Computing the Smallest Color-Spanning Axis-Parallel Square

Payam Khanteimouri[1], Ali Mohades[1],
Mohammad Ali Abam[2], and Mohammad Reza Kazemi[1]

[1] Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran
[2] Sharif University of Technology, Tehran, Iran

Abstract. For a given set of n colored points with k colors in the plane, we study the problem of computing the smallest color-spanning axis-parallel square. First, for a dynamic set of colored points on the real line, we propose a dynamic structure with $O(\log^2 n)$ update time per insertion and deletion for maintaining the smallest color-spanning interval. Next, we use this result to compute the smallest color-spanning square. Although we show there could be $\Omega(kn)$ minimal color-spanning squares, our algorithm runs in $O(n \log^2 n)$ time and $O(n)$ space.

Keywords: Computational Geometry, Algorithm, Color-Spanning Objects.

1 Introduction

Background. Suppose there are k different types of facilities like banks, police offices, etc. and we are given n facilities of these types. A basic problem arising here is to find a region in which there is at least one representative from each type of facilities. This suggests the problems of computing *the smallest area/perimeter color-spanning objects*. A region is said to be *color-spanning* if it contains at least one point from each color. Another motivation comes from *discrete imprecise data*. In this context, each imprecise point is defined with a set of discrete possible locations. So, for a given k imprecise points we have a set of n points with k colors, where n is the number of possible locations for all imprecise points. The basic problem on a set of imprecise points is to locate each imprecise point within its defining set in which a measure becomes optimized [9]. This is equivalent to choose exactly k points with different colors in which a property e.g. diameter, closest pair, bounding box, etc. gets minimized or maximized. Beside these two applications, this problem has other applications in statistical clustering, pattern recognition and generalized range searching [7,10,12].

Related Works. There are several works on colored points. Motivated from imprecise data, Fan et al. [5] showed some results on hardness of *the largest closest pair* and *the minimum planar spanning tree*. Fan et al. [6] also proposed an $O(n^2 \log n)$ time algorithm to compute the expected area of convex hulls of

L. Cai, S.-W. Cheng, and T.-W. Lam (Eds.): ISAAC2013, LNCS 8283, pp. 634–643, 2013.

the color-spanning sets. In addition, Consuegra et al. [3] considered the same setting and solved several other problems.

In the view of location problems, for a given set of n colored points with k colors in the plane, one of the most studied problems is to find *the smallest color-spanning rectangle*. For the axis-parallel case, Abellanas et al. [1] showed there are $\Theta\big((n-k)^2\big)$ minimal rectangles in the worst case and proposed an algorithm of $O\big((n-k)^2 \log^2 k\big)$ running time and $O(n)$ space to solve the problem. The algorithm has been recently improved to $O(n^2 \log n)$ time by Das et al. [4]. For arbitrary oriented case, Das et al. [4] proposed an algorithm in $O(n^3 \log k)$ time and $O(n)$ space. Furthermore, they solved the problem of computing *the smallest color-spanning strip* in $O(n^2 \log n)$ time and $O(n)$ space using the dual of the given points. The results are near efficient with respect to testing all minimal objects. A *minimal color-spanning object* contains at least one point from each color and any sub-region of it does not contain all colors.

For the problem of computing *the smallest color-spanning circle*, Abellanas et al. [2] proposed an algorithm with $O\big(n^2 \alpha(k) \log k\big)$ time using *farthest colored Voronoi diagram (FCVD)*. For a given set of colored points with k colors in the plane, the FCVD is a subdivision of the plane in which for any region R there is a unique site p such that any color-spanning circle centered at a point in R contains p. Indeed, a vertex of the FCVD which is adjacent to three regions with different colors of their sites, is a candidate of being the center of the smallest color-spanning circle. Therefore, a simple algorithm is to compute the FCVD and test circles centered at the vertices of FCVD. The other approach mentioned by Abellanas et al. [1] is to obtain the smallest color-spanning circle and the smallest color-spanning axis-parallel square in $O(kn \log n)$ time and $O(n)$ space using the upper envelope of Voronoi surfaces [8].

