# Recognizing Pemutations Generated by Sweeps of Planar Point Sets

Hanspeter Bieri* and Peter-Michael Schmidt†

January 1993

## 1 Introduction

The *plane-sweep method* is a well-known general technique in Computational and Combinatorial Geometry with a steadily growing number of applications. In the 2-dimensional case the translational or rotational sweep of $\mathbb{R}^2$ by a straight line and the rotational sweep by a straight ray represent the most common cases. The basic idea consists in stopping the sweep at a finite number of event points in order to solve there local problems. The resulting partial solutions form then together the solution of the initial problem. In our case, the set of event points $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^2$ is given at the beginning. We assume $P$ to be *in general position* in the following sense: Let $\mathbf{H}(P)$ denote the set of all lines joining two points of $P$, then it is assumed that no two lines of $\mathbf{H}(P)$ are parallel and no three lines of $\mathbf{H}(P)$ meet in one point. Now, sweeping the plane $\mathbb{R}^2$ by a translational or rotational sweep the points of $P$ are met in a certain order. In [Schm92] it is examined which of the $n!$ orders of $P$ can be generated by a translational sweep and it is proved that their number is in $\Theta(n^2)$. In [BiSc93] the analogous problem is studied for rotational line- and ray-sweeps, and it is proved that in both cases the corresponding numbers of orders is in $O(n^4)$. Hence for $n$ „large" only „few" of the $n!$ possible orders of $n$ points $\in \mathbb{R}^2$ in general position can be obtained by performing a translational or rotational sweep. In the following we study two „inverse" problems: Let $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^2$ in general position and $\pi$ a permutation of $\{1, \ldots, n\}$. Can the order $\left(p_{\pi(1)}, \ldots, p_{\pi(n)}\right)$ be generated by a translational or rotational sweep, respectively ? This question is called the *decision problem*. If yes, what are the actual sweeps generating this order (*search problem*). For both problems we will present efficient solutions.

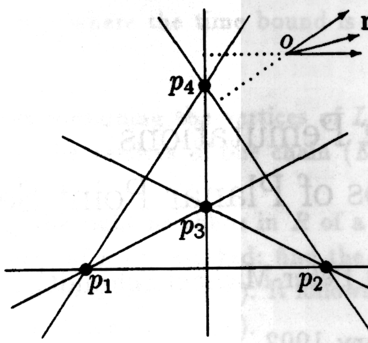
## 2 Translational and rotational sweeps

### 2.1 Translational sweep

A translational sweep will be specified by a normal vector $\mathbf{n} = (\cos\varphi, \sin\varphi)$ indicating the direction of the sweep and by a real parameter $\tau$ indicating the actual position of

*Universität Bern, Institut für Informatik und angewandte Mathematik, Länggassstrasse 51, CH-3012 Bern, Switzerland, bieri@iam.unibe.ch

†Friedrich-Schiller-Universität, Fakultät für Mathematik und Informatik, Universitätshochhaus, 17. OG, O-6900 Jena, Germany, schmidt@mathematik.uni-jena.dbp.de




$\pi_1 = (1342)$ $\pi_7 = (2431)$

$\pi_2 = (1324)$ $\pi_8 = (4231)$

$\pi_3 = (1234)$ $\pi_9 = (4321)$

$\pi_4 = (2134)$ $\pi_{10} = (4312)$

$\pi_5 = (2314)$ $\pi_{11} = (4132)$

$\pi_6 = (2341)$ $\pi_{12} = (1432)$

Figure 1: All permutations generated by translational sweeps of a 4–point set.

the sweep–line. That is, the sweep–line is defined at any time by $T(\mathbf{n},\tau) := \{\mathbf{x} \in I\!R^2 : <\mathbf{n},\mathbf{x}> = \tau\}$, where $<\cdot,\cdot>$ denotes the scalar product. Each point $p_i \in P$ determines uniquely a parameter $\tau_i$ by means of the condition $p_i \in T(\mathbf{n},\tau_i)$. The permutation $\pi$ of $\{1,\ldots,n\}$ can be generated by a translational sweep in direction $\mathbf{n}$ iff $\tau_{\pi(1)} < \tau_{\pi(2)} < \ldots < \tau_{\pi(n)}$. Figure 1 shows all permutations $\pi_i$ of a given set $P = \{p_1,\ldots,p_4\}$ in the form $(\pi_i(1)\ldots\pi_i(n))$. Only 12 of the 24 permutations of $\{1,\ldots,4\}$ can be generated in this case. Figure 1 also shows the directions $\mathbf{n}$ of all those sweeps which generate $\pi_1$. [Schm92] presents an optimal $O(n^3)$ algorithm for reporting all permutations generated by sweeping $n$ points by a translational sweep. For every such permutation the interval of all directions $\mathbf{n} = (\cos\varphi, \sin\varphi)$ leading to it is also reported.

