Point location in zones of k-flats in arrangements

Point Location in Zones of k–Flats in Arrangements

M.T. de Berg, M. van Kreveld, J. Snoeyink

RUU-CS-91-09
April 1991

Utrecht University

Department of Computer Science

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

Point Location in Zones of k–Flats in Arrangements

M.T. de Berg, M. van Kreveld, J. Snoeyink

Technical Report RUU-CS-91-09
April 1991

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924–3275

Point Location in Zones of k-Flats in Arrangements*

Mark de Berg Marc van Kreveld Jack Snoeyink†

Abstract

Let $\mathcal{A}(H)$ be the arrangement of a set H of n hyperplanes in d-space. A k-flat is defined to be a k-dimensional affine subspace of d-space. The *zone* of a k-flat f with respect to H is the closure of all cells in $\mathcal{A}(H)$ that intersect f. In this paper we study some problems on zones of k-flats. Our most important result is a data structure for point location in the zone of a k-flat. This structure uses $O(n^{\lfloor d/2 \rfloor + \epsilon} + n^{k+\epsilon})$ preprocessing time and space and has a query time of $O(\log^2 n)$. We also show how to test efficiently whether two flats are visible for each other with respect to a set of hyperplanes.

1 Introduction

The subdivision of d-space into connected pieces — usually called faces — of various dimension, induced by a set H of hyperplanes, is called the *arrangement* $\mathcal{A}(H)$ of H. This concept was introduced to computational geometry by Edelsbrunner, O'Rourke and Seidel [8] (see also [7]). They showed how to construct an arrangement optimally, and proved the so-called *zone theorem*, a combinatorial bound on the maximum complexity of the zone of a hyperplane. The *zone* of a hyperplane h consists of all faces of cells in $\mathcal{A}(H)$ that are supported by h. See Figure 1 for an example in 2-space. The zone of the line h consists of all bold-face segments and vertices, together with the shaded cells. Zones are important in several contexts, as the efficiency of some algorithms depends on the size of zones. In [8], the bound on the complexity of the zone of a hyperplane guarantees optimal construction time of arrangements in d-space. In [3], a bound on the complexity of the vertical decomposition of the zone of a plane in 3-space is used to improve range searching in some cases. As a third application, observe that the zone of a hyperplane h defines exactly the region that is visible from h, where the other hyperplanes are the obstacles. Therefore, zones are suitable for solving some visibility problems. Lately, an upper bound on the *zone of algebraic hypersurfaces* has been shown [1]. This result has the following important application to lines in 3-space. To distinguish between the $O(n^4)$ isotopy classes induced by n lines, the Plücker coordinates of a line and the Plücker hypersurface in 5-space can be used. The total complexity of the zone of the Plücker hypersurface is $O(n^4 \log n)$, which gives an $O(n^{4+\epsilon})$ size data structure that solves the problem of distinguishing between isotopy classes of lines [2].

*This research was supported by the ESPRIT Basic Research Action No. 3075 (project ALCOM). Research of the first author was also supported by the Dutch Organization for Scientific Research (N.W.O.).
Authors address: Department of Computer Science, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, the Netherlands.

†On leave from the Department of Computer Science of the University of British Columbia.

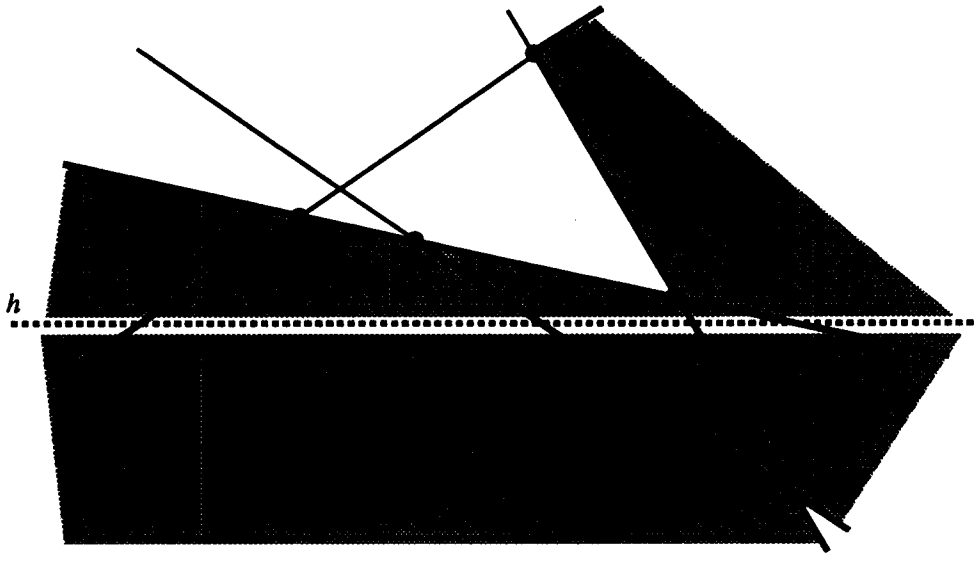

Figure 1: The zone of a line h in the plane.

In this paper we generalize the notion 'zone of a hyperplane' to 'zone of a k-flat', where a k-flat is defined to be the intersection of $d-k$ hyperplanes with linearly independent normal vectors ($0 \leq k \leq d-1$), see [7].

