Point Pattern Matching

Gerald Weber

December 21, 1995

Abstract

We consider the problem of finding (possibly transformed) copies of a point pattern B in a larger point set A. We discuss several special versions of this problem.

Contents

Introduction and Motivation

Pattern matching in pictures is an important problem in automatic image processing. The problem is to find fast algorithms which decide whether a certain pattern appears in a given image. We study here a certain class of pattern matching probems, called point pattern matching problems: We consider the pattern as well as the picture to be finite point sets in a certain fixed dimension. Our aim is to find algorithms with small asymptotic running time in the number of picture points n and pattern points m in the algebraic model of computation. We can state the problem in the following generic form:

> Given a set A of n points and a set B of m points, $2 \leq m \leq n$, find all geometric transformations in a given class $\mathcal{C}$ that map B into A. A is called the picture, B the pattern.

In this report we discuss the problem from a theoretical point of view. We can consider instances of the problem for points in an arbitrary dimension d and for different possible classes of allowed geometric transformations, typically subclasses of the class of similarity mappings (translations, stretch-rotations with fixed center, isometries, arbitrary similarity mappings). We will abbreviate "Point Pattern Matching" in the following with "PPM".

We give algorithms for some instances of this problem, which are connected with applications or simply of theoretical interest. It will turn out that even for one-dimensonal PPM under isometries there is a gap between the upper and lower bounds. In fact, the only known lower bound for PPM in arbitrary dimensions is an $\Omega(n \log n)$ lower bound, which is presented in [19] and can be found by a Ben-Or argument.

In one dimension we focus on isometries as allowed transformations. We first establish an $\mathcal{O}(nm + n \log n)$ general upper bound in Theorem 1. For patterns with points in a certain kind of general position we can present in Theorem 2 an $\mathcal{O}(n \log^2 n)$ bound, independent of the pattern size. It is proved using an interesting lemma (Lemma 2.1) on special graphs. For one-dimensional PPM under similarity mappings only an $\mathcal{O}(n^2 m \log(n/m))$ algorithm is known (see Subsection 2).

Two-dimensional PPM under translations or under rotations can easily be reduced to the one-dimensional case. Theorem 5 reports a well known result for the two-dimensional problem for isometries. It shows that this problem can be solved in $\mathcal{O}(n^{4/3} m \log n)$ time. This bound rely heavily on the best known results for the problems, how many unit distances can appear among n points in the plane and how fast can they be reported. The algorithm for the two-dimensional problem for similarity mappings is very similar to the corresponding one-dimensional algorithm. Our investigations on better algorithms failed, but we found the interesting Theorems 6 and 7.

One general remark: All classes that we consider are closed under reflection, like isometries or similarity mappings. For simplicity we present each time only an algorithm for the subclass of nonreflected mappings, e.g. *even* isometries and *even* similarity mappings. This suffices, because the problem for the whole class can in each case be solved with the same algorithm by running it twice, once with the original pattern and once with a pattern, that is the reflection of the original one (on an arbitrary hyperplane).

This article studies only exact matching. Another approach to pattern matching is the computation of the (nonsymmetric) Hausdorff distance. [8] gives solutions for minimal Hausdorff distance computations under Euclidean motion, [1] discusses pseudooptimal solutions by use of a reference point. In [14] methods are given to compute the Hausdorff distance of digital images using special-purpose graphics hardware (z-buffer). A topic similar to point pattern matching is the determination of congruences and symmetries of point sets, which is treated in [2].

I want to thank Helmut Alt, who supported me in the process of this work.

1 One-Dimensional PPM under Even Isometries

In one dimension even isometries and translations are the same. We consider one-dimensional PPM with respect to translations in the following form:

> Given two sets $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_m\}$ of real numbers, determine whether there exists a real number t with $B + t \subseteq A$ (and find all such numbers).

First we state the upper bound yet known for arbitrary patterns:

Theorem 1 *One-dimensional PPM is solvable in $\mathcal{O}(nm + n \log n)$ time.*

Proof (by Algorithm):

Because of the $\mathcal{O}(n \log n)$-term we can consider the input to be given as sorted sequences (sorted arrays) $A = (a_1, \ldots, a_n)$ and $B = (b_1, \ldots, b_m)$. We give here a simple algorithm for finding all pattern occurrences, which we describe informally (see Figure 1). It translates the pattern consecutively with its first point on each point of the picture and checks whether the pattern occurs at this position. The work which must be done on one position we call a *check* on this position. In each check the algorithm performs for consecutive points of the pattern a search in the picture to determine if the translated pattern point matches some picture point, until the first search fails. We describe later how this search is done. We will call a search "good" if the pattern point is found in the picture, and "bad" otherwise. Thus, in each check there will be at most one bad search, after which the check terminates with negative answer.

