Divide and Conquer Method for k-Set Polygons

Wael El Oraiby and Dominique Schmitt

Laboratoire MIA, Université de Haute-Alsace
4, rue des Frères Lumière, 68093 Mulhouse Cedex, France
{Wael.El-Oraiby,Dominique.Schmitt}@uha.fr

Abstract. The k-sets of a set V of points in the plane are the subsets of k points of V that can be separated from the rest by a straight line. In order to find all the k-sets of V, one can build the so called k-set polygon whose vertices are the centroids of the k-sets of V. In this paper, we extend the classical convex-hull divide and conquer construction method to build the k-set polygon.

1 Introduction

Given a finite set V of $|V| = n$ points in the Euclidean plane (no three of them being collinear) and an integer k ($0 < k \leq n$), the k-set polygon $g^k(V)$ of V is the convex hull of the centroids $g(T)$ of all the subsets T of k elements of V. Andrzejak and Fukuda [1] showed that the vertices of $g^k(V)$ are the centroids of the k-sets of V, i.e. of the subsets of k points of V that can be strictly separated from the rest by a straight line. Thus, determining k-sets comes down to constructing k-set polygons. Counting and constructing k-sets is an important problem in computational geometry. Cole, Sharir, and Yap [4] have given an algorithm to determine the k-sets by generalizing the convex hull algorithm of Jarvis [7]. Their algorithm can be implemented to run in $O(n \log n + m \log^2 k)$ time, where m is the size of the output. In [6] we extended the incremental convex hull construction algorithm to construct k-set polygons. The algorithm performance was in $O(n \log n + k(n-k) \log^2 k)$ time and we showed that incremental algorithms constructing k-set polygons may have to construct $\Omega(k(n-k))$ edges.

In this paper we extend another classical convex hull construction method to the k-set polygon, namely the divide and conquer method. Our algorithm works similarly in that it starts by dividing the set of points V recursively into subsets of relatively equal size, then recursively constructs their k-set polygons, and finally merges the two polygons. We first characterize the edges to remove, which form two connected lines on the k-set polygons to merge. We also show that the edges to create can be obtained by considering k-set polygons of only $2k$ points. This leads to an algorithm that constructs the k-set polygon of n points in $O(n \log n + m \log^2 k \log(n/k))$ time, where m is the worst case size of the output.

2 Preliminaries

Throughout this paper we will consider the boundary of the k-set polygon $g^k(V)$ of V to be oriented in counter clockwise direction. We denote by st the closed

H. Ito et al. (Eds.): KyotoCGGT 2007, LNCS 4535, pp. 166–177, 2008.

oriented line segment with s as startpoint and t as endpoint, by (st) the oriented straight line generated by st, and by $(st)^+$ (resp. $(st)^-$) the open half plane on the left (resp. right) of (st). $\overline{(st)^+}$ and $\overline{(st)^-}$ denote the closures of $(st)^+$ and $(st)^-$.

Let us first recall two important properties of the vertices and edges of k-set polygons given by Andrzejak and Fukuda [1], and by Andrzejak and Welzl [2].

Proposition 1. *The centroid $g(T)$ of a subset T of k points of V is a vertex of $g^k(V)$ if and only if T is a k-set of V. Moreover, the centroids of distinct k-sets are distinct vertices.*

Proposition 2. *$g(T)g(T')$ is an oriented edge of $g^k(V)$ if and only if there exist two points s and t of V and a subset P of $k-1$ points of V such that $T = P \cup \{s\}$, $T' = P \cup \{t\}$, and $V \cap (st)^- = P$.*

Such an oriented edge will be denoted by $e_P(s,t)$; $g(P \cup \{s\})$ will be called its start vertex, and $g(P \cup \{t\})$ its end vertex. Note that $e_P(s,t)$ is parallel to (st) (see Fig. 1).

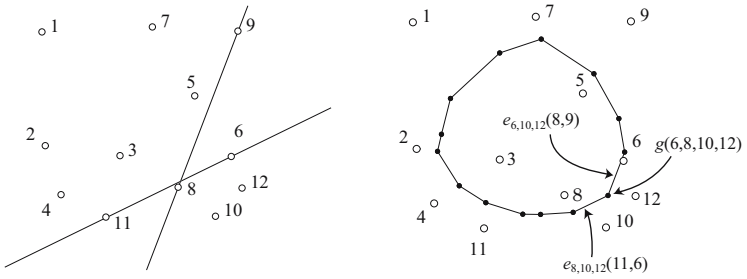

Fig. 1. Edges and vertices of a 4-set polygon of 12 points

From the above propositions we can easily obtain the following:

Proposition 3. *If $e_P(s,t)$ and $e_{P'}(s',t')$ are two consecutive edges of $g^k(V)$, then the line segments st and $s't'$ intersect.*

Propositions 1 and 2, lead to an efficient data structure to store k-set polygons. Indeed, the boundary of $g^k(V)$ can be stored in a circular list L whose elements represent the edges of $g^k(V)$. To any element e of L, which represents an edge $e_P(s,t)$, are associated the two elements of L that represent the predecessor and the successor of $e_P(s,t)$ on the boundary of $g^k(V)$, as well as the two points s and t of V. Note that, from Proposition 2, the k-sets defining two consecutive vertices of $g^k(V)$ differ from each other by one point and thus it suffices to know one k-set T of V and one edge with endpoint $g(T)$ to be able to generate the whole k-sets of V while traversing L. It follows that a k-set polygon with c edges can be stored in a data structure of size $O(c+k)$. In this paper we will store in the data structure the two k-sets whose centroids are the leftmost and the rightmost vertices of $g^k(V)$.

