NBSIR 87-3679

Expected O(N) and O($N^{4/3}$) Algorithms for Constructing Voronoi Diagrams in Two and Three Dimensions

Javier Bernal

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Center for Applied Mathematics
Mathematical Analysis Division
Gaithersburg, MD 20899

November 1987

U.S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

NBSIR 87-3679

EXPECTED O(N) AND O($N^{4/3}$) ALGORITHMS FOR CONSTRUCTING VORONOI DIAGRAMS IN TWO AND THREE DIMENSIONS

Javier Bernal

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Center for Applied Mathematics
Mathematical Analysis Division
Gaithersburg, MD 20899

November 1987

U.S. DEPARTMENT OF COMMERCE, C. William Verity, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

Expected $O(N)$ and $O(N^{4/3})$ algorithms for constructing Voronoi diagrams in two and three dimensions

Javier Bernal

National Bureau of Standards, Gaithersburg, MD 20899.

Abstract. Bentley, Weide and Yao have proposed an expected $O(N)$ cell technique for computing Voronoi diagrams in two dimensions that does not generalize readily to three. In this paper their work is further developed and generalized to produce expected $O(N)$ and $O(N^{4/3})$ algorithms for constructing Voronoi diagrams in two and three dimensions, respectively. Computational experience is presented for the algorithm in two dimensions.

Key words. algorithm, computational geometry, computational complexity, Voronoi diagram, expected time analysis

AMS(MOS) subject classifications. 68U05

1. Introduction

Consider a finite set S of points in Euclidean space, and let P be any point in S. The *Voronoi polyhedron* $V(P, S)$ *of* P *relative to* S is the set of all points in the space such that P is as close to any point in this set as is any other point in S. $V(P, S)$ is the result of a *bisection process for* P *with respect to* S; i. e., $V(P, S)$ is the intersection of the closed half-spaces that contain P and that are determined by the perpendicular bisectors of the line segments connecting P with the other points in S. It follows that

$V(P, S)$ is a closed convex polyhedron, possibly unbounded, which contains P in its interior. The Voronoi polyhedra of the points in S (relative to S if not stated otherwise) fill the space without common interior points, and the union of their boundaries forms a diagram, the *Voronoi diagram for S*, which partitions the space into the Voronoi polyhedra. A point in the space that is a vertex of some Voronoi polyhedron is said to be a *vertex* of the Voronoi diagram for S. In k-dimensional Euclidean space a vertex of the Voronoi diagram for a set S is always the common vertex of at least $k+1$ Voronoi polyhedra of points in S. It is called a *degenerate vertex* whenever it is a vertex of more than $k+1$ Voronoi polyhedra.

Bentley, Weide and Yao [1] have shown how the Voronoi diagram for a set S of N points, chosen independently from a uniform distribution on a square in the plane, can be constructed in linear expected time. With M defined as the largest integer less than or equal to $N^{1/2}$, i. e. the *floor* of $N^{1/2}$, Bentley, *et al.* divide the square into M^2 equal-sized square *cells*. With $L(N)$ defined as the floor of $\log N$ (here log denotes the natural logarithm), cells not contained in the outermost $L(N)$ layers of cells of the square are called 'inner cells', the rest 'outer cells'. Essentially, the algorithm takes three steps. In the first step, each point in S is assigned (in constant time) to a cell in which it is contained. In the second step, the Voronoi polygons of points assigned to inner cells are constructed. Given a point P assigned to an inner cell, a search is conducted through each of the layers of cells surrounding P for points assigned to cells in these layers. This search procedure, called a *spiral search around P*, starts with the cell assigned to P, and then proceeds in outward direction to each of the layers of cells surrounding this cell. $V(P, S)$ is progressively built through a bisection process for P with respect to the set of points in S found through the search. A geometric test is available for deciding whether $V(P, S)$ has been obtained. In most cases $V(P, S)$ is obtained after examining only a small number of cells and points so that the expected time for the second step is $O(N)$. Finally, in the third step, the Voronoi polygons of those points assigned to outer cells are constructed by applying to them an $O(K \log K)$ worst case algorithm, e. g. Shamos' [3]. The expected time for this last step is also $O(N)$.

In this paper, we present a version of this algorithm in which the third step is replaced by steps that also require expected time $O(N)$ but that have the advantage of generalizing to three dimensions. Again, with M defined as the floor of $N^{1/2}$, the square is divided into M^2 equal-sized cells. We define inner cells and outer cells differently from [1]. Namely, cells contained in the outermost two layers of cells of the square are now called *outer cells*, the rest *inner cells*. As in [1], each point in S is assigned to a cell in which it is contained. Voronoi polygons of points assigned to inner cells are constructed by applying the Bentley, *et al.* cell-based method together with a generalization of its geometrical test. Finally, a modified version of Bowyer's insertion process [2] is used to handle points assigned to outer cells.

An *insertion process* is a method for updating the Voronoi diagram for a set of $k-1$ points for an additional k^{th} point in order to obtain the Voronoi diagram for all k points. The first step in Bowyer's algorithm consists of identifying a vertex of the Voronoi diagram for the $k-1$ points that will not be a vertex of the Voronoi diagram for the k points. To do this, Bowyer uses a 'walk' that starts near the centroid of the $k-1$ points and that ends at a point in whose Voronoi polygon the k^{th} point lies. The point thus found is clearly a point among the $k-1$ points that is closest to the k^{th} point. We have modified this step by taking advantage of the cell structure. In place of a walk we use a spiral search around the k^{th} point and, as described in [1], find a point among the $k-1$ points that is closest to it. The remainder of Bowyer's algorithm, in which vertices of the Voronoi diagram for the $k-1$ points which lie in the Voronoi polygon of the k^{th} point are deleted, and the vertices of the Voronoi polygon of the k^{th} point are added, is left essentially unaltered.

In Section 2, the two-dimensional algorithm is outlined and justified. In Section 3, we prove the algorithm is of expected $O(N)$ complexity. In Section 4, the algorithm is generalized to three dimensions and shown to be of expected $O(N^{4/3})$ complexity. Finally, in Section 5, computational experience with the implementation of the two-dimensional algorithm is presented.

2. The two-dimensional algorithm

In the following, cells are divided into four classes of cells (Figure 1). Inner cells not contained in any of the outermost $L(N)$ layers of cells of the square are called *class* 1 *cells*. Inner cells within $L(N)$ layers of cells from exactly one side of the square are called *class* 2 *cells*. Inner cells within $L(N)$ layers

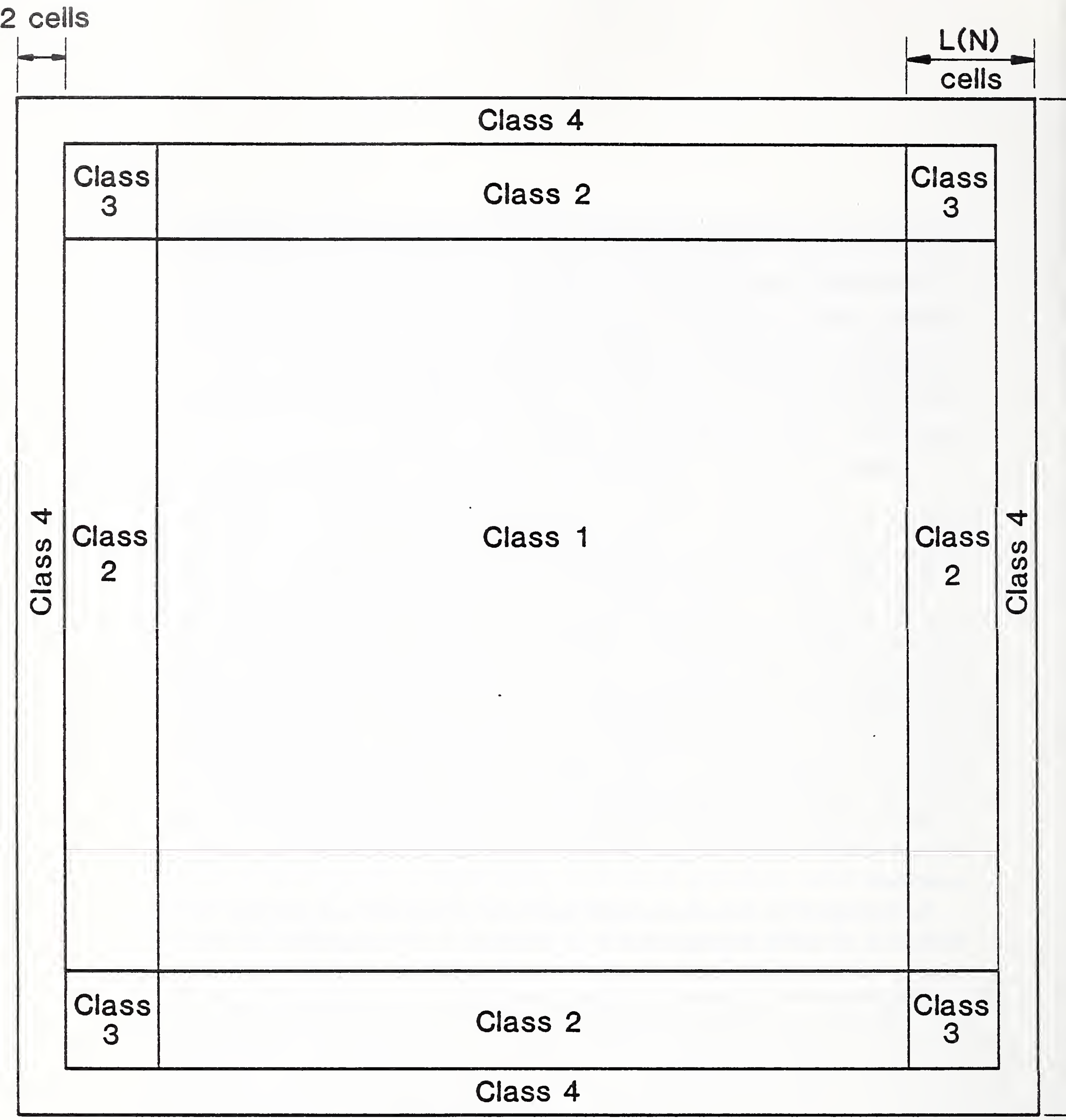

Figure 1: Regions of square that contain the four classes of cells. Class 1, 2, and 3 cells are inner cells, and class 4 cells are outer cells.

of cells from two sides of the square are called *class* 3 *cells*. Finally, all outer cells are called *class* 4 *cells*. We assume that N is large enough so that none of the four classes is empty. We define $S_1 \subseteq S$ as the set of points assigned to class 1 cells. S_2, S_3, S_4 are analogously defined with respect to class 2, class 3, class 4 cells, respectively.

Throughout the outline of the algorithm that follows, we say a point in S has been *processed* if its Voronoi polygon has been constructed. We say a cell is *empty* if no points of S are assigned to it. We say an inner cell has been *activated* if it has been found to be empty or if each point of S assigned to the cell has been processed. We select a *centermost* class 1 cell. The order in which inner cells are activated by our algorithm is determined by proceeding through the layers of cells in the square in a spiral-like fashion around the centermost cell. We say an inner cell is *the currently active cell* if the points assigned to it are currently being processed. We say a point in the currently active cell is *the currently active point* if the point is currently being processed. We say a *cell has been searched* during a spiral search around a point in S if all points assigned to the cell have already been found through the search.

We say that points P and P' in S are *Voronoi neighbors in S* if $V(P,S)$ and $V(P',S)$ are contiguous in the Voronoi diagram for S. For each P in S we let $\overline{V(P,S)}$ represent the set of Voronoi neighbors of P in S, and we assume $\overline{V(P,S)}$ can be readily extracted from $V(P,S)$. During the execution of the algorithm, for each P in S that has not been processed we define $Z^v(P)$ as the set of points assigned to inner cells that are *known Voronoi neighbors of P in S*; i. e., the set of points in $S \setminus S_4$ that have been processed and found to be Voronoi neighbors of P in S. Every point assigned to an inner cell is processed by our algorithm with a spiral search around the point. Given P, one such point, and assuming it is the currently active point, we let d_t represent the radius of the largest circle centered at P whose interior only intersects cells that have been searched up to the current time. We also let Z^t be the set of points in S that have been found up to the current time during the search. Then, as in [1], we define the *tentative Voronoi polygon V^t of P* at the current time as the Voronoi polygon of P relative to $\{P\} \cup Z^t \cup Z^v(P)$ (the plane if Z^t and $Z^v(P)$ are empty). Accordingly, when the search starts and Z^t is still empty, we define the *initial tentative Voronoi polygon V^i of P* as the tentative Voronoi polygon of P at that time. Also as in [1], each V^t is constructed by *updating* a previous one in such a way that each V^t is exactly the result of a bisection process for P with respect to $Z^t \cup Z^v(P)$.

