Maximal Covering by Two Isothetic Unit Squares

Priya Ranjan Sinha Mahapatra [*] Partha P. Goswami [*] Sandip Das [†]

Abstract

Let P be the point set in two dimensional plane. In this paper, we consider the problem of locating two isothetic unit squares such that together they cover maximum number of points from P. In case of overlapping, the points in their common zone are counted once. To solve the problem, we propose an algorithm that runs in $O(n^2 \log^2 n)$ time using $O(n \log n)$ space.

1 Introduction

Encloser problems of many variations involving a point set $P = \{p_1, p_2, \ldots, p_n\}$ have been extensively studied in computational geometry. Problems of computing smallest enclosing circle [15], triangle [4, 11, 14], square and rectangle [17] are well known. The problem of finding the smallest enclosing convex polygon is the famous convex hull problem.

Finding the smallest region of given type that contains k points of P, that is, the problem of computing smallest k enclosing region is an important variation of enclosure problem. Efrat et al. [9, 12] studied the problem of computing smallest k-enclosing circle and k-enclosing homothetic copy of a given convex polygon. Eppstein and Erickson [10] studied a number of extensions including finding subsets of size k from the given set P that minimize area, perimeter, diameter, and circumradius. Problems of computing k-enclosing rectangles and squares are also studied [1, 5, 8, 10, 16] extensively.

A closely related problem is to find the placement of one or more copies of a given region to maximize the size k of the subset covered. In other words, instead of fixing k and computing an optimal enclosing region, the problem is to maximize the number of points covered by the given region(s) of fixed size and shape. This type of problem has similar applications as the problems mentioned above. These so called problems of maximal covering by convex objects has also received attention of many researchers. Barequet et al. [2] proposed an algorithm to cover maximum number of points from a planar point set P by a given convex polygon with m vertices in $O(nk \log(mk) + m)$ time using $O(m + n)$ space. In the context of bichromatic planar point set, Diaz-Banez et al. [7] proposed algorithms for maximal covering by two disjoint isothetic unit squares and circles in $O(n^2)$ and $O(n^3 \log n)$ time respectively. They later improved the complexities to $O(n \log n)$ and $O(n^{8/3} \log^2 n)$ time respectively [6]. The optimal $O(n \log n)$ time algorithm for the maximal covering by two disjoint isothetic unit squares was proposed by Mahapatra et al. [13].

In this paper we consider another natural variation of the maximal covering problem. We study the problem of computing two isothetic unit squares, which may not be disjoint, such that together they cover maximum number of points from P. In case they are overlapping, points in their common zone are counted once. Our proposed algorithm for the problem runs in $O(n^2 \log^2 n)$ time and uses $O(n \log n)$ space.

2 Overview

Let P be the set of n points in two dimensional plane and R_1 and R_2 be two isothetic unit squares. Our objective is to compute placement of R_1 and R_2 such that the number of points in the region $R_1 \cup R_2$ is maximized. Number of points contained by a region R is denoted by $|R|$. In the optimal placement, following cases may occur.

- The squares are disjoint.
- The squares are overlapping and the common zone is empty.
- The squares are overlapping and the common zone is nonempty.

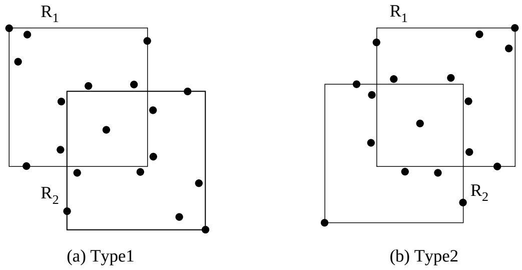
In case the optimal solution belongs to first two cases, the solution can be reported using the algorithms proposed by Mahapatra et al. [13]. When the optimal solution is overlapping, our proposed algorithm returns the optimal pair. We must concentrate in locating the optimal pair of squares on the region where there is a possibility of overlap between two squares. Note that, a pair of squares having same top boundary or same left boundary cannot be the optimal pair. In case R_1 and R_2 have an overlapping region, then depending upon the position of the overlapped region, the placement of R_1 and R_2 can be classified into two types as depicted in Figure 1. We tackle each case separately.