3 Edge Removal

For the sake of simplicity of the exposition, we will assume that no two points of V belong to a same vertical line and that no four distinct points of V belong to two parallel lines.

Suppose now that the points of V are sorted with respect to the x-axis (from left to right). Suppose also that we are given two subsets V_l and V_r of at least k points of V each, having at most $k-1$ common points, and such that there exists a vertical strip containing $V_l \cap V_r$. $V_l \setminus V_r$ and $V_r \setminus V_l$ are respectively on the left and on the right side of the strip. More precisely, there exist two vertical straight lines Δ_l and Δ_r oriented upwards such that $\Delta_l \subset \Delta_r^+$, $V_r \subset \Delta_l^-$, $V_l \subset \Delta_r^+$, and $V_l \cap V_r = (\Delta_l^- \cap \Delta_r^+) \cap (V_l \cup V_r)$. Suppose furthermore that $g^k(V_l)$ and $g^k(V_r)$ are given and that we want to construct $g^k(V_l \cup V_r)$. As in the construction of the classical convex hull, the method consists in finding the edges to remove on the given k-set polygons $g^k(V_l)$ and $g^k(V_r)$ and afterwards determining the new edges to create in order to get $g^k(V_l \cup V_r)$ (see Fig. 2).

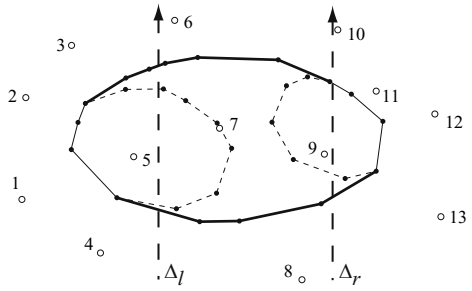

Fig. 2. Merging the 5-set polygons $g^5(1, ..., 9)$ and $g^5(6, ..., 13)$

In this section we focus on the edges to remove. We notably characterize the two connected lines that they form on $g^k(V_l)$ and $g^k(V_r)$.

Lemma 1. *(i) If $g^k(V_l)$ (resp. $g^k(V_r)$) is not reduced to a unique vertex, at least one of the edges incident in its rightmost (resp. leftmost) vertex is to be removed.*

(ii) The leftmost vertex of $g^k(V_l)$ and the rightmost vertex of $g^k(V_r)$ are vertices of $g^k(V_l \cup V_r)$.

From now on we will consider $g^k(V)^+$ (resp. $g^k(V)^-$) to be the oriented polygonal line of the edges of $g^k(V)$ that connects the rightmost to the leftmost (resp. leftmost to rightmost) vertex of $g^k(V)$ in counter clockwise direction.

Obviously, an edge st precedes an edge $s't'$ in counter clockwise direction on $g^k(V)^+$ if and only if the angle $\theta(st)$ of the oriented line (st) with the x-axis (oriented from left to right) is smaller than the angle $\theta(s't')$ of $(s't')$ with the x-axis. In the remainder of this paper the three following notations will be equivalent: $\theta(st) < \theta(s't')$, $(st) <_\theta (s't')$, and $st <_\theta s't'$.

Lemma 2. *Let $e_P(s,t)$ and $e_{P'}(s',t')$ be two edges of the line $g^k(V_l)^+$ such that $e_P(s,t) <_\theta e_{P'}(s',t')$ and let r be a point of $V_r \setminus V_l$. If $r \in (s't')^-$ then $r \in (st)^-$.*

Proof. If $e_P(s,t)$ and $e_{P'}(s',t')$ are two consecutive edges of $g^k(V_l)^+$, then from Proposition 3, the line segments st and $s't'$ intersect and, since $V_l \subset \Delta_r^+$, their intersection point belongs to Δ_r^+. Moreover, since $st <_\theta s't'$, $(s't')^- \cap \Delta_r^- \subset (st)^-$. It follows that, if a site r of $V_r \setminus V_l$ belongs to $(s't')^-$, it also belongs to $(st)^-$. By an elementary induction, the result holds for any edges $e_P(s,t)$ and $e_{P'}(s',t')$ of $g^k(V_l)^+$ such that $e_P(s,t) <_\theta e_{P'}(s',t')$. □

Similar results hold for $g^k(V_l)^-$, $g^k(V_r)^+$, and $g^k(V_r)^-$. Thus, using Proposition 2:

Theorem 1. *The edges to remove on $g^k(V_l)$ (resp. $g^k(V_r)$) form a connected line which contains the rightmost vertex of $g^k(V_l)$ (resp. leftmost vertex of $g^k(V_r)$).*

Denote respectively by $\mathcal{D}^+_{left}$ and $\mathcal{D}^+_{right}$ the lines to remove on $g^k(V_l)^+$ and $g^k(V_r)^+$, oriented in counter clockwise direction (see Fig. 3). That is, the rightmost vertex of $g^k(V_l)$ and the leftmost vertex of $g^k(V_r)$ are respectively the start vertex of $\mathcal{D}^+_{left}$ and the end vertex of $\mathcal{D}^+_{right}$.

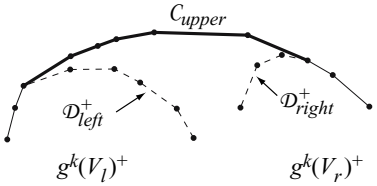

Fig. 3. The upper lines handled in Algorithms 1 and 2

We show now that the edges of $\mathcal{D}^+_{left}$ can be found efficiently by only traversing the edges to remove on the k-set polygons.