Given points Q and Q' in the plane, we let $\overline{QQ'}$ and $dist(Q,Q')$ represent, respectively, the closed linear segment that connects Q and Q', and the distance from Q to Q'. In the outline of the algorithm, given Q, a point in the plane, and W, a finite or infinite subset of the plane, we let $dmax(Q,W)$ represent the maximum value of $dist(Q,Q')$ for Q' in W. Given rays $\vec{r}_1$ and $\vec{r}_2$ of common origin, we let $m(\vec{r}_1,\vec{r}_2)$ represent the size of the angle produced by a clockwise rotation from $\vec{r}_1$ to $\vec{r}_2$. Given points Q and Q' in the plane, we let $Q\vec{Q}'$ represent the ray through Q' of origin Q. Given a point P in S_2, we let both $\vec{l}_1(P)$ and $\vec{l}_2(P)$ represent the ray of origin P that is perpendicular to the side of the square closest to P. Given a point P in S_3, we let $\vec{l}_1(P)$ and $\vec{l}_2(P)$ represent the two rays of origin P that are perpendicular to the two sides of the square closest to P with $m(\vec{l}_1(P),\vec{l}_2(P))$ equal to 90^0. Given P, a point in S_2 or S_3, and assuming it is the currently active point, we let m_{t1} and m_{t2} represent at the current time the smallest positive values of $m(P\vec{P}',\vec{l}_1(P))$ and $m(\vec{l}_2(P),P\vec{P}')$, respectively, for P' in $Z^t \cup Z^v(P)$. We say points P_1 and P_2 in $Z^t \cup Z^v(P)$ *determine* m_{t1} and m_{t2}, respectively, if m_{t1} equals $m(P\vec{P}_1,\vec{l}_1(P))$ and m_{t2} equals $m(\vec{l}_2(P),P\vec{P}_2)$. Assuming points P_1 and P_2 determine m_{t1} and m_{t2}, respectively, we let C^t represent the interior of that region of the plane obtained by a clockwise rotation from $P\vec{P}_1$ to $P\vec{P}_2$. We are now ready to formulate the algorithm.

Start of algorithm.

Step 1. *Assign points to cells and select first class 1 cell to be activated.*

Let M be the floor of $N^{1/2}$.
Partition the square into M^2 equal-sized square cells.
Determine inner and outer cells.
Assign each point in S to a cell.
For each cell, list the points assigned to it.
Determine the centermost cell.
If the centermost cell is empty then go to Step 2.
Else designate this cell as the currently active cell.
Go to Step 3.

Step 2. *Select next inner (class 1, 2, or 3) cell.*

If all inner cells have been activated then go to Step 8.
Else choose the next inner cell to be activated.
If this cell is empty then go to Step 2.
Else designate this cell as the currently active cell.
Determine class for currently active cell.
If class 1 then go to Step 3.
Else go to Step 4.

Step 3. *Construct Voronoi polygon of a point in S_1.*
Let P be a point assigned to the currently active cell that has not been processed.
Designate P as the currently active point.
Start spiral search around P and construct V^i.
Update V^i and each subsequent V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t)$ and d_t.
Terminate search when one of the following criteria is met.

1. $2\,D_t < d_t$.
2. All cells in the square have been searched.

Upon termination go to Step 7.

Step 4. *Begin construction of Voronoi polygon of a point in S_2 or S_3.*
Let P be a point assigned to the currently active cell that has not been processed.
Designate P as the currently active point.
Determine $\vec{l}_1(P)$ and $\vec{l}_2(P)$.
Start spiral search around P and construct V^i.
Update V^i and each subsequent V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t)$, d_t, m_{t1}, and m_{t2}, and determine C^t.
Terminate search when one of the following criteria is met.

1. $2\,D_t < d_t$.
2. All cells in the square have been searched.
3. All cells that intersect C^t have been searched.

Upon termination, if neither criterion 1 nor criterion 2 has been met then go to Step 5.
Else go to Step 7.

Step 5. *Continue construction of Voronoi polygon of a point in S_2 or S_3.*
Let $C = C^t$.
Let P_1 and P_2 be points in S that determine m_{t1} and m_{t2}, respectively.
Compute $d' = dmax(P, \{P_1, P_2\})$.
Resume spiral search around P.
Update each V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t)$ and d_t.
Terminate search when one of the following criteria is met.

1. $2\,D_t < d_t$.
2. All cells in the square have been searched.
3. $d' < d_t$

Upon termination, if neither criterion 1 nor criterion 2 has been met then go to Step 6.
Else go to Step 7.

Step 6. *Complete construction of Voronoi polygon of a point in S_2 or S_3.*
Resume spiral search around P.
Update each V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t \setminus C)$ and d_t.
Terminate search when one of the following criteria is met.

1. $2\,D_t < d_t$.
2. All cells in the square have been searched.

Upon termination go to Step 7.

Step 7. *Save Voronoi polygon of a point assigned to an inner cell.*
Identify $V(P, S)$ with V^t.
Mark P as processed and save $V(P, S)$.
For each P' in $\overline{V(P, S)}$ that has not been processed let
$Z^v(P') = Z^v(P') \cup \{P\}$.

Determine whether currently active cell has been activated.
If activated then go to Step 2.
Else if P is in S_1 then go to Step 3.
Else go to Step 4.

Step 8. *Construct and save Voronoi polygons of points in S_4.*
Determine S_4.
If S_4 is empty then stop.
Else perform insertion process on S_4.
Perform for each P in S_4 a bisection process with respect to $\overline{V(P, S_4)} \cup Z^v(P)$ and identify $V(P, S)$ with result of process.
For each P in S_4 mark P as processed and save $V(P, S)$.
Stop.

End of algorithm.

Justification of algorithm. As established in [1], the Voronoi polygons of points in S_1 can be constructed with Step 3 of the algorithm. Let P, V^t, D_t, d_t be as defined in Step 3. Let P' be a point in S with $2\,D_t < dist(P, P')$. It follows that P' can not affect V^t since the perpendicular bisector of $\overline{PP'}$ does not intersect V^t. Thus, during the spiral search around P, we may conclude that $V(P, S)$ is equal to V^t as soon as $2\,D_t < d_t$.

We justify, with the aid of Figure 2, that we can produce the Voronoi polygons of points in S_2 or S_3 with Steps 4, 5, 6 of the algorithm. Let P be as defined in Step 4, and let C, P_1, P_2, d' be as defined in Step 5. Let V be the portion of $V(P, \{P, P_1, P_2\})$ that is contained in C (shaded region in Figure 2). Let P' be a point in S that does not lie in C or in the interior of the circles with centers $(P + P_1)/2$, $(P + P_2)/2$, and diameters $dist(P, P_1)$, $dist(P, P_2)$, respectively, as shown in Figure 2. It follows that P' does not affect V as indicated in Figure 2 by the perpendicular bisector b of $\overline{PP'}$. We note that points in S contained in C are found through the spiral search around P with Step 4, and that points in S contained in the circle with center P and radius d' are found with Step 5 (this circle contains the two circles mentioned above and is easier to search). Thus, with Step 6, in which V is not considered during the geometrical test and through which only points that do not affect V are found, we can produce $V(P, S)$.

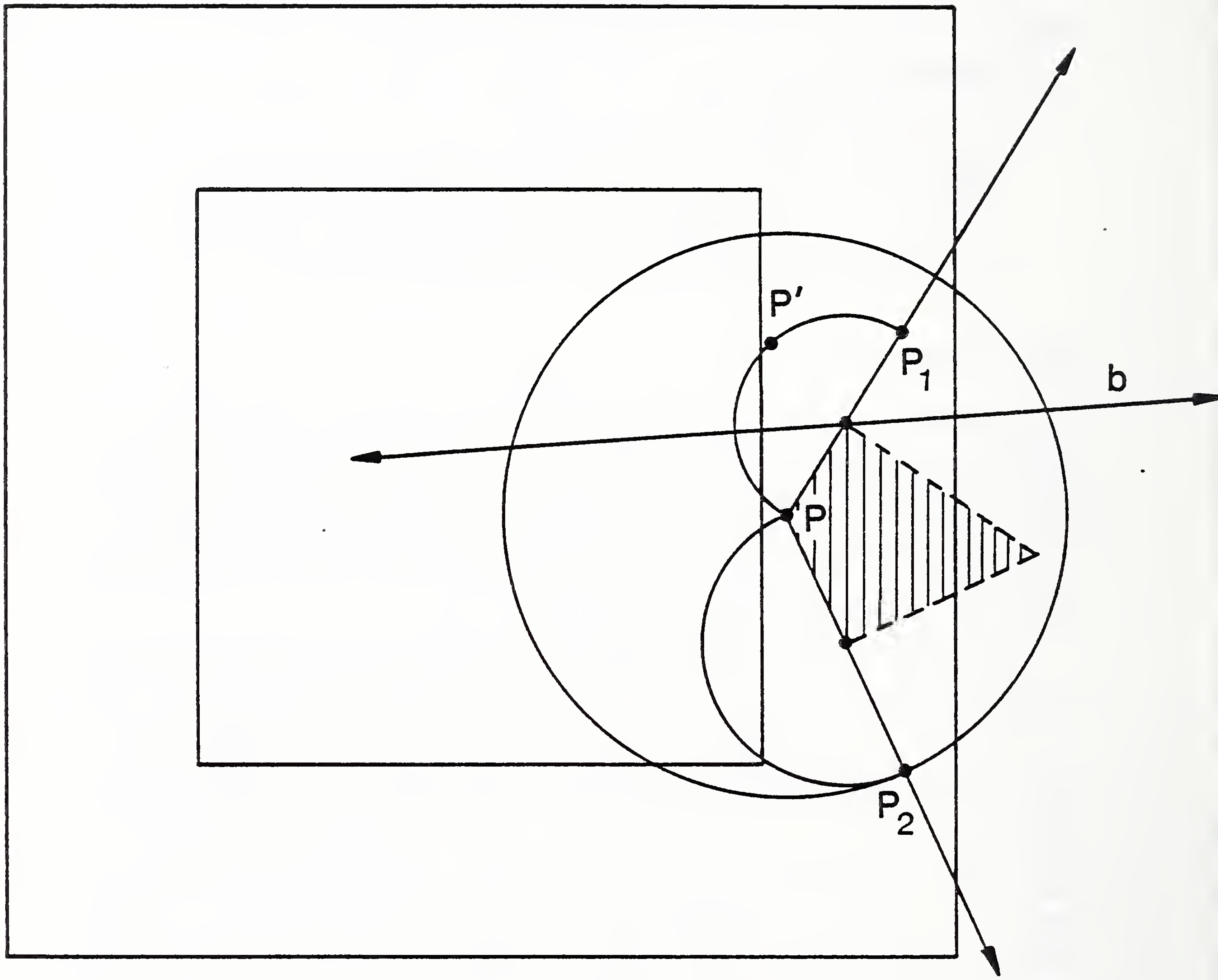

Figure 2: P', a point in S that does not affect V (shaded region).

Finally, we can construct the Voronoi polygons of points in S_4 with Step 8 of the algorithm since two points in S_4 that are Voronoi neighbors in S must be Voronoi neighbors in S_4.

3. Proof of complexity

In this section, we assume that the points in S have been chosen from a uniform distribution on the square, and prove that the algorithm presented in Section 2 has expected $O(N)$ execution time. First, we state the following theorem by Bentley, Weide and Yao [1]. Here, the time involved is defined as the number of cells plus the number of points examined.

Theorem *Let P be a point in the square, and let P' be a point in S closest to P. Then the expected time required to find P' through a spiral search around P is constant.*

By modifying the proof of the theorem, Bentley, *et al.* also establish the following two observations crucial to our proof of optimality.

1. Let P be a point in S such that at least one point in S is contained in each of the octants around P shown in Figure 3. Then the expected time required to find at least one point in each octant through a spiral search around P is constant.

2. Under similar assumptions, let P_i, $i = 1, \ldots, 8$, be the first eight points in S obtained through a spiral search around P such that they are contained in the octants around P, one per octant. Let d and D be the values of $dmax(P, \{P_1, \ldots, P_8\})$ and $dmax(P, V(P, \{P, P_1, \ldots, P_8\}))$, respectively. $V(P, S)$ can be constructed by searching only those cells intersecting the interior of the circle with center P and radius $2\,D$ (Figure 3). Since $2\,D \leq \sqrt{2}\,d$, it follows from Observation 1 that the expected time required to search all of these cells through a spiral search around P is constant.