3 Characterization

Consider two arrays L_x and L_y containing the points of P in ascending order with respect to their x- and y-coordinates respectively. Let the x-coordinate of the i-th entry of L_x be x_i and similarly the y-coordinate of

[*] University of Kalyani, Kalyani, India

[†] Indian Statistical Institute, Kolkata, India



(a) Type1

(b) Type2

Figure 1: Placement of R_1 and R_2

the i-th entry of L_y be y_i, $1 \leq i \leq n$. We use the implicit grid obtained by drawing vertical and horizontal lines through each point of the given set P. The grid point (x_i, y_j), $(1 \leq i, j \leq n)$ is generated by the intersection of the vertical line through the point in the i-th entry of L_x and the horizontal line through the point in the j-th entry of L_y. The coordinate of a generic point p is denoted by (p_x, p_y).

A square can be specified using its top left corner. Here, $S(i, j)$ denotes a square whose top left corner is at (x_i, y_j). Consider a horizontal line l_α on the grid having y-coordinate y_α above the square $S(i, j)$ for some i, j, $1 \leq i, j \leq n$. Initially we are trying to identify a square S' whose top boundary is aligned with l_α such that the square S' together with $S(i, j)$ cover maximum number of points and whenever they overlap, configuration of overlapping region is of Type-1 (See Figure 1(a)). Hence we can obtain the optimal pair by choosing all possible $S(i, j)$ and α. Note that, the placement of the upper square S' depends upon the choice of α and the lower square $S(i, j)$. Upper square S' has the following characteristics.

(1) Given α and a lower square $S(i, j)$, the upper square S' belongs to the set $\{S(1, \alpha), S(2, \alpha), \ldots, S(i, \alpha)\}$

(2) If $|S(a, \alpha)| \geq |S(b, \alpha)|$ where $a < b \leq i$ then $|S(a, \alpha) \cup S(i, j)| \geq |S(b, \alpha) \cup S(i, j)|$.

3.1 Matching

Given an index α, the matching of $S(i, j)$ with respect to α $(\alpha \geq j)$ is defined as a square $S(k, \alpha)$ such that $|S(k, \alpha) \cup S(i, j)| > |S(k', \alpha) \cup S(i, j)|$ for $k' = 1, 2, \ldots k-1$ and $|S(k, \alpha) \cup S(i, j)| \geq |S(k', \alpha) \cup S(i, j)|$ for $k' = k+1, k+2, \ldots i$.

Lemma 1 *For a given α and j, $\alpha \geq j$, let the matching of $S(b, j)$ be $S(k, \alpha)$ and the matching of $S(c, j)$, $b < c \leq n$ be $S(k', \alpha)$. Then $k \leq k'$.*

Proof. As the matching of $S(b, j)$ is $S(k, \alpha)$, $|S(t, \alpha) - S(b, j)| < |S(k, \alpha) - S(b, j)|$ for $t = 1, 2, \ldots, k-1$. Again $(S(b, j) - S(c, j)) \cap S(t, \alpha) \subseteq (S(b, j) - S(c, j)) \cap S(k, \alpha)$. This implies $|S(t, \alpha) - S(c, j)| = |S(t, \alpha) - S(b, j)| + |(S(b, j) - S(c, j)) \cap S(t, \alpha)| < |S(k, \alpha) - S(b, j)| + |(S(b, j) - S(c, j)) \cap S(k, \alpha)| = |S(k, \alpha) - S(c, j)|$ for $t = 1, 2, \ldots, k-1$. Hence the result follows. □