Definition 1 *The* zone of a k-flat f *with respect to* H, *denoted by* $zone(H, f)$, *is the subarrangement of* $\mathcal{A}(H)$ *formed by the closure of all cells of* $\mathcal{A}(H)$ *that are intersected by* f.

Thus for a point p, the $zone(H, p)$ is the convex polytope formed by the intersection of the closed halfspaces that contain p and are bounded by the hyperplanes of H. For a hyperplane h, the $zone(H, h)$ is the zone of a hyperplane as defined in [7, 8].

We concentrate on algorithmic aspects of zones of k-flats rather than the combinatorial side. We obtain an efficient algorithm for point location in the zone of a k-flat. The data structure has preprocessing time and space $O(n^{\lfloor d/2 \rfloor + \epsilon} + n^{k+\epsilon})$ (for any $\epsilon > 0$), where n is the size of the set H of hyperplanes. Notice that the first term of the size of the structure is close to the size of one single cell in the arrangement, and the second term is close to the number of cells in the k-flat itself. For almost all k this is considerably less than the size of the zone itself! With this structure it is possible to determine in $O(\log^2 n)$ time if a query point lies in the zone and, if so, in which cell of the zone it lies. The problems of point location in a full arrangement and in a convex polytope have been studied before [4, 5]. The preprocessing time and space of these structures is $O(n^{d+\epsilon})$ and $O(n^{\lfloor d/2 \rfloor + \epsilon})$, respectively. The query time is $O(\log n)$.

We also investigate the following problem: Given a k_1-flat f_1, a k_2-flat f_2, and a set H of n hyperplanes in d-space, determine whether f_1 and f_2 can see each other with respect to H. In other words, determine whether there are points $p_1 \in f_1$ and $p_2 \in f_2$, such that the segment p_1p_2 does not properly intersect any hyperplane in H. We obtain efficient algorithms

for this problem using two techniques of reducing the dimension of the problem, together with linear programming and the structure for point location in the zone. The precise bounds of the algorithm are given in Corollary 1.

Both our point location and visibility algorithms are based on sampling. Lately this technique has been used to solve a variety of problems [2, 3, 4, 5, 9, 14]. The rough idea of random sampling is as follows. Let H be a set of n hyperplanes in d-space. Choose a random subset R of H of size r. Consider the arrangement $\mathcal{A}(R)$, and triangulate it (that is, partition the cells into simplices). The triangulated arrangement has complexity $O(r^d)$, and with high probability, any simplex is intersected by only $O(n \log r/r)$ hyperplanes of H. This property can be used to define subproblems of the original problem recursively. For a more detailed treatment of random sampling and its applications we refer to [4, 5]. Recently, the following important result on sampling has been shown. A sample R of size $r < n^{1-\delta}$, for any fixed $\delta > 0$, can be constructed deterministically in $O(nr^d)$ time, such that any simplex in the triangulated arrangement $\mathcal{A}(R)$ is intersected by only $O(n/r)$ hyperplanes of H [10]. The main feature of our algorithms lies in the fact that we do not triangulate $\mathcal{A}(R)$ in d-space, but in the k-flat. This causes the efficiency of the algorithms.

The rest of this paper is organized as follows. In Section 2 we show some easy properties and notation for zones of k-flats. In Section 3 the algorithm for point location in the zone of a k-flat is given, and its correctness proved. Section 4 deals with visibility among k-flats. Conclusions and open problems are given in Section 5.

2 Basic properties

In this section we present some easy properties on zones of k-flats. We start with a lemma that gives an alternative definition of the zone of a k-flat, and which will be used in the next section. We also give an upper bound and a lower bound on the maximum size of the zone,

Lemma 1 *A point q lies in the zone of a k-flat f with respect to H if and only if there is a point p in f such that the line segment pq does not properly intersect any hyperplane of H.*

Proof: Follows immediatly from the convexity of the cells in $\mathcal{A}(H)$. □

Such a point p is called a *witness* for q. The lemma shows that zones are useful for visibilty problems, where the obstacles are hyperplanes.

Lemma 2 *For the maximum complexity of the zone of a k-flat f with respect to a set H of n hyperplanes in d-space, $z_d^k(n)$, we have: $z_d^k(n) = \Omega(n^{\lfloor (d+k)/2 \rfloor})$ and $z_d^k(n) = O(\min(n^{\lfloor d/2 \rfloor + k}, n^{d-1}))$.*

Proof: Observe that there are $O(n^k)$ cells in the zone of a k-flat. Each cell has complexity $O(n^{\lfloor d/2 \rfloor})$ by the upper bound theorem [7]. Furthermore, the zone of a $(d-1)$-flat has complexity $O(n^{d-1})$ [7, 8].[1] This proves the upper bound.

To prove the lower bound, construct a grid-like structure using k groups of hyperplanes perpendicular to the k-flat f, and perpendicular to hyperplanes in other groups. Each group

[1] We are aware of the fact that the original proof in [7, 8] is not correct. Recently, this proof has been repaired, and new proofs for the zone theorem have been found.

has size $\lfloor n/(2k) \rfloor$. Clearly, there are $\Theta(n^k)$ cells intersected by f. Consider a $(d-k)$-flat g perpendicular to f. Construct a polytope in g with $n/2$ facets, and with $\Theta(n^{\lfloor (d-k)/2 \rfloor})$ vertices (see [7]). Place the polytope such that the intersection between f and g is a point in the interior of the polytope. Then choose for each facet a hyperplane that contains it, and which is parallel to the k-flat f. Every vertex of the polytope extends to a k-flat parallel to f, and each one of them is in the boundary of each cell of the grid-like structure. Therefore, the zone of f contains $\Theta(n^{\lfloor (d+k)/2 \rfloor})$ k-faces. □

The construction for the lower bound in the above lemma results in a degenerate arrangement of hyperplanes. This restriction can easily be removed by a slight perturbation of each hyperplane.