Our Results. In this paper, first we consider the problem of maintaining the smallest color-spanning interval for a dynamic set of colored points with k colors on the real line. We propose a data structure which spends $O(\log^2 n)$ update time per insertion and deletion using the structure designed by Overmars and van Leeuwen [11]. Next, we concentrate on the problem of computing the smallest color-spanning axis-parallel square. The algorithm sweeps the points and keeps the smallest color-spanning interval for the projection of the points on the sweep line when insertion or deletion occurs. We use the mentioned dynamic structure for maintaining the smallest color-spanning interval to solve the problem. Our algorithm runs in $O(n \log^2 n)$ time and $O(n)$ space. The algorithm does not test every minimal candidates (in fact, we show that there may be $\Theta(kn)$ minimal color-spanning axis-parallel squares in the worst case). So, this result is an improvement to the result proposed by Abellanas et al. [1] using the upper envelope of Voronoi surfaces [8].

This paper is organized as follows. In Section 2 we show how to maintain the smallest color-spanning interval for a dynamic set of colored points on the real line. Next, in Section 3 we propose an algorithm to compute the smallest color-spanning square using the results of Section 2. Finally we conclude in Section 4.

2 Dynamic Maintenance of Minimal Color-Spanning Intervals

In this section, we concentrate on color-spanning intervals for a set of colored points with k colors on the real line. For ease of the presentation, we assume that points are in general position which means point coordinates are different. We first show some properties of color-spanning intervals for a static set of colored points on the real line. Next, we consider the problem of maintaining the smallest color-spanning interval for dynamic points in which the points are permitted to be inserted or deleted.

2.1 Minimal Color-Spanning Intervals for Static Points

A minimal color-spanning interval for a set of colored points on the real line is an interval containing all colors and any sub-interval of it does not contain all colors. As a simple observation, the endpoints of a minimal color-spanning interval have different colors and their colors are unique in the interval.

Suppose we are given a set $\mathcal{P}$ of n colored points with k colors on the real line. It is easy to show that the number of minimal color-spanning intervals is linear and they can be found with a simple algorithm in linear time and space apart from sorting. The algorithm sweeps the points from left to right with two sweep lines which stop at the endpoints of an interval. It uses an array for keeping the number of points from each color and a variable for the number of different colors between the two sweep lines. Since the sweep lines never go back, the algorithm takes $O(n)$ time and space. We omit the details due to the simplicity and conclude the following theorem.

Theorem 1. *For a given set of n points with k colors on the real line, the smallest color-spanning interval can be computed in $O(n)$ time and space apart from sorting.*

In the following, we show a new view of minimal color-spanning intervals. Gupta et al. [7] use a transformation to perform generalized range reporting/counting for a given set of colored points on the real line. Indeed, they map the original given points on the real line to points in the transformed plane and perform the standard known 3-sided range reporting/counting in the transformed plane. We exploit the same transformation to give a new view of minimal color-spanning intervals. Let $\mathcal{P} = \{p_1, p_2, \cdots, p_n\}$ be a given set of colored points with k colors on the real line. For each color c, first we sort the points with color c (c-colored points, for short) in an increasing order. Then, for an arbitrary point p_i with color c, let $pred(p_i)$ be the previous point of p_i in the list of the ordered c-colored points. In addition, we set $pred(p_j) = -\infty$ if p_j is the left-most point with color c. Moreover, we insert k additional points from each color at $+\infty$—see Figure 1. A point p_i is mapped to the point $p_i^* = \big(p_i, pred(p_i)\big)$ in the transformed plane. Let $\mathcal{P}^* = \{p_1^*, p_2^*, \cdots, p_{n+k}^*\}$ be the transformed points—Figure 3 shows the transformed points of Figure 1. Furthermore, an interval $I = [l, r]$ on the real line is mapped to the point $I^* = (r, l)$. This transformation has several interesting properties.