Assigning the points in S to the appropriate cells can be accomplished in $O(N)$ time [1]. Thus, it will suffice to show that the expected time involved in constructing with the algorithm all Voronoi polygons of points in S_i, for each i, $i = 1, \ldots, 4$, is bounded above by $O(N)$. Let P be a point in $S \setminus S_4$ so that $V(P, S)$ is constructed with the algorithm through a spiral search around P. In what follows, we let w denote the time involved in constructing $V(P, S)$,

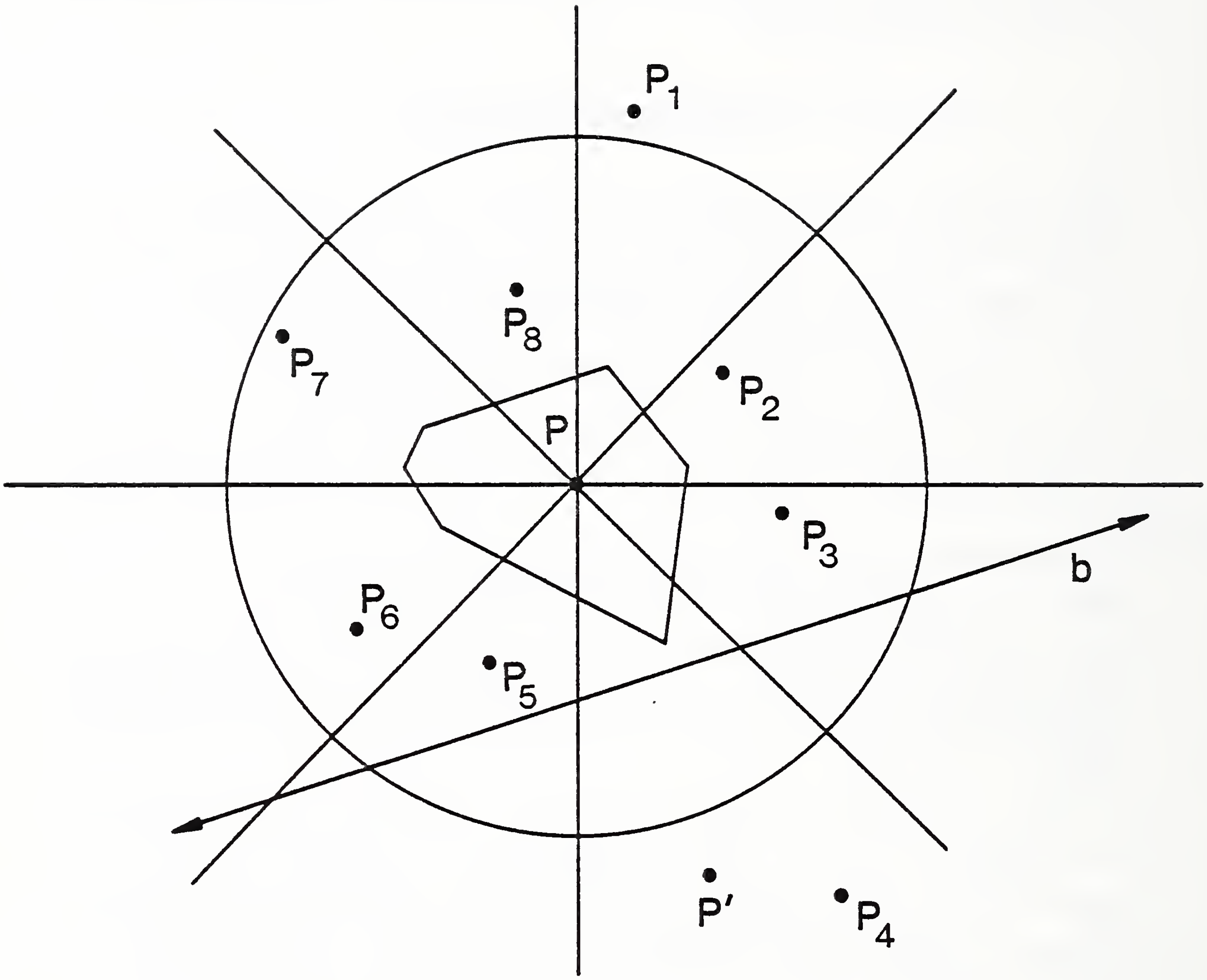

Figure 3: Points P_i, $i = 1, \ldots, 8$, contained in octants around P. P' is outside the circle of radius $2D$ so it does not affect $V(P, \{P, P_1, \ldots, P_8\})$ as indicated by the perpendicular bisector b of $\overline{PP'}$.

and use the fact that w is bounded above by

$$O(j + \sum_{i=1}^{k} v_i),$$

where j is the number of cells examined with the search, k is the number of points in those j cells, and v_i is the number of vertices in the tentative Voronoi polygon of P when the i^{th} point is found through the search. Finally, we let $E(w)$ denote the expected value of w, i. e. the expected time involved in constructing $V(P, S)$.

Proof for S_1. Let P be a point in S_1. As in [1], we say P is closed if at least one point in S is contained in each of the octants around P as shown in Figure 3.

Let p_1 be the probability that P is closed, and t_1 the expected number of points examined while constructing $V(P, S)$ with the algorithm when P is closed. p_2 and t_2 are similarly defined, respectively, for P not closed.

If t is the expected number of points examined while constructing $V(P, S)$ with the algorithm, then $t = p_1 \cdot t_1 + p_2 \cdot t_2$.

Of course $p_1 \leq 1$, and from Observation 2 above, $t_1 = O(1)$. Since at most all points are examined when P is not closed, it follows that $t_2 \leq O(N)$.

Next, we find an upper bound for p_2 using an argument of [1]. If no points are found in a given octant, then at least $O(L(N)^2)$ cells must be empty. The probability of the octant being empty is then bounded above by $e^{-O(L(N)^2)}$. It follows that $p_2 \leq 8\, e^{-O(L(N)^2)}$.

Therefore,

$$t = 1 \cdot O(1) + 8\, e^{-O(L(N)^2)} \cdot O(N) = O(1).$$

A similar argument can be used to show that the expected number of cells examined while constructing $V(P, S)$ with the algorithm is constant. Finally, since the number of vertices in any tentative Voronoi polygon of P is at most the number of points examined while constructing $V(P, S)$, the expected number of vertices in any tentative Voronoi polygon of P is also constant.

It follows, then, that

$$E(w) \leq O(1) + O(1) \cdot O(1) = O(1)$$

for each point in S_1.

Since at most N points are contained in S_1 then the expected time required for S_1 is

$$N \cdot O(1) = O(N).$$

Proof for S_2. Let P be a point in S_2 and, without any loss of generality, assume P is within $L(N)$ layers of cells from the right-hand side of the square. Let $\vec{l}$ represent the ray that both $\vec{l_1}(P)$ and $\vec{l_2}(P)$ represent. As shown in Figure 4, we say P is closed if within the first $L(N)$ layers of cells that surround P at least one point in S is contained in each of six octants around P, octants I through VI, and at least one point in S (which may be one of the points in octants I or VI) is found in each of the upper and lower portions of the outermost layer of the square.

If P is closed let P_1 and P_2 be points in S within the first $L(N)$ layers of cells that surround P in the upper and lower portions of the outermost layer of the square, respectively, with the smallest positive values of $m(\vec{PP_1}, \vec{l})$ and $m(\vec{l}, \vec{PP_2})$. Let C be the interior of the region of the plane obtained by a clockwise rotation from $\vec{PP_1}$ to $\vec{PP_2}$.

Clearly, since P is further than two cells from all sides of the square, $\vec{PP_1}$ and $\vec{PP_2}$ intersect the boundary of the square at points within the first $2\,L(N)$ layers of cells that surround P. Therefore, by examining the first $2\,L(N)$ layers of cells that surround P, all cells intersecting C are examined.

Let d' be the value of $dmax(P, \{P_1, P_2\})$. Then the circle of radius d' with center P is also contained in the first $2\,L(N)$ layers of cells that surround P. Finally, let U be the Voronoi polygon of P relative to those points within the first $L(N)$ layers of cells that surround P. Note that points outside C and the circle of radius d' with center P do not affect the part of U that is contained in C. Accordingly, let D' be the value of $dmax(P, U \setminus C)$. Then, since P is closed, the circle of radius $2\,D'$ with center P is also contained in the first $2\,L(N)$ layers of cells that surround P.

It follows from these observations that the Voronoi polygon of a closed point can be constructed with the algorithm by examining no more than the first $2\,L(N)$ layers of cells that surround the point.

Let p_1 be the probability that P is closed, and t_1 the expected number of points examined while constructing $V(P,S)$ with the algorithm when P is closed. p_2 and t_2 are similarly defined, respectively, for P not closed.

If t is the expected number of points examined while constructing $V(P,S)$ with the algorithm, then $t = p_1 \cdot t_1 + p_2 \cdot t_2$.

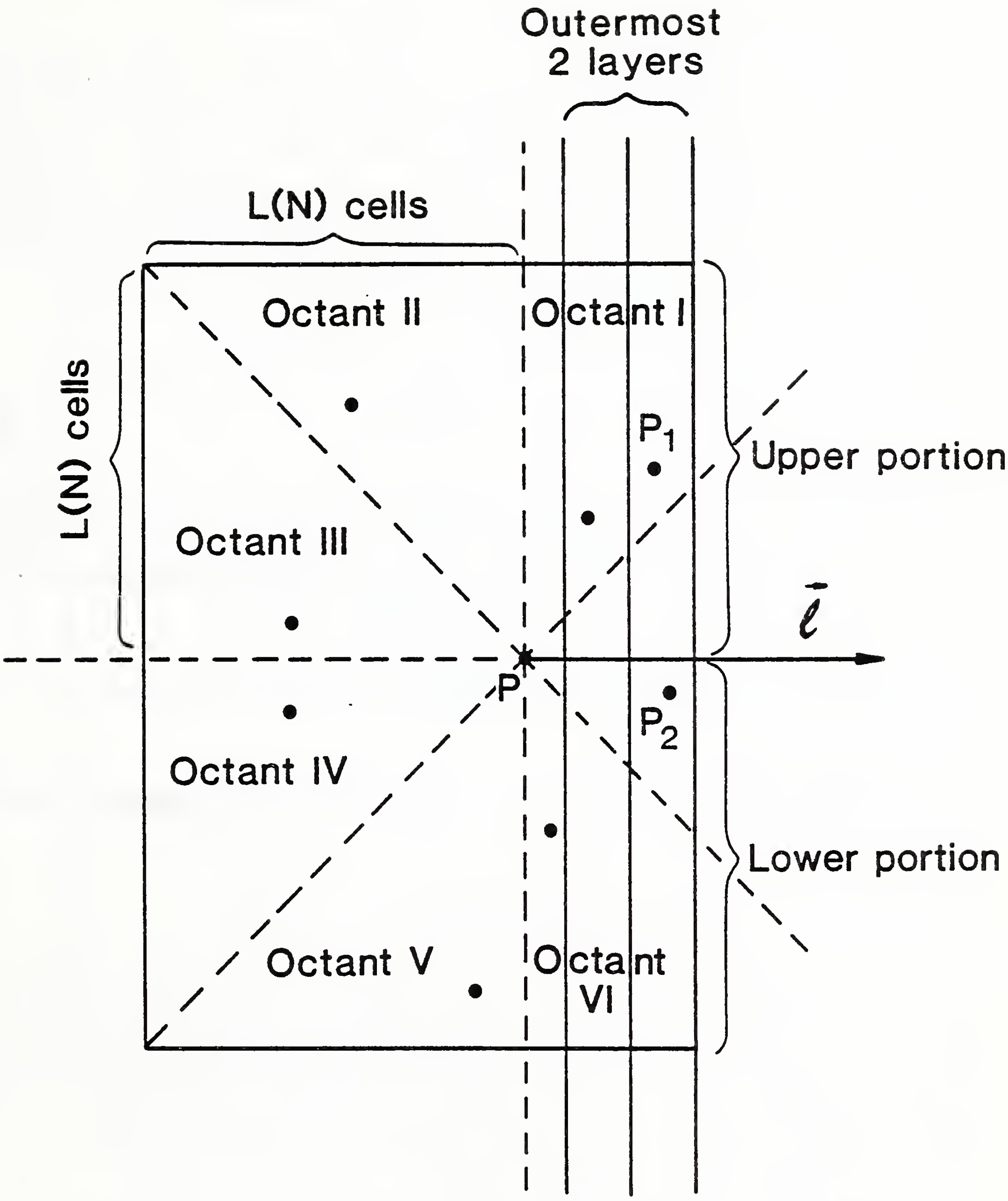

Figure 4: A closed point in S_2. One point is contained in each of the octants I through VI. P_1 and P_2 are contained in the upper and lower portions of the outermost layer of the square, respectively.

Of course $p_1 \leq 1$, $t_2 \leq O(N)$, and from the above discussion $O(L(N)^2)$ is an upper bound for t_1.

In order to find an upper bound for p_2, we argue as follows. If no points are found in one of the octants II through V, then at least $O(L(N)^2)$ cells must be empty. If no points are found in one of the octants I and VI, then at least $O(L(N))$ cells must be empty. Finally, if either the upper or the lower portion of the outermost layer of the square is empty, then $O(L(N))$ cells must be empty. It follows that $p_2 \leq 4\,e^{-O(L(N)^2)} + 4\,e^{-O(L(N))}$.

Thus,

$$\begin{aligned} t &\leq 1 \cdot O(L(N)^2) + (4\,e^{-O(L(N)^2)} + 4\,e^{-O(L(N))}) \cdot O(N) \\ &= O(L(N)^2) + O(1) = O(L(N)^2). \end{aligned}$$

An argument similar to that used for points in S_1 can be used to show that the expected number of cells examined while constructing $V(P, S)$ with the algorithm is $O(L(N)^2)$, and that the expected number of vertices in any tentative Voronoi polygon of P is also $O(L(N)^2)$. Thus,

$$E(w) \;\leq\; O(L(N)^2) + O(L(N)^2) \cdot O(L(N)^2) = O(L(N)^4)$$

for each point in S_2.

The expected number of points in S_2 is $O(N^{1/2}L(N))$. Hence, the expected time required for S_2 is

$$O(N^{1/2}L(N)) \cdot O(L(N)^4) = O(N).$$

Proof for S_3. Let P be a point in S_3 and, without any loss of generality, assume P is within $L(N)$ layers of cells from the right-hand and bottom sides of the square. Let $\vec{l_1}$ and $\vec{l_2}$ represent the rays that $\vec{l_1}(P)$ and $\vec{l_2}(P)$ represent, respectively. As shown in Figure 5, we say P is closed if within the first $L(N)$ layers of cells that surround P at least one point in S is contained in each of four octants around P, octants I through IV, and the right-hand and bottom portions of the outermost layer of the square.

That $O(N)$ is an upper bound for the expected time required for S_3 now follows by an argument similar to the one used for S_2.