Lemma 2 *Given indices α and j, $\alpha \geq j$, let the matching of both $S(a, j)$ and $S(b, j)$ be $S(k, \alpha)$, $1 \leq a, b \leq n$. If $a < b$ then the matching of each $S(i, j)$, $a \leq i \leq b$ is $S(k, \alpha)$.*

Proof. Let the matching of $S(i, j)$ be $S(k', \alpha)$ for some i, $a < i < b$. Since the matching of $S(a, j)$ is $S(k, \alpha)$ and $a < i$, from Lemma 1, we get $k \leq k'$. Similarly, as the matching of $S(b, j)$ is $S(k, \alpha)$ and $i < b$ so $k' \leq k$. This implies $k = k'$. □

Given α and j $(\alpha > j)$, for computing matching of all $S(i, j)$, $1 \leq i \leq n$, we can reduce the search space from the fact stated in Lemma 1. Observe that the matching of $S(i, j)$, for all i can be computed in $O(n \log n)$ time. This implies, for a given α, the matching of all $S(i, j)$'s, $1 \leq i \leq n$ and $\alpha > j$, can be computed in $\mathrm{O}(n^2 \log n)$ time. Hence the matching of all $S(i, j)$'s, $1 \leq i \leq n$ and $1 \leq j \leq n$, can be computed in $\mathrm{O}(n^3 \log n)$ time. In this paper, we propose an algorithm to compute the matching of all possible squares in $\mathrm{O}(n^2 \log^2 n)$ time. Below we describe some more characterizations to achieve the sub-cubic complexity of the proposed algorithm.

From Lemma 2, we conclude that for given α and j $(\alpha \geq j)$, there exists an interval $[a, b]$ such that all $S(i, j)$'s for $a \leq i \leq b$ are matched with $S(k, \alpha)$ and the matching of each $S(i, j)$ for $b < i \leq n$ or $1 \leq i < a$ is different from $S(k, \alpha)$. Here we denote such an interval $[a, b]$ using notation $\mathcal{F}(S(k, \alpha), j)$. An interval $\mathcal{F}(S(k, \alpha), j)$ is empty whenever there does not exist any square $S(i, j)$, $1 \leq i \leq n$, whose matching is $S(k, \alpha)$.

Observation 1 (a) *Intervals $\mathcal{F}(S(k, \alpha), j)$ and $\mathcal{F}(S(k', \alpha), j)$ are disjoint for $k \neq k'$, $1 \leq k, k' \leq n$.* (b) *Moreover, if both the intervals are not $\emptyset$ and $k < k'$ then interval $\mathcal{F}(S(k', \alpha), j)$ is on the right side of the interval $\mathcal{F}(S(k, \alpha), j)$.* (c) $\bigcup_{k=1}^{n} \mathcal{F}(S(k, \alpha), j) = [1, n]$.

Proof. (a) Note that for a given α and j, the matching of $S(i, j)$, $1 \leq i \leq n$ is unique and hence $\mathcal{F}(S(k, \alpha), j) \cap \mathcal{F}(S(k', \alpha), j) = \emptyset$ for $k \neq k'$. Statement (b) follows from Lemma 1. □

Lemma 3 *For a given α and j, $\alpha > (j+1)$, if the matchings of $S(i, j)$ and $S(i, j+1)$ are $S(k, \alpha)$ and $S(k', \alpha)$ respectively then $k \geq k'$.*

Proof. As the matching of $S(i, j)$ is $S(k, \alpha)$, $|S(t, \alpha) - S(i, j)| \leq |S(k, \alpha) - S(i, j)|$ for $t = 1, 2, \ldots, k-1, k+1, \ldots, i$. Again $S(t, \alpha) \cap S(i, j) \supset S(k, \alpha) \cap S(i, j)$ for $t = k+1, \ldots, i$ and $S(t, \alpha) \cap S(i, j+1) - S(i, j) \supset S(k, \alpha) \cap S(i, j+1) - S(i, j)$ for $t = k+1, \ldots, i$. Hence, we have $|S(t, \alpha) - S(i, j+1)| \leq |S(k, \alpha) - S(i, j+1)|$ for $t = k+1, \ldots, i$. □