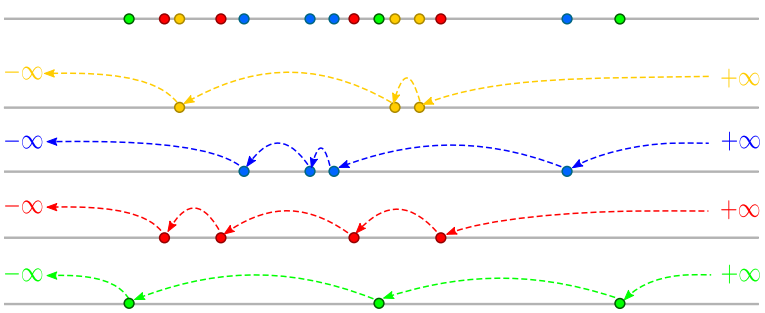

Fig. 1. The ordered lists of c-colored points for each color c

Consider an arbitrary point p in the plane. The vertical and the horizontal lines passing through p divide the plane into four quadrants. Let $\sigma(p)$ be the right-bottom quadrant; precisely $\sigma(p) = \{q \in \mathbb{R}^2 | p_x \leq q_x, p_y \geq q_y\}$. In the following we give some related definitions.

Definition 1. *For a set of points $\mathcal{Q} = \{q_1, \cdots, q_n\}$ and a point q in the plane, $\sigma(q)$ is empty with respect to $\mathcal{Q}$ if there is no point $q_i \in \mathcal{Q}$ in the interior of $\sigma(q)$.*

Definition 2. *For a set $\mathcal{Q}$ of points in the plane, a point $q \in \mathbb{R}^2$ is maximal with respect to $\mathcal{Q}$ if $\sigma(q)$ is empty and there is no other empty $\sigma(p)$ for some point $p \in \mathbb{R}^2$ such that $\sigma(q) \subset \sigma(p)$.*

Definition 3. *For a set of points $\mathcal{Q}$ in the plane, a point $q_i \in \mathcal{Q}$ is a skyline point if $\sigma(q_i)$ is empty.*

In order to see how the maximal points in the plane are related to the minimal color-spanning intervals, we present the following lemma.

Lemma 1. *For a given set $\mathcal{P}$ of colored points on the real line, $I = [l, r]$ is a minimal color-spanning interval if and only if the point $I^* = (r, l)$ is maximal with respect to the points $\mathcal{P}^*$ in the transformed plane.*

Proof. First, suppose $I = [l, r]$ is a minimal color-spanning interval. For the sake of contradiction, assume $I^* = (r, l)$ is not maximal. We distinguish two cases: (1) $\sigma(I^*)$ is not empty, (2) $\sigma(I^*)$ is empty but it is not maximal. In case (1) assume point $\big(p_i, pred(p_i)\big)$ is inside $\sigma(I^*)$ where both p_i and $pred(p_i)$ have color c—see Figure 2. This gives us $l > pred(p_i)$ and $r < p_i$ which means $[l, r]$ is a proper subinterval of $\big[pred(p_i), p_i\big]$. Therefore, $[l, r]$ does not contain any point of color c which is a contradiction. Now, consider case (2). In this case, there is a point $q^* = (r', l')$ such that $\sigma(I^*) \subset \sigma(q^*)$. This means there is a smaller color-spanning interval $q = [l', r']$ contained in $I = [l, r]$ which is a contradiction. Moreover, additional points at infinity makes any minimal color-spanning interval covered by maximal points in the transformed plane. The converse implication can be proved in a similar way. □

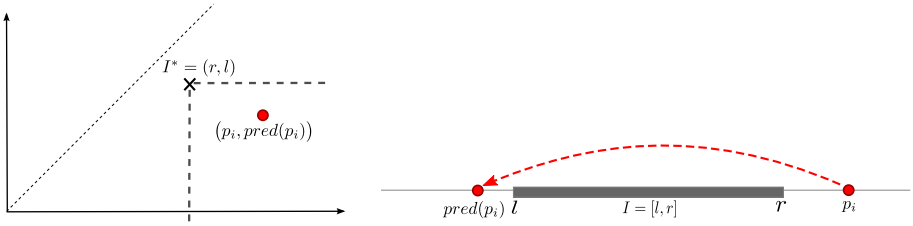

Fig. 2. A not color-spanning interval $I = [l, r]$ where $\sigma(I^*)$ is not empty