Proof for S_4. The expected time required by the insertion process of Step 8 of the algorithm is at most proportional to the product of the expected number

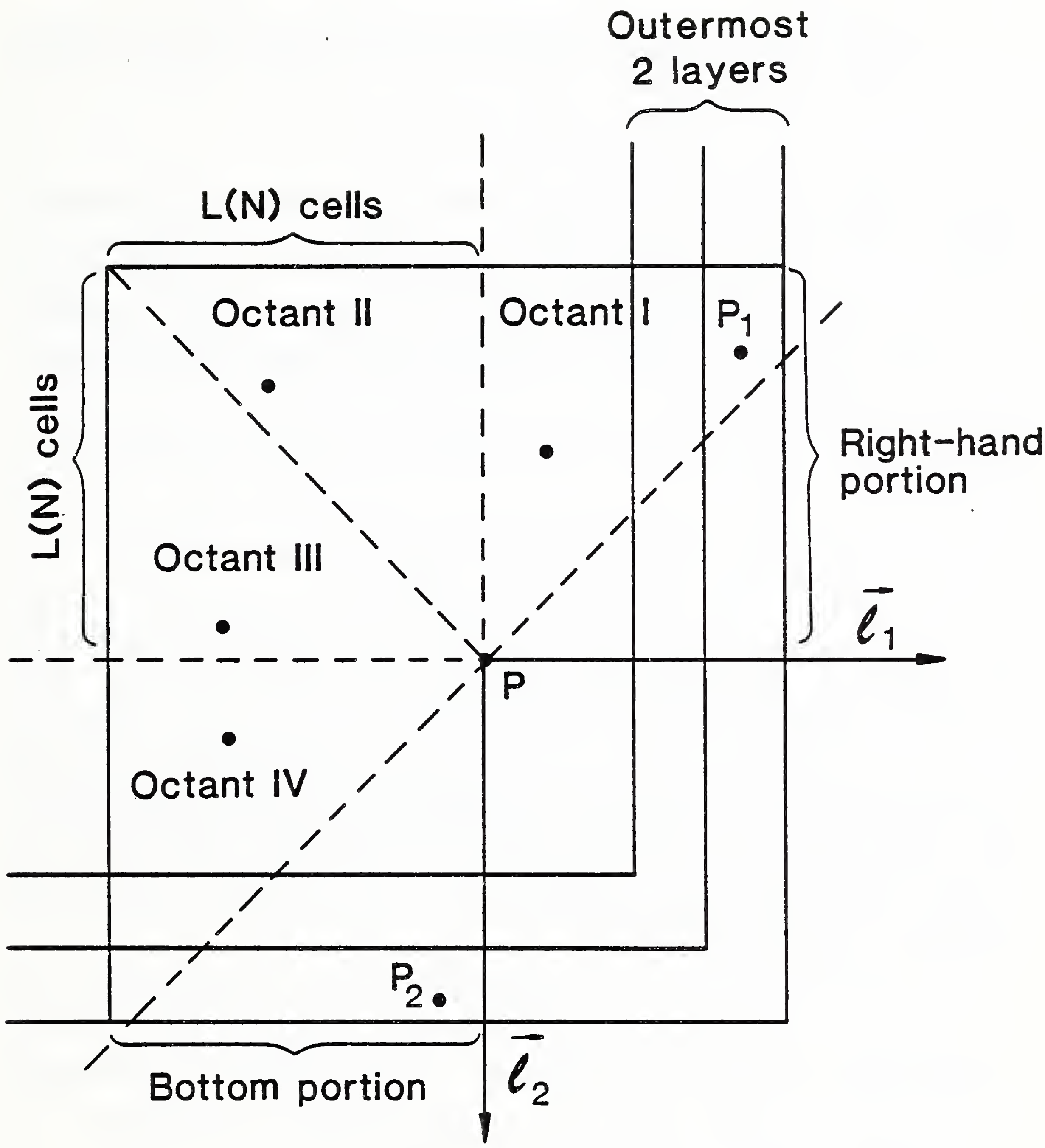

Figure 5: A closed point in S_3. One point is contained in each of the octants I through IV. P_1 and P_2 are contained in the right-hand and bottom portions of the outermost layer of the square, respectively.

of points in S_4 and the expected maximum number of vertices in the Voronoi diagram for any subset of S_4. Since the expected number of points in S_4 is $O(N^{1/2})$, it follows from the Euler-Poincaré formula that this time is at most

$$O(N^{1/2}) \cdot O(N^{1/2}) = O(N).$$

Finally, it suffices to show that the expected time required by the bisection process of Step 8 is also at most $O(N)$. Let r be the number of points in S_4. Let P_i, $i = 1, \ldots, r$, be the points in S_4. For each i, $i = 1, \ldots, r$, define w_i as the number of points in $\overline{V(P_i, S_4)}$, u_i as the final number of points in $Z^v(P_i)$, and v_i as the maximum number of vertices of any polygon obtained during the bisection process for P_i.

It follows that the time required by the bisection process is at most proportional to

$$\sum_{i=1}^{r}(w_i + u_i) \cdot v_i \leq \sum_{i=1}^{r}(w_i + u_i)^2 = \sum_{i=1}^{r}(w_i^2 + 2\, w_i u_i + u_i^2).$$

Again, r has expected value $O(N^{1/2})$, so that by the Euler-Poincaré formula, $\sum_{i=1}^{r} w_i$ has expected value $O(N^{1/2})$.

In order to calculate an upper bound for the expected value of each u_i, $i = 1, \ldots, r$, we proceed as follows. Given i, $1 \leq i \leq r$, let P' be a point in S not contained in S_4 such that P' is outside the first $2\, L(N)$ layers of cells that surround P_i. As previously proven, P' is a Voronoi neighbor of P_i in S with probability at most proportional to $e^{-O(L(N))}$, so that the expected value for u_i is bounded above by

$$O(L(N)^2) \cdot 1 + N \cdot e^{-O(L(N))} = O(L(N)^2) + O(1) = O(L(N)^2).$$

It follows now that the expected value of $\sum_{i=1}^{r}(w_i^2 + 2\, w_i u_i + u_i^2)$ is at most

$$O(N) + O(N^{1/2}) \cdot O(L(N)^2) + O(N^{1/2}) \cdot O(L(N)^4) = O(N);$$

i. e., the expected time required by the bisection process is at most $O(N)$.

4. The three-dimensional algorithm

We now present an algorithm for constructing the Voronoi diagram for a set S of N points contained in a cube in three-dimensional Euclidean space. First, with M defined as the floor of $N^{1/3}$ we divide the cube into M^3 equal-sized cubic cells. Cells contained in the outermost two layers of cells of the cube we call *outer cells*, the rest *inner cells*. Next, as in the two-dimensional case, each point in S is assigned to a cell in which it is contained. Finally, the Voronoi polyhedron of each point in S is constructed according to its cell assignment by generalizing the two-dimensional algorithm of Section 2.

In order to outline the algorithm, cells are further divided into five classes of cells. With $L(N)$ defined again as the floor of $\log N$, inner cells not contained in any of the outermost $L(N)$ layers of cells of the cube are called *class* 1 *cells.* Inner cells within $L(N)$ layers of cells from exactly one face of the cube are called *class* 2 *cells.* Inner cells within $L(N)$ layers of cells from exactly two faces of the cube ared called *class* 3 *cells.* Inner cells within $L(N)$ layers of cells from three faces of the cube are called *class* 4 *cells.* Finally, all outer cells are called *class* 5 *cells.* We assume N is large enough so that none of the five classes is empty. We define $S_1 \subseteq S$ as the set of points assigned to class 1 cells. S_2, S_3, S_4, S_5 are analogously defined with respect to class 2, class 3, class 4, class 5, respectively.

Throughout the following, definitions and meaning of terminology, such as Z^t, $Z^v(P)$, $V(P,S)$, are as in the two-dimensional case with the words polyhedron and space replacing the words polygon and plane, respectively, when necessary. However, points in S_2, S_3, S_4 require some additional definitions and terminology which we present separately for the purpose of clarity. Most importantly, we define symbols $C''(P)$ and d'_t for each point P in $S_2 \cup S_3 \cup S_4$, and describe what it means to say that P 'has been closed.'

Definitions and terminology for S_2. Let P be a point in S_2. Let F be the face of the cube closest to P. Let $\vec{l}$ be the ray with origin P that is perpendicular to F. Let P' be the point at which $\vec{l}$ intersects F. Let m be a line through P' that is perpendicular to an edge of the cube in F. Let H'' be the plane parallel to F that contains the point $(P + P')/2$. We define $C''(P)$ as that closed half-space determined by H'' that contains P.

Assume P is the currently active point, and P'_i, $i = 1, \ldots, 8$, are points in F contained in the octants around P' as shown in Figure 6. We say that at the current time P *has been closed* and that $\{P'_i, i = 1, \ldots, 8\}$ *closes* P if there exist points P_i, $i = 1, \ldots, 8$, in $Z^t \cup Z^v(P)$ such that the rays $\vec{PP_i}$, $i = 1, \ldots, 8$, intersect F at the points P'_i, $i = 1, \ldots, 8$, respectively. Assuming P has been closed we define d'_t at the current time as the smallest value of $dmax(P, \{P'_i, i = 1, \ldots, 8\})$ for $\{P'_i, i = 1, \ldots, 8\}$ in the family of sets that close P.

Definitions and terminology for S_3. Let P be a point in S_3. Let F_j, $j = 1, 2$, be the two faces of the cube closest to P. For each j, $j = 1, 2$, let $\vec{l_j}$ be the ray with origin P that is perpendicular to F_j. For each j, $j = 1, 2$, let P'_j be the point at which $\vec{l_j}$ intersects F_j. Let P''_{j0}, $j = 1, 2$, be the vertices of the cube common to F_1 and F_2 in the order shown in Figure 7. Let m be the line that contains the edge of the cube common to F_1 and F_2. For each j, $j = 1, 2$, let m_j be the line through P'_j perpendicular to m. Let m_0 represent the same line that m_2 represents. For each j, $j = 1, 2$, let $E_{j-1,1}$ and E_{j0} be the closed half-planes determined by m_{j-1} and m_j, respectively, that contain P''_{j0}. For each j, $j = 1, 2$, let H''_j be the plane parallel to F_j that contains $(P + P'_j)/2$. We define $C''(P)$ as the intersection of the closed half-spaces determined by H''_1 and H''_2 that contain P.

Assume P is the currently active point, and P'_{ji}, $j = 1, 2$, $i = 0, \ldots, 7$, are points such that with $P'_{07} = P'_{27}$, for each j, $j = 1, 2$, P'_{ji}, $i = 1, \ldots, 6$, are points in F_j contained in the six octants around P'_j as shown in Figure 7, and $P'_{j-1,7}$ and P'_{j0} are points in $E_{j-1,1}$ and E_{j0}, respectively. We say that at the current time P *has been closed* and that $\{P'_{ji}, j = 1, 2, i = 0, \ldots, 7\}$ *closes* P if there exist points P_{ji}, $j = 1, 2$, $i = 0, \ldots, 6$, in $Z^t \cup Z^v(P)$ such that for each j, $j = 1, 2$, the rays $\vec{PP_{ji}}$, $i = 1, \ldots, 6$, intersect F_j at the points P'_{ji}, $i = 1, \ldots, 6$, respectively, and the ray $\vec{PP_{j0}}$ intersects $E_{j-1,1}$ and E_{j0} at the points $P'_{j-1,7}$ and P'_{j0}, respectively. Assuming P has been closed we define d'_t at the current time as the smallest value of $dmax(P, \{P'_{ji},\ j = 1, 2,\ i = 0, \ldots, 7\})$ for $\{P'_{ji}, j = 1, 2, i = 0, \ldots, 7\}$ in the family of sets that close P.

Definitions and terminology for S_4. Let P be a point in S_4. Let F_j, $j = 1, 2, 3$, be the three faces of the cube closest to P in the order shown in Figure 8. Let F_0 represent the same face that F_3 represents. For each j, $j = 1, 2, 3$, let $\vec{l_j}$ be the ray with origin P perpendicular to F_j. For each j, $j = 1, 2, 3$, let P'_j be the point at which $\vec{l_j}$ intersects F_j. Let P'_0 represent the same point that P'_3 represents. Let P''_0 be the vertex of the cube common to F_1, F_2, and

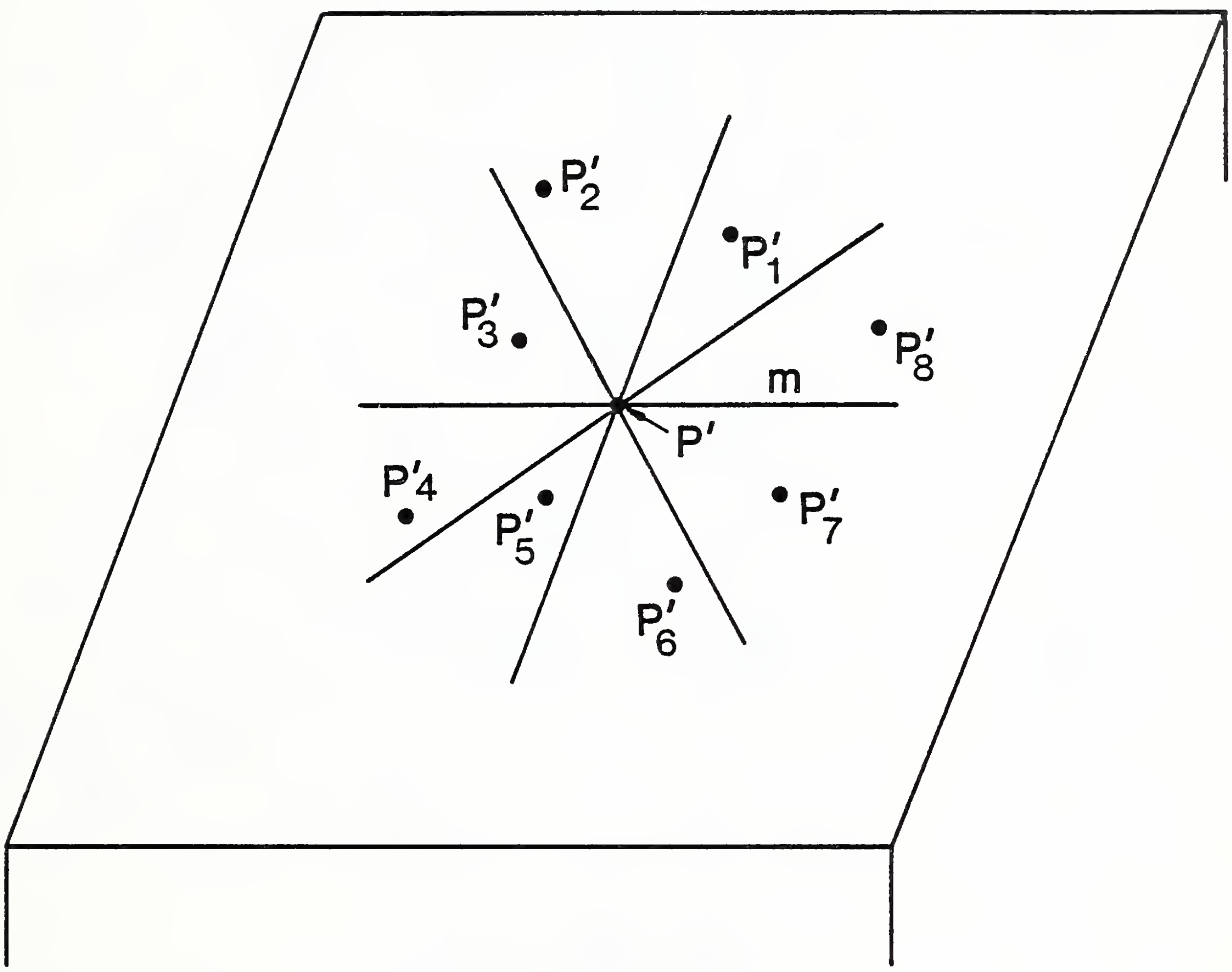

Figure 6: View of the face closest to a point in S_2 that has been closed.