To actually construct the zone of a k-flat, one could construct the full arrangement $\mathcal{A}(H)$ and then remove the faces that are not in the zone. There are alternative ways to compute the zone of a k-flat that are more efficient. However, in Section 3 we give a superior technique to perform point location in the zone without actually constructing it.

For the sake of completeness we define the notion 'zone of a polytope P in a k-flat'. This will be important in the next section.

Definition 2 *The* zone of a polytope P in a k-flat f *with respect to* H *is the subarrangement of* $\mathcal{A}(H)$ *formed by the closure of all cells of* $\mathcal{A}(H)$ *that intersect* P.

3 Point location in the zone

For a set H of n hyperplanes and a k-flat f in d-space, the problem of point location in the zone is defined as follows. Preprocess H and f, such that for any given query point q, one can determine efficiently whether q lies in $zone(H, f)$ and, if so, in which cell of the zone q lies. If q lies on the boundary of one or more cells of the zone, then one of these cells should be found. Point location in the zone of a polytope P in a k-flat f is defined analogously.

In this section we solve these problems using sampling. The algorithm returns a witness if the query point lies in the zone. This witness is such that it uniquely determines a cell of the zone that contains this query point. Rather than constructing the whole zone, we construct a tree that uses considerably less preprocessing time and space. Let H be a set of hyperplanes and let f be a fixed k-flat in d-space. Then $\overline{H}$ denotes the set $\{\overline{h} \mid \overline{h} = h \cap f \text{ where } h \in H\}$, and $\mathcal{A}(\overline{H})$ is the arrangement in f formed by $\overline{H}$.

Construct a sample $\overline{R}$ of size r, where $\overline{R} \subset \overline{H}$, and r is a sufficiently large constant. We triangulate the arrangement $\mathcal{A}(\overline{R})$, for instance with a bottom-vertex triangulation. The triangulated arrangement $\mathcal{A}(\overline{R})$ consists of $O(r^k)$ simplices, and $\overline{R}$ can be constructed deterministically such that each simplex is intersected by $O(n/r)$ $(k-1)$-flats of $\overline{H}$, and, thus, hyperplanes of H. Each simplex s partitions H into two subsets H_s and H'_s. The subset H_s contains the hyperplanes of H that properly intersect s, and H'_s contains the remaining hyperplanes. Notice that $R \subseteq H'_s$. Let c_s be that cell of $\mathcal{A}(H'_s)$ that contains the interior of the simplex s. An illustration of these definitions is given in Figure 2, where the upper part of the cell c_s in 3-space is shown. The planes h_1 and h_2 are in H'_s, and they contribute to c_s. The plane h_3 is in H_s, because it intersects the shaded triangle s properly.

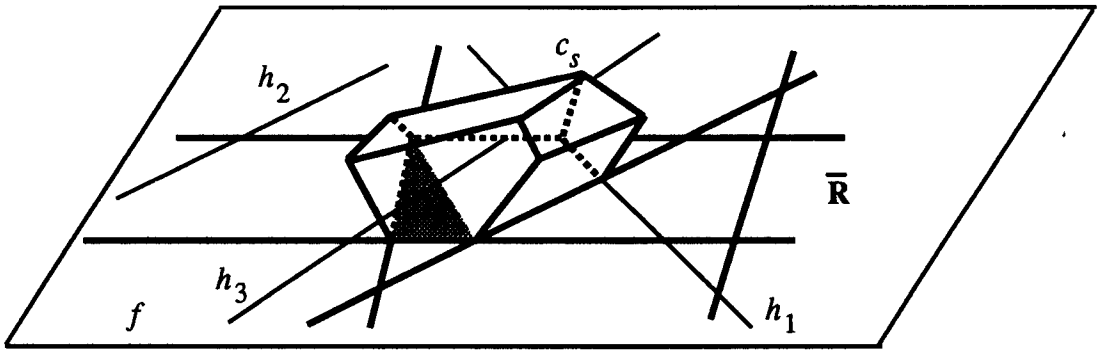

Figure 2: Situation for a 2-flat in 3-space: the polytope c_s for the shaded triangle.

Lemma 3

(i) *If a point q lies in $zone(H, f)$ then there is a simplex s in the triangulated arrangement $\mathcal{A}(\overline{R})$ such that q lies in c_s.*

(ii) *For all simplices s in the triangulated arrangement $\mathcal{A}(\overline{R})$, such that $q \in c_s$, we have: $q \in zone(H, f)$ if and only if $q \in zone(H_s, f \cap c_s)$.*

Proof:

(i) Suppose that q lies in $zone(H, f)$. By Lemma 1 there is a witness $q' \in f$, such that the line segment qq' does not properly intersect any hyperplane of H. Consequently, q and q' lie in the same cell of $\mathcal{A}(H)$. Let s be the simplex in the triangulated arrangement $\mathcal{A}(\overline{R})$ that contains q'. Then $q \in c_s$, because q and q' lie in the same cell of $\mathcal{A}(H)$, and this cell is contained in c_s.

(ii) Let s be a simplex in the triangulated arrangement $\mathcal{A}(\overline{R})$, such that $q \in c_s$.
$\Rightarrow$: Suppose that $q \in zone(H, f)$. Then there is a witness $q' \in f$, such that the line segment qq' does not properly intersect any hyperplane of H. Because $q \in c_s$, so must q'. Thus $q' \in f \cap c_s$, and no hyperplane of $H_s \subseteq H$ properly intersects the line segment qq'. Hence, q' is a witness for q, and q lies in $zone(H_s, f \cap c_s)$.
$\Leftarrow$: Suppose that q lies in $zone(H_s, f \cap c_s)$. Then there is a witness $q' \in f \cap c_s$, such that the line segment qq' does not intersect any hyperplane of H_s properly. Because q and q' both lie in c_s, and all hyperplanes in H'_s do not intersect c_s, it follows that the line segment qq' does not properly intersect any hyperplane of $H = H_s \cup H'_s$. Furthermore, $q' \in f \cap c_s \subseteq f$, thus q' is a witness for q, and $q \in zone(H, f)$.

□

When we consider the zone of a polytope P in a k-flat f we define $\mathcal{A}_P(\overline{H})$ to be that subarrangement of $\mathcal{A}(\overline{H})$ that is the closure of all cells of $\mathcal{A}(\overline{H})$ that have a non-empty intersection with P. Lemma 3 remains valid after replacing f by P and $\mathcal{A}$ by $\mathcal{A}_P$.

Lemma 3 gives the recursive property on which the structure for point location in the zone of f is based. The structure for $zone(H, f)$ is a tree $\mathcal{T}$ of degree $O(r^k)$. Let δ be the root of $\mathcal{T}$. The root δ has one child γ_s for every simplex s in the triangulated arrangement $\mathcal{A}(\overline{R})$. With γ_s we store the simplex s and an associated structure for deciding whether a point lies in c_s. The child γ_s is the root of a recursively defined tree for $zone(H_s, f)$. If the number of hyperplanes in H is smaller than some constant, we do not take a sample, but we store the full arrangement $\mathcal{A}(H)$.

This recursive definition does not completely correspond to the recursive property of Lemma 3 (ii): the subtree rooted at γ_s is defined for $zone(H_s, f)$ instead of for $zone(H_s, f \cap c_s)$. To remedy this, we will fill in witnesses at some cells in the arrangements stored in the leaves in such a way, that any query point finds a witness if and only if it lies in the zone. To this end, impose an arbitrary order on the children of each node. A query with a point q should continue in the *first* child γ_s for which the query point lies in the polytope c_s. To finish the preprocessing, consider the arrangement $\mathcal{A}(\overline{H})$, and for each cell, take one point p in its interior. Locate p in the arrangement of the leaf where the search ends, and store p as a witness with the cell of this arrangement that contains p. (It may take too much preprocessing time to search with p for all children for which $p \in c_s$.)

A query with a point q is performed as follows. Start at the root δ. Find the first child γ_s (with respect to the chosen order on children) for which p lies in c_s; this is determined by searching in the associated structure of each child of δ. Continue the search recursively at this child. If q does not lie in c_s for any child, then q does not lie in the zone. If the current node is a leaf, then we locate q in a cell in the associated arrangement. If there is a witness stored at the cell, then q lies in the zone and we return this witness. Otherwise, q does not lie in the zone. The filling in of witnesses is correct, because a query point that lies in the zone will follow the same path down $\mathcal{T}$ as its witness. A query point that does not lie in the zone will, at some point, take a different path down $\mathcal{T}$ than any witness.

Theorem 1 *For any $\epsilon > 0$, a set H of n hyperplanes and a k-flat f (or polytope P in a k-flat) in d-space can be preprocessed in $O(n^{\lfloor d/2 \rfloor + \epsilon} + n^{k+\epsilon})$ time and space, such that point location queries in the zone can be performed in $O(\log^2 n)$ time.*

Proof: Testing whether a query point lies in a convex cell determined by the intersection of n halfspaces in d-space can be performed in $O(\log n)$ time, after $O(n^{\lfloor d/2 \rfloor + \epsilon})$ preprocessing time and space (for any $\epsilon > 0$), see [4]. Therefore, the initial preprocessing $S(n)$ of our structure satisfies the following recurrence:

$$S(n) = O(r^k) \cdot S(n/r) + O(r^k) \cdot O(n^{\lfloor d/2 \rfloor + \epsilon})$$

$$S(O(1)) = O(1).$$

This solves to $S(n) = O(n^{\lfloor d/2 \rfloor + \epsilon} + n^{k+\epsilon})$ preprocessing for any $\epsilon > 0$, if r is a large enough constant. Additionally, $O(n^k)$ queries, each taking $O(\log^2 n)$ time (see below) are required to fill in the witnesses in $\mathcal{T}$.

The query time $Q(n)$ satisfies the following recurrence:

$$Q(n) = Q(n/r) + O(r^k) \cdot O(\log n)$$

$$Q(O(1)) = O(1),$$

which solves to $Q(n) = O(\log^2 n)$ time. □

4 Visibility among k-flats

Since the zone of a k-flat f defines the region from which f is visible (without looking through a hyperplane), it is natural to use zones to solve visibility problems in arrangements of hyperplanes. In particular, we consider the problem: given a k_1-flat f_1 and a k_2-flat f_2, with $k_1 \leq k_2$, can f_1 and f_2 see each other?

Definition 3 *Flats f_1 and f_2 are* visible *(for each other) with respect to H if and only if there are points $q_1 \in f_1$ and $q_2 \in f_2$ such that the segment q_1q_2 does not properly intersect any hyperplane of H. The points q_1 and q_2 are called* witnesses.

Before we consider solutions that use our query structure, we note some useful facts about affine spaces. Recall that a k-flat is an affine space of dimension k, which is the translation of a linear subspace spanned by k linearly independent vectors. The *join* of flats f_1 and f_2 is the affine space of smallest dimension that contains both flats. The join of a k_1-flat and a k_2-flat has dimension at most $k_1 + k_2 + 1$; it is formed by taking the linear subspace spanned by the vectors defining flats f_1 and f_2, together with a vector from a point in f_1 to a point in f_2. This set of vectors is translated to contain a point in f_1. The next lemma shows that if k_1 and k_2 are small compared to the dimension d, then one can solve the visibility problem in the join of f_1 and f_2.

Lemma 4 *Given an instance of the visibility problem with k_1-flat and k_2-flat in d-space, we can ensure that $d \leq k_1 + k_2 + 1$ in linear time.*

Proof: The join of a k_1-flat and a k_2-flat is an affine subspace of dimension $d' \leq k_1 + k_2 + 1$ that contains both flats. Since any line segment in d-space that is a witness to visibility is contained in the join, it is enough to solve the problem in the join. To reduce the hyperplane equations to this subspace in linear time is straightforward. □

The lemma above can also be used when the visibility problem for *polytopes in k-flats* is considered. If one is investigating the visibility of entire flats, one can always reduce dimension of the space and possibly the dimension of the flats to the case where $d = k_1 + k_2 + 1$.

Lemma 5 *Given an instance of the visibility problem with a k_1-flat and a k_2-flat in d-space, if $k_1 + k_2 + 1 \neq d$ then one can find in linear time an equivalent instance of the visibility problem with fewer dimensions.*

Proof: If $d > k_1 + k_2 + 1$, then Lemma 4 can be used to reduce the dimension.

If $d < k_1 + k_2 + 1$, then the join of the k_1-flat f_1 and the k_2-flat f_2 has dimension strictly less than $k_1 + k_2 + 1$; either the flats intersect or the vectors that define them are not linearly independent. We can test intersection in constant time.

If f_1 and f_2 do not intersect, we find a direction v that is contained in both flats (by Gaussian elimination) in constant time. Let us call v the vertical direction. We can partition

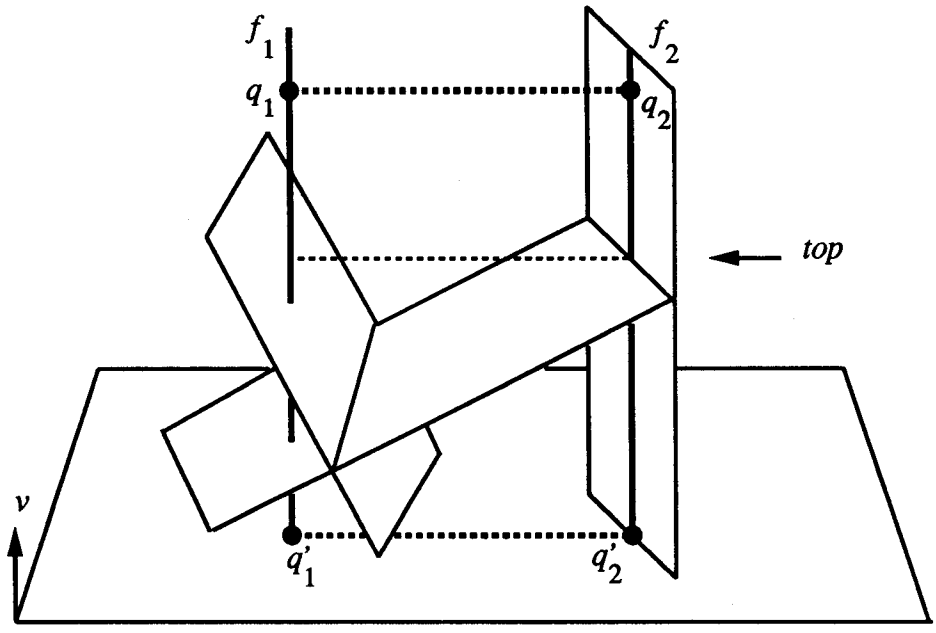

Figure 3: Illustration of the proof of Theorem 5.

the set of hyperplanes into the vertical hyperplanes H_v, those containing a line parallel to v, and the non-vertical hyperplanes $H_{non-v} = H - H_v$. Notice that every non-vertical hyperplane intersects every vertical line.

In linear time, we can project the flats f_1 and f_2 and the vertical hyperplanes H_v in the direction v to form an instance of the visibility problem with a $(k_1 - 1)$-flat f'_1, a $(k_2 - 1)$-flat f'_2 and a set H'_v of hyperplanes in $(d-1)$-space. We can show that this reduced problem is equivalent to the original: If f'_1 and f'_2 can see each other, then there are witness points q'_1 and q'_2 for the reduced problem. The vertical lines that project to q'_1 and q'_2 are contained in f_1 and f_2; any two points on these lines can see each other with respect to H_v. Furthermore, above some height *top*, both lines are above all hyperplanes of H_{non-v}. Choose two points, q_1 and q_2, above *top* as witnesses that these lines see each other with respect to H_{non-v} (see Figure 3). Thus the flats f_1 and f_2 see each other with respect to H. We can determine this height *top* in linear time. On the other hand, if the flats f'_1 and f'_2 cannot see each other, then f_1 and f_2 cannot see each other with respect to $H_v \subseteq H$. □

We present three methods for solving the visibility problem. These methods should be used after reducing the dimension with the above lemma, if possible. We first show how to use linear programming to determine visibility in $O(n^{k_1+1})$ time using $O(n^{k_1})$ space. When $k_1 = k_2$ our point location structure gives a better solution with $O(n^{k_1+\epsilon})$ space and time. Finally, when $k_1 = k_2 - 1$ we obtain a solution — by one more use of sampling — that is somewhat more efficient.

First the linear programming solution. In d-space, a point $p = (x_1, x_2, \ldots, x_d)$ is contained in a hyperplane h if $a_1x_1 + \cdots + a_dx_d = b_h$, where $(a_1, \ldots, a_d)^T$ is the normal vector of h. By definition, a given k-flat is the intersection of $d - k$ hyperplanes with linearly independent normal vectors. Thus, a point p in a k-flat satisfies the matrix equation $M \cdot p = b$, where the rows of M are the normal vectors of the hyperplanes defining the k-flat, and b is the column vector with the corresponding b_h's. With such a matrix representation of a k-flat, we can prove:

Theorem 2 *Let H be a set of n hyperplanes in d-space, let f_1 be a k_1-flat and let f_2 be a k_2-flat. One can decide in $O(n^{k_1+1})$ time and $O(n^{k_1})$ space whether f_1 and f_2 see each other with respect to H.*

Proof: The hyperplanes H partition the k_1-flat f_1 into $O(n^{k_1})$ cells. We can construct them in $O(n^{k_1})$ time and space by building the arrangement of $\overline{H} = \{\overline{h} \mid \overline{h} = h \cap f_1 \text{ where } h \in H\}$ in the flat f_1.

For a given cell, take a candidate witness point q_1 from its interior. For each hyperplane $h \in H$, choose the sign of the hyperplane equation so that the halfspace $\{p \mid (a_1, \ldots, a_d) \cdot p \geq b_h\}$ contains q_1; expressed as an $n \times d$ matrix inequality, $M_H \cdot q_1 \geq b_H$.

We now wish to find a witness $q_2 \in f_2$ that lies in the intersection of these halfspaces. Thus, we wish to find a feasible solution to $M_H \cdot p \geq b_H$ and the matrix equation defining f_2, which is $M_{f_2} \cdot p = b_{f_2}$. This is a linear program with $n + d - k_2$ equations in d variables; it can be solved by techniques of Meggido [11, 12] in $O(n)$ time (see also [14]).

Repeating this for each cell determines if f_1 and f_2 can see each other in $O(n^{k_1+1})$ time and $O(n^{k_1})$ space. □

When both flats have the same dimension, our query structure gives a better solution than the linear programming approach.

Theorem 3 *Let H be a set of n hyperplanes, and f_1 and f_2 two k-flats in d-space. For any $\epsilon > 0$, one can decide in $O(n^{k+\epsilon})$ time whether the two k-flats can see each other with respect to H.*

Proof: By Lemma 4, we can assume that $d \leq 2k + 1$. Let f_1 and f_2 be the two k-flats. Preprocess f_1 and H for point location queries in the zone of f_1 in $O(n^{\lfloor d/2 \rfloor + \epsilon} + n^{k+\epsilon}) = O(n^{k+\epsilon})$ time and space. Then construct in f_2 the arrangement of $\overline{H} = \{\overline{h} \mid \overline{h} = h \cap f_2 \text{ where } h \in H\}$. Choose one point in the interior of each cell, and perform a point location query in the zone with it. Flats f_1 and f_2 can see each other if and only if at least one of the chosen points of f_2 lies in the zone of f_1. □

The approach of Theorem 3 gives an $O(n^{\lfloor d/2 \rfloor + \epsilon} + n^{k_2} \log^2 n)$ time method to test visibility among a k_1-flat f_1 and a k_2-flat f_2, with $k_2 > k_1$. We improve upon this with a sampling based algorithm. We choose a single sample of size $r = \Theta(n^{1/2})$.

Let $\overline{R}$ be a sample of size r of the set $\overline{H} = \{\overline{h} \mid \overline{h} = h \cap f_1 \text{ where } h \in H\}$. Triangulate $\mathcal{A}(\overline{R})$, and determine for each simplex s the subset H_s of H of hyperplanes that properly intersect s. Let $H'_s = H - H_s$, let c_s be the cell of $\mathcal{A}(H'_s)$ that contains s, let $\overline{\overline{c_s}} = c_s \cap f_2$, and let $\overline{\overline{H}} = \{\overline{\overline{h}} \mid \overline{\overline{h}} = h \cap f_2 \text{ where } h \in H\}$.

Lemma 6

(i) *If q_2 is a witness in f_2, then any point q_2' in the same cell of $\mathcal{A}(\overline{\overline{H}})$ is a witness in f_2.*

(ii) *For any simplex s in the triangulated arrangement $\mathcal{A}(\overline{R})$: in s lies a witness q_1 if and only if there is a point $q_2 \in \overline{\overline{c_s}} \subset f_2$ such that $q_2 \in zone(H_s, s)$.*

Proof: (i) is an easy consequence of the definition of visibility, and of the convexity of the cells in $\mathcal{A}(H)$.
(ii) $\Rightarrow$: Assume q_1 is a witness in $s \subset f_1$. Then by definition there is a witness $q_2 \in f_2$, and q_1 and q_2 lie in the same cell of $\mathcal{A}(H)$. In particular, they lie in the same cells in $\mathcal{A}(H_s')$ and in $\mathcal{A}(H_s)$. Hence, q_2 lies in $zone(H_s, s)$.
(ii) $\Leftarrow$: Let s be a simplex in the triangulated arrangement $\mathcal{A}(\overline{R})$, and assume $q_2 \in \overline{\overline{c_s}}$ and $q_2 \in zone(H_s, s)$. The last assumption implies that there is a point $q_1 \in s$, such that the segment q_1q_2 does not intersect any hyperplane of H_s. Since $q_1 \in s \subset c_s$ and $q_2 \in c_s$, no hyperplane of H_s' intersects q_1q_2. □

For every simplex s of the triangulated arrangement $\mathcal{A}(\overline{R})$, we construct a structure for point location in $zone(H_s, s)$, as in the previous section. Furthermore, we compute a relevant set of points in f_2 with which to query. This set will contain all vertices of the arrangement $\mathcal{A}(\overline{\overline{H_s}})$ that lie in $\overline{\overline{c_s}}$, together with the vertices that are obtained by intersecting the boundary of $\overline{\overline{c_s}}$ with $\mathcal{A}(\overline{\overline{H_s}})$. If the simplex s contains a witness, then one of the vertices will lie in $zone(H_s, s)$.

We compute the cell $\overline{\overline{c_s}}$ and preprocess it in $O(n^{\lfloor k_2/2 \rfloor + \epsilon})$ preprocessing time and space for $O(\log n)$ point location [4, 5]). Then we compute all $O((n/r)^{k_2})$ vertices of the arrangement $\mathcal{A}(\overline{\overline{H_s}})$, and perform a point location query in $\overline{\overline{c_s}}$ to find the ones in this cell. To find the vertices on the boundary of $\overline{\overline{c_s}}$, we do the following. Take every possible subset of i $(0 \leq i \leq k_2 - 2)$ hyperplanes of $\overline{\overline{H_s}}$, and consider the $(k_2 - i)$-flat formed by them. Compute the vertices that lie in the intersection of $\overline{\overline{c_s}}$ with the $(k_2 - i)$-flat, using the dual of a $(k_2 - i)$-dimensional convex hull algorithm that takes logarithmic time per face [13]. It remains to compute the vertices that are the intersections of $k_2 - 2$ hyperplanes of $\overline{\overline{H_s}}$ with $\overline{\overline{c_s}}$, which are 2-dimensional convex polygons. The remaining vertices that we have to compute are intersections of these polygons with one more hyperplane. Hence, we preprocess these polygons in linear time for $O(\log n)$ line intersection queries [6]. We query each polygon with the lines that are the intersections of all the other hyperplanes of $\overline{\overline{H_s}}$ with the 2-flat containing the polygon. This gives us the vertices that lie in one hyperplane of $\overline{\overline{H_s'}}$ and $k_2 - 1$ hyperplanes of $\overline{\overline{H_s}}$. (We have to take this extra effort, because in 1-space, the computation of the convex hull is not roughly proportional to its size, as is the case in all other dimensions.) To summarize, we find a set of

$$O((n/r)^{k_2} + \sum_{i=0}^{k_2-1} \binom{n/r}{i} n^{\lfloor (k_2-i)/2 \rfloor}) = O((n/r)^{k_2} + n^{\lfloor k_2/2 \rfloor} + n^{\lfloor (k_2+1)/2 \rfloor}/r)$$

points, and it takes logarithmic time to obtain each point. We query with every point of the set in the point location structure for $zone(H_s, s)$. If any query is successful, then f_1 and f_2 are visible with respect to H. When all simplices s have been processed, and no query was successful, then f_1 does not contain a witness, thus f_1 and f_2 are not visible with respect to H.

Theorem 4 *Let H be a set of n hyperplanes, f_1 a k_1-flat and f_2 a k_2-flat in d-space, where $1 \leq k_1 \leq k_2$. For any $\epsilon > 0$, one can decide whether f_1 and f_2 can see each other with respect to H in time $O(n^{k_1/2+k_2/2+\epsilon})$.*

Proof: By Lemma 4, we can assume that $d \leq k_1 + k_2 + 1$. For a sample $\overline{R}$ of r hyperplanes, the triangulated arrangement $\mathcal{A}(\overline{R})$ has $O(r^{k_1})$ simplices. A sample $\overline{R}$ can be constructed in time $O(nr^{k_1-1})$ such that any simplex s is intersected by $O(n/r)$ hyperplanes of H [10]. Thus, the size of H_s is $O(n/r)$. For every simplex s, the point location structure for the zone is constructed in $O((n/r)^{\lfloor d/2 \rfloor+\epsilon} + (n/r)^{k_1+\epsilon})$ time, by Theorem 1. We can preprocess the cell $\overline{\overline{c_s}}$ for point location in time $O(n^{\lfloor k_2/2 \rfloor+\epsilon})$ [5]. To find all the vertices and to query with them in the point location structure for $zone(H_s, s)$ takes time $O((n/r)^{k_2}+n^{\lfloor k_2/2 \rfloor}+n^{\lfloor (k_2+1)/2 \rfloor}/r)\cdot O(\log^2 n)$. Adding it all up, the time $T(n)$ taken by the algorithm is

$$T(n) = O(nr^{k_1-1}) + O(r^{k_1}) \cdot [O((n/r)^{\lfloor d/2 \rfloor+\epsilon}+$$

$$(n/r)^{k_1+\epsilon} + n^{\lfloor k_2/2 \rfloor+\epsilon} + (n/r)^{k_2}\log^2 n + n^{\lfloor k_2/2 \rfloor}\log^2 n + n^{\lfloor (k_2+1)/2 \rfloor}\log^2 n/r].$$

The first term is dominated by the others. Furthermore, $k_2 \geq k_1$ and $k_2 \geq \lfloor d/2 \rfloor$. If we take $r = c \cdot n^{1/2}$ for an appropriate constant c, we get $T(n) = O(n^{k_1/2+k_2/2+\epsilon})$. □

The sampling method is asymptotically faster than the linear programming solution when $k_2 - 1 \leq k_1 \leq k_2$. All time and space bounds still hold when we consider polytopes in flats rather than whole flats. To summarize the results of this section, we state:

Corollary 1 *For any fixed $\epsilon > 0$, one can check if a k_1-flat and a k_2-flat can see each other with respect to a set of n hyperplanes in d dimensions in time*

$O(n^{k_1+\epsilon})$	*if* $k_1 = k_2$
$O(n^{k_1+1/2+\epsilon})$	*if* $k_1 = k_2 - 1$
$O(n^{k_1+1})$	*always.*

5 Conclusions and open problems

In this paper we introduced the notion of a zone of a k-flat in an arrangement of hyperplanes in d-space, and studied two algorithmic aspects. Firstly, we presented a structure for $O(\log^2 n)$ time point location in the zone, which in many cases uses less space than the zone itself. Secondly, an efficient algorithm was given to determine whether two flats are visible for each other with respect to a set of hyperplanes.

Some open algorithmic problems that remain are the optimal construction of the zone itself, and the improvement of our algorithms. One can also think of a generalization to arrangements of hyperspheres or curves.

The main open combinatorial problem is to find sharp upper and lower bounds on the complexity of the generalized notion of the zone.

Acknowledgements

The authors thank Otfried Schwarzkopf for helpful discussions.

References

[1] Aranov, B., and M. Sharir, On the Zone of an Algebraic Surface in a Hyperplane Arrangement, *manuscript*, 1991.

[2] Chazelle, B., H. Edelsbrunner, L. J. Guibas, M. Sharir, and J. Stolfi, *Lines in Space: Combinatorics and Algorithms*, Techn. Rep. CS-TR-294-90, Dept. of Comp. Science, Princeton University, 1990.

[3] Chazelle, B., M. Sharir, and E. Welzl, Quasi-Optimal Upper Bounds for Simplex Range Searching and New Zone Theorems, *Proc. 6th ACM Symp. on Comp. Geom.* (1990), pp. 23-33.

[4] Clarkson, K.L., New Applications of Random Sampling in Computational Geometry, *Discr. & Comp. Geom.* **2** (1987), pp. 195-222.

[5] Clarkson, K.L., and P.W. Shor, Applications of Random Sampling in Computational Geometry II, *Discr. & Comp. Geom.* **4** (1989), pp. 387-422.

[6] Dobkin, D.P., and D.G. Kirkpatrick, Fast Detection of Polyhedral Intersection, *Theor. Comp. Science* **27** (1983), pp. 241-253.

[7] Edelsbrunner, H., *Algorithms in Combinatorial Geometry*, Springer-Verlag, 1987.

[8] Edelsbrunner, H., J. O'Rourke, and R. Seidel, Constructing Arrangements of Lines and Hyperplanes with Applications, *SIAM J. Comput.* **15** (1986), pp. 341-363.

[9] Haussler, D., and E. Welzl, ϵ-Nets and Simplex Range Queries, *Discr. & Comp. Geom.* **2** (1987), pp. 127-151.

[10] Matoušek, J., Approximations and Optimal Geometric Divide-and-Conquer, *manuscript*, 1990.

[11] Meggido, N., Linear-Time Algorithms for Linear Programming in $\mathbf{R}^3$ and Related Problems, *SIAM J. Comput.* **12** (1983), pp. 759-776.

[12] Meggido, N., Linear Programming in Linear Time when the Dimension is Fixed, *J. ACM* **31** (1984), pp. 114-127.

[13] Seidel, R., Constructing Higher-Dimensional Convex Hulls at Logarithmic Cost per Face, *Proc. 18th STOC* (1986), pp. 404-413.

[14] Seidel, R., Linear Programming and Convex Hulls Made Easy, *Proc. 6th Symp. on Comp. Geom.* (1990), pp. 211-215.