To summarize, a minimal color-spanning interval I is a maximal point in the transformed plane. In fact, vertical and horizontal rays extending to $+y$ and $-x$ directions starting at skyline points define the maximal points—see Figure 3 for more illustration. Consider I^* as a maximal point between two consecutive skyline points $p^*_{c_i} = (p_{c_i}, p_{c_{i-1}})$ and $p^*_{c'_j} = (p_{c'_j}, p_{c'_{j-1}})$ in the transformed plane. The minimal color-spanning interval defined by I^* is $I = [p_{c'_{j-1}}, p_{c_i}]$. Furthermore, the length of an interval is the vertical distance of its transformed point to the line $y = x$. The smallest color-spanning interval is *the minimum maximal point* in the transformed plane—see Figure 3.

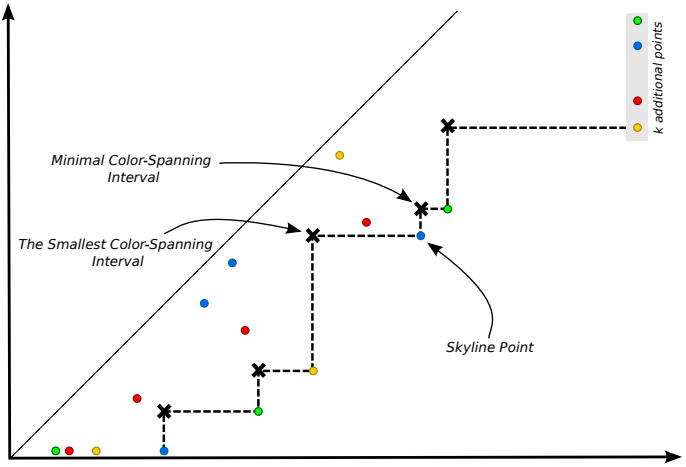

Fig. 3. The smallest color-spanning interval on the staircase of maximal points

2.2 Minimal Color-Spanning Intervals for Dynamic Points

In this section, we assume the colored points on the real line are dynamic, i.e., they can be inserted or deleted. Our goal here is to maintain the minimal color-spanning intervals and, specifically, the smallest color-spanning interval. This is the main ingredient of our algorithm to compute the smallest color-spanning square for a set of colored points in the plane.

In the previous section, we showed the minimal color-spanning intervals for a set of colored points on the real line can be considered as the maximal points in the transformed plane. Therefore, maintaining the minimal color-spanning intervals reduces to maintaining the skyline points in the transformed plane. Overmars and van Leeuwen [11] proposed a data structure for maintaining the skyline. In the following we briefly go over their data structure.

Let $\mathcal{Q}$ be a set of n points in the plane. Overmars and van Leeuwen [11] used a binary search tree T which stores all points in its leaves in the sorted order by their y-coordinates. Moreover, an internal node v is augmented with a concatenable tree, e.g. a 2-3 tree, which stores the skyline of points in the subtree rooted at v that are not contained in the skyline of points in the subtree rooted at v's parent. The skyline of all points in $\mathcal{Q}$ is augmented in the root of T. In addition, the cut point where the skyline split is stored in v.

If a point p is inserted, a procedure $Down(T, p)$ goes down the tree to locate p and construct the skyline of points for subtrees of children of each internal node u on the path. Suppose $Down(T, p)$ is running and the skyline of u's subtree has been computed in u's parent in the previous step. Then, from the cut point of u's children, it is possible to split the u's skyline at that point and to merge split parts with the lists stored in augmented trees of u's children. After inserting point p to a leaf of tree T, the other procedure $Up(T, p)$ goes up the tree to the root of T and reconstruct the augmented trees and cut points for children of each internal node placed on the path. If a point is deleted, the procedures work similarly. These procedures split or merge two concatenable queues in $O(\log n)$ time in each internal node on the path from root to inserted or deleted leaf and totally needs $O(\log^2 n)$ time. In addition, they showed their data structure and algorithms use linear space [11].