Observation 2 *For a given α and j, $\alpha > (j+1)$, let the matching of $S(i, j)$ be $S(k, \alpha)$ and $(S(i, j+1) \cap S(k, \alpha)) - S(i, j) = \emptyset$. Then the matching of $S(i, j+1)$ is $S(k, \alpha)$.*

Proof. Matching of $S(i,j)$ is $S(k,\alpha)$ and therefore, $|S(k,\alpha)-S(i,j)| > |S(t,\alpha)-S(i,j)|$ for $t=1,2,\dots k-1$, and $|S(k,\alpha)-S(i,j)| \geq |S(t,\alpha)-S(i,j)|$ for $t = k+1, k+2, \dots i$. Now, $(S(i,j+1) \cap S(k,\alpha)) - S(i,j) = \emptyset$ implies $|S(k,\alpha)-S(i,j+1)| = |S(k,\alpha)-S(i,j)|$. Hence, $|S(k,\alpha)-S(i,j+1)| > |S(t,\alpha)-S(i,j)| \geq |S(t,\alpha)-S(i,j+1)|$ for $t=1,2,\dots k-1$, and $|S(k,\alpha)-S(i,j+1)| \geq |S(t,\alpha)-S(i,j+1)|$ for $t=k+1,k+2,\dots i$. Hence the result follows. □

Lemma 4 *For $\mathcal{F}(S(k,\alpha),j) \neq \emptyset$, $\alpha > j+1$, either one of the following is true.*

(1) $\mathcal{F}(S(k,\alpha),j+1) = \emptyset$

(2) $\mathcal{F}(S(k,\alpha),j+1) = \mathcal{F}(S(k,\alpha),j)$

(3) $\mathcal{F}(S(k,\alpha),j+1) \subset \mathcal{F}(S(k,\alpha),j)$

(4) $\mathcal{F}(S(k,\alpha),j+1) \supset \mathcal{F}(S(k,\alpha),j)$

There exists at most one value of k, $1 \leq k \leq n$, that satisfy Case (3) *and similarly, there exist at most one value of k, $1 \leq k \leq n$, that satisfy Case* (4).

Proof. Let $p(x_m, y_{j+1})$ be a point in P. Suppose $\mathcal{F}(S(k,\alpha),j) = [a,b]$. If $|(S(i,j+1) \cap S(k,\alpha)) - S(i,j)| = 0$ for some i's, $a \leq i \leq b$, then $|(S(r,j+1) \cap S(k,\alpha)) - S(r,j)| = 0$ for $r = i, i+1, \dots, b$. In this case, from Observation 2, the matching of $S(r,j+1)$ is $S(k,\alpha)$. Therefore, if $p \notin S(k,\alpha)$ then $i = a$ and $\mathcal{F}(S(k,\alpha),j+1) \supseteq \mathcal{F}(S(k,\alpha),j)$. Moreover, if matching of $S(b+1,j)$ is $S(k',\alpha)$ and $|(S(b+1,j+1) \cap S(k',\alpha)) - S(b+1,j)| = 0$, the matching of $S(b+1,j+1)$ is $S(k',\alpha)$ and then it implies $\mathcal{F}(S(k,\alpha),j+1) = \mathcal{F}(S(k,\alpha),j)$.

