ELSEVIER

Information Processing Letters 69 (1999) 189–195

Information Processing Letters

On detecting spatial regularity in noisy images

Gabriel Robins *, Brian L. Robinson, Bhupinder S. Sethi

Department of Computer Science, University of Virginia, Charlottesville, VA 22903, USA

Received 4 June 1998; received in revised form 9 November 1998
Communicated by D. Gries

Abstract

Detecting spatial regularity in images arises in computer vision, scene analysis, military applications, and other areas. In this paper we present an $O(n^{5/2})$ algorithm that reports all maximal equally-spaced collinear subsets. The algorithm is robust in that it can tolerate noise or imprecision that may be inherent in the measuring process, where the error threshold is a user-specified parameter. Our method also generalizes to higher dimensions. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Algorithms; Combinatorial problems; Computational geometry; Pattern recognition

1. Introduction

Spatial regularity detection is an important problem in a number of domains such as computer vision, scene analysis, and landmine detection from infrared terrain images [5]. This paper addresses the problem of recognizing equally-spaced collinear subsets of a given pointset, where there may be imprecision in the input data. Kahng and Robins [5] gave an optimal $O(n^2)$-time algorithm for the *exact* version of this problem (i.e., where no noise is allowed in the input data); their algorithm was later parallelized in [2]. However, inaccuracies inherent in the imaging technology may corrupt the data and eliminate exact collinearity and equally-spaced relationships. Thus, methods which do not compensate for these phenomena (e.g., those of [2] and [5]) are inadequate in practice.

To address this issue, we develop a robust algorithm for spatial regularity detection that can tolerate inaccuracies in the data relative to a user-specified error threshold. For an input pointset, our algorithm will report all equally-spaced collinear subsets, where points are allowed to drift from their ideal locations by up to a given amount. Our algorithm runs within $O(n^{5/2})$ time in the worst case, and we prove a lower bound of $\Omega(n^2 \log n)$ on the time complexity for this problem.

2. Problem formulation

Given a finite set of distinct points $P \subset E^2$, a subset $\overline{P} \subseteq P$ is *collinear* if $|\overline{P}| \geqslant 2$ and all points of $\overline{P}$ lie on the same line. A collinear subset $\overline{P} \subseteq P$ is *equally-spaced* if $|\overline{P}| \geqslant 3$ and the points of $\overline{P}$ are equally-spaced along their containing line. [1] We use the term *regular* to refer to a sequence that is both collinear and equally-spaced. A sequence of points is

* Corresponding author. Email: robins@cs.virginia.edu, http://www.cs.virginia.edu/~robins/. Professor Gabriel Robins is supported by a Packard Foundation Fellowship and NSF Young Investigator Award MIP-9457412.

[1] That is, a sequence $\overline{P} = (\overline{p}_1, \overline{p}_2, \ldots, \overline{p}_n)$ is equally-spaced if $\overline{p}_i - \overline{p}_{i-1} = \overline{p}_{i+1} - \overline{p}_i$ for $2 \leqslant i \leqslant n-1$.

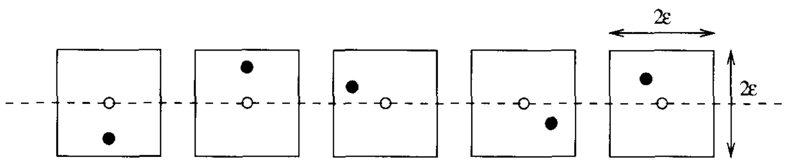

Fig. 1. An ε-regular sequence (solid dots), whose points are within ε of the corresponding (ideal) points of a regular sequence (hollow dots).