We use the described structure to maintain the smallest color-spanning interval which is the maximal point with the minimum distance to the line $y = x$ (the minimum maximal point) in the transformed plane. Furthermore, we define some pointers to maintain the minimum maximal point per insertion and deletion. For each internal node v in augmented trees, we set two pointers p^*_{m1} and p^*_{m2} to the leaves corresponding to the consecutive skyline points which together define the minimum maximal point in the subtree rooted at v. In fact, p^*_{m1} and p^*_{m2} indicate the endpoints of the smallest color-spanning interval for the points in the subtree rooted at v. Moreover, let p^*_l and p^*_r be pointers respectively to the leftmost and the rightmost leaves in the subtree rooted at v. We use the above pointers to update the data structure in $O(\log^2 n)$ time and maintaining the minimum maximal point. We omit the details due to the space constraint.

Theorem 2. *For a given set of n points with k colors on the real line, the smallest color-spanning interval can be maintained in $O(\log^2 n)$ update time per insertion and deletion and $O(n)$ space.*

Proof. When a new point p with color c is inserted between the points $p_{c_{i-1}}$ and p_{c_i} in the c-colored list, the point $p^*_{c_i} = (p_{c_i}, p_{c_{i-1}})$ should be deleted from the transformed plane and two new points $p^*_{c_i} = (p_{c_i}, p)$ and $p^* = (p, p_{c_{i-1}})$ are

inserted. Deletion of a point similarly needs a constant number of insertions and deletions in the transformed plane. □

We showed how we can maintain the smallest color-spanning interval for dynamic points on the real line. This is an important tool which helps us in the next section to compute the smallest color-spanning square for a given set of colored points in the plane.

3 The Smallest Color-Spanning Square

In this section, we focus on computing the smallest area/perimeter color-spanning axis-parallel square. Suppose we are given a set of n points with k colors in the plane. The smallest color-spanning square is an axis-parallel square which contains all colors and its area/perimeter is minimum.

Recall that this problem can be solved using the upper envelope of Voronoi surfaces [8] in $O(kn \log n)$ time. In fact, all the previously used methods for computing the smallest color-spanning objects such as rectangle, circle, etc. generally test all the minimal objects. We show that there are $\Omega(kn)$ minimal color-spanning squares in the worst case. This indicates that any algorithm testing all minimal color-spanning squares runs in $\Omega(kn)$ time. Next, we present an algorithm running in $O(n \log^2 n)$ time that computes the smallest color-spanning square without testing all minimal color-spanning squares.

We first explain how a minimal color-spanning square can be represented. As illustrated in Figure 4, a minimal color-spanning square is defined with two, three or four points with different colors on its edges under the assumption that point coordinates are different (recall that this assumption is just for the ease of the presentation). In all cases the minimal color-spanning square is bounded by two points on opposite sides. So, it suffices to consider the minimal color-spanning squares of case 1 instead of all. Moreover, we are interested in counting the minimal color-spanning squares with different contained points. We first give a lower bound for the number of minimal color-spanning squares.

Lemma 2. *There is a configuration of n points with k colors in the plane in which there are $\Omega(kn)$ minimal color-spanning squares.*

Proof. Let's place a set of $2k$ points with k colors in the configuration shown in Figure 5. In this pattern, we have to fix the left and the right edges in a way that they span the half of the colors (1 to $\frac{k}{2}$). So, there are $\frac{k}{2} + 1$ choices for fixing the right and the left edges. Similarly, for a pair of fixed left and right edges, it is possible to obtain $\frac{k}{2} + 1$ pairs to be the top and the bottom sides. Therefore, there are $\Omega(k^2)$ minimal color-spanning squares in the pattern of Figure 5 using $2k$ points. By repeating this pattern $\frac{n}{2k}$ times, we achieve $\Omega(kn)$ minimal color-spanning squares for a set of n points with k colors. □

Now, we start describing the steps of our main algorithm. Suppose the points are sorted in a descending order according to their y coordinates. The algorithm