Given an oriented straight line Δ, we say that a set T is Δ-separable from V if T is a subset of V such that $\Delta^- \cap V = T$. Moreover, T is said to be $//_\Delta$-separable from V if there exists a straight line Δ', parallel to Δ and oriented as Δ, such that T is Δ'-separable from V.

Lemma 3. *An edge $e_P(s,t)$ of $g^k(V_l)^+$ is also an edge of $g^k(V_l \cup V_r)$ if and only if the k-element set T_r which is $//_{(st)}$-separable from V_r is such that $T_r \setminus V_l \subset (st)^+$.*

Proof. If $(T_r \setminus V_l) \cap (st)^- \neq \emptyset$ then, from Proposition 2, $e_P(s,t)$ is not an edge of $g^k(V_l \cup V_r)$. Conversely, if $T_r \setminus V_l \subset (st)^+$, let Δ be a straight line parallel to (st), oriented as (st), and such that T_r is Δ-separable from V_r. Since $|T_r \setminus V_l| \geq 1$, at least one point of T_r belongs to $(st)^+$. It follows that $\Delta \subset (st)^+$, that is, $V_r \setminus T_r \subset (st)^+$. Hence, from Proposition 2, $e_P(s,t)$ is an edge of $g^k(V_l \cup V_r)$. □

Thus, finding the edges of $\mathcal{D}^+_{left}$ comes down to finding the first edge $e_P(s,t)$ of $g^k(V_l)^+$ that verifies Lemma 3. Now, given a straight line Δ, the k-element set $//_\Delta$-separable from V_r can be found thanks to the following lemma:

Lemma 4. *If the k-set polygon $g^k(V_r)$ is not reduced to a unique vertex, let $g(T_0)$, ..., $g(T_m)$ and $e_{P_1}(s_1,t_1)$, ..., $e_{P_m}(s_m,t_m)$ be the vertices and the edges of $g^k(V_r)^+$, given in counter clockwise direction. Let Δ be an oriented straight line with $\theta(\Delta) \in [\pi/2, 3\pi/2]$. T_i is $//_\Delta$-separable from V_r if and only if,*
- either $i = 0$ and $\Delta <_\theta (s_1t_1)$,
- or $i \in \{1,\ldots,m-1\}$ and $(s_it_i) <_\theta \Delta <_\theta (s_{i+1}t_{i+1})$,
- or $i = m$ and $(s_mt_m) <_\theta \Delta$.

Proof. Since $g(T_0)$ is the rightmost vertex of $g^k(V_r)$, T_0 can be separated from V_r with a straight line Δ_0 such that $\theta(\Delta_0) = \pi/2$. The same, T_m can be separated from V_r with a straight line Δ_{m+1} such that $\theta(\Delta_{m+1}) = 3\pi/2$. For every $i \in \{1,\ldots,m\}$, let $\Delta_i = (s_it_i)$. Then, from Proposition 2, for every $i \in \{0,\ldots,m\}$, $T_i \subset \overline{\Delta_i^-} \cap \overline{\Delta_{i+1}^-}$ and $V_r \setminus T_i \subset \overline{\Delta_i^+} \cap \overline{\Delta_{i+1}^+}$. For every straight line Δ such that $\Delta_i <_\theta \Delta <_\theta \Delta_{i+1}$, the line Δ', parallel to Δ, oriented as Δ, and passing through $x = \Delta_i \cap \Delta_{i+1}$ is such that $\overline{\Delta_i^-} \cap \overline{\Delta_{i+1}^-} \subset \overline{\Delta'^-}$ and $\overline{\Delta_i^+} \cap \overline{\Delta_{i+1}^+} \subset \overline{\Delta'^+}$. It follows that T_i is $//_\Delta$-separable from V_r (if $x \in V_r$, it suffices to move slightly Δ' parallely to itself such that it strictly separates T_i from V_r).

The converse follows directly from the fact that, for a given straight line Δ, there exists at most one k-set T_i $//_\Delta$-separable from V_r. □

To avoid dealing with the special cases $i = 0$ and $i = m$ in the algorithm, we add two anchor-edges to the upper line $g^k(U)^+$ of the k-set polygon of any subset U of V in the following way: If $g(T)$ is the rightmost vertex of $g^k(U)^+$, insert an anchor-edge $e_P(s,t)$ with end vertex $g(T)$, such that t is the leftmost point of T and s is any point in the plane having the same x-coordinate as t and a smaller y-coordinate (note that, from the assumptions on V, $s \notin V$). Clearly, $\theta(st) = \pi/2$. In the same way, if $g(T)$ is the leftmost vertex of $g^k(U)^+$, insert an anchor-edge $e_P(s,t)$ with start vertex $g(T)$, such that s is the rightmost point of T and t is any point in the plane having the same x-coordinate as s and a smaller y-coordinate (i.e. $\theta(st) = 3\pi/2$). Note that if $g^k(U)^+$ is reduced to a unique vertex, this vertex will be incident to both anchor-edges.

The edges of $\mathcal{D}^+_{left}$ can then be found by the following algorithm:

Algorithm 1 – Find $\mathcal{D}^+_{left}$

let $e_P(s,t)$ be the edge of $g^k(V_l)^+$ with start vertex the rightmost vertex of $g^k(V_l)^+$
let $g(T)$ be the leftmost vertex of $g^k(V_r)$
let $e_{P'}(s',t')$ be the edge of $g^k(V_r)^+$ with end vertex $g(T)$ (i.e. $P' \cup \{t'\} = T$)
(1) *while $T \setminus V_l \not\subset (st)^+$*
  $e_P(s,t) \longleftarrow$ successor of $e_P(s,t)$ on $g^k(V_l)$
(2) *while $st <_\theta s't'$*
  $e_{P'}(s',t') \longleftarrow$ predecessor of $e_{P'}(s',t')$ on $g^k(V_r)$
  (3) *while $(P' \cup \{t'\}) \setminus V_l \not\subset (st)^+$*
    $e_P(s,t) \longleftarrow$ successor of $e_P(s,t)$ on $g^k(V_l)$
return $e_P(s,t)$

For any polygonal line $\mathcal{L}$, let $|\mathcal{L}|$ be the number of edges of $\mathcal{L}$.