ε-regular if each of its points can be displaced by at most ε along each axis to yield a regular sequence (Fig. 1); i.e., given a fixed $\varepsilon \geqslant 0$, a sequence of points $P = (p_1, p_2, \ldots, p_n) \subset E^2$ is ε-regular if there exists a regular sequence $\overline{P} = (\overline{p}_1, \overline{p}_2, \ldots, \overline{p}_n)$ such that $|x_i - \overline{x}_i| \leqslant \varepsilon$, $|y_i - \overline{y}_i| \leqslant \varepsilon$ for all $1 \leqslant i \leqslant n$, where $p_i = (x_i, y_i)$ and $\overline{p}_i = (\overline{x}_i, \overline{y}_i)$. We refer to points in $\overline{P}$ as the *ideal* points, and to points in P as the *actual* points. Note that a regular sequence is simply an ε-regular sequence with $\varepsilon = 0$. Finally, a *maximal* ε-regular sequence is one that is not properly contained as a contiguous subsequence in any other ε-regular sequence.[2]

Our problem can now be formulated as follows.

Maximal ε-regular Sequences Problem. Given $\varepsilon \geqslant 0$ and a finite pointset $P \subset E^2$, find all maximal ε-regular sequences contained in P.

Unfortunately, this formulation is intractable since it allows an exponentially large output size. We therefore propose a modification in order to make possible an efficient algorithm:

Stipulation. *Each interpoint distance in the input data must be greater than* 8ε.

As we will see, this stipulation precludes cases with exponentially-sized output. If the input does not satisfy this condition,[3] a slight variation on our proposed algorithm will still produce a correct solution (although a polynomial runtime is no longer guaranteed). Existing methods for detecting collinearity, such as ones based on Hough transforms [1,3,7], are not sensitive to equally-spaced constraints and are thus not applicable to our problem. Some algorithms for dealing with imprecision in computational geometry were introduced in [4], but they do not address our problem either. This work builds upon the approach introduced in Robins and Robinson [8].

3. Tools and techniques

Our overall strategy is based on starting with a pair of points and repeatedly extending it into an ε-regular sequence, until it is no longer possible to do so. Each extension involves finding a new point to convert the last pair of the current ε-regular sequence into an ε-regular triple, and then verifying if the current sequence with the new point added is still ε-regular. The rest of this section develops techniques that will perform these operations efficiently.

3.1. Finding ε-regular triples

Given a pair of points (p_1, p_2), a naive algorithm to extend it into an ε-regular triple would be to consider each of the remaining $n - 2$ points in turn and check if any of them formed an ε-regular triple with the given pair. Each check would take constant time, resulting in a linear-time algorithm. We now develop a more efficient way of finding all possible triples, using the following observation.

Lemma 1. *Given the first two points of an ε-regular triple, (x_1, y_1) and (x_2, y_2), the third point (x_3, y_3) of the triple must lie in the range $((2x_2 - x_1) \pm 4\varepsilon, (2y_2 - y_1) \pm 4\varepsilon)$.*

[2] Thus, according to this definition, if (a, b, c, d, e) is an ε-regular sequence, then so is (a, c, e). Since detecting all maximal ε-regular sequences does not require asymptotically more time than detecting only those not properly contained in other ε-regular sequences, this definition suffices and also results in a streamlined exposition and analysis (note that any "redundant" subsequences may later be removed in a post-processing step). This approach is also consistent with the application area of landmine detection, where false positives are preferable to false negatives.

[3] This assumption is not unrealistic for applications such as landmine detection, since the sizes of mines and their potential drift are considerably smaller than typical inter-mine separation.

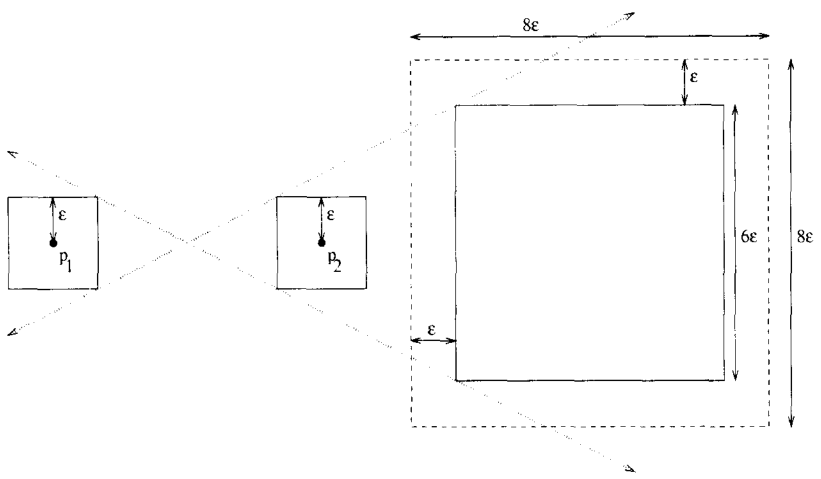

Fig. 2. The dashed square on the right is the region in which p_3 must lie if the actual points (p_1, p_2, p_3) are an ε-regular sequence. The large solid square on the right denotes the region in which the ideal point corresponding to p_3 must lie.

Proof. Given the first two points of the triple, $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$, let the corresponding ideal points be $(\overline{x}_1, \overline{y}_1)$ and $(\overline{x}_2, \overline{y}_2)$ (Fig. 2). Then we have:

$$x_1 - \varepsilon \leqslant \overline{x}_1 \leqslant x_1 + \varepsilon, \quad x_2 - \varepsilon \leqslant \overline{x}_2 \leqslant x_2 + \varepsilon,$$
$$y_1 - \varepsilon \leqslant \overline{y}_1 \leqslant y_1 + \varepsilon, \quad y_2 - \varepsilon \leqslant \overline{y}_2 \leqslant y_2 + \varepsilon.$$

Since the ideal points are equally-spaced and collinear, we have, $\overline{x}_3 - \overline{x}_2 = \overline{x}_2 - \overline{x}_1$ and $\overline{y}_3 - \overline{y}_2 = \overline{y}_2 - \overline{y}_1$. The third ideal point is given by $(\overline{x}_3, \overline{y}_3) = (2\overline{x}_2 - \overline{x}_1, 2\overline{y}_2 - \overline{y}_1)$. Substituting for $\overline{x}_1$, $\overline{x}_2$, $\overline{y}_1$ and $\overline{y}_2$, we obtain:

$$(2x_2 - x_1) - 3\varepsilon \leqslant \overline{x}_3 \leqslant (2x_2 - x_1) + 3\varepsilon,$$
$$(2y_2 - y_1) - 3\varepsilon \leqslant \overline{y}_3 \leqslant (2y_2 - y_1) + 3\varepsilon.$$

The third actual point, (x_3, y_3) can be displaced from the third ideal point by as much as ε in any direction. That is, $\overline{x}_3 - \varepsilon \leqslant x_3 \leqslant \overline{x}_3 + \varepsilon$ and $\overline{y}_3 - \varepsilon \leqslant y_3 \leqslant \overline{y}_3 + \varepsilon$. Hence, we have:

$$(2x_2 - x_1) - 4\varepsilon \leqslant x_3 \leqslant (2x_2 - x_1) + 4\varepsilon,$$
$$(2y_2 - y_1) - 4\varepsilon \leqslant y_3 \leqslant (2y_2 - y_1) + 4\varepsilon. \qquad \square$$

Our 8ε stipulation ensures that for every pair, the range that needs to be searched contains at most a single point, which can be found out in $O(\log n)$ time using range searching techniques.

3.2. Linear programming

Given an ε-regular sequence $(p_1, p_2, \ldots, p_t)$ and an ε-regular triple containing the last pair, (p_{t-1}, p_t, p_{t+1}), the sequence $(p_1, p_2, \ldots, p_t, p_{t+1})$ is not necessarily ε-regular, since additional constraints need to be satisfied. Fortunately, these constraints can be expressed as a low-dimensional linear program (LP).

For simplicity, let us first look at the one-dimensional case. If a given sequence $P = (p_0, p_1, \ldots, p_{n-1})$ is to be ε-regular, there must exist a corresponding regular sequence of ideal points

$$\overline{P} = (\overline{p}_0, \overline{p}_1, \ldots, \overline{p}_{n-1})$$
$$= \big(\overline{p}_0, \overline{p}_0 + d, \overline{p}_0 + 2d, \ldots, \overline{p}_0 + (n-1)d\big),$$

where d is the distance between successive points in $\overline{P}$. Since $\overline{p}_i - \varepsilon \leqslant p_i \leqslant \overline{p}_i + \varepsilon$, we obtain:

$$\overline{p}_0 + id - \varepsilon \leqslant p_i, \quad \overline{p}_0 + id + \varepsilon \geqslant p_i.$$

Thus to check for ε-regularity of a sequence $P = (p_0, p_1, p_2, \ldots, p_{n-1})$, we create an LP consisting of the inequalities above for all $0 \leqslant i \leqslant n-1$, and solve it to determine if feasible values for $\overline{p}_0$ and d exist. If so, the sequence is ε-regular.

Moving into two-dimensions, we observe that a sequence is regular iff its orthogonal projections onto the x and y axes are also regular. Recall that in an ε-regular sequence, the projection of an actual point

p_i onto any axis must be within distance ε of the projection of the corresponding ideal point $\overline{p}_i$. We thus formulate the following constraints in our LP, where $d_x, d_y, \overline{p}_{0_x}$ and $\overline{p}_{0_y}$ are variables:

$$\overline{p}_{0_x} + id_x - \varepsilon \leqslant p_{i_x}, \quad \overline{p}_{0_x} + id_x + \varepsilon \geqslant p_{i_x},$$
$$\overline{p}_{0_y} + id_y - \varepsilon \leqslant p_{i_y}, \quad \overline{p}_{0_y} + id_y + \varepsilon \geqslant p_{i_y}.$$

Since the constraints along one axis are completely independent of the constraints along the other axis, this system of constraints can be expressed as a pair of two-dimensional linear programs. Linear programming in fixed dimension (i.e., the number of variables) can be accomplished in time linear in the number of constraints [6]. Since the dimension of each of our LPs is fixed at two, the time required for verification of ε-regularity is proportional to the length of the sequence examined.

4. An efficient algorithm

Our algorithm repeatedly extends point pairs into maximal ε-regular sequences by adding points which form an ε-regular triple with the last pair of the current sequence, and then verifies the ε-regularity of the complete sequence. We utilize Lemma 1 in conjunction with geometric range searching to find new points to be added, and we rely on linear programming to verify the ε-regularity of the sequence.

4.1. Preprocessing

The preprocessing phase involves verifying that the 8ε stipulation holds (which can easily be done in $O(n^2)$ time), and then suitably organizing the data. In order to apply Lemma 1, we need to be able to query efficiently for the existence of a point in the required 8ε region. To accomplish this, we conceptually partition the plane into a grid of granularity 8ε. For each point we compute the coordinates of the grid cell which it occupies.[4] These coordinates are stored along with the points and used to index the points in subsequent searching. Finally, we sort all the points by their cell coordinates, which enables logarithmic-time binary search queries. The preprocessing cost is dominated by the $O(n^2)$ verification step.

4.2. Finding all maximal ε-regular sequences

Our approach is to extend a pair of points into a maximal ε-regular sequence. At each step of the algorithm we seek a single additional point which may be included into the current sequence while preserving the ε-regularity of the sequence. We refer to this process as *extending* the sequence, since it finds a point which would form an ε-regular triple with the last pair of the current sequence. The preprocessing ensures that this operation can be performed in $O(\log n)$ time. If such a point exists, verifying the ε-regularity of the complete sequence entails solving an LP which takes time proportional to the length of the sequence. A single extension step in the worst case thus takes time $O(\log n + k)$, where k is the length of the current sequence. An extension fails if a suitable point cannot be found, and a maximal ε-regular sequence is detected when no further extension is possible in either direction.

All maximal sequences could be enumerated by starting from each of the $\binom{n}{2}$ pairs and repeating the process above; however, to avoid redundant computations, we exploit two observations. First, we note that any contiguous subsequence of an ε-regular sequence is itself ε-regular; thus, we do not need to reconstruct any subsequence which is a contiguous portion of a previously-found maximal ε-regular sequence. Specifically, after extending a so-called *seed* pair into an *initial* maximal sequence, we remove the leftmost[5] point of the sequence, leaving an ε-regular subsequence which we then extend to the right.

Each time a maximal sequence is found, we remove the leftmost point and extend to the right as much as possible. When it is no longer possible to continue in this manner (i.e., when the remaining subsequence contains only a single point), we repeat this entire process starting with the current initial sequence, except that this time we remove the rightmost point and

[4] If $\varepsilon = 0$, the cell coordinates are equal to the coordinates of the points themselves and no grid is constructed.

[5] One end of the sequence is chosen arbitrarily to be *leftmost*, with the the other end being *rightmost*. Thus, the terms "left" and "right" are not related to X or Y coordinates, but rather are used to refer to the two opposite directions along a sequence, and as such the terms "left" and "right" are symmetric and interchangeable.

```
Maximal ε-regular Sequences Algorithm
Input: A pointset $P \subset E^2$
Output: LIST of all maximal ε-regular sequences
1: LIST = ∅
2: For all unmarked pairs $S = \{p_i, p_j\}$
3:    Call Procedure Initialize S
4:    Let INIT = S, AUXLIST = {S}
      /* INIT is now the initial sequence */
5:    Call Procedure March(right)
6:    Let S = INIT
7:    Call Procedure March(left)
8:    Mark all pairs in all elements of AUXLIST
9:    Let LIST = LIST ∪ AUXLIST

Procedure Initialize S
1: Mark S
2: While S can be extended to the left Do
      Extend S to the left
3: While S can be extended to the right Do
      Extend S to the right

Procedure March(direction)
/* ⟨opposite direction⟩ is 'right' if ⟨direction⟩ is 'left'
   and vice-versa */
1: Remove the ⟨opposite direction⟩-most point from S
2: While |S| > 1
3:    If S is extendable to the ⟨direction⟩ Then
      /* Extendable means there exists a point
      such that extended sequence is ε-regular and
      has no marked pair */
4:       Repeat (Extend S to the ⟨direction⟩)
            Until S cannot be extended to the ⟨direction⟩
5:       Let AUXLIST = AUXLIST ∪ {S}
6:    Remove the ⟨opposite direction⟩-most point
         from S
```

Fig. 3. An algorithm for enumerating all maximal ε-regular sequences.

extend to the left. We refer to this process of removing and extending as *marching*. Thus, the process of marching begins by operating on an initial sequence, and repeatedly (1) removing an endpoint of the current maximal ε-regular sequence, and (2) extending the resulting sequence in the opposite direction until another maximal ε-regular sequence is obtained.

Our second observation is that contiguous pairs in an already-found maximal ε-regular sequences need not be considered as seed pairs in future sequence searches. We thus explicitly *mark* such pairs and ignore them from further consideration (all pairs are *unmarked* when the algorithm starts). The complete algorithm therefore involves repeated iterations of the marching process described above, starting with an unmarked pair as the seed each time, until there are no unmarked pairs left. A formal description of the algorithm is shown in Fig. 3.

5. Lower bound on output size

Consider a pointset distributed along an arc in such a way that although every contiguous subsequence of $\frac{1}{2}n$ points is ε-regular, there is no ε-regular sequence of length greater than $\frac{1}{2}n$ (Fig. 4). Such a pointset has $\frac{1}{2}n$ ε-regular sequences of length $\frac{1}{2}n$, $\frac{1}{2}n$ ε-regular sequences of length $\frac{1}{4}n$, $\frac{1}{2}n$ ε-regular sequences each of length $\frac{1}{6}n$, etc. In general, $\frac{1}{2}n$ ε-regular sequences of length $\frac{1}{2}n/k$ are determined by starting from any of the first $\frac{1}{2}n$ points and selecting every kth point on the arc. Thus, the sum of the lengths of the maximal ε-regular sequences in the example of Fig. 4 is given by $\frac{1}{2}n(\frac{1}{2}n+\frac{1}{4}n+\frac{1}{6}n+\cdots+3)=\Theta(n^2\log n)$, and the output size for this pointset is $\Omega(n^2\log n)$.

6. Time complexity

Each extension in the marching process results in the insertion of a new point (if the extension succeeds), or the removal of a point (if the extension fails). In the worst case, each extension requires a search time of $O(\log n)$, and a verification time of $O(k)$ where k is the length of the ε-regular sequence under consideration; we assign this cost of extension to the point being added or deleted.

Define a p-initiated sequence as an ε-regular sequence beginning with point p. Each time a point p is encountered and added during the extension process, the resulting sequence is a p-initiated sequence. Similarly, when a point p is removed while marching, the sequence before deletion is also a p-initiated sequence. Moreover, the sequence in question is *maximal* among all p-initiated sequences. Each time the point p incurs a cost, the associated maximal p-initiated sequence is unique. The search time is bounded by $O(\log n)$, while the verification time is proportional to the length of the associated p-initiated

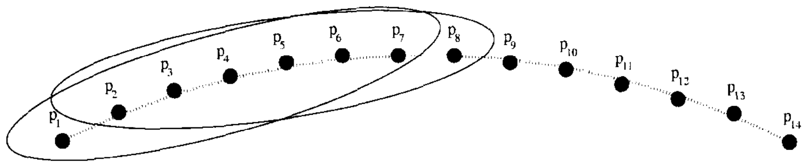

Fig. 4. In this example, the maximal ε-regular sequences are $(p_1, p_2, \ldots, p_7)$, $(p_2, p_3, \ldots, p_8), \ldots, (p_8, p_9, \ldots, p_{14})$.

sequence. If the maximum number of maximal p-initiated sequences is m, the total cost incurred by all searches is bounded by $O(m \log n)$, while the time for all verifications is bounded by the sum of the lengths of all the maximal p-initiated sequences.

Because of the 8ε stipulation, each of the other $n-1$ points can occur in the second position in at most one maximal p-initiated sequence. Thus, the number of p-initiated sequences is always less than n. Hence $m < n$, and the total search time for all maximal p-initiated sequences is bounded by $O(n \log n)$. We now prove an upper bound on the cost to verify each new point.

Lemma 2. *There exists a total order on the points such that all p-initiated sequences are subsequences in this order.*

Proof. We order the points by their distance to p, with ties broken arbitrarily. Since every p-initiated sequence starts with p and every successive point is at least as far from p as the previous one, a p-initiated sequence is a subsequence of this total order.[6] □

Each maximal p-initiated sequence can then be expressed as a series of numbers $(a_1, a_2, \ldots, a_q)$, where a_i is the rank of the ith point of the sequence in the total order with respect to p and q is the sequence length. Since the sequence is a subsequence of the total order, $a_i < a_{i+1}$ for all $1 \leqslant i < q$, and since the sequence is p-initiated, a_1 is always 1.

We represent a sequence $(a_1, a_2, \ldots, a_q)$ as a series of "hops" $(a_2 - a_1, a_3 - a_2, a_4 - a_3, \ldots, a_q - a_{q-1})$. Since a_1 is always 1, the points in the sequence can be regenerated from this sequence of hops. For a maximal p-initiated sequences stored as a series of hops $(b_1, b_2, \ldots, b_r)$, the length of the sequence is $r+1$. Since the last point has an rank less than $n+1$ in the total order, the sum of the hops $b_1 + b_2 + \cdots + b_r < n$.

Consider a multiset S containing all the hops of all maximal p-initiated sequences. Since there are no more than n such sequences and the sum of hops in each is less than n, the sum of all the hops in S cannot exceed n^2. Since the number of hops in each sequence is one less than the sequence length, the size of S is at most n less than the sum of the lengths of all maximal p-initiated sequences. Since a contiguous pair can occur in only one of the maximal p-initiated sequence, there are no more than $n-1$ possible hops of size 1 in the total order, no more than $n-2$ possible hops of size 2, etc. Under these constraints, we seek to maximize the size of S, in order to determine an upper bound on the sum of lengths of all maximal p-initiated sequences.

Let S contain h_i hops of size i, for $1 \leqslant i \leqslant n-1$. The constraints above can then be expressed as

(1) $h_i \leqslant n - i$ for all $1 \leqslant i < n$,

(2) $\sum_{i=1}^{n-1} i h_i \leqslant n^2$,

and we seek the maximum possible value of $h_1 + h_2 + \cdots + h_{n-1}$. In the interest of simplicity we relax the first constraint to

(1′) $h_i \leqslant n$, for $1 \leqslant i < n$,

which is permissible since it can only increase the value of the upper bound.

Lemma 3. *There is a solution to the maximization problem above having the form $h_i = n$ for $0 \leqslant i \leqslant l$, and $h_i = 0$ for $l < i < n$.*

Proof. Consider an optimal solution $(h_1, h_2, \ldots, h_n)$ not having the above form. If there exist i, j with $i < j$, such that $h_i < n$ and $h_j > 0$, then, h_i can be

[6] Note that this lemma does not actually require explicit sorting inside the algorithm; rather, we use the implicit ordering established by the lemma as an aid in deriving an upper bound on the time complexity of the algorithm.

incremented by one and h_j can be decremented by one, while satisfying both the constraints of the problem and not decreasing the function to be maximized. This can be done repeatedly until there exists t such that $h_i = n$ for $i \leqslant t$. For $n = \binom{l}{2}$, $t = l$ and $h_i = 0$ for $i > t$, there exists a new optimal solution having the required form. The second constraint becomes an equality and no more increment–decrement operations are possible. □

Without loss of generality, let $n = \binom{l}{2}$ for some l. Thus $l = \mathrm{O}(\sqrt{n})$ and

$$\sum_{i=1}^{n-1} h_i = \sum_{i=1}^{l-1} h_i = \sum_{i=1}^{l-1} n = \mathrm{O}(n\sqrt{n}).$$

Thus, the sum of all maximal p-initiated sequences cannot exceed $\mathrm{O}(n\sqrt{n})$. Since the total cost incurred by a single point is bounded by $\mathrm{O}(n \log n + n\sqrt{n}) = \mathrm{O}(n\sqrt{n})$, the overall cost for all points is bounded by $\mathrm{O}(n^{5/2})$, which is the time complexity of the algorithm.

Although the above analysis holds for only specific values of $n = \binom{l}{2}$, we can always add dummy points to the input pointset in order to ensure that $n = \binom{r}{2}$ for some r; in such a case a maximal sequence is listed in the output iff it does not contain any dummy point. Since $\binom{r+1}{2} - \binom{r}{2} = r$, we never have to add more than n dummy points, and the overall time complexity remains unchanged.

Finally, note that even if the input does not satisfy the 8ε stipulation, a slight variation on our main algorithm will still produce a correct solution, although a polynomial time bound is no longer guaranteed. In such a case, a pair of points may participate in more than two distinct triples; more than one point may therefore be a suitable candidate for extension of an ε-regular sequence under consideration. All the possibilities would then have to be explored, which may result in a non-polynomial runtime. However, for typical inputs, such pathological behavior is very unlikely.

References

[1] D. Ben-Tzvi, M.B. Sandler, A combinatorial hough transform, Pattern Recognition Lett. 11 (1990) 167–174.

[2] L. Boxer, R. Miller, Parallel algorithms for all maximal equally-spaced collinear sets and all maximal regular coplanar lattices, Pattern Recognition Lett. 14 (1993) 14–20.

[3] R. Duda, P. Hart, Use of the Hough transform to detect lines and curves in pictures, Comm. ACM 15 (1972) 11–15.

[4] L. Guibas, D. Salesin, J. Stolfi, Constructing strongly convex approximate hulls with inaccurate primitives, Algorithmica 9 (1993) 534–560.

[5] A.B. Kahng, G. Robins, Optimal algorithms for extracting spatial regularity in images, Pattern Recognition Lett. 12 (1991) 757–764.

[6] N. Megiddo, Linear programming in linear time when the dimension is fixed, J. Assoc. Comput. Mach. 31 (1984) 114–127.

[7] T. Risse, Hough transform for line recognition: Complexity of evidence accumulation and cluster detection, Computer Vision 46 (1989) 327–345.

[8] G. Robins, B.L. Robinson, Landmine detection from inexact data, in: Proc. International Symposium on Aerospace/Defense Sensing and Dual-Use Photonics, Vol. 2496, Orlando, FL, April 1995, pp. 568–574.