## 2.2 Rotational sweep

Any $q = (\eta_1,\eta_2) \in I\!R^2 \setminus \bigcup \mathbf{H}(P)$ may be a center of rotation. For a rotational line–sweep around $q$ the sweep line is defined at any time by $R(q,\tau) = \{x = (\eta_1 + \rho\cos\tau, \eta_2 + \rho\sin\tau) : \rho \in I\!R\}$. We assume the sweep to start at $p_1 \in P$, i.e. $\tau$ passes through $[\tau_1, \tau_1 + \pi)$ where $\tau_1$ is given by $p_1 \in R(q,\tau_1)$. A rotational ray–sweep around $q$ is defined analogously: In the expression for $R(q,\tau)$ the set $I\!R$ is replaced by $I\!R^+$, and $\tau$ passes now through $[\tau_1, \tau_1 + 2\cdot\pi)$. As with a translational sweep a permutation $\pi$ of $\{1,\ldots,n\}$ can be generated by a rotational sweep iff $\tau_{\pi(1)} < \tau_{\pi(2)} < \ldots < \tau_{\pi(n)}$. By assumption $\pi(1) = 1$ always holds. Let $\mathbf{A}(P)$ be the arrangement defined by $\mathbf{H}(P)$. $\mathbf{A}(P)$ partitions $I\!R^2$ into finite number of cells of dimensions 0, 1 and 2. All centers of rotation lying in the same 2–dimensional cell of $\mathbf{A}(P)$ lead to rotational sweeps generating the same permutation $\pi$. [BiSc93] presents an $O(n^5 \log n)$–algorithm for reporting all permutations generated by sweeping $n$ points by a rotational line–sweep or ray–sweep, respectively. For every such permutation the 2–dimensional cells leading to it are also reported. Figure 2 shows an arrangement $\mathbf{A}(P)$ and states for each 2–dimensional cell $E$ which permutation is generated by a line–sweep around any center of rotation $q \in E$.

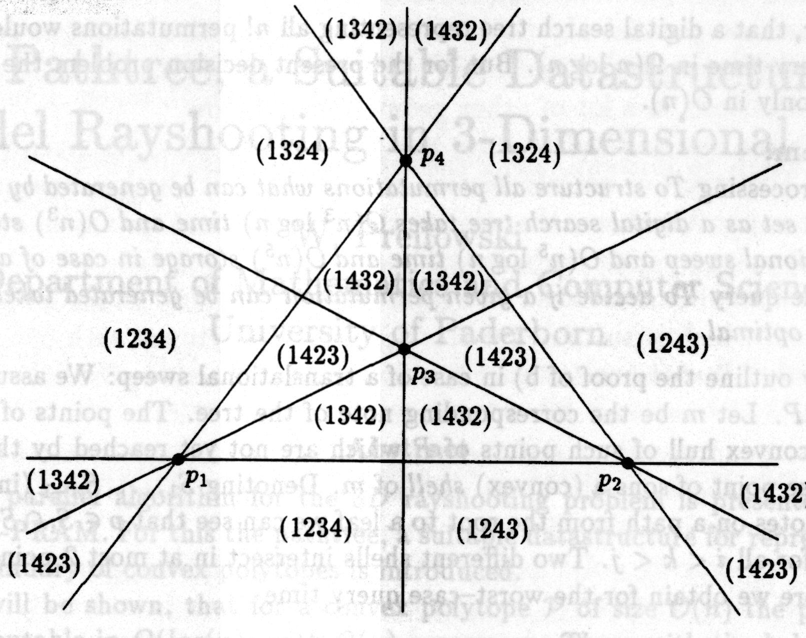



Figure 2: The cells of $\mathbf{A}(P)$ and the corresponding permutations generated by rotational line–sweeps.

## 3 A solution to the decision problem

In order to decide efficiently if a given permutation can be generated by a translational line–sweep, a rotational line–sweep or a rotational ray–sweep, respectively, we first perform the following preprocessing: All permutations what can be generated are determined by means of the respective algorithms from [Schm92] or [BiSc93]. Then they are sorted lexicographically and structured as a digital search tree (cf. [TeAu86],[Mehl84]). Figure 3 shows the digital search tree representing the permutations of Figure 1. Each path from the root to a leaf represents a permutation. Now, in order to answer a query, i.e. to decide if a given permutation can be generated, we execute the following tree search: Starting from the root we find at each next level the appropriate note by means of a binary search. When arriving at a leaf note the query can be answered by string comparison. As many queries can be answered after only a very small number of steps the digital search tree seems to be a very natural data structure for our purpose. We should bear in mind,