Proposition 4. *(i) At the end of the algorithm, $g(P \cup \{s\})$ is the end vertex of $\mathcal{D}^+_{left}$.*

(ii) $\mathcal{D}^+_{left}$ can be found in $O(k + |\mathcal{D}^+_{left}| \log k + |\mathcal{D}^+_{right}|)$ time.

Proof. (i.1) Every edge $e_P(s,t)$ of $g^k(V_l)$ traversed by the algorithm, except possibly the last one, is to be removed since at least one of the points of $V_r \setminus V_l$ belongs to $(st)^-$, as checked in loops (1) and (3) conditions. Note that these loops necessarily stops at the latest on an anchor-edge.

(i.2) If $e_P(s,t)$ and $e_{P'}(s',t')$ are the last edges encountered by the algorithm respectively on $g^k(V_l)^+$ and on $g^k(V_r)^+$ then, from loop (2) condition, there exist two consecutive edges $e_{P'_1}(s'_1,t'_1)$ and $e_{P'_2}(s'_2,t'_2)$ of $g^k(V_r)^+$ such that $s't' \leq_\theta s'_1t'_1 <_\theta st <_\theta s'_2t'_2$. From Lemma 4, $P'_1 \cup \{t'_1\}$ is then $/\!/_{(st)}$-separable from V_r. Moreover, from loops (1) and (3) conditions, there exists an edge $e_{P_1}(s_1,t_1)$ of $g^k(V_l)^+$ such that $s_1t_1 \leq_\theta st$ and $(P'_1 \cup \{t'_1\}) \setminus V_l \subset (s_1t_1)^+$. From Lemma 2, $(P'_1 \cup \{t'_1\}) \setminus V_l \subset (st)^+$ and, from Lemma 3, $e_P(s,t)$ is an edge of $g^k(V_l \cup V_r)$. Since $\mathcal{D}^+_{left}$ is connected, it follows from (i.1) that $g(P \cup \{s\})$ is the end vertex of $\mathcal{D}^+_{left}$.

(ii.1) From (i.1), within a margin of one, only the edges to remove are traversed on $g^k(V_l)^+$. Moreover, for every edge $e_{P'}(s',t')$ of $g^k(V_r)$ traversed by the algorithm, except for the last one, there exists an edge $e_P(s,t)$ of $g^k(V_l)^+$ such that $st <_\theta s't'$ (loop (2) condition) and $(P' \cup \{t'\}) \setminus V_l \subset (st)^+$ (loops (1) and (3) conditions). Since, $(P' \cup \{t'\}) \setminus V_l$ contains at least one point and since this point belongs to $\Delta_r^- \cap (st)^+$, $(st) \cap (s't') \in \Delta_r^-$. Thus $\overline{(st)^-} \cap \Delta_l^+ \subset (s't')^-$ and, since $\overline{(st)^-} \cap \Delta_l^+$ contains at least one point of $(P \cup \{s,t\}) \setminus V_r$, $e_{P'}(s',t')$ is not an edge of $g^k(V_l \cup V_r)$. It follows that, within a margin of one, only the edges to remove are traversed on $g^k(V_r)^+$.

(ii.2) In loop (1), $g(T)$ is the leftmost vertex of $g^k(V_r)$ and, by hypothesis, T is stored in the data structure containing $g^k(V_r)$. To check whether $T \setminus V_l \subset (st)^+$, it suffices to compute the straight line passing through s, tangent to the convex hull $conv(T \setminus V_l)$ at a point r, and such that $conv(T \setminus V_l) \subset \overline{(rs)^+}$. $(T \setminus V_l) \subset (st)^+$ is then equivalent to $r \in (st)^+$. Since the points of V are sorted from left to right, $T \setminus V_l$ can be obtained in $O(k)$ time and $conv(T \setminus V_l)$ can also be computed in $O(k)$ time. Any tangent to $conv(T \setminus V_l)$ can then be found in $O(\log k)$ time (see for example [9]). The time complexity of loop (1) can thus be bounded by $O(k + |\mathcal{D}^+_{left}| \log k)$.

(ii.3) From (ii.1), the test $(P' \cup \{t'\}) \setminus V_l \not\subset (st)^+$ in loop (3) condition is done at most $|\mathcal{D}^+_{left}| + |\mathcal{D}^+_{right}| + 2$ times. From Lemma 2, given a set $P' \cup \{t'\}$, if an edge $e_P(s,t)$ of $g^k(V_l)^+$ is such that $(P' \cup \{t'\}) \setminus V_l \subset (st)^+$, then all the successors of $e_P(s,t)$ verify the same inclusion. Furthermore, if $e_{P''}(s'',t'')$ is the predecessor of an edge $e_{P'}(s',t')$ of $g^k(V_r)$, then $P'' \cup \{t''\} = (P' \cup \{t'\}) \setminus \{t'\} \cup \{s'\}$, from Proposition 2. It follows that, for two consecutive passes in loop (2), the considered sets $(P' \cup \{t'\}) \setminus V_l$ differ from each other by at most one point and the test $(P' \cup \{t'\}) \setminus V_l \not\subset (st)^+$ can be achieved in constant time. It is the same with all the other instructions in loop (2), which are all together in $O(|\mathcal{D}^+_{left}| + |\mathcal{D}^+_{right}|)$; hence the result. □