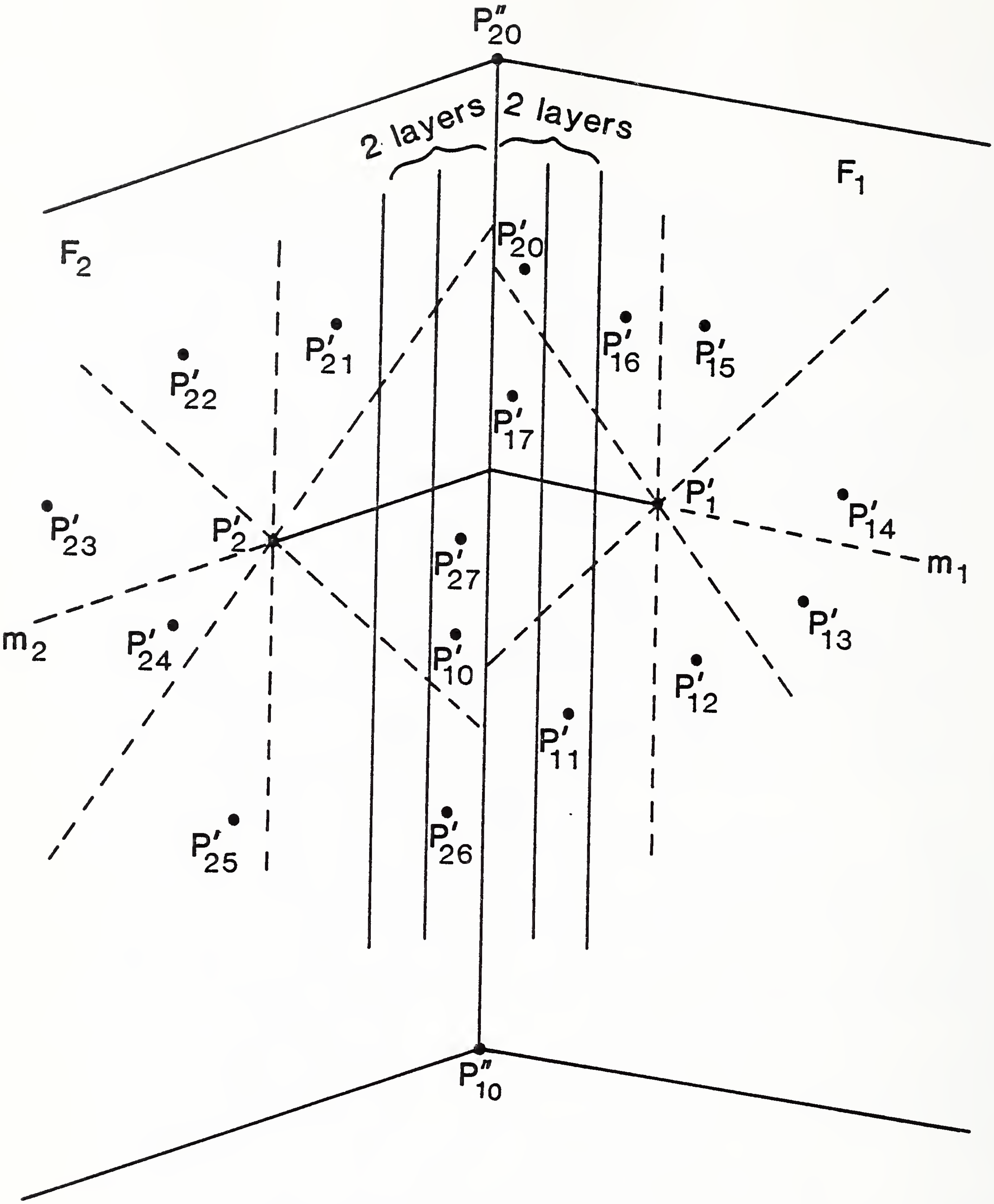

Figure 7: View of the two faces closest to a point in S_3 that has been closed.

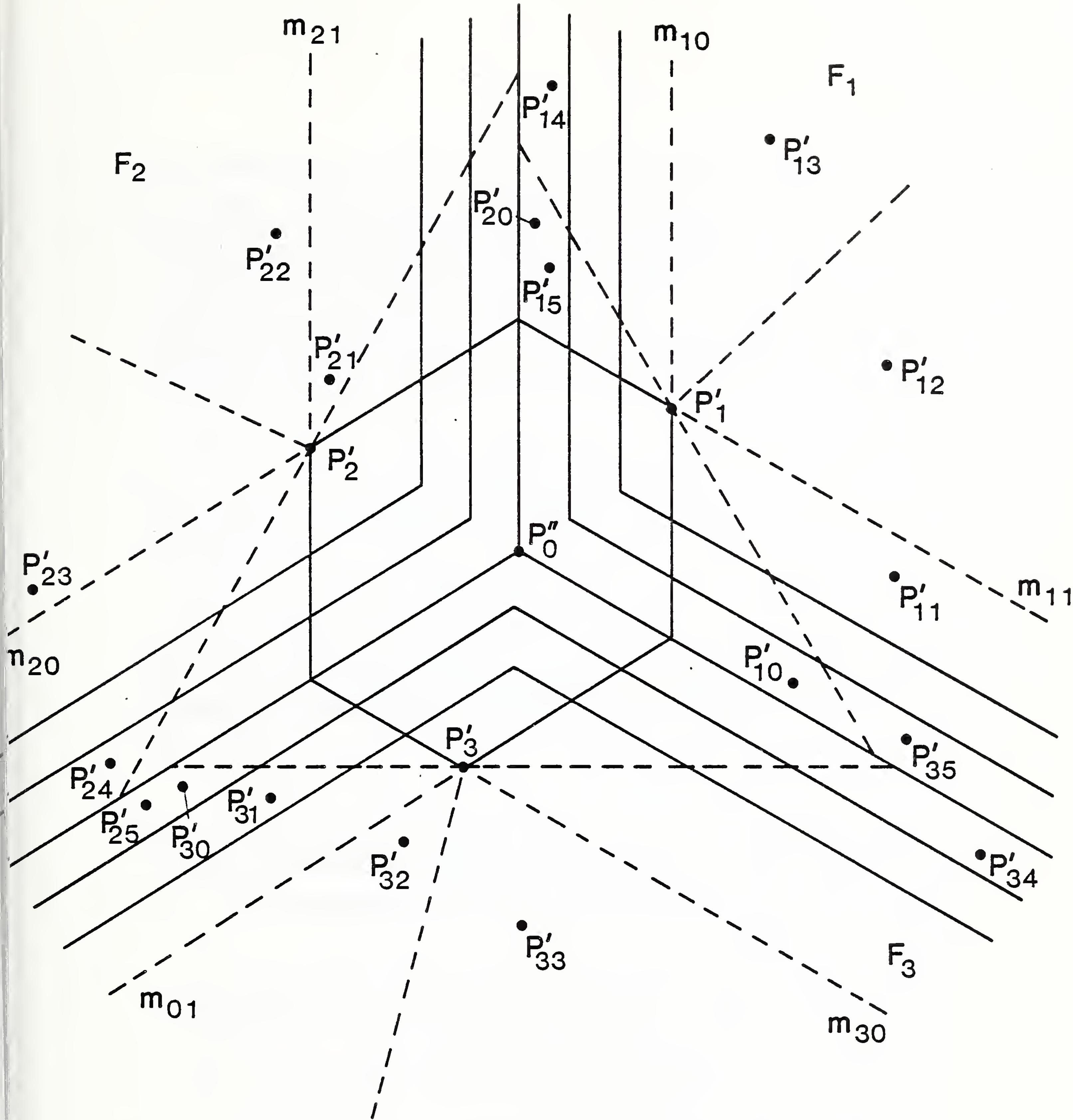

Figure 8: View of the three faces closest to a point in S_4 that has been closed.

F_3. For each j, $j = 1,2,3$, let m_j be the line that contains the edge of the cube common to F_{j-1} and F_j. For each j, $j = 1,2,3$, let $m_{j-1,1}$ and m_{j0} be the lines through P'_{j-1} and P'_j, respectively, perpendicular to m_j. For each j, $j = 1,2,3$, let $E_{j-1,1}$ and E_{j0} be the closed half-planes determined by $m_{j-1,1}$ and m_{j0}, respectively, that do not contain P''_0, and that are contained in the planes that contain F_{j-1} and F_j, respectively. For each j, $j = 1,2,3$, let H''_j be the plane parallel to F_j that contains $(P + P'_j)/2$. We define $C''(P)$ as the intersection of the closed half-spaces determined by H''_1, H''_2, and H''_3 that contain P.

Assume P is the currently active point, and P'_{ji}, $j = 1,2,3$, $i = 0,\ldots,5$, are points such that with $P'_{05} = P'_{35}$, for each j, $j = 1,2,3$, P'_{ji}, $i = 1,\ldots,4$, are points in F_j contained in the four octants around P'_j as shown in Figure 8, and $P'_{j-1,5}$ and P'_{j0} are points in $E_{j-1,1}$ and E_{j0}, respectively. We say that at the current time P *has been closed* and that $\{P'_{ji}, j = 1,2,3, i = 0,\ldots,5\}$ *closes* P if there exist points P_{ji}, $j = 1,2,3$, $i = 0,\ldots,4$, in $Z^t \cup Z^v(P)$ such that for each j, $j = 1,2,3$, the rays $\vec{PP_{ji}}$, $i = 1,\ldots,4$, intersect F_j at the points P'_{ji}, $i = 1,\ldots,4$, respectively, and the ray $\vec{PP_{j0}}$ intersects $E_{j-1,1}$ and E_{j0} at the points $P'_{j-1,5}$ and P'_{j0}, respectively. Assuming P has been closed we define d'_t at the current time as the smallest value of $dmax(P, \{P'_{ji}, j = 1,2,3, i = 0,\ldots,5\})$ for $\{P'_{ji}, j = 1,2,3, i = 0,\ldots,5\}$ in the family of sets that close P.

A modified version of Bowyer's three-dimensional insertion process [2] is used in what follows. It is the obvious generalization to three dimensions of the modified version of Bowyer's two-dimensional insertion process.

Start of algorithm.

Step 1. *Assign points to cells and select first class 1 cell to be activated.*

Let M be the floor of $N^{1/3}$.
Partition the cube into M^3 equal-sized cubic cells.
Determine inner and outer cells.
Assign each point in S to a cell.
For each cell, list the points assigned to it.
Determine the centermost cell.
If the centermost cell is empty then go to Step 2.
Else designate this cell as the currently active cell.

Go to Step 3.

Step 2. *Select next inner (class 1, 2, 3, or 4) cell.*
If all inner cells have been activated then go to Step 8.
Else choose the next inner cell to be activated.
If this cell is empty then go to Step 2.
Else designate this cell as the currently active cell.
Determine class for currently active cell.
If class 1 then go to Step 3.
Else go to Step 4.

Step 3. *Construct Voronoi polyhedron of a point in* S_1.
Let P be a point assigned to the currently active cell that has not been processed.
Designate P as the currently active point.
Start spiral search around P and construct V^i.
Update V^i and each subsequent V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t)$ and d_t.
Terminate search when one of the following criteria is met.

1. $2\,D_t < d_t$.
2. All cells in the cube have been searched.

Upon termination go to Step 7.

Step 4. *Begin construction of Voronoi polyhedron of a point in* S_2, S_3, *or* S_4.
Let P be a point assigned to the currently active cell that has not been processed.
Designate P as the currently active point.
Start spiral search around P and construct V^i.
Update V^i and each subsequent V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t)$ and d_t.
Terminate search when one of the following criteria is met.

