Quadratic-Time Linear-Space Algorithms for Generating Orthogonal Polygons with a Given Number of Vertices ⋆

Ana Paula Tomás[1] and António Leslie Bajuelos[2]

[1] DCC-FC & LIACC, University of Porto, Portugal
apt@ncc.up.pt
[2] Dept. of Mathematics & CEOC - Center for Research in Optimization and Control, University of Aveiro, Portugal
leslie@mat.ua.pt

Abstract. We propose Inflate-Paste – a new technique for generating orthogonal polygons with a given number of vertices from a unit square based on gluing rectangles. It is dual to Inflate-Cut – a technique we introduced in [12] that works by cutting rectangles.

1 Introduction

To test and evaluate geometric algorithms we may need to construct samples of random geometric objects. The main motivation for our work was the experimental evaluation of the algorithm described in [11]. In addition, the generation of random geometric objects raises interesting theoretical questions.

In the sequel, polygon stands for *simple polygon without holes* and sometimes it refers to a polygon together with its interior. P denotes a polygon and r the number of reflex vertices. A polygon is *orthogonal* if its edges meet at right angles. As usual, H and V are abbreviations of horizontal and vertical, respectively, e.g., H-edge, V-edge, H-ray and so forth. For every n-vertex orthogonal polygon (n-*ogon*, for short), $n = 2r + 4$, e.g. [7]. Generic orthogonal polygons may be obtained from a particular kind of orthogonal polygons, that we call *grid orthogonal polygons* (see Fig. 1). A *grid n-ogon* is any n-ogon in general position defined in a $\frac{n}{2} \times \frac{n}{2}$ square grid. P is in general position iff it has no collinear edges. We assume the grid is defined by the H-lines $y = 1, \ldots, y = \frac{n}{2}$ and the V-lines $x = 1, \ldots, x = \frac{n}{2}$ and that its northwest corner is (1,1). Every grid n-ogon has exactly one edge in every line of the grid. Each n-ogon which is not in general position may be mapped to an n-ogon in general position by ϵ-perturbations, for a sufficiently small constant $\epsilon > 0$. Hence, we may restrict generation to n-ogons in general position. Each n-ogon in general position is mapped to a unique grid n-ogon through top-to-bottom and left-to-right sweeping. Reciprocally, given a

⋆ Partially funded by LIACC through *Programa de Financiamento Plurianual, Fundação para a Ciência e Tecnologia (FCT)* and *Programa POSI*, and by CEOC (Univ. of Aveiro) through *Programa POCTI, FCT, co-financed by EC fund FEDER.*

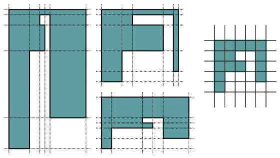


Fig. 1. Three 12-ogons mapped to the same grid 12-ogon.

grid n-ogon we may create an n-ogon that is an instance of its class by randomly spacing the grid lines in such a way that their relative order is kept.

The Paper's Contribution. We propose two methods that generate grid n-ogons in polynomial time: INFLATE-CUT and INFLATE-PASTE. The former has been published in [12]. There we mention two programs for generating random orthogonal polygons, one by O'Rourke (developed for the evaluation of [8]) and another by Filgueiras[3]. O'Rourke's program constructs such a polygon by gluing together a given number of *cells* (i.e., unit squares) in a board, starting from a seed cell. The cells are chosen in a random way using heuristics. Filgueiras' method shares a similar idea though it glues rectangles of larger areas and allows them to overlap. Neither of these methods allows to control the final number of vertices of P. The major idea in INFLATE-PASTE is also to glue rectangles. Nevertheless, it strongly restricts the positions where rectangles may be glued. In this way, not only the algorithm becomes simpler and elegant, but also controls the final number of vertices and guarantees that P is in general position. The INFLATE transformation is crucial. INFLATE-PASTE may be implemented so as to run in quadratic-time in the worst-case using linear-space in n. For the INFLATE-CUT method we had the same space complexity, but could only guarantee average quadratic-time complexity, because CUT may fail. In addition, INFLATE-PASTE allows to understand the combinatorial structure of orthogonal polygons much better [2].

In the next section we describe the INFLATE-PASTE transformation and recall INFLATE-CUT. In Sect. 3 we give a formal proof that both these techniques are complete. Finally, Sect. 4 is devoted to implementation and complexity issues.