Obviously, $\mathcal{D}^+_{right}$ can be found in a symmetric way and it is the same with the lines to remove on $g^k(V_l)^-$ and $g^k(V_r)^-$. Hence the theorem:

Theorem 2. *The edges to remove on $g^k(V_l)$ and $g^k(V_r)$ can be found in $O(k + d \log k)$ time, where d is the total number of edges to remove.*

4 Edge Construction

In the preceding section we have seen that the edges to remove form two connected lines on the left and on the right k-set polygons. Since the k-set polygon to construct is convex, the edges to create form also two connected lines, an upper and a lower one (see Fig. 2 and Fig. 3). We denote by $\mathcal{C}_{upper}$ the oriented upper line to create and by $\mathcal{C}_{lower}$ the lower line. $\mathcal{C}_{upper}$ connects the start vertex of $\mathcal{D}^+_{right}$ to the end vertex of $\mathcal{D}^+_{left}$ (obtained by Algorithm 1).

We show now that the edges of $\mathcal{C}_{upper}$ can be found by considering k-set polygons of at most $2k$ points of $V_l \cup V_r$.

Lemma 5. *$e_P(s,t)$ is an edge of $\mathcal{C}_{upper}$ if and only if the k-element sets T_l and T_r, which are $//_{(st)}$-separable from V_l and V_r respectively, are such that:*
– $e_P(s,t)$ is an edge of $g^k(T_l \cup T_r)^+$,
– $g(T_l)$ is a vertex of $\mathcal{D}^+_{left}$ and $g(T_r)$ is a vertex of $\mathcal{D}^+_{right}$.

Proof. (i) If $e_P(s,t)$ is an edge of $\mathcal{C}_{upper}$, $P \cup \{s,t\}$ contains at least one point u of $V_l \setminus V_r$ (otherwise it would be an edge of $g^k(V_r)$). If there were a point v of $V_r \setminus T_r$ in $\overline{(st)^-}$ then, since there exists an oriented straight line Δ parallel to (st) such that T_r is Δ-separable from V_r, we would have $\Delta \subset \overline{(st)^-}$ and thus $T_r \subset \overline{(st)^-}$. Hence $T_r \cup \{u,v\} \subset \overline{(st)^-}$. This is impossible, from Proposition 2, and thus $V_r \setminus T_r \subset (st)^+$. In the same way, $V_l \setminus T_l \subset (st)^+$. It follows that $P \cup \{s,t\} \subseteq T_l \cup T_r$ and that $e_P(s,t)$ is an edge of $g^k(T_l \cup T_r)$. More precisely, since $e_P(s,t)$ is an edge of $g^k(V_l \cup V_r)^+$, it is also an edge of $g^k(T_l \cup T_r)^+$.

Moreover, if $g(T_r)$ is the leftmost vertex of $g^k(V_r)$, from Lemma 1, it belongs to $\mathcal{D}^+_{right}$. Otherwise, from Lemma 4, the edge $e_{P'}(s',t')$ of $g^k(V_r)$ with start vertex $g(T_r)$ is such that $st <_\theta s't'$. Then $e_{P'}(s',t')$ cannot be an edge of $g^k(V_l \cup V_r)$ since it should precede $e_P(s,t)$ on $g^k(V_l \cup V_r)^+$. It follows that $g(T_r)$ is a vertex of $\mathcal{D}^+_{right}$. In the same way, $g(T_l)$ is a vertex of $\mathcal{D}^+_{left}$.

(ii) Conversely, let T_l and T_r be two k-element sets $//_\Delta$-separable from V_l and V_r respectively (with a same straight line Δ) and such that $g^k(T_l \cup T_r)^+$ admits an edge $e_P(s,t)$ parallel to Δ. Since $|T_l| = k$ and $|P| = k-1$, at least one point of T_l belongs to $\overline{(st)^+}$. It follows that if Δ' is a straight line parallel to Δ such that T_l is Δ'-separable from V_l, we have $(st) \subset \Delta'^-$. Hence $V_l \setminus T_l \subset \Delta'^+ \subset (st)^+$. In the same way, $V_r \setminus T_r \subset (st)^+$. From Proposition 2, $e_P(s,t)$ is then an edge of $g^k(V_l \cup V_r)$. Moreover, since $e_P(s,t)$ belongs to $g^k(T_l \cup T_r)^+$, it also belongs to $g^k(V_l \cup V_r)^+$. Furthermore, since $g(T_r)$ belongs to $\mathcal{D}^+_{right}$, every edge $e_{P'}(s',t')$ of $g^k(V_r)^+$ that belongs to $g^k(V_l \cup V_r)$ is such that $s't' <_\theta st$. The same, since $g(T_l)$ belongs to $\mathcal{D}^+_{left}$, every edge $e_{P''}(s'',t'')$ of $g^k(V_l)^+$ that belongs to $g^k(V_l \cup V_r)$ is such that $st <_\theta s''t''$. It follows that $e_P(s,t)$ is an edge of $\mathcal{C}_{upper}$. □

It follows from this proposition that, to construct $\mathcal{C}_{upper}$, we have to consider all the couples of vertices $(g(T_l), g(T_r))$, where $g(T_l)$ and $g(T_r)$ belong to $\mathcal{D}^+_{left}$ and $\mathcal{D}^+_{right}$ respectively and such that T_l and T_r are $//_\Delta$-separable from V_l and V_r with a same straight line Δ. Then it suffices, for each of these couples, to compute the k-set polygon of $T_l \cup T_r$ and to extract some of its edges. We give now an algorithm that generates these couples efficiently, by using the result of Lemma 4.