1. $2\,D_t < d_t$.
2. All cells in the cube have been searched.

3. P has been closed.

Upon termination, if neither criterion 1 nor criterion 2 has been met then go to Step 5.
Else go to Step 7.

Step 5. *Continue construction of Voronoi polyhedron of a point in S_2, S_3, or S_4.*
Resume spiral search around P.
Update each V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t)$, d_t, and d'_t.
Terminate search when one of the following criteria is met.

1. $2\, D_t < d_t$.
2. All cells in the cube have been searched.
3. $\sqrt{2}\, d'_t < d_t$.

Upon termination, if neither criterion 1 nor criterion 2 has been met then go to Step 6.
Else go to Step 7.

Step 6. *Complete construction of Voronoi polyhedron of a point in S_2, S_3, or S_4.*
Determine $C''(P)$.
Resume spiral search around P.
Update each V^t as appropriate.
For each V^t compute $D_t = dmax(P, V^t \cap C''(P))$ and d_t.
Terminate search when one of the following criteria is met.

1. $2\, D_t < d_t$.
2. All cells in the cube have been searched.

Upon termination go to Step 7.

Step 7. *Save Voronoi polyhedron of a point assigned to an inner cell.*
Identify $V(P, S)$ with V^t.
Mark P as processed and save $V(P, S)$.
For each P' in $\overline{V(P, S)}$ that has not been processed let

$Z^v(P') = Z^v(P') \cup \{P\}$.
Determine whether currently active cell has been activated.
If activated then go to Step 2.
Else if P is in S_1 then go to Step 3.
Else go to Step 4.

Step 8. *Construct and save Voronoi polyhedra of points in* S_5.
Determine S_5.
If S_5 is empty then stop.
Else let $Z_5 = \cup_{P \epsilon S_5} Z^v(P)$.
Perform insertion process on $S_5 \cup Z_5$.
For each P in S_5 identify $V(P, S)$ with $V(P, S_5 \cup Z_5)$.
For each P in S_5 mark P as processed and save $V(P, S)$.
Stop.

End of algorithm.

Justification of algorithm. Because of similarities with the two dimensional method, we need not justify that the above algorithm constructs the Voronoi polyhedra of points in S_1 or S_5. In addition, because of similarities among S_2, S_3, and S_4, we only justify that it constructs the Voronoi polyhedra of points in S_2.
Let F, $\vec{l}$, P', m, H'', $C''(P)$ be as defined above for a point P in S_2. Assume that P is the currently active point and that it has been closed. In addition, assume that at the current time $\sqrt{2}\, d_t' < d_t$ and that a set $\{P_i', i = 1, \ldots, 8\}$ of points in F closes P with $d_t' = dmax(P, \{P_i', i = 1, \ldots, 8\})$. Let V be that part of V^t that is not contained in $C''(P)$. Assume $S \setminus Z^t \cup Z^v(P) \cup \{P\}$ is not empty and Q' is a point in this set. Let H' be the plane that perpendicularly bisects $\overline{Q'P}$, and let C' be the open half-space determined by H' that contains P. We show that C' contains V, so that Q' does not affect V, and thus the termination criteria may change from those in Step 5 to those in Step 6.
Let P_i, $i = 1, \ldots, 8$, be points in $Z^t \cup Z^v(P)$ such that the rays $\vec{PP_i}$, $i = 1, \ldots, 8$, intersect F at the points P_i', $i = 1, \ldots, 8$, respectively. We assume $P_i' = P_i$ for each i, $i = 1, \ldots, 8$, since, at worst, V is just a larger set. We assume $P_i' \neq P'$, $i = 1, \ldots, 8$, since, otherwise, V equals the empty set. Finally, we assume P is not a convex combination of P' and Q'.

We first argue that we may assume that Q' is contained in F. For suppose Q' is not in F. Let Q'_0 be the point in the plane that contains F such that Q' and Q'_0 are equidistant from P, and Q'_0 lies in the half-plane that contains Q' and which is determined by a line through P perpendicular to F. Such a Q'_0 exists because the perpendicular distance from P to F is less than d'_t and $dist(Q', P) > \sqrt{2}\, d'_t$. Let H'_0 be the plane that is the perpendicular bisector of $\overline{Q'_0 P}$. Let C'_0 be the open half-space determined by H'_0 that contains P. It follows, as shown in Figure 9, that if C'_0 contains V, which lies above H'', then so does C'. Thus, without any loss of generality, we assume that Q' is contained in F. Further, we assume that $P'\vec{P'_1} \neq P'\vec{P'_2}$ and that Q' is contained in the region in F obtained by a clockwise rotation along F from $P'\vec{P'_2}$ to $P'\vec{P'_1}$ (refer back to Figure 6).

Let H'_1 and H'_2 be the planes that are the perpendicular bisectors of $\overline{P'_1 P}$ and $\overline{P'_2 P}$, respectively. Let R be the region which is the intersection of the closed half-space determined by H'' that does not contain P and the closed half-spaces determined by H'_1 and H'_2 that contain P. Clearly R contains V. We will show that C' contains V by showing that R is the convex hull of a region (K'') and a ray ($\vec{r}''$), both of which lie in C'. Since C' is convex, C' then contains R and the result follows.

The following definitions will be useful. Let H''' be the plane that contains P and is parallel to H''. Let Q''', P'''_1, and P'''_2 be the perpendicular projections onto H''' of Q', P'_1, and P'_2, respectively. Let d'_0, d'_1, d'_2, d'''_0, d'''_1, and d'''_2 be the values of $dist(Q', P)$, $dist(P'_1, P)$, $dist(P'_2, P)$, $dist(Q''', P)$, $dist(P'''_1, P)$, and $dist(P'''_2, P)$, respectively. Let l''_0, l''_1, and l''_2 be the lines that are the intersections of H'' with H', H'_1, and H'_2, respectively. Let l'''_0, l'''_1, and l'''_2 be the lines in H''' that perpendicularly bisect $\overline{Q'''P}$, $\overline{P'''_1 P}$, and $\overline{P'''_2 P}$, respectively. Let P'' be the perpendicular projection of P onto H''. The region K'' is defined to be the wedge obtained by intersecting the half-planes in H'' determined by l''_1 and l''_2 that contain P'' (Figure 10). Another region, K''', is similarly defined with respect to H''', l'''_1, l'''_2, and P (Figure 11).

We begin the task of showing that C' contains the region K'' by arguing that $d'''_0 > \sqrt{2}\, dmax(P, \{P'''_1, P'''_2\})$.

$(Q' + P)/2$, $(P'_1 + P)/2$, $(P'_2 + P)/2$, $(Q''' + P)/2$, $(P'''_1 + P)/2$, $(P'''_2 + P)/2$ are contained in l''_0, l''_1, l''_2, l'''_0, l'''_1, l'''_2, respectively, so that from the definitions of Q''', P'''_1, and P'''_2, it follows by similar triangles that l'''_0, l'''_1, and l'''_2 are the perpendicular projections onto H''' of l''_0, l''_1, and l''_2, respectively. If h' is the

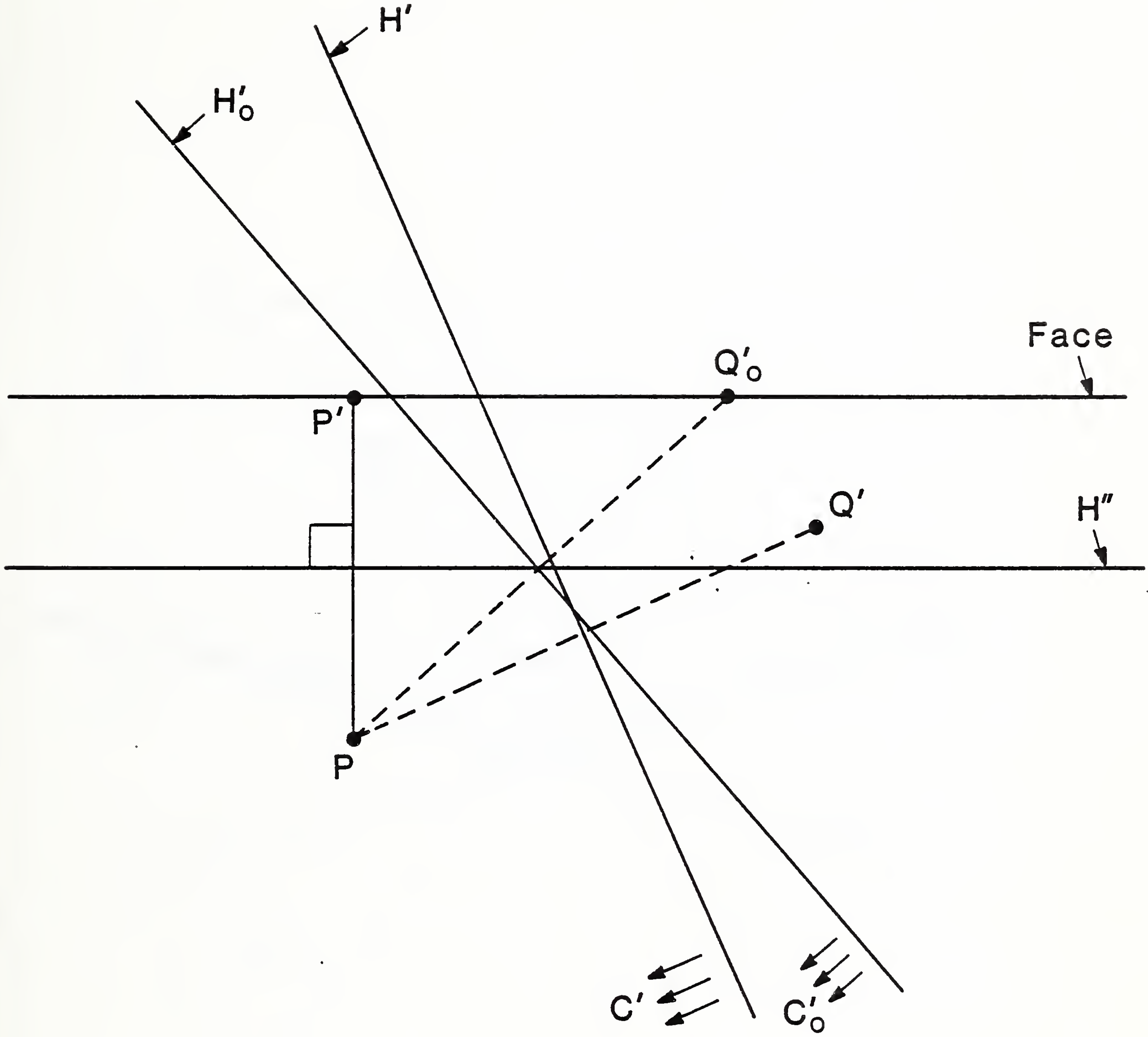

Figure 9: Perpendicular view of unique plane that contains P, Q'_0, and Q'.

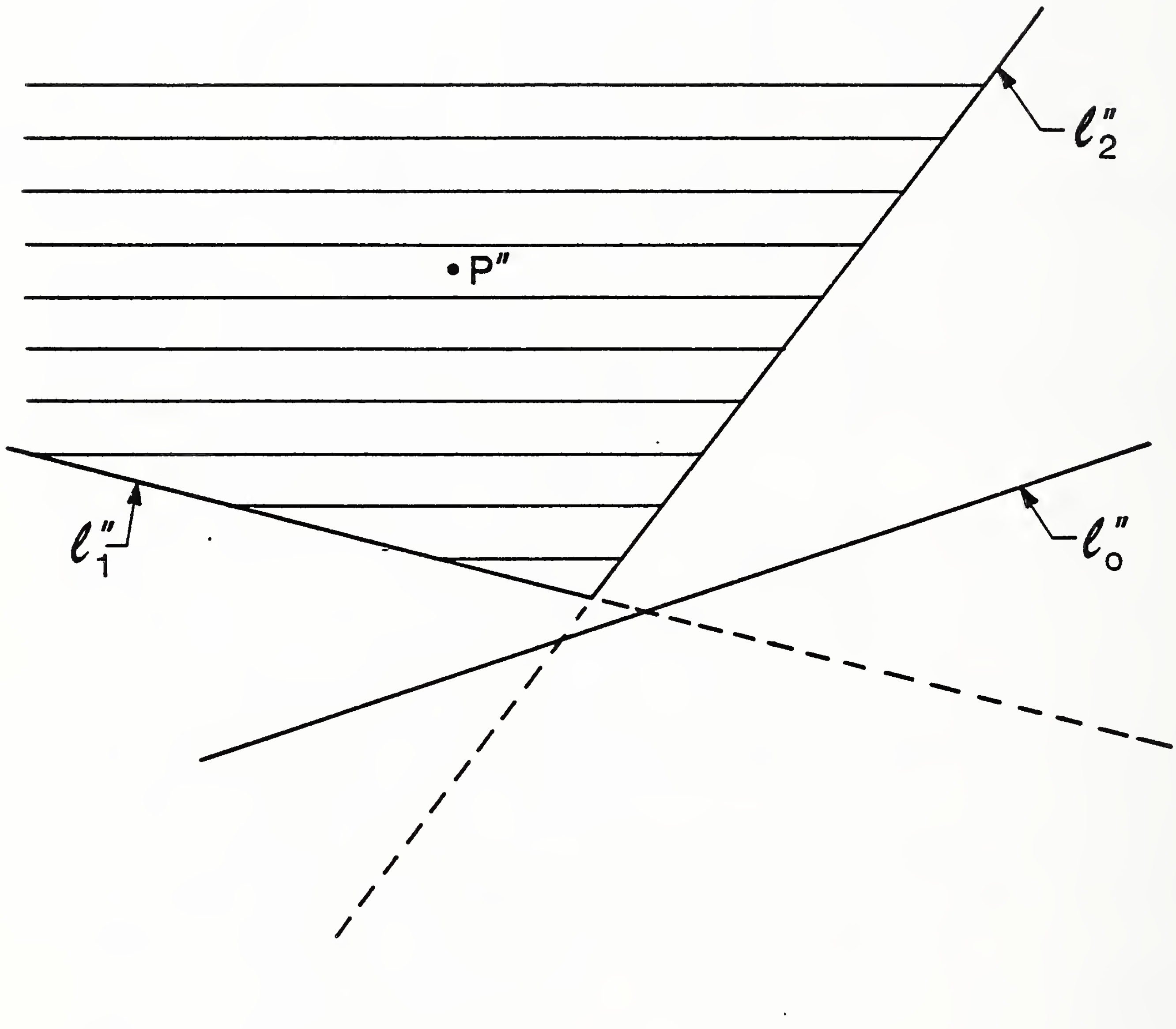

Figure 10: Perpendicular view of H''. The shaded area is the wedge K''.