At this point the correctness of the algorithm follows from the fact, that the algorithm tests all possible matching positions. We will now achieve the claimed time bound by performing all searches for one b_j in the different checks altogether in only $\mathcal{O}(n)$ time. We use for each b_j a pointer c_j with initial value zero which stores the position (the index) of the search for b_j in A. All linear searches for b_j can be performed now without decreasing c_j: After a good search for b_j, c_j contains the index of the matching point in A, and after a bad search, it contains the smallest index k for which a_k is bigger than the translated b_j. The next linear search for b_j can start at this position. Figure 1 illustrates this argument. Thus the time needed for all searches for one b_j is altogether $\mathcal{O}(n)$, and the running time of the whole algorithm (after the sorting step) is $\mathcal{O}(nm)$.

□

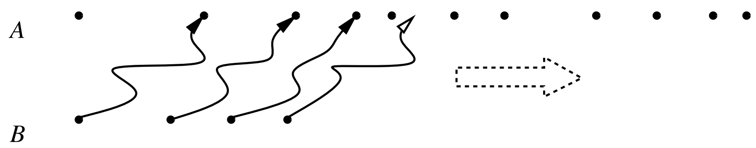

Figure 1: The principle of the algorithm.

The worst case running time of our algorithm is also $\Theta(nm)$, and it is reached for example if $A = \{1, 2, \ldots, n\}$ and $B = \{1, 2, \ldots, m-1, m-\frac{1}{2}\}$: For this input, the algorithm will find on each possible position all but the last pattern points. This gives also the reason why it suffices to perform the searches as linear searches: In our example, each search takes only constant time, so we can not do better. This particular worst case example gives rise to the question: Do all worst case inputs have some regular structure? In a certain sense, the answer will be yes: We give here a definition for patterns without regularities.

Definition 1 *We call a pattern B in* **general position** *if it is affinely independent in the vector space* $\mathbf{R} : \mathbf{Q}$, *i.e. for the set*

$$D := \{b_2 - b_1, b_3 - b_1, ..., b_m - b_1\}$$

there are no linear combinations with integer coefficients that sum up to zero, except the trivial one. (This means also, D is linearly independent in $\mathbf{R} : \mathbf{Q}$*.)*

For example, patterns of real numbers, drawn with a continuous probability distribution, are in general position. Exactly the patterns which can be interpreted as finite subsets of an one-dimensional grid are not in general position. In fact, for patterns in general position, a significantly better bound can be shown:

Theorem 2 *For patterns B in general position, one-dimensional PPM is solvable in $\mathcal{O}(n \log^2 n)$ time.*

Proof:

We again consider A and B as sorted sequences (sorted arrays). We change our algorithm in the following way: We replace the linear searches for the pattern points in the picture by so called exponential-binary searches, i.e. we first go one step, then two, then four and so on, until we exceed our target, and then perform a binary search between the last two positions. Where linear search takes k steps, an exp.-bin. search needs only $\mathcal{O}(\log k)$ steps. So the bound from Theorem 1 still holds.

Now, given a picture A and a pattern B in general position, we want to show that our algorithm achieves the claimed time bound. Because the time for each exp.-bin. search is in any case $\mathcal{O}(\log n)$, it suffices to show that there will be only $\mathcal{O}(n \log n)$ such searches. In each of the n checks there can be only one bad search (because we leave the check after a bad search), therefore it remains to prove that there are at most $\mathcal{O}(n \log n)$ good searches.

Given an arbitrary good search, the difference between two picture points (the one is the point on which B is positioned with b_1 and the other is the point which is found by the search) is an element of $D := \{b_2 - b_1, b_3 - b_1, \ldots, b_m - b_1\}$. The good searches are therefore mapped injectively into the edge set of the graph $G(A, E)$ defined by:

$$\forall u, v \in A : uv \in E \Leftrightarrow v - u \in D,$$

with D as above. Each edge $uv \in E$ is naturally labeled with $v - u$, which is an element of D. An example is shown in Figure 1.

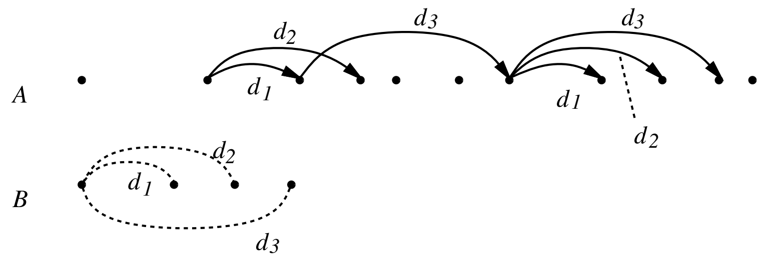

Figure 2: The Graph G

It now suffices to show :

$$|E| \leq \frac{n \log n}{2}$$

We can assume that G is connected in the undirected sense, and $a_0 = 0$. (If G is disconnected in the undirected sense, we can translate the vertices of one connected component of G until a new edge between this and a second component appears; so G does not have a maximal number of edges.) We take an arbitrary a_i and an arbitrary undirected path from a_0 to a_i. If we sum up the labels on the edges of the path (taking the negative label if the edge is passed backwards), we get a representation of a_i as an integer linear combination of D. We now use the affine independency of B and can conclude that this linear combination depends only on a_i, but not on the particular path chosen.

