdituston, SIAM J. Comput., 15 (1986), 317-340
[EGSh] Edelsbrunner. H., Guibas. L.J., Sharir, M. The complezity of many faces in arrangements of lines and of segments, Proc 4th Ann. ACM Sympos. Comput. Geom. (1988), to appear.
[EA] ElGindy, H., Avis, D A linear algorithm for computing the visibility polygon from a point, J of Algorithms, 2 (1981). 186-197.
[HW] Haussler, D., Welz], E. Epsilon-nets and simplex range queries, Discrete Comput Geom. 2 (1987).
[HHK] Bofmann, C.M., Hopcroft, J.E., Karasick. M.S. Touards Implementing Robust Geometric Computations, Proc. 4th Ann. ACM Sympos. Comput. Geom. (1988), to appear.
[Ma] Mabler, K. An inequality for the discriminant of a polynomial, Michigan Math. J., 11 (1964), 257-262.
[Mc] McKenna, M. The brggest stick problem, First Computational Geometry Day, New York University, Sept. 1986.
[MO] McKenna, M., O'Rourke, J. Arrangements of lines in 3-space: a data structure with applications, Proc. 4th And. ACM Sympos. Comput. Geom. (1988), to appear.
[Mi] Milnor, J. On the Betti numbers of real varmeties, Proc Amer Math Soc. 15 (1904), 275-280
[M] Monk, L Elementary recurstue decision procedures, PbD thesis, University of Califorma, Berkeley, 1974
[PS] Preparata, F.P., Shamos, MI Computational geometry: an introduction, Springer-Verlag. New York, NY, 1985.
[ST] Sarnak, N., Tarjan, R.E Planar point location using persistent search trees, Comm. ACM 29 (1986), 669-679
[S] Seidenberg, A A neu decasion method for elementary algebra, Annals of Math, 60 (1954), 365-374
[SS] Schwartz. J.T.. Sharir, M On the "piano movers" problem II: Generaliechniques for computing topological properties of real algebrazc manifolds. Ady in Appl. Math. 4 (1983), 298-351
[T] Tarski, A. A decision method for elementary algebro and geometry. Univ of Calif Press. 1948. 2nd edition, 1951
[TV'] Tarjan, R.E.. Van Wyk, C.J An $O(n \log \log n)$-time algorithm for trangulating simple polygons, SIAM J. Comput., 17 (1988). 143-178
[W] van der Wiaerden. B.L Modern Algebra. Ungar Co.. New York. 1953
[Y] Yao, A C. On constructing minimum spanning tree in $k$-dimensional spoce and related problems. SIAM J Comput.. 11 (1982). 721-736

MY] Yao. A.C., Yao, F.F. A general approach to d-dimensional geometric queries. Proc. Iith Ann. ACM Symp. on Theory of Computing (1985), 163-168


NYU COMPSCI TR-369
Chazelle, Bernard
An algorithm for
generalized point location and its applications

NYU COMPSCI TR-369
Chazelle, Bernard

- An algorithm for
generalized point location
- and its applications


This hook may be kept
FOURTEEN DAYS
A fine will be chased for enct day the book is kept overtime.