2 Inflate, Cut and Paste Transformations

Let $v_i = (x_i, y_i)$, for $i = 1, \ldots, n$, be the vertices of a grid n-ogon P, in CCW order.

2.1 Inflate

INFLATE takes a grid n-ogon P and a pair of integers (p, q) with $p, q \in [0, \frac{n}{2}]$, and yields a new n-vertex orthogonal polygon $\tilde{P}$ with vertices $\tilde{v}_i = (\tilde{x}_i, \tilde{y}_i)$ given by

[3] Personal communication, DCC-LIACC, 2003.

$\tilde{x}_i = x_i$ if $x_i \leq p$ and $\tilde{x}_i = x_i + 1$ if $x_i > p$, and $\tilde{y}_i = y_i$ if $y_i \leq q$ and $\tilde{y}_i = y_i + 1$ if $y_i > q$, for $i = 1, \ldots, n$. INFLATE augments the grid creating two free lines, namely $x = p + 1$ and $y = q + 1$.

2.2 Inflate-Cut

Fig. 2 llustrates this technique. Let C be a unit cell in the interior of P, with

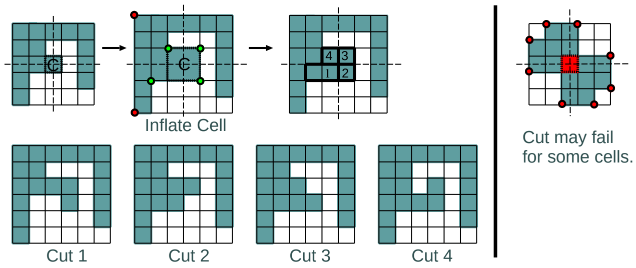


Fig. 2. The INFLATE-CUT transformation. The two rectangles defined by the center of C and the vertices of the leftmost V-edge $((1, 1), (1, 7))$ cannot be cut and so there remain the four possibilities shown. On the right we see a situation where CUT fails.

center c and northwest vertex (p, q). When we apply INFLATE using (p, q), c is mapped to $\tilde{c} = (p+1, q+1)$, that is the center of inflated C. The goal of CUT is to introduce $\tilde{c}$ as a reflex vertex of the polygon. To do that, it cuts one rectangle (defined by $\tilde{c}$ and a vertex $\tilde{v}_m$ belonging to one of the four edges shot by the H- and V-rays that emanate from $\tilde{c}$). We allow such a rectangle to be cut iff it contains no vertex of $\tilde{P}$ except $\tilde{v}_m$. If no rectangle may be cut, we say that CUT fails for C.

So, suppose that $\tilde{s}$ is the point where one of these rays first intersects the boundary of $\tilde{P}$, that $\tilde{v}_m$ is one of the two vertices on the edge of $\tilde{P}$ that contains $\tilde{s}$ and that the rectangle defined by $\tilde{c}$ and $\tilde{v}_m$ may be cut. CUT cuts this rectangle from $\tilde{P}$ and replaces $\tilde{v}_m$ by $\tilde{s}$, $\tilde{c}$, $\tilde{s}'$ if this sequence is in CCW order (or $\tilde{s}'$, $\tilde{c}$, $\tilde{s}$, otherwise), with $\tilde{s}' = \tilde{c} + (\tilde{v}_m - \tilde{s})$. We may conclude that $\tilde{s}$, $\tilde{c}$, $\tilde{s}'$ is in CCW order iff $\tilde{s}$ belongs to the edge $\overline{\tilde{v}_{m-1}\tilde{v}_m}$ and in CW order iff it belongs to $\overline{\tilde{v}_m\tilde{v}_{m+1}}$. CUT always removes a single vertex of the grid ogon and introduces three new ones. CUT never fails if C has an edge that is part of an edge of P. Hence, the INFLATE-CUT transformation may be always applied to any P.

2.3 Inflate-Paste

We first imagine the grid n-ogon merged in a $(\frac{n}{2}+2) \times (\frac{n}{2}+2)$ square grid, with the top, bottom, leftmost and rightmost grid lines free. The top line is $x = 0$ and the leftmost one $y = 0$, so that $(0, 0)$ is now the northwest corner of this extended grid. Let $e_{\mathrm{H}}(v_i)$ represent the H-edge of P to which v_i belongs.