Suppose the matching of $S(i,j)$ and $S(i,j+1)$ are $S(k,\alpha)$ and $S(k',\alpha)$ respectively. From Lemma 3, $k' \leq k$. Let p be in $((S(i,j+1) \cap S(k,\alpha)) - S(i,j)$. If $\mathcal{F}(S(k',\alpha),j) = (c,d]$ and $k' < k$, then $p \notin S(k',\alpha)$ and the matching of $S(r,j+1)$ is $S(k',\alpha)$ for $r = d+1, d+2, \dots i$. Here, $\mathcal{F}(S(k,\alpha),j+1) \subset \mathcal{F}(S(k,\alpha),j)$ and $\mathcal{F}(S(k',\alpha),j+1) \supset \mathcal{F}(S(k',\alpha),j)$. In case, $i = b$, $\mathcal{F}(S(k,\alpha),j+1) = \emptyset$. Hence the lemma follows. □

Suppose the matching of $S(i-1,j)$ and $S(i,j)$ are $S(k',\alpha)$ and $S(k,\alpha)$ respectively with $k' < k$. Let the point $p(x_m, y_{j+1})$ lie inside the region $((S(i,j+1) \cap S(k,\alpha)) - S(i,j)$ but not inside the region $((S(i-1,j+1) \cap S(k',\alpha)) - S(i-1,j)$. Then using similar arguments as in the proof of Lemma 4, $\mathcal{F}(S(k',\alpha),j+1) \supseteq \mathcal{F}(S(k',\alpha),j)$ and for all $k'' < k'$, $\mathcal{F}(S(k'',\alpha),j+1) = \mathcal{F}(S(k'',\alpha),j)$. If $\mathcal{F}(S(k',\alpha),j+1) = \mathcal{F}(S(k',\alpha),j)$, then $\mathcal{F}(S(r,\alpha),j+1) = \mathcal{F}(S(r,\alpha),j)$ for $r = 1,2,\dots,n$. When $\mathcal{F}(S(k',\alpha),j+1) \supset \mathcal{F}(S(k',\alpha),j)$, there exist an integer v, $0 \leq v \leq n$, such that the matching of $S(v,j+1)$ is $S(k',\alpha)$ but the matching $S(v+1,j+1)$ is not $S(k',\alpha)$. Suppose the matching of $S(v+1,j)$ is $S(h,\alpha)$, then from Lemma 4 along with similar arguments, we can conclude that $\mathcal{F}(S(r,\alpha),j+1) = \emptyset$ for $r = k'+1, k'+2, \dots, h-1$, $\mathcal{F}(S(h,\alpha),j+1) \subseteq \mathcal{F}(S(h,\alpha),j)$ and $\mathcal{F}(S(r,\alpha),j+1) = \mathcal{F}(S(r,\alpha),j)$ for $r = h+1, h+2, \dots, n$.

Observation 3 *For given indices α and β, $\alpha \geq \beta$, with $(y_\alpha - y_\beta) \geq 1$, all nonempty $\mathcal{F}(S(k,\alpha),\beta)$ can be computed in linear time.*

4 Algorithm

Here, we are looking for a pair of overlapping squares where top boundary of the upper square (R_1) is at y_α and top left corner of the lower square (R_2) is inside the interior of upper square such that together they cover maximum number of points of P. Note that R_1 and R_2 may together contain less number of points than a non-overlapping pair with top boundary of upper square at y_α. Below, we describe the algorithm to report the maximum number of points covered by a pair of squares with top boundary of the upper square at y_α and the top boundary of the lower square is above the bottom boundary of the upper square in $O(n \log^2 n)$ time.

Let the function $left(p_{x_i})$ output the minimum entry in L_x, say p_{x_j}, such that $x_i - x_j$ is less than or equal to unity. Then we can compute $left(p_{x_i}), 1 \leq i \leq n$ in $O(\log n)$ time. Note that $left(p_{x_1}) = p_{x_1}$. In a similar way, we define *right*(.), *bottom*(.), and *top*(.). For orthogonal range searching on a given points set P in $2D$, we construct range tree R [3] that reports the number of points in a query rectangle in $O(\log n)$ time. The construction time and the space for range tree R are both $O(n \log n)$.