If $e_{P_l}(s_l, t_l)$ and $e_{P'_l}(s'_l, t'_l)$ are the edges of $g^k(V_l)^+$ respectively starting and ending in $g(T_l)$ and if $e_{P_r}(s_r, t_r)$ and $e_{P'_r}(s'_r, t'_r)$ are the edges of $g^k(V_r)^+$ respectively starting and ending in $g(T_r)$, we denote by $\theta(g(T_l), g(T_r))$ the interval $[max(\theta(s'_l t'_l), \theta(s'_r t'_r)), min(\theta(s_l t_l), \theta(s_r t_r))]$.

We suppose that the upper line of any k-set polygon is completed with the same two anchor-edges as in Algorithm 1.

Algorithm 2 – Construct $\mathcal{C}_{upper}$

let $e_{P_{min}}(s_{min}, t_{min})$ *be the edge of* $g^k(V_r)^+$ *ending at the start vertex of* $\mathcal{D}^+_{right}$
let $e_{P_{max}}(s_{max}, t_{max})$ *be the edge of* $g^k(V_l)^+$ *starting at the end vertex of* $\mathcal{D}^+_{left}$
let $e_{P_l}(s_l, t_l)$ *be the edge of* $g^k(V_l)^+$ *starting at the rightmost vertex of* $\mathcal{D}^+_{left}$
(1) *while* $s_l t_l <_\theta s_{min} t_{min}$
    $e_{P_l}(s_l, t_l) \longleftarrow$ *successor of* $e_{P_l}(s_l, t_l)$ *on* $g^k(V_l)^+$
$T_l \longleftarrow P_l \cup \{s_l\}$
$e_{P_r}(s_r, t_r) \longleftarrow$ *successor of* $e_{P_{min}}(s_{min}, t_{min})$ *on* $g^k(V_r)^+$
$T_r \longleftarrow P_r \cup \{s_r\}$
$T \longleftarrow T_r$
(2) *do*
    let Θ *be the interval* $\theta(g(T_l), g(T_r))$
    (3) *let* $e_P(s, t)$ *be the edge of* $g^k(T_l \cup T_r)$ *starting at* $g(T)$
    (4) *while* $\theta(st) \in \Theta$
        insert $e_P(s, t)$ *in* $g^k(V_l \cup V_r)^+$ *such that it starts at* $g(T)$
        (5) $T \longleftarrow P \cup \{t\}$
        let $e_P(s, t)$ *be the edge of* $g^k(T_l \cup T_r)$ *starting at* $g(T)$
    (6) *if* $s_l t_l <_\theta s_r t_r$
        $e_{P_l}(s_l, t_l) \longleftarrow$ *successor of* $e_{P_l}(s_l, t_l)$ *on* $g^k(V_l)$
        $T_l \longleftarrow P_l \cup \{s_l\}$
    else
        $e_{P_r}(s_r, t_r) \longleftarrow$ *successor of* $e_{P_r}(s_r, t_r)$ *on* $g^k(V_r)$
        $T_r \longleftarrow P_r \cup \{s_r\}$
while $\theta(s_{max} t_{max}) \notin \Theta$

Proposition 5. *The algorithm constructs* $\mathcal{C}_{upper}$ *and can be implemented to run in* $O((k + |\mathcal{D}^+_{right}| + |\mathcal{D}^+_{left}| + |\mathcal{C}_{upper}|) \log^2 k)$ *time.*

Proof. (i) We first show that the algorithm constructs $\mathcal{C}_{upper}$. Since $g^k(V_l \cup V_r)^+$ is convex, the angles of the edges of $\mathcal{C}_{upper}$ with the x-axis belong to the interval $[\theta(s_{min}, t_{min}), \theta(s_{max}, t_{max})]$. Now, the intervals $\theta(g(T_l), g(T_r))$ considered

in loop (2) partition $[\theta(s_{min}, t_{min}), \theta(s_{max}, t_{max})]$. It follows, from Lemmas 4 and 5, that the edges of $\mathcal{C}_{upper}$ are the edges $e_P(s,t)$ of $g^k(T_l \cup T_r)^+$ such that $\theta(s,t) \in \theta(g(T_l), g(T_r))$, for all couples (T_l, T_r) treated in loop (2). Moreover, since the intervals $\theta(g(T_l), g(T_r))$ are treated in increasing angular order, the edges extracted from two consecutive k-set polygons $g^k(T_l \cup T_r)$ will appear consecutively on $\mathcal{C}_{upper}$.