Definition 1. *Given a grid n-ogon P merged into a $(\frac{n}{2}+2)\times(\frac{n}{2}+2)$ square grid and a convex vertex v_i of P, the* free staircase neighbourhood *of v_i, denoted by FSN(v_i), is the largest staircase polygon in this grid that has v_i as vertex, does not intersect the interior of P and its base edge contains $e_{\mathrm{H}}(v_i)$ (see Fig. 3).*

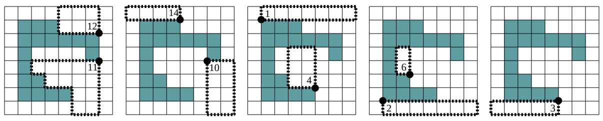


Fig. 3. A grid n-ogon merged into a $(\frac{n}{2}+2)\times(\frac{n}{2}+2)$ square grid and the free staircase neighbourhood for each of its convex vertices.

FSN(v_i) is the intersection of a particular quadrant (with origin at v_i) with the polygon formed by the external points that are rectangularly visible from v_i. This quadrant is determined by $e_{\mathrm{H}}(v_i)$ and a V-ray emanating from v_i to the exterior of P. So, FSN(v_i) may be computed in linear time by adapting Lee's algorithm [4, 5] or a sweep based method given by Overmars and Wood in [9]. We say that two points a and b are rectangularly visible if the axes-aligned rectangle that has a and b as opposite corners does not intersect the interior of P.

To transform P by Inflate-Paste (see Fig. 4) we first take a convex vertex v_i of P, select a cell C in FSN(v_i) and apply Inflate using the nortwest corner (p, q) of C. As before, the center of C is mapped to $\tilde{c} = (p+1, q+1)$, that will

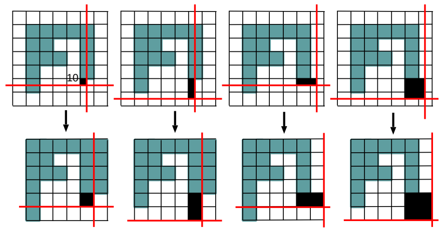


Fig. 4. At the bottom we see the four grid 14-ogons that may result when Inflate-Paste is applied to the given 12-ogon, extending the V-edge that ends in vertex 10.

now be a convex vertex of the new polygon. Paste glues the rectangle defined by $\tilde{v}_i$ and $\tilde{c}$ to $\tilde{P}$, augmenting the number of vertices by two. If $e_{\mathrm{H}}(v_i) \equiv \overline{v_i v_{i+1}}$ then Paste removes $\tilde{v}_i = (\tilde{x}_i, \tilde{y}_i)$ and inserts the chain $(\tilde{x}_i, q+1)$, $\tilde{c}$, $(p+1, \tilde{y}_i)$ in its place. If $e_{\mathrm{H}}(v_i) \equiv \overline{v_{i-1} v_i}$, Paste replaces $\tilde{v}_i$ by the chain $(p+1, \tilde{y}_i)$, $\tilde{c}$, $(\tilde{x}_i, q+1)$. Clearly, Paste never fails, in contrast to Cut, because the interior of FSN(v_i) is nonempty, for every convex vertex v_i of P.

3 Inflate-Cut and Inflate-Paste Methods

In [12], we show that every grid n-ogon may be generated from the unit square (i.e., from the grid 4-ogon) using r Inflate-Cut transformations. We may now show exactly the same result for Inflate-Paste. At iteration k, both methods construct a grid $(2k+4)$-ogon from the grid $(2(k-1)+4)$-ogon obtained in the previous iteration, for $1 \leq k \leq r$. The Inflate-Cut method yields a random grid n-ogon, if cells and rectangles are chosen at random. This is also true for Inflate-Paste, though now for the selections of v_i and of C in $\mathrm{FSN}(v_i)$. These algorithms are described in more detail in Sect. 4.

3.1 Correctness and Completeness

It is not difficult to see that both Inflate-Cut and Inflate-Paste yield grid ogons. In contrast, the proof of their completeness is not immediate, as suggested by the examples given in Figs. 5 and 6.

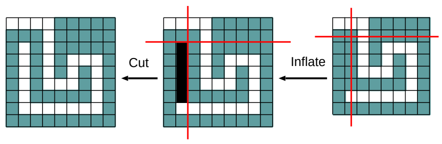


Fig. 5. The rightmost polygon is the unique grid 16-ogon that gives rise to this 18-ogon, if we apply Inflate-Cut.