4.1 Data structure and initialization

Let y_γ be the *bottom*(y_α) $(1 \leq \alpha \leq n)$. Consider an index β such that $|y_\alpha - y_\beta| \geq 1$ and compute $\mathcal{F}(S(k,\alpha),\beta)$ for all k, $1 \leq k \leq n$, which are nonempty. Using observation 3, this computation can be done in linear time. Let the intervals be $[a_1,b_1],[a_2,b_2],\dots,[a_\nu,b_\nu]$ from left to right such that $\mathcal{F}(S(k_i,\alpha),\beta) = [a_i,b_i]$ for $i = 1,2,,\dots,\nu$.

Now we initialize β by γ and subsequently the value β varies from $\gamma+1, \gamma+2, \dots, \alpha-1$.

Consider a balanced binary search tree T constituted by $x_1, x_2, \dots, x_n$, where each value x_i corresponds to a leaf node, and the search is guided by the x-values. The leaf node corresponding to x_i keeps information about $|S(i,\beta) \cup S(k,\alpha)|$, where the matching of $S(i,\beta)$ is $S(k,\alpha)$. Two counter variables $\mathcal{M}$ and $\mathcal{C}$ are attached with nodes of T for computing the number of points covered by $S(i,\beta)$ along with its matching square for all i. To start with, the variables $\mathcal{M}$ and $\mathcal{C}$ corresponding to all nodes are initialized with zero. Given an interval, *Increment* operation modifies T such that count of all squares with top-left corner within the interval is incremented by one. Similarly, we can define the *Decrement* operation. Detail algorithms for *Increment* and *Decrement* operations are discussed by Mahapatra et al. [13]. At any instance with α and β, given a lower square X, *Report* operation [13] is able to report the maximum number of points covered by a pair of squares where the lower square must be X and the upper square have the top boundary at y_α. It can also report the number of maximum covered points of a pair of squares whose top boundaries are aligned with y_α and y_β.

Construct another balanced binary search tree T' constituted by disjoint intervals $[x_{a_1}, x_{b_1}]$, $[x_{a_2}, x_{b_2}]$, $\ldots$, $[x_{a_\nu}, x_{b_\nu}]$ as leaf nodes. A variable W is attached with each leaf node to keep information of a square. All squares having left boundary within the interval corresponding to that leaf node are matched with W. Given a point p, we can report the interval containing p in $O(\log n)$ time. Insertion, deletion and update of an interval can be done in $O(\log n)$ time. We now describe main steps of the algorithm.

For $\beta = \gamma + 1, \gamma + 2, \ldots, \alpha - 1$, execute the following steps:

1. Suppose the β-th entry of L_y be the point p with coordinates, say (x_m, y_β).
2. Find the interval $[a, b]$ from T' containing p and let the corresponding square be $S(k, \alpha)$.
3. Find the inorder-predecessor of $S(k, \alpha)$, say $S(k', \alpha)$ and the corresponding interval is $[a', b']$.
    - 3.1 $p \notin S(k, \alpha)$: Perform *Increment* operation for the interval $[left(x_m), x_m]$ into T.
    - 3.2 $p \in S(k, \alpha)$ and $p \notin S(k', \alpha)$: Compute $|S(a, \beta) \cup S(k, \alpha)|$ and $|S(a, \beta) \cup S(k', \alpha)|$ by rectangular range searching from range tree R. If $|S(a, \beta) \cup S(k', \alpha)| \geq |S(a, \beta) \cup S(k, \alpha)|$,
        - 3.2.1 the matching of $S(a, \beta)$ is $S(k', \alpha)$;
        - 3.2.2 use divide and conquer method within interval $[a, b]$, select an index c such that $\mathcal{F}(S(k, \alpha), \beta)$ is $[c, b]$;
        - 3.2.3 update the interval $[a, b]$ in T' with $[c, b]$ and $[a', b']$ with $[a', c - 1]$;
        - 3.2.4 perform *Increment* operation for the interval $[left(x_m), x_{c-1}]$ and update the current optimal count with the value at the root of T.
    - 3.3 $p \in S(k, \alpha)$ and $p \in S(k', \alpha)$: Identify k'' such that $p \notin S(k'', \alpha)$ but the inorder successor $S(l, \alpha)$ of $S(k'', \alpha)$ contains p. Go to Step 3.2 with $k \leftarrow l$, $k' \leftarrow k''$.