In order to show that the algorithm works, it remains to prove that instruction (3) is valid, that is, that $g(T)$ is really a vertex of the considered k-set polygon $g^k(T_l \cup T_r)$ (even if none of its edges belongs to $g^k(V_l \cup V_r)^+$). By construction, $g(T)$ is the end vertex of the already constructed part of $\mathcal{C}_{upper}$. Denote now by $e_{P_1}(s_1, t_1)$ and $e_{P_2}(s_2, t_2)$ the edges of $g^k(V_l \cup V_r)^+$ respectively ending and starting in $g(T)$. On the one hand, T_l and T_r are such that, when instruction (3) is executed, $\theta(s_1 t_1)$ is smaller than the lower bound θ_{min} of $\theta(g(T_l), g(T_r))$, since $e_{P_1}(s_1, t_1)$ has been constructed before the couple (T_l, T_r) was considered. On the other hand, $\theta(s_2 t_2)$ is greater than θ_{min} since $e_{P_2}(s_2, t_2)$ has still to be constructed. Let Δ be a straight line such that $\theta(\Delta) \in \theta(g(T_l), g(T_r))$ and $\theta(\Delta) < \theta(s_2 t_2)$ (Δ can always be chosen in such a way that it is not parallel to any straight line passing through two points of V). Since $\theta(s_1 t_1) < \theta(\Delta) < \theta(s_2 t_2)$, T is $//_\Delta$-separable from $V_l \cup V_r$, by Lemma 4. Let now $g(T')$ be the vertex of $g^k(T_l \cup T_r)$ such that T' is $//_\Delta$-separable from $T_l \cup T_r$. If Δ' is the oriented straight line parallel to Δ such that T' is Δ'-separable from $T_l \cup T_r$, then the same reasoning as in proof of Lemma 5 shows that no point of $V_l \setminus T_l$ and of $V_r \setminus T_r$ belongs to Δ'^-. It follows that $g(T')$ is also a vertex of $g^k(V_l \cup V_r)$. More precisely, $g(T')$ is the vertex of $g^k(V_l \cup V_r)^+$ such that T' is $//_\Delta$-separable from $V_l \cup V_r$, that is, $g(T') = g(T)$. Hence, $g(T)$ is really a vertex of $g^k(T_l \cup T_r)$.

(ii) The essential step of the algorithm, given a vertex $g(T)$ of $g^k(T_l \cup T_r)$, is to determine the edge $e_P(s,t)$ of $g^k(T_l \cup T_r)$ starting at $g(T)$. If the convex hulls of T and $(T_l \cup T_r) \setminus T$ are given, it suffices to find the common oriented tangent Δ of these convex hulls such that $T \subset \overline{\Delta^-}$, $(T_l \cup T_r) \setminus T \subset \overline{\Delta^+}$, and $s = T \cap \Delta$ precedes $t = ((T_l \cup T_r) \setminus T) \cap \Delta$ on Δ. Indeed, from Proposition 2, $e_{T \setminus \{s\}}(s,t)$ is then the edge of $g^k(T_l \cup T_r)$ starting at $g(T)$. We have thus to maintain the convex hulls of T and of $(T_l \cup T_r) \setminus T$ all along the algorithm.

At the beginning of the algorithm, $g(T) = g(T_r)$ is the start vertex of $\mathcal{D}^+_{right}$. The convex hull of T can then be obtained in the following way: If $g(T')$ is the leftmost vertex of $g^k(V_r)$, the convex hull of T' can be directly computed since the points of T' are stored in the data structure containing $g^k(V_r)$. Let CH be the data structure containing this convex hull. $\mathcal{D}^+_{right}$ can then be traversed from $g(T')$ to $g(T)$ and, for each traversed edge $e_P(s,t)$, t is removed from CH and s is inserted in CH. When arriving in $g(T)$, CH contains the convex hull of T. Using the fully dynamic convex hull data structure of Overmars and van Leeuwen [8], the convex hull of T' can be stored in CH in $O(k \log^2 k)$ time and every insertion or deletion in CH can be done in $O(\log^2 k)$ time (the same operations can be achived in $O(\log k)$ amortized time with the data structure of [3]). The convex hull of the set T at the beginning of the algorithm can thus be obtained in $O((k + |\mathcal{D}^+_{right}|) \log^2 k)$ time. Since, at the beginning of the algorithm, $T = T_r$,

the convex hull of $(T_l \cup T_r) \setminus T = T_l \setminus T$ can be computed in the same way (in a dynamic structure CH') while traversing $\mathcal{D}^+_{left}$ in loop (1). In order to place in CH' only the points of T_l that are not in T, it suffices to mark the points of T (for example, while constructing their convex hull). During the execution of the algorithm, the set T is only modified by instruction (5). To update CH, we have just to remove s and to insert t. Since instruction (5) happens exactly once per edge created on $\mathcal{C}_{upper}$, the overall complexity of all the updates of CH is $O(|\mathcal{C}_{upper}| \log^2 k)$. The set $(T_l \cup T_r) \setminus T$ is modified by instructions (5) and (6). In the same way, for each of these instructions, at most one point is removed from CH' and at most one point is inserted (a point that already belongs to CH is neither removed nor inserted in CH'). Since the total number of passes in loop (2) is at most $|\mathcal{D}^+_{right}| + |\mathcal{D}^+_{left}|$ and since the total number of passes in loop (4) is equal to the number of edges of $\mathcal{C}_{upper}$, it follows that the overall complexity of the updates of CH' is $O((|\mathcal{D}^+_{right}| + |\mathcal{D}^+_{left}| + |\mathcal{C}_{upper}|) \log^2 k)$. Since a common tangent of T and $(T_l \cup T_r) \setminus T$ can also be found in $O(\log^2 k)$ time using CH and CH', $\mathcal{C}_{upper}$ can be constructed in $O((k + |\mathcal{D}^+_{right}| + |\mathcal{D}^+_{left}| + |\mathcal{C}_{upper}|) \log^2 k)$ time. □

Obviously, the lower polygonal line can be constructed similarly and we get the following result:

Theorem 3. *The edges to construct while merging $g^k(V_l)$ and $g^k(V_r)$ can be found in $O((k + d + c) \log^2 k)$ time, where d and c are the numbers of edges to delete and to create.*

The divide and conquer construction of the k-set polygon of a set V of at least k points is then as follows:

Algorithm 3 – Construct $g^k(V)$

if $|V| \leq k + 1$
  construct directly $g^k(V)$
else
  if $|V| < 2(k + 1)$
    divide V into two non-disjoint subsets V_l and V_r of k or $k + 1$ points each such that $V_l \cap V_r$ belong to a vertical strip separating $V_l \setminus V_r$ and $V_r \setminus V_l$
  else
    divide V into two disjoint subsets V_l and V_r of $\lceil |V|/2 \rceil$ and $\lfloor |V|/2 \rfloor$ points separable by a vertical straight line
  construct recursively $g^k(V_l)$ and $g^k(V_r)$
  merge $g^k(V_l)$ and $g^k(V_r)$ with Algorithms 1 and 2

Theorem 4. *Algorithm 3 constructs the k-set polygon of n points in $O(n \log n + m \log^2 k \log(n/k))$ time, where m is the worst case size of the output.*

Proof. If $n \leq k + 1$, the algorithm directly constructs the k-set polygon of V. If $n = k$, this k-set polygon is reduced to the centroid $g(V)$. If $n = k + 1$, from

Proposition 1, a vertex of the k-set polygon of V is the centroid of a subset of k points of V separable from the last one by a straight line. This last point is then a vertex of the convex hull of V and it follows that constructing the k-set polygon of V comes to constructing its convex hull. This can be done in $O(k)$ time since V is sorted.

If $n > k+1$, V is divided into two subsets V_l and V_r such that $|V_l| = \lceil n/2 \rceil$ and $|V_r| = \lfloor n/2 \rfloor$ if $n \geq 2(k+1)$, and $|V_l| \leq k+1$ and $|V_r| \leq k+1$ otherwise. The k-set polygons of V_l and V_r are then recursively constructed and, finally, merged with Algorithms 1 and 2 in $O((k+d+c)\log^2 k)$ time, where d and c are the total numbers of edges deleted and constructed in the merging step.

Now, Dey [5] and Tóth [10] have shown that the size of a k-set polygon of n points is in $O(n\beta(k))$, with $2^{\Omega(\sqrt{\log k})} \leq \beta(k) \leq O(k^{1/3})$. It follows that d and c are bounded by $O(n\beta(k))$ and that the complexity of the merging is $O(n\beta(k)\log^2 k)$. Hence the induction relation that gives the complexity $T(n)$ of the algorithm (without the sorting step):

$$\begin{array}{ll} T(n) \leq T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + O(n\beta(k)\log^2 k) & \text{if } n \geq 2(k+1) \\ T(n) \leq 2T(k+1) + O(n\beta(k)\log^2 k) & \text{if } k+1 < n < 2(k+1) \\ T(n) = O(k) & \text{if } n \leq k+1 \end{array}$$

Solving this relation, we get $T(n) = O(n\beta(k)\log^2 k \log(n/k))$. Thus, the overall complexity of Algorithm 3, including sorting, is $O(n\log n + m\log^2 k \log(n/k))$, where $m = O(n\beta(k))$ is the worst case size of a k-set polygon of n points. □

5 Conclusion

In this paper we have applied a classical algorithmic method to the construction of k-set polygons in the plane: The divide and conquer method. To this aim we have characterized the edges that are removed and the ones that are created when two k-set polygons are merged.

In the worst-case time complexity of the final divide and conquer algorithm appears an additional $\log(n/k)$ factor, in comparison with the algorithm of [4]. We suspect that this factor comes from over-estimates in the complexity computation. Indeed, in this computation, we suppose that the number of edges removed and created at each merging step is linear with the worst case sizes of the merged k-set polygons. Applied to the recursive convex hull construction (i.e. for $k = 1$), this way of computing leads to a total number of $O(n\log n)$ created edges whereas it is well known that this algorithm only constructs $O(n)$ edges in all.

The aim now is to find a finer analysis method of our algorithm to remove the $\log(n/k)$ factor. A second aim is to extend other classical convex hull algorithms to the construction of k-set polygons and namely the Quick-Hull algorithm in order to check if its good practical performances hold for k-set polygon construction.

References

1. Andrzejak, A., Fukuda, K.: Optimization over k-set polytopes and efficient k-set enumeration. In: Dehne, F., Gupta, A., Sack, J.-R., Tamassia, R. (eds.) WADS 1999. LNCS, vol. 1663, pp. 1–12. Springer, Heidelberg (1999)
2. Andrzejak, A., Welzl, E.: In between k-sets, j-facets, and i-faces: (i, j)-partitions. Discrete Comput. Geom. 29, 105–131 (2003)
3. Brodal, G.S., Jacob, R.: Dynamic planar convex hull. In: Proc. 43rd Annu. Sympos. Found. Comput. Science, pp. 617–626 (2002)
4. Cole, R., Sharir, M., Yap, C.K.: On k-hulls and related problems. SIAM J. Comput. 16, 61–77 (1987)
5. Dey, T.K.: Improved bounds on planar k-sets and related problems. Discrete Comput. Geom. 19, 373–382 (1998)
6. El Oraiby, W., Schmitt, D.: k-sets of convex inclusion chains of planar point sets. In: Královič, R., Urzyczyn, P. (eds.) MFCS 2006. LNCS, vol. 4162, pp. 339–350. Springer, Heidelberg (2006)
7. Jarvis, R.A.: On the identification of the convex hull of a finite set of points in the plane. Inform. Process. Lett. 2, 18–21 (1973)
8. Overmars, M.H., van Leeuwen, J.: Maintenance of configurations in the plane. J. Comput. Syst. Sci. 23, 166–204 (1981)
9. Preparata, F.P., Shamos, M.I.: Computational Geometry: An Introduction. Springer, New York (1985)
10. Toth, G.: Point sets with many k-sets. In: Proc. 16th Annu. ACM Sympos. Comput. Geom, pp. 37–42 (2000)