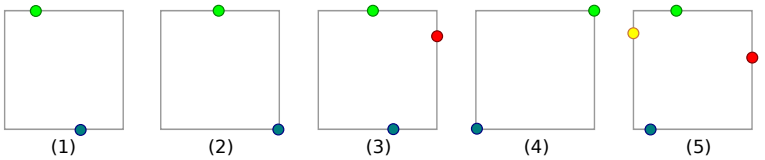

Fig. 4. The cases of a minimal color-spanning square

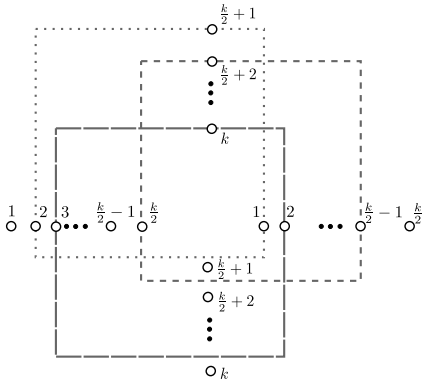

Fig. 5. $\Omega(k^2)$ minimal color-spanning squares with $2k$ points

sweeps the points with two lines L_b and L_t from top to bottom and variable d denotes the vertical distance of the sweep lines—see Figure 6 for more illustration. In fact, the lines L_b and L_t, respectively, define the bottom and the top edges of the desired square. At the beginning, L_b and L_t pass through the topmost point. We move down the sweep lines as follows:

- Move down the line L_t to the next point if there exist at least one color-spanning square in the strip bounded by L_b and L_t—see Figure 6(a).
- Move down the line L_b to the next point if there is no color-spanning square in the strip bounded by L_b and L_t—see Figure 6(b).

Now, it remains to show how we can find out if there is a minimal-color-spanning square in the strip of lines L_b and L_t. Suppose the points in the strip are projected onto the real line; let $\mathcal{P}$ be the union of this set of projected points and the additional points from each color at infinity as defined in the previous section. The following simple but important observation describes the necessary and sufficient conditions that a minimal color-spanning square exists inside the strip bounded by L_b and L_t.

Observation 1. *There is a minimal color-spanning square in the strip of lines L_b and L_t if and only if for the points in $\mathcal{P}$ the length of the smallest color-spanning interval is at most d.*

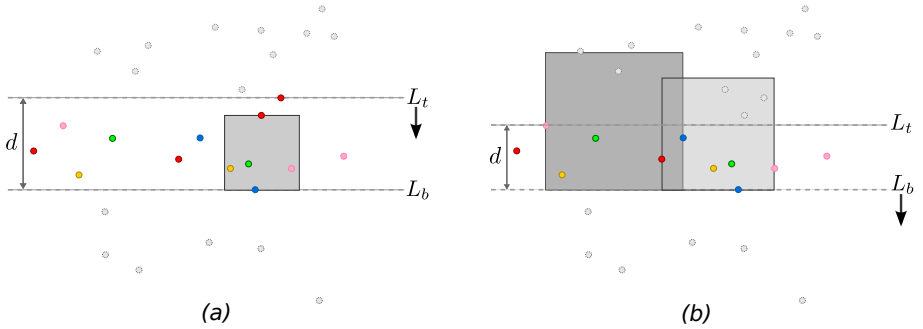

Fig. 6. Sweeping the points with two lines L_b and L_t

According to this observation, the algorithm only considers the projected points $\mathcal{P}$. Indeed, when L_b reaches a new point or a point goes out of the strip when L_t moves down, we insert or respectively delete a point from the dynamic structure of maintaining the smallest color-spanning interval that we described in the previous section. In addition, while L_t is moving down, at the time the smallest color-spanning interval becomes greater than d, the solution should be updated. We give the following theorem.

Theorem 3. *For a given set of n points with k colors in the plane, the smallest color-spanning axis-parallel square can be computed in $O(n \log^2 n)$ time and $O(n)$ space.*