Finally, the count of the optimal pair with top boundary of the upper square at y_α can be obtained at the root of T and the corresponding pair can be identified.

4.2 Complexity of the Algorithm

Theorem 5 *Given a set P of n points in $\mathcal{R}^2$, two disjoint or overlapping isothetic unit squares covering maximum number of points can be found in $O(n^2 \log^2 n)$ time using $O(n \log n)$ space.*

References

[1] A. Aggarwal, H. Imai, N. Katoh, and S. Suri, Finding k points with minimum diameter and related problems. *Journal of Algorithms*, 12, 38–56, 1991.

[2] Gill Barequet, Matthew Dickerson, and Petru Pau, Translating a convex polygon to contain a maximum number of points. *Computational Geometry Theory and Applicaions*, 8, 167–179, 1997.

[3] M. de Berg, M. Van Kreveld, M. Overmars, and O. Schwarzkopf, Computational Geometry, Algorithms and Applications. *Springer*, Berlin, 1997.

[4] S. Chandran and D. Mount, A parallel algorithm for enclosed and enclosing triangles. *International Journal of Computational Geometry and Applications*, 2, 191–214, 1992.

[5] A. Datta, H.-P. Lenhof, C. Schwarz, and M. Smid, Static and dynamic algorithms for k-point clustering problems. *Journal of Algorithms*, 19, 474–503, 1995.

[6] Sergio Cabello, J. Miguel Diaz-Banez, Carlos Seara, J. Antoni Sellares, Jorge Urrutia, Inmaculada Ventura, Covering point sets with two disjoint disks or squares. *Computational Geometry Theory and Applicaions*, 40, 195–206, 2008.

[7] J. Miguel Diaz-Banez, Carlos Seara, J. Antoni Sellares, Jorge Urrutia, and Imma Ventura, Covering Points Sets with Two Convex Objects. *Proc. 21st European Workshop on Computational Geometry*, 2005.

[8] Sandip Das, Partha P. Goswami, and Subhas C. Nandy, Smallest k-point enclosing rectangle and square of arbitrary orientation. *Information Processing Letters*, 95, 259–266, 2005.

[9] A. Efrat, M. Sharir, and A. Ziv Computing the smallest k-enclosing circle and related problems. *Computational Geometry*, 4, 119–136, 1994.

[10] D. Eppstein and J. Erickson, Iterated nearest neighbors and finding minimal polytopes. *Discrete and Computational Geometry*, 11, 321–350, 1994.

[11] V. Klee and M.L. Laskowski, Finding the smallest triangles containing a given convex polygon. *Journal of Algorithms*, 6, 359–375, 1985.

[12] J. Matousek On geometric optimization with few violated constraints. *Discrete and Computational Geometry*, 14, 365–384, 1995.

[13] Priya Ranjan Sinha Mahapatra, Partha P. Goswami, and Sandip Das, Covering Points by Isothetic Unit Squares. *Proc. 19th Canadaian Conference on Computational Geometry*, 169–172, 2007.

[14] J. O'Rourke, A. Aggarwal, S. Maddila, and M. Baldwin, An optimal algorithm for finding minimal enclosing triangles. *Journal of Algorithms*, 7, 258–269, 1986.

[15] F.P. Preparata and M.I. Shamos, Computational Geometry: An Introduction. *Springer*, Berlin, (1985).

[16] M. Segal and K. Kedem, Enclosing k points in smallest axis parallel rectangle. *Information Processing Letters*, 65, 95–99, 1998.

[17] G.T. Toussaint, Solving geometric problems with the rotating calipers. *Proc. IEEE MELECON*, 1983.