If we map each point a_i from A to the $(m-1)$-tuple of coefficients of that unique linear combination representing a_i, we get a mapping of A into the $(m-1)$-dimensional unit grid Z^{m-1}. The edges of G are mapped onto edges of length 1 between neighbouring grid points. Such a graph we will call a grid-graph:

Definition 2 *A graph G(V, E) is called a* **grid-graph** *if the following conditions hold:*

1. *V is a subset of $\mathbf{Z}^d$ for some $d \in \mathbf{N}$,*

2. *$\forall u, v \in V$: $uv \in E \Leftrightarrow u$ and v have Euclidean distance 1.*

We can therefore complete the proof by using the following lemma, which is of interest not only for this application:

Lemma 1 *Let G(V, E) be a grid-graph. Then $|E| \leq \frac{n \log n}{2}$, where $n = |V|$.*

Proof (by induction):

$n = 1$: clear.

$(1 \ldots n-1) \rightarrow n$: We take an arbitrary hyperplane h parallel to $d-1$ coordinate axes that cuts $c \geq 1$ edges of G. Taking away these edges, we get two grid-graphs, with k and $n-k$ vertices, say $k \leq n/2$, for which the claim holds by induction. Furthermore $c \leq k$ (the number of vertices in the smaller part is the maximum number of edges that can cross h). It follows:

$$
\begin{aligned}
|E| &\leq \frac{k \log k}{2} + \frac{(n-k)\log(n-k)}{2} + c \\
&\leq \frac{k \log k}{2} + \frac{(n-k)\log(n-k)}{2} + k
\end{aligned}
$$

The right hand side is a function of k for fixed n. By considering the derivatives, it turns out that it has its maximum for $k = n/2$. We get:

$$
\begin{aligned}
|E| &\leq \frac{n/2 \log(n/2)}{2} + \frac{n/2 \log(n/2)}{2} + \frac{n}{2} \\
&= \frac{n \log(n/2)}{2} + \frac{n}{2} \\
&= \frac{n \log n}{2}.
\end{aligned}
$$

$\square$

Remarks

1. In Lemma 1 the bound on the number of edges of a grid-graph does not depend on the dimension in which the vertices live. An intuitive explanation of this surprising property is that n vertices have most grid edges in common if they are packed as a (possibly incomplete) hypercube in $\lceil \log n \rceil$ dimensions.

2. The algorithm described in the proof of Theorem 2 is not numerically totally instable, as it may seem because of the very strong condition for the patterns. First of all, the correctness does not depend on this condition; the algorithm works for arbitrary patterns. Secondly, one could use a considerably weaker condition for the patterns in the proof. For example, we could assume only that the linear combination representing each point of A is independent of the particular path chosen to generate this combination (this is exactly the point where the affine independence is needed). However, this new condition is no more only a property of the pattern, but of both the pattern and the picture; therefore the meaning of the theorem could be more difficult to understand in this new version.

3. The bound established by Theorem 2 does not seem to be tight for the discussed special case, because the graph G seems to be significantly larger than the subgraph the algorithm works on.

   In short, G contains all pattern *fractions* occurring in A, while the algorithm works only on pattern *prefixes* occuring in A. More formally, the mapping beween good searches and edges of G is only into E, but not onto E. An edge, on which a search is mapped will be referred to as *visited*.

   Now it is an interesting question how many visited edges can occur for a given picture size n. So far it is still open whether the number $v(n)$ of visited edges in G is $o(n \log n)$. On the other hand we show with the following example that the number of visited edges can be superlinear. (This seems to be moreover a worst case example for the number of visited edges).

   We can characterize the visited edges as follows: Let e be an edge. The vertex incident on e which has the lower coordinate value is called the *base vertex*. We call the set of edges which have a vertex a as base vertex and lie in ascending dimensions, starting with the first dimension, *an ascending n-pod* on a. The visited edges are exactly the edges which are contained in ascending n-pods.

   We define a family of grid-graphs $G(n)$ with n vertices for which the number of visited edges is $v(n) = \Theta(n \log \log n)$, by giving their vertex sets $A(n)$. We define $A(n)$ for $n = 2^{2^k}$, $k \in \mathbf{N}_0$ in a recursive fashion as it is shown in Figure 3. Each $A(2^{2^k})$ lives in the $k+1$-dimensional unit grid. $A(2^{2^0})$ is the set $\{(0,0),(1,0)\}$, therefore the corresponding grid-graph has one edge between the two points.

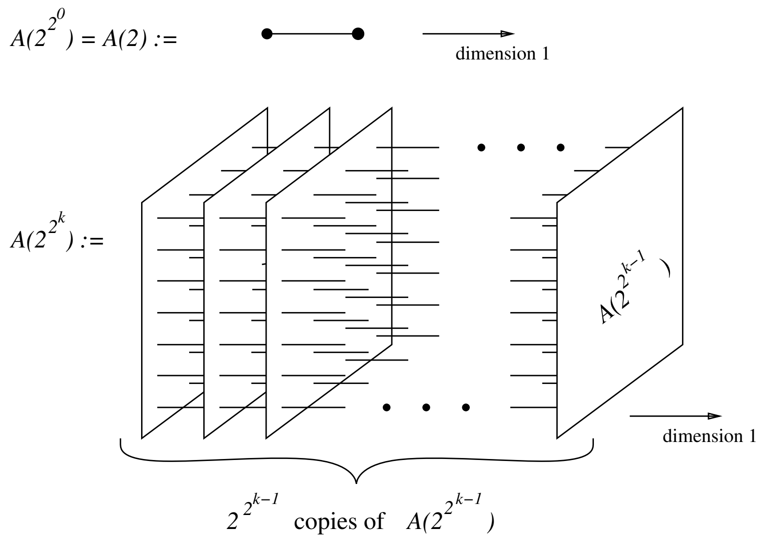

Figure 3: The Definition of $A(2^{2^k})$

Theorem 3 *Let $v(n)$ be the number of visited edges in $A(n)$, then:*

$$v(n) = \Theta(n \log\log n)$$

Proof:

We prove for $n = 2^{2^k}$:

$$\tfrac{1}{4}k2^{2^k} \le v(2^{2^k}) \le (k+1)2^{2^k}$$

- Proof of the left inequality (by induction):
  The claim holds for $k = 0$. Thus let $k \ge 1$.
  The visited edges in $A(2^{2^k})$ are either in the direction of the first dimension, or they are visited edges in the first $(2^{2^{k-1}} - 1)$copies of $A(2^{2^{k-1}})$.
  Therefore we get:

$$\begin{aligned}
v(2^{2^k}) &= (2^{2^k} - 2^{2^{k-1}}) + (2^{2^{k-1}} - 1)\ v(2^{2^{k-1}}) \\
&\ge (2^{2^k} - 2^{2^{k-1}}) + (2^{2^{k-1}} - 1)\ \tfrac{1}{4}(k-1)\ 2^{2^{k-1}} \\
&\ge \tfrac{1}{2}2^{2^k} + (2^{2^{k-1}} - 1)\ \tfrac{1}{4}(k-1)\ 2^{2^{k-1}} \\
&\ge \tfrac{1}{2}2^{2^k} + 2^{2^{k-1}}\ \tfrac{1}{4}(k-2)\ 2^{2^{k-1}} \\
&= \tfrac{1}{2}2^{2^k} + \tfrac{1}{4}(k-2)\ 2^{2^k} \\
&= \tfrac{1}{4}k\ 2^{2^k}
\end{aligned}$$

- Proof of the right inequality: Let $e(2^{2^k})$ be the total number of edges of $G(2^{2^k})$. We charge each edge to its base vertex. Each vertex is charged by at most $k+1$ edges, since $G(2^{2^k})$ lives in $k+1$ dimensions. We therefore have
$$v(2^{2^k}) \leq e(2^{2^k}) \leq (k+1)2^{2^k}$$
edges in total.

From the proven inequality follows the conjecture.

□

We can construct from each set A(n) immediately two sets A and B such that the algorithm performs $\Theta(n \log\log n)$ good searches in A: We must choose an arbitrary set B with $|B| = \log\log n$ in general position and define A as the set of all integer linear combinations of the set D_B whose coordinate vectors represents elements of $A(n)$. This means, we perform the inversion of the mapping described in the proof of Theorem 2.

2 One-Dimensional PPM under Even Similarity Mappings

A straightforward algorithm solving the PPM-problem for similarity mappings is the following: We take two arbitrary points $p < q$ from B. For each pair $s < t$ of points from A there is a unique even similarity mapping M, that maps (p, q) onto (s, t). We check now for each of these $\mathcal{O}(n^2)$ mappings, if it maps B into A.

How long does the check take for one particular mapping M? We must search the m points of $M(B)$ in A. We use again exponential-binary search: We start the search for $M(b_1)$ at a_1. If $M(b_i)$ is found in A, we can start the search for $M(b_{i+1})$ at the position of $M(b_i)$ in A. The worst case is the case that all searches are succesful. Let k_i be the difference between startindex and stopindex of the search for $M(b_i)$ in A. We have:
$$k_1 + \ldots + k_m \leq n \tag{1}$$
The time needed for the search of $M(b_i)$ is $c \log k_i)$ for some constant c. Hence the time needed for the check is
$$c(\log k_1 + \ldots + \log k_m)$$
If $\bar{k}$ is the average of $k_1, \ldots, k_m$, then because of the convexity of the logarithm this term is at least
$$c(\log \bar{k} + \ldots + \log \bar{k})$$
This term is $\mathcal{O}(m \log(n/m))$ because of 1. The time complexity of the algorithm is therefore $\mathcal{O}(n^2 m \log(n/m))$.

3 Two-Dimensional PPM under Translations or Scaled Rotations

Theorem 4 *Two-dimensional PPM with only translations or with only scaled rotations (i.e. combinations of scalings and rotations around a given fixpoint) allowed can be solved in time $\mathcal{O}(nm + n \log n)$.*

Proof: We reduce both problems to the one-dimensional problem with isometries (translations). For the two-dimensional problem with translations we consider the points to be given as cartesian coordinate vectors. For the two-dimensional problem with scaled rotations we consider the points to be given as polar coordinate vectors around the given fixpoint, i.e. given is the angle and the logarithm of the radius. The problem to solve is now the following:

> Given two sets A and B of two-dimensional real vectors, determine whether there exists a two-dimensional real vector t with $B + t \subseteq A$ (and find all such vectors).

Every addition or lexicographical comparison of the two-dimensional vectors costs $O(1)$ time. The problem can be solved with the same algorithm as for the one-dimensonal problem with isometries (translations), except that our input sets A and B are now finite subets of the two-dimensional real vector space and the operations in the one-dimensional algorithm must be replaced by the corresponding vector operations.

□

4 Two-Dimensional PPM under Even Isometries

We consider now PPM in the plane with even isometries as allowed geometric transformations. For this problem instance, we describe here an already known algorithm. This algorithm is a straightforward application of deep results on another problem, the problem of counting and reporting all unit distances in a set of n points.

Theorem 5 *2-dim PPM with respect to isometries can be solved in $\mathcal{O}(n^{4/3} m \log n)$ time.*

Proof: The algorithm has the following steps:

1. Chose a pair (p, q) of points from B.
2. Search for all sorted pairs (s, t) in A which have the same distance as (p, q) (i.e. if (s, t) is such a pair, then also (t, s)).

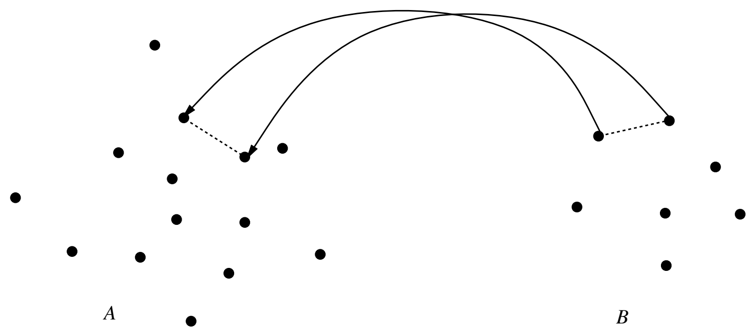

Figure 4: The mapping between the two pairs defines the whole mapping

3. For each such (s, t) the mapping from (p, q) onto (s, t) can be uniquely extended to a rigid motion M (even isometric mapping) of the plane. (see Figure 4).
   For each such rigid motion M:
   Check whether $M(B) \in A$.
   If yes, report M.

We are now interested in an efficient algorithm to compute all occurrences of a given distance in a set of n points in the plane. Moreover we want to know the maximum number $g(n)$ of such distances (or occurrences of a given distance). Both problems have turned out to be hard. The question of bounds for $g(n)$ was posed first by Paul Erdös in [9]. He established:

$$n^{1+c/\log\log n} \leq g(n) \leq n^{3/2}$$

The best upper bound known so far is $\mathcal{O}(n^{4/3})$, which was first stated by Spencer, Szemeredi and Trotter in [22]. A simpler proof of the same bound, which also leads to a better constant, was presented by Clarkson, Edelsbrunner, Guibas, Sharir and Welzl in [7].

To give an efficient algorithm for computing all unit distances in a finite planar point set is also very difficult. The problem can be considered as an incidence problem: Compute all incidences between n points and n unit circles. It is very similar to a problem called Hopcroft's Problem: Compute all incidences beween n points and n lines in the plane.

Matoušek has presented in [16] a result, which leads to an algorithm for computing all unit distances in time $h(n) = n^{4/3} 2^{\mathcal{O}(\log^* n)}$. We can use this algorithm to perform the second step of our algorithm.

In the third step, for each M the check if $M(B) \in A$ can be performed in time $\mathcal{O}(m \log n)$ as follows: Test for each $b \in B$, if $M(b) \in A$. Having A properly preprocessed, this works in time $\log n$ for each $b \in B$. A proper preprocessing of A at the beginning of the algorithm would be the lexicographically sorting of A. Hence the total time needed by our algorithm is:

$$\mathcal{O}(h(n) + g(n) m \log n) = \mathcal{O}(n^{4/3} m \log n)$$

□

5 Two-Dimensional PPM under Even Similarity Mappings

We use an algorithm similar to that for the corresponding onedimensional case: We take two arbitrary points p, q from B. For each pair (s, t) of points from A each of the two mappings of (p, q) onto (s, t) can be uniquely extended to a even similarity mapping of the plane. We check now for each pair (s, t) of points from A whether one of the two even similarity mappings M_1, M_2 that map (p, q) onto (s, t) maps B into A. As explained in the proof to Theorem 5 this check requires $\mathcal{O}(m \log n)$ time. Since each pair (p, q) can be mapped onto any pair (s, t) between points in A by even similarity mappings, the total running time of this algorithm is $\mathcal{O}(n^2 m \log n)$.

In contrast to point pairs, a triangle pqr can be mapped only onto similar triangles stu in A. So we would obtain a faster algorithm if the set of triangles similar to a given one within a set of n points would be $o(n^2)$. Unfortunately, this is not the case. This is stated in the following two theorems, which are of general interest.

Theorem 6 *Let PQR be an arbitrary triangle. Then for $n \in \mathbf{N}$ there are point sets A with $\mathcal{O}(n)$ elements for which there are $\Omega(n^2)$ 3-subsets that are similar to PQR.*

Proof: In Figure 5 we give an example for a set A with the desired property.
G_1 is the set of all points of the unit grid with an Euclidean distance to the origin of at most $\sqrt{n}$ and G_2 the set of all points of the unit grid with an Euclidean distance to the origin of at most $2\sqrt{n}$. We refer to the unit grid as U. We assume that the point P of our triangle is the origin.
Let M_R be the similarity mapping that fixes R and maps P onto Q. We define $G_Q := M_R(G_1)$. Clearly $G_Q \subset M_R(U)$.
Let M_Q be the similarity mapping that fixes Q and maps P onto R. We define $G_R := M_Q(G_1)$. Clearly $G_R \subset M_Q(U)$.
Now we define $A := G_2 \cup G_Q \cup G_R$.
Remark: Let p, q, r be the lengths of the edges of PQR. Then the grid edges of $M_R(U)$ (and thus of G_Q) have length p/q, and the grid edges of $M_Q(U)$ (and thus of G_R) have length p/r.

There are $\Omega(n^2)$ triangles $P'Q'R'$ similar to PQR (according to the point names and under an even similarity mapping) with $Q' \in G_Q$ and $R' \in G_R$. Therefore it suffices to prove that for all these triangles $P' \in G_2$. Now let $P'Q'R'$ be such a triangle similar to PQR and with $Q' \in G_Q$ and $R' \in G_R$. The proof works by considering an intermediate triangle $\bar{P}QR'$ where $\bar{P}$ is the point, for which $\bar{P}QR'$ is similar to PQR (see Figure 6).

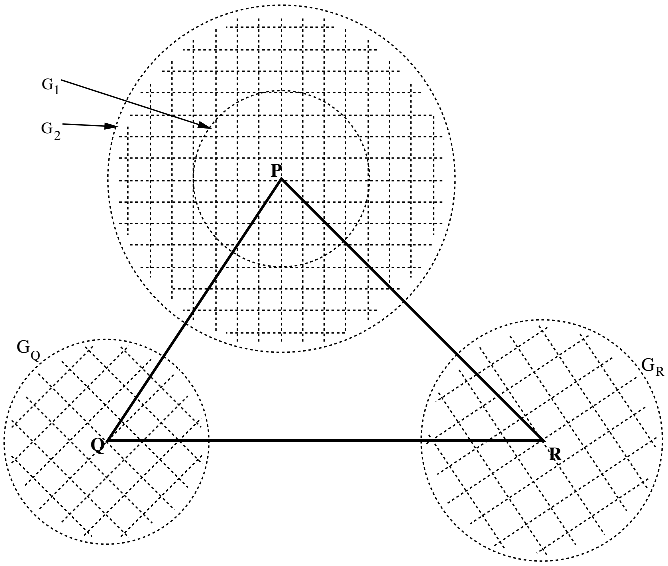

Figure 5: Illustration to Theorem 6.

- We first prove: $P' \in U$. In order to do this we first remark that $\bar{P} \in U$ because of the definition of G_R and $R' \in G_R$. The very reason for $P' \in U$ is now, that U is mapped onto $M_R(U)$ under the similarity mapping $M_{R'}$, that fixes R' and maps $\bar{P}$ onto Q. (This is so because $M_{R'}(U)$ must contain Q and must have grid edges of length q/p and the same inclination as $M_R(U)$. Thus both grids must be the same.) Therefore $Q, Q' \in M_{R'}(U)$ and from this follows immediately $P' \in U$.

- It remains to prove that the Euclidean Distance $d(P, P')$ is at most $2\sqrt{n}$. We have $d(P, \bar{P}) = \frac{r}{p} d(R, R') \leq \sqrt{n}$ and $d(\bar{P}, P') = \frac{q}{p} d(Q, Q') \leq \sqrt{n}$. With the triangle inequality we get:

$$d(P, P') \leq d(P, \bar{P}) + d(\bar{P}, P') \leq 2\sqrt{n}$$

□

Even for a k-element subset S of a grid, the number of occurrences in a picture A can be quadratic in $|A|$, but the constant depends on S:

Theorem 7 *Let S be an arbitrary k-subset of the unit grid and L the greatest absolute value of a coordinate in S, but at least 1. Then for $n \in \mathbf{N}$ there is a point set A with n elements for which there are $\Omega(n^2/L^4)$ k-subsets that are similar to S.*

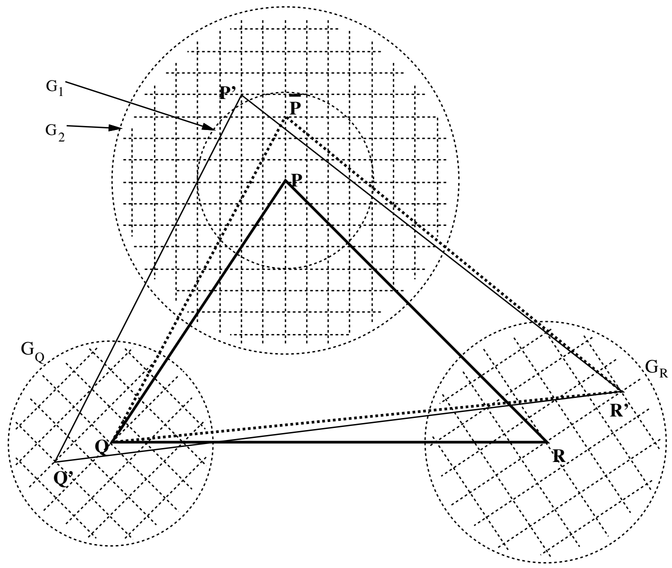

Figure 6: The intermediate triangle $\bar{P}QR'$ is drawn with dotted lines.

Proof:

Let n be fixed. We define A to be simply the set of all points of the grid with Euclidean distance less than $\sqrt{n}$ from the origin (see Figure 7). Let T be the set of all points with distance less than $\sqrt{n}/4L$. To prove, that A has the claimed property, it will suffice to show: Each similarity mapping M that maps $(0,0)$ and $(1,0)$ onto points in T maps S into A. Let M be such a mapping. First we remark that by definition of L all points of S have euclidean distances of at most $\sqrt{2}L$ from $(0,0)$. Since the distance of $M(0,0)$ and $M(1,0)$ will be at most $\sqrt{n}/2L$ (see definition of T), all point of $M(S)$ have distances of at most $\sqrt{n}/\sqrt{2}$ from $M(0,0)$. From $M(0,0) \in T$ and the triangle inequality follows that all points of $M(S)$ have distances of at most $\sqrt{n}/\sqrt{2} + \sqrt{n}/4L \leq \sqrt{n}$ from $(0,0)$ and are therefore in A.

T has $\Omega(n/L^2)$ elements, so there are $\Omega(n^2/L^4)$ such similarity mappings M.

□

Remark: The conjecture holds for each finite set S of points with rational coordinates in the plane, since each such S is similar to a subset of the unit grid.

References

[1] H. Alt, O. Aichholzer, G. Rote, "Matching Shapes with a Reference Point", Proc. 10th Annu. ACM Sympos. Comput. Geom., 1994, 85–92.

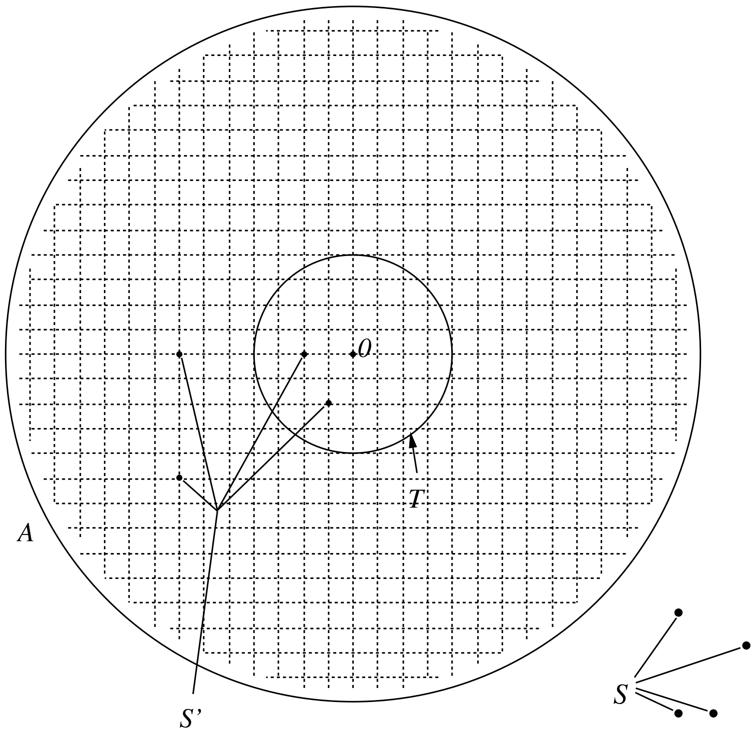

Figure 7: Illustration to Theorem 7.

[2] H. Alt, K. Mehlhorn, H. Wagener and E. Welzl, "Congruence, Similarity, and Symmetries of Geometric Objects", *J. on Discr. Comp. Geom.*, **3** (1988), pp. 237-256.

[3] E. M. Arkin, K. Kedem, J. S. B. Mitchell, J. Sprinzak and M. Werman, "Matching Points into Noise Regions: Combinatorial Bounds and Algorithms.", Proc. 2nd ACM-SIAM Symposium on Discrete Algorithms, 1991, pp. 42-51.

[4] M. J. Atallah, "Checking Similarity of Planar Figures", *Internat. J. Comput. Inform. Science*, 13 (1984), pp. 279-290.

[5] D. Baldini, M. Barni, G. Benelli, A. Foggi and A. Mecocci, "A New Star-Constellation *(sic!)* Matching Algorithm for Satellite Attitude Determination", *ESA Journal* 1993, Vol. 17, pp. 185-198.

[6] M. Ben-Or, "Lower Bounds For Algebraic Computation Trees", *Proc. Symp. on Theory of Computation*, 1983, pp. 80-86.

[7] K. L. Clarkson, H. Edelsbrunner, L. J. Guibas, M. Sharir and E. Welzl, "Combinatorial Complexity Bounds for Arrangements of Curves and Spheres", *J. on Discr. Comp. Geom.* **5** (1990), pp. 99-160.

[8] L. P. Chew, M. T. Goodrich, D. P. Huttenlocher, K. Kedem, J. M. Kleinberg and D. Kravets, "Geometric Pattern Matching under Euclidean Motion", Proceedings Can. Conf. on Comp. Geom., 1993, pp. 151-155.

[9] P. Erdös, "On Sets of Distances of n Points", *American Mathematical Monthly*, Vol. 53, 1946, pp. 248-250.

[10] H. Edelsbrunner. *Algorithms in Computational Geometry*, Springer-Verlag, Heidelberg, 1987.

[11] R. L. Graham, D. E. Knuth and O. Patashnik, *Concrete Mathematics*, Addison-Wesley, 1989.

[12] L. J. Guibas, D. E. Knuth and M. Sharir, "Randomized Incremental Construction of Delaunay and Voronoi Diagrams", *Algorithmica*, **7** (1992), pp. 381-413.

[13] K. Imai, S. Sumino and H. Imai, "Minimax Geometric Fitting of Two Corresponding Sets of Points.", Proc. 5th ACM Symp. on Comp. Geom., 1989, pp. 266-275.

[14] D. P. Huttenlocher, G. A. Klanderman and W. J. Rucklidge, "Comparing Images Using the Hausdorff Distance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15** (1993), pp. 850-863.

[15] P. J. Heffernan and S. Schirra, "Approximate Decision Algorithms for Point Set Congruence.", Proceedings, 8th Annual ACM Symp. on Computational Geometry, 1992, pp. 93-101.

[16] J. Matoušek, "Range searching with efficient hierarchical cuttings", Proc. 8th ACM Symp. on Computational Geometry, 1992, pp. 276-285.

[17] Roger Penrose, *The Emperor's New Mind*, Oxford University Press, 1989.

[18] F. Preparata and M. Shamos, *Computational Geometry - An Introduction*, Springer-Verlag, Berlin 1985.

[19] G. Rote, "Computing the minimal Hausdorff distance between two point sets on a line under translation", *Information Processing Letters*, **38** (1991), pp. 123-127.

[20] R. Seidel, "Backwards Analysis of Randomized Geometric Algorithms", Technical Report ICSI Berkley, TR-92-014, 1992.

[21] U. Renner, B. Lübke-Ossenbeck and P. Butz, "TUBSAT, Low Cost Access to Space Technology", Proceedings, Symp. (international) Small Satellites Systems and Services, 1992, Arcachon.

[22] J. Spencer, E. Szemeredi and W. T. Trotter, Jr., “Unit Distances in the Euclidean Plane”, *Graph Theory and Combinatorics*, 1984, pp. 293-303.

[23] G. Weber, L. Knipping, H. Alt, “An Application of Point Pattern Matching in Astronautics ”, Technical Report B 93-16, Institut für Informatik, FU Berlin, 1993.