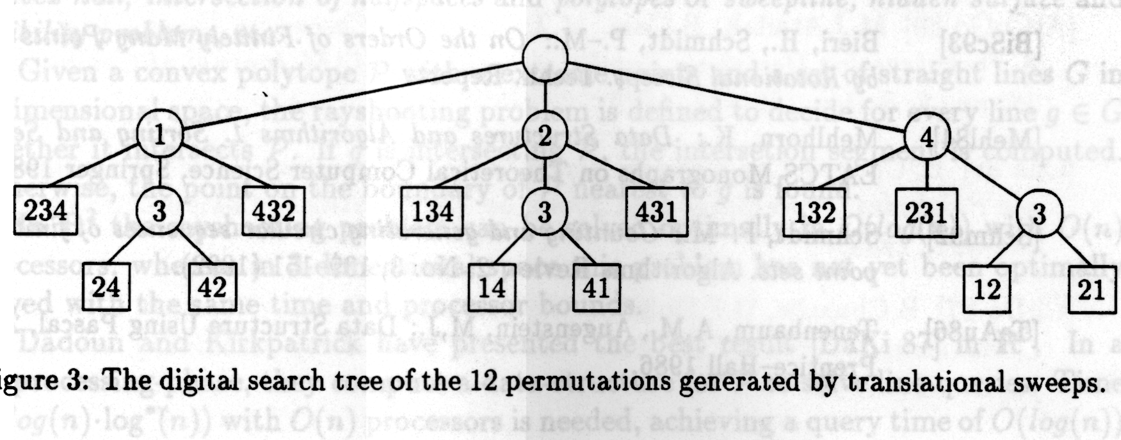



Figure 3: The digital search tree of the 12 permutations generated by translational sweeps.

however, that a digital search tree representing all $n!$ permutations would lead to a worst-case query time in $\Omega(n \log n)$. But for the present decision problem the worst-case query time is only in $O(n)$.

**Theorem:**
a) Preprocessing *To structure all permutations what can be generated by sweeping a planar n-point set as a digital search tree takes $O(n^3 \log n)$ time and $O(n^3)$ storage in case of a translational sweep and $O(n^5 \log n)$ time and $O(n^5)$ storage in case of a rotational sweep.*
b) Single query *To decide if a given permutation can be generated takes then time $O(n)$, what is optimal.*

We only outline the proof of b) in case of a translational sweep: We assume a sweep stops at $p_i \in P$. Let $m$ be the corresponding note of the tree. The points of the sons of $m$ lie on the convex hull of such points of $P$ which are not yet reached by the sweep line. We call these point of sons a (convex) *shell* of $m$. Denoting $S_1, \ldots, S_{n-1}$ (in this order) shells of the notes on a path from the root to a leaf we can see that $p \in S_i \cap S_j$ for $i < j$ implies $p \in S_k$ for all $i < k < j$. Two different shells intersect in at most 2 points. Therefore we obtain for the worst-case query time

$$c \cdot \sum_{i=1}^{n-1} \log |S_i| < c \cdot (n-1) \cdot \log \left( \frac{1}{n-1} \sum_{i=1}^{n-1} |S_i| \right) \in O(n) \text{ using } \sum_{i=1}^{n-1} |S_i| \leq n + 2(n-1)$$

and the Jensen inequality ($c$ is the constant arising from the binary search in $S_1, \ldots, S_{n-1}$).

## 4 Outlook to arbitrary dimension

Using linear programming the decision problem for translational sweeps in $\mathbb{R}^d$ can also be solved in linear time: A query whether or not there is a normal vector $\mathbf{n} \in \mathbb{R}^d$ such that $\tau_{\pi(1)} < \tau_{\pi(2)} < \ldots < \tau_{\pi(n)}$ with $\tau_j = <\mathbf{n}, p_j>$ for $j = 1, \ldots, n$ is equivalent to the consideration of the existence of a solution $\mathbf{n}$ of the system of linear inequalities $<\mathbf{n}, p_{\pi(k+1)} - p_{\pi(k)}> > 0$, $k = 1, \ldots, n-1$. The disadvantage of this method is that the implementation is difficult and the constants in the $O$-notation grows exponentially in $d$. The task is the design of a more geometric algorithm which is in addition applicable to the rotational sweep.

## References


[BiSc93] Bieri, H., Schmidt, P.-M.: *On the Orders of Finitely Many Points Induced by Rotational Sweeps.* Techn. Report.

[Mehl84] Mehlhorn, K.: *Data Structures and Algorithms 1. Sorting and Searching.* EATCS Monographs on Theoretical Computer Science, Springer 1984.

[Schm92] Schmidt, P.-M.: *Counting and generating circular sequences of finite planar point sets.* Algorithms Review 2, No. 3, 139–151 (1992).

[TeAu86] Tenenbaum, A.M., Augenstein, M,J.: Data Structure Using Pascal, 2nd edn. Prentice-Hall 1986.