Proof. Let s be the smallest color-spanning square which its top and bottom sides are defined with points p_t and p_b respectively. Suppose t_1 is the time L_b reaches p_b and t_2 is the time L_t stops at point p_t. There are two possibilities here. The case $t_1 < t_2$ means L_b is at point p_b while L_t is above the point p_t. Since there is at least one color-spanning square between the sweep lines, L_t moves down until it reaches point p_t. When L_t leaves out p_t, the length of the smallest color-spanning interval becomes greater than d and s has been visited. Otherwise s is not the solution which is a contradiction. Similarly, if $t_1 > t_2$, L_t stops at point p_t while L_b moves down until it reaches to the point p_b. Since s is the solution, L_b must stop at point p_b. Therefore, we visit the smallest color-spanning square in both cases. To analyse the running time of our algorithm, we noted that each point p enters the strip and is eliminated from it once when L_b and respectively L_t reach it. Therefore, in total we have $O(n)$ insertions and deletions and by Theorem 2 each operation spends $O(\log^2 n)$ time to maintain the smallest color-spanning interval for the points in $\mathcal{P}$. So, the algorithm runs in $O(n \log^2 n)$ time and $O(n)$ space. □

4 Conclusion

For a set of colored points in the plane, the problem of computing the smallest color-spanning axis-parallel square was studied in this paper. The previous

methods for the smallest color-spanning objects test all minimal candidates. We showed although there could be $\Omega(kn)$ minimal color-spanning squares, there exists an algorithm that compute the smallest color-spanning square without testing all minimal ones. As the main ingredient of our algorithm, we first presented a dynamic data structure to maintain the smallest color-spanning interval for dynamic points on the real line. Then we used this dynamic structure to run our algorithm in $O(n \log^2 n)$ time and $O(n)$ space for computing the smallest color-spanning axis-parallel square. Our algorithm is independent from number of colors and faster than the algorithm mentioned by Abellanas et al.[1] (running in $O(kn \log n)$ time) which is based on computing the upper envelope of Voronoi surfaces [8].

References

1. Abellanas, M., Hurtado, F., Icking, C., Klein, R., Langetepe, E., Ma, L., Palop, B., Sacristán, V.: Smallest Color-Spanning Objects. In: Meyer auf der Heide, F. (ed.) ESA 2001. LNCS, vol. 2161, pp. 278–289. Springer, Heidelberg (2001)
2. Abellanas, M., Hurtado, F., Icking, C., Klein, R., Langetepe, E., Ma, L., Palop, B., Sacristán, V.: The Farthest Color Voronoi Diagram and Related Problems. Tech. Report. University of Bonn (2006)
3. Consuegra, M.E., Narasimhan, G., Tanigawa, S.: Geometric Avatar Problems. Tech. Report TR-2013-02-25. Florida International University (2013)
4. Das, S., Goswami, P.P., Nandy, S.C.: Smallest Color-Spanning Object Revisited. Int. J. Comput. Geometry Appl. 19, 457–478 (2009)
5. Fan, C., Ju, W., Luo, J., Zhu, B.: On Some Geometric Problems of Color-Spanning Sets. In: Atallah, M., Li, X.-Y., Zhu, B. (eds.) FAW-AAIM 2011. LNCS, vol. 6681, pp. 113–124. Springer, Heidelberg (2011)
6. Fan, C., Luo, J., Zhong, F., Zhu, B.: Expected Computations on Color Spanning Sets. In: Fellows, M., Tan, X., Zhu, B. (eds.) FAW-AAIM 2013. LNCS, vol. 7924, pp. 130–141. Springer, Heidelberg (2013)
7. Gupta, P., Janardan, R., Smid, M.H.M.: Further Results on Generalized Intersection Searching Problems: Counting, Reporting, and Dynamization. J. Algorithms 19(2), 282–317 (1995)
8. Huttenlocher, D.P., Kedem, K., Sharir, M.: The Upper Envelope of Voronoi Surfaces and Its Applications. Discrete Computational Geometry 9, 267–291 (1993)
9. Löffler, M.: Data Imprecision in Computational Geometry. Ph.D. Thesis. Utrecht University (2009)
10. Matoušek, J.: On Enclosing k Points by a Circle. Information Processing Letters 53(4), 217–221 (1995)
11. Overmars, M.H., van Leeuwen, J.: Maintenance of Configurations in the Plane. J. Comput. Syst. Sci. 23, 166–204 (1981)
12. Smid, M.H.M.: Finding k Points With a Smallest Enclosing Square. MPI-I-92-152, Max-Planck-Institut Inform., Saarbrücken, Germany (1992)