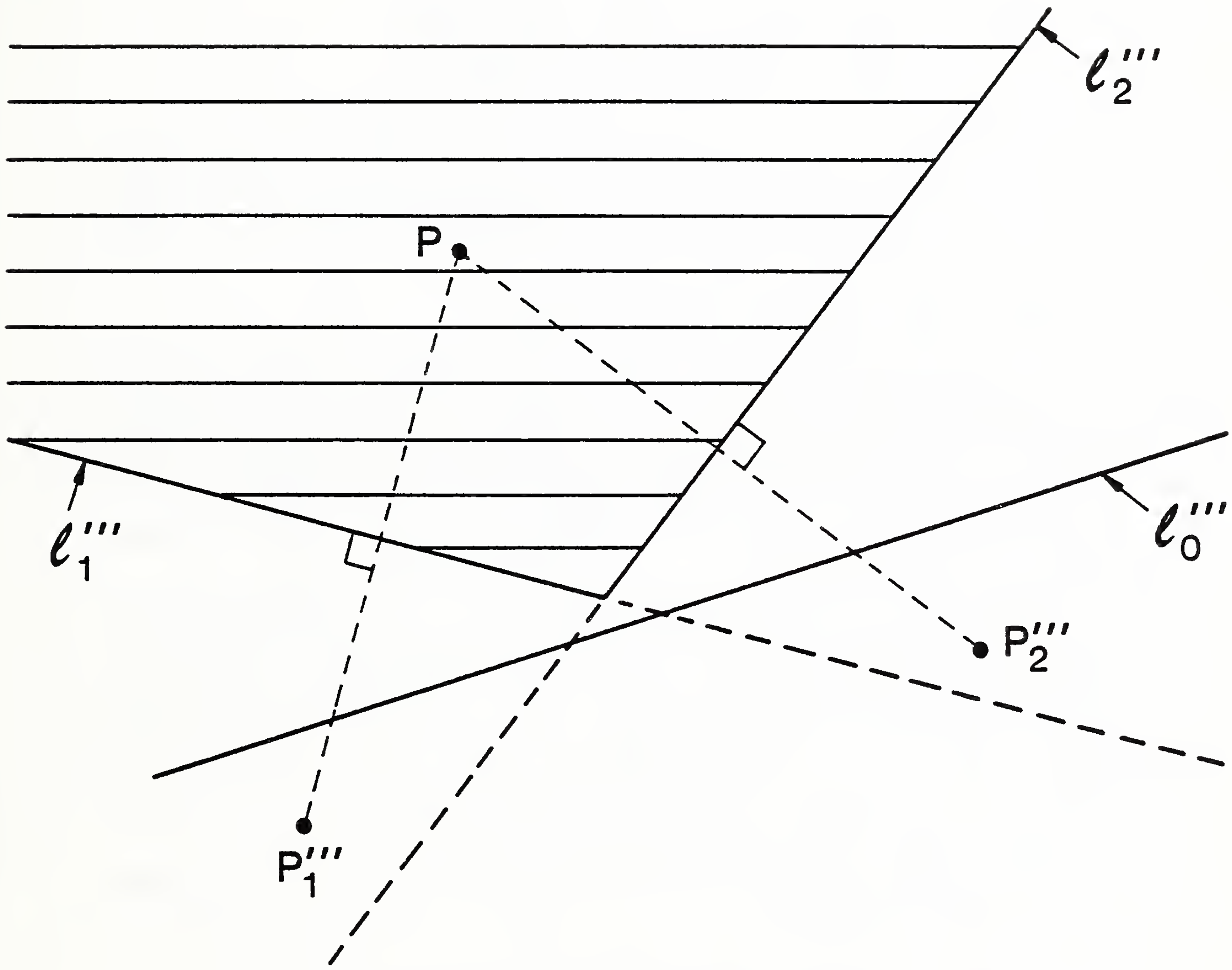

Figure 11: Perpendicular view of H'''. The shaded area is the wedge K'''.

value of $dist(P', P)$ then, since $d_0' > \sqrt{2}\, d_t'$, we have

$$\begin{aligned}
(d_0''')^2 + (h')^2 &= (d_0')^2 \\
&> 2\,(d_t')^2 \\
&\geq 2\,(d_1')^2 \\
&= 2\,((d_1''')^2 + (h')^2) \\
&> 2\,(d_1''')^2 + (h')^2.
\end{aligned}$$

Thus, $d_0''' > \sqrt{2}\, d_1'''$. Similarly $d_0''' > \sqrt{2}\, d_2'''$, which shows

$$d_0''' > \sqrt{2}\, dmax(P, \{P_1''', P_2'''\}).$$

Since P_1' and P_2' are in contiguous octants, it follows that l_0''' does not intersect K'''. In addition, since K''' is the perpendicular projection onto H''' of K'', it also follows that l_0'' does not intersect K'' (Figures 10 and 11). In other words, by the definition of l_0'', C' contains K''.

Next, we define $\vec{r}''$ and show it is contained in C'.

Let H^* be the unique plane that contains P, P_1', and P_2'. Let C^* be the closed half-space determined by H^* that contains P'. Since, in particular, $d_0' > dmax(P, \{P_1', P_2'\})$, it follows that some convex combination, say T', of P_1' and P_2' is also a convex combination of P' and Q' with $T' \neq Q'$ (Figure 12). Thus, since P' is an interior point of C^*, and T' is contained in H^*, it follows that Q' does not belong to C^*.

Let l'' be the line that is the intersection of the planes H_1' and H_2'. Let H^{**} be the plane that contains P and Q' and that is perpendicular to H^*. Let l^{**} be the perpendicular projection onto H^{**} of l''. Since l'' is perpendicular to H^* (Figure 13), so is l^{**}. Thus, as shown in Figure 14, l^{**} contains a ray $\vec{r}^{**}$ that lies wholly in $C' \cap C^*$. Therefore, by the definition of l^{**}, it follows that l'' contains a ray $\vec{r}''$ that is also contained in $C' \cap C^*$.

Since both K'' and $\vec{r}''$ lie in C', their convex hull R lies in the convex region C', and our assertion is proved.

Proof of complexity. We now assume that the points in S have been chosen from a uniform distribution in the cube. Under this assumption, we claim that the expected time involved with the above algorithm is $O(N)$ and $O(N^{4/3})$ for obtaining all Voronoi polyhedra of points in $S \setminus S_5$ and S_5, respectively. Again, because of similarities with the two-dimensional algorithm, we only present the proof for S_5.

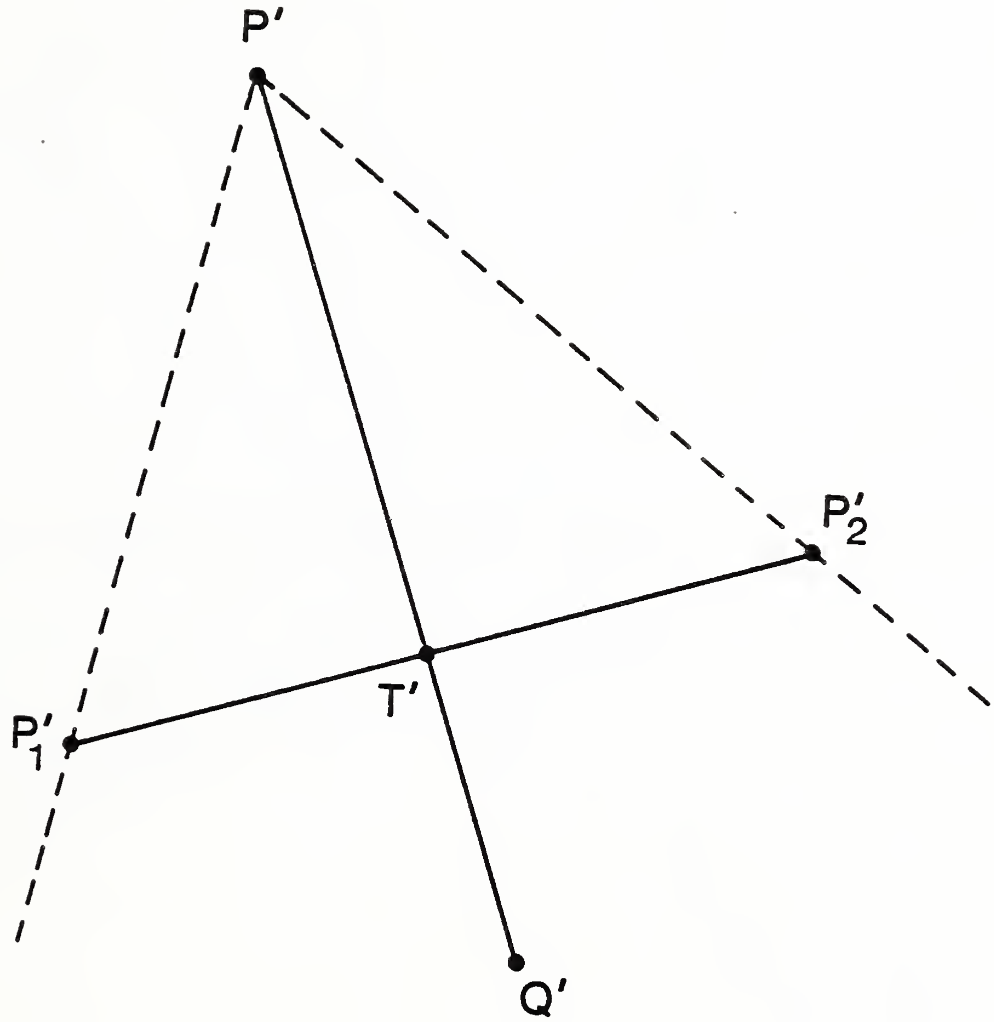

Figure 12: Perpendicular view of face of cube closest to P. $\overline{P_1'P_2'}$ and $\overline{P'Q'}$ intersect at the point T'.

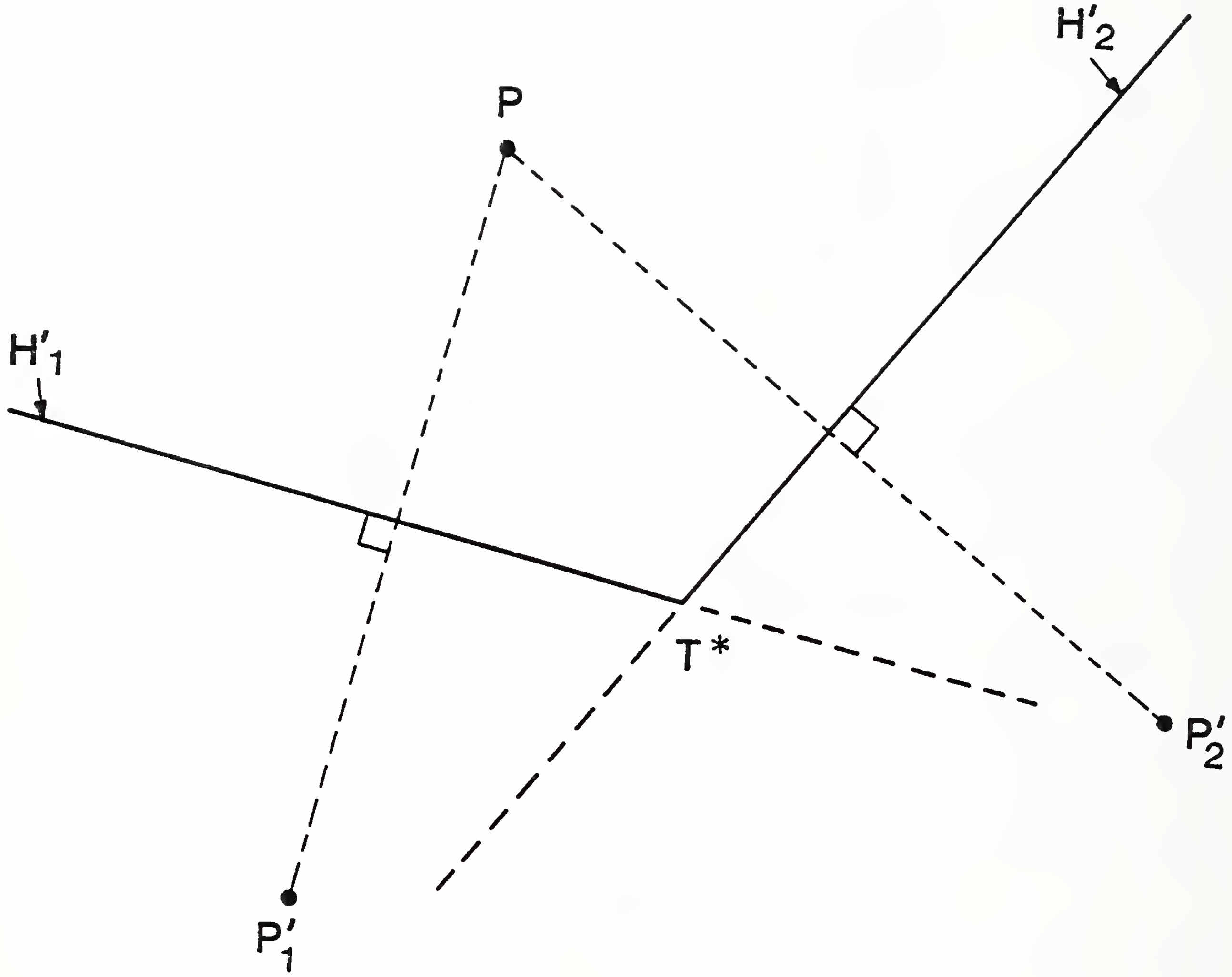

Figure 13: Perpendicular view of plane H^*. l'' is perpendicular to H^* through the point T^*.

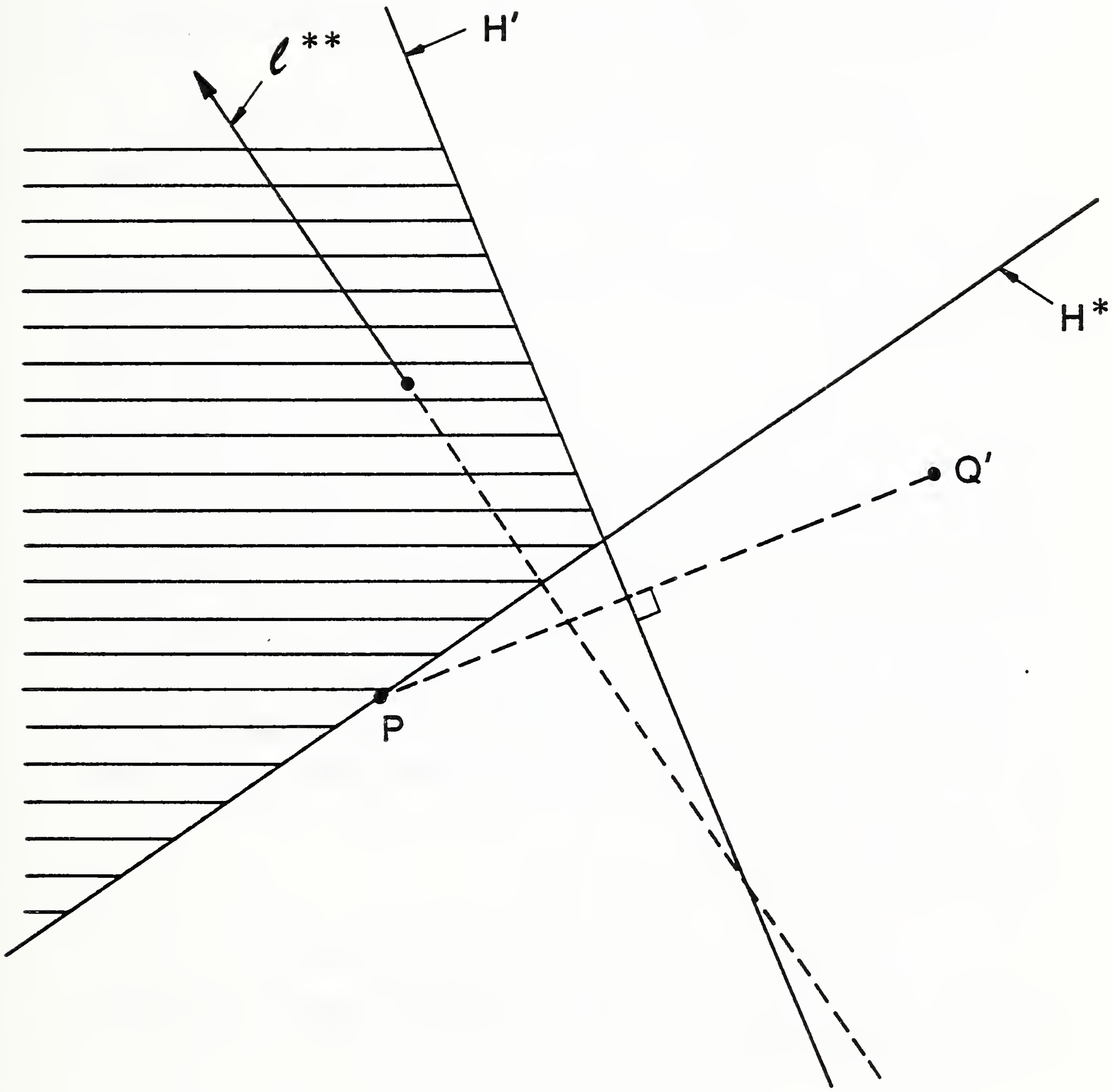

Figure 14: Perpendicular view of plane H^{**}. The shaded area is $C' \cap C^*$. The ray $\vec{r}^{**}$ is the solid portion of the line l^{**}.

Let Z_5 be as defined in Step 8 of the algorithm. We first show that the expected number of points in $S_5 \cup Z_5$ is $O(N^{2/3})$.
Clearly, the expected number of points in S_5 is $O(N^{2/3})$. Thus, it suffices to prove that the expected number of points in Z_5 is $O(N^{2/3})$.
In what follows, given G, a finite nonempty set in Euclidean space, and G_1, G_2, nonempty subsets of G, we define $N(G, G_1, G_2)$ as the number of points in G_2 that are Voronoi neighbors in G of points in G_1. Accordingly, $E(N(G, G_1, G_2))$ is defined as the expected value of $N(G, G_1, G_2)$.
The following observation is crucial to our proof:
Let X, Y be finite nonempty sets in Euclidean space. Let X' be a nonempty subset of X. Then

$$N(X, X', X) \leq N(X \cup Y, X', X) + N(X \cup Y, Y, X).$$

We prove a stronger result. Let P and P' be points in X and X', respectively. We show that if P' is a Voronoi neighbor in X of P then either P' is a Voronoi neighbor in $X \cup Y$ of P or there exists at least one point in Y which is a Voronoi neighbor in $X \cup Y$ of P. If not, all Voronoi neighbors in $X \cup Y$ of P lie in X, none of which is P'. But this implies that the Voronoi polyhedron of P relative to X coincides with the Voronoi polyhedron of P relative to $X \cup Y$, so that P' cannot be a Voronoi neighbor in X of P, a contradiction.
Let B be the cube from which the set S has been chosen. Let B' be the cube obtained by surrounding B with $L(N) + 1$ additional layers of cells. Define cells in $B' \setminus B$ as *exterior cells*. Let Z be the set whose members are the centroids of the exterior cells.
It follows from the observation above that

$$N(S, S_5, S) \leq N(S \cup Z, S_5, S) + N(S \cup Z, Z, S).$$

Thus, since each term above is a random variable that depends solely on the choice of S, we must have

$$E(N(S, S_5, S)) \leq E(N(S \cup Z, S_5, S)) + E(N(S \cup Z, Z, S)).$$

Let Z' be the set of points in Z contained in the first layer of exterior cells that surrounds B. From the definition of Z, points in Z' are the only points in Z that can be Voronoi neighbors in $S \cup Z$ of points in S. Clearly, points in $S_5 \cup Z'$ are not contained in any of the outermost $L(N)$ layers of cells of B', so that the expected number of Voronoi neighbors in $S \cup Z$ of a point in $S_5 \cup Z'$ is $O(1)$ [1]. Since the expected number of points in $S_5 \cup Z'$ is $O(N^{2/3})$, this implies

$$E(N(S \cup Z, S_5, S)) = E(N(S \cup Z, Z, S)) = O(N^{2/3}),$$

so that

$$E(N(S, S_5, S)) \leq O(N^{2/3}) + O(N^{2/3}) = O(N^{2/3}).$$

Since $E(N(S, S_5, S))$ is by definition an upper bound on the expected number of points in Z_5, our assertion follows.

Finally, we show that the expected time required to perform the insertion process of Step 8 is bounded above by $O(N^{4/3})$.

Assume, inductively, that the Voronoi diagram for $k-1$ points in $S_5 \cup Z_5$ has been constructed. Since the expected number of points in $S_5 \cup Z_5$ is $O(N^{2/3})$, it follows that $O(N^{2/3})$ is an upper bound on the expected time required to find that one of the $k-1$ points closest to an additional k^{th} point, and that the expected number of vertices of any polyhedron in the Voronoi diagram for the $k-1$ points is at most $O(N^{2/3})$. Thus, the expected time required to find a vertex of the Voronoi diagram for the $k-1$ points that does not belong to the Voronoi diagram for all k points is at most $O(N^{2/3})$. This implies that the expected time required by the first step of the insertion process, as described in the introduction, is $O(N^{2/3})$ for each point in $S_5 \cup Z_5$, and

$$O(N^{2/3}) \cdot O(N^{2/3}) = O(N^{4/3})$$

for all of $S_5 \cup Z_5$.

Next, we show that the expected time required by the remainder of the insertion process is also $O(N^{4/3})$ for all of $S_5 \cup Z_5$. As described in the introduction, and again assuming that the Voronoi diagram for $k-1$ points in $S_5 \cup Z_5$ has been constructed, the remainder of the insertion process consists of deleting those vertices in the Voronoi diagram for the $k-1$ points that lie in the Voronoi polyhedron of the k^{th} point, and adding to the diagram the vertices of the Voronoi polyhedron of the k^{th} point. Let r be the number of points in $S_5 \cup Z_5$, and for each k, $1 \leq k \leq r$, let $d(k)$ and $a(k)$ be the number of vertices that are deleted and added, respectively, when updating for the k^{th} point in $S_5 \cup Z_5$ the Voronoi diagram for the first $k-1$ points. Clearly, for each k, $1 \leq k \leq r$, the expected value of $a(k)$ is at most $O(N^{2/3})$, so that the expected value of $\sum_{k=1}^{r} a(k)$ is at most $O(N^{4/3})$. Also, since deleted vertices come from the set of previously added vertices, we must have

$$\sum_{k=1}^{r} d(k) \leq \sum_{k=1}^{r} a(k),$$

and the desired result follows.

5. Computational experience

The two-dimensional algorithm was implemented in standard FORTRAN on a Control Data Cyber 205[1] at the National Bureau of Standards. The storage of the data structure and the treatment of degenerate vertices were based on considerations from [2]. Table 1 shows the computing times per

N	*Proposed algorithm*	*Bowyer's algorithm*
8,100	1.27×10^{-3}	1.07×10^{-3}
14,400	1.37×10^{-3}	1.29×10^{-3}
52,900	1.30×10^{-3}	1.97×10^{-3}
102,400	1.28×10^{-3}	2.59×10^{-3}
122,500	1.28×10^{-3}	2.80×10^{-3}

Table 1: Computing time per Voronoi polygon.

Voronoi polygon for the algorithm when applied to several regular nontrivial configurations. The times obtained when applying the modified version of Bowyer's algorithm to the same sets are also shown. The unit of computing time is given in CPU seconds.

Acknowledgments

The author wishes to express his deep gratitude to Dr. Sally E. Howe who strongly influenced this work and who provided invaluable assistance in reviewing several versions of the manuscript. He also thanks Dr. Christoph

[1]Any references to commercial products do not imply recommendation or endorsement by the National Bureau of Standards of these products.

Witzgall for reading the manuscript and offering his criticism and helpful suggestions. Finally, his gratitude also goes to Dr. Francis Sullivan for his encouragement, and to Mrs. Rosemary Massengill and Mrs. Ellen Sullivan for their help in typing the first version of the manuscript.

References

[1] J. L. Bentley, B. W. Weide and A. C. Yao, Optimal expected-time algorithms for closest point problems, ACM Trans. Math. Software, 6, 4 (1980), pp. 563-580.

[2] A. Bowyer, Computing Dirichlet tessellations, Comput. J., 24, 2 (1981), pp. 162-166.

[3] M. I. Shamos, Computational geometry, Ph.D. dissertation, Yale Univ., New Haven, Conn., 1978.

BS-114A (REV. 2-8C)

U.S. DEPT. OF COMM.

BIBLIOGRAPHIC DATA SHEET *(See instructions)*

1. PUBLICATION OR REPORT NO. NBSIR 87-3679

2. Performing Organ. Report No.

3. Publication Date NOVEMBER 1987

4. TITLE AND SUBTITLE

Expected O(N) and O($N^{4/3}$) algorithms for constructing Voronoi diagrams in two and three dimensions

5. AUTHOR(S)

Javier Bernal

6. PERFORMING ORGANIZATION *(If joint or other than NBS, see instructions)*

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS *(Street, City, State, ZIP)*

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)*

Bentley, Weide and Yao have proposed an expected O(N) cell technique for computing Voronoi diagrams in two dimensions that does not generalize readily to three. In this paper their work is further developed and generalized to produce expected O(N) and O($N^{4/3}$) algorithms for constructing Voronoi diagrams in two and three dimensions, respectively. Computational experience is presented for the algorithm in two dimensions.

12. KEY WORDS *(Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)*

algorithm; computational geometry; computational complexity; Voronoi diagram; expected time analysis; Voronoi polyhedra

13. AVAILABILITY

[X] Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

[X] Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES 43

15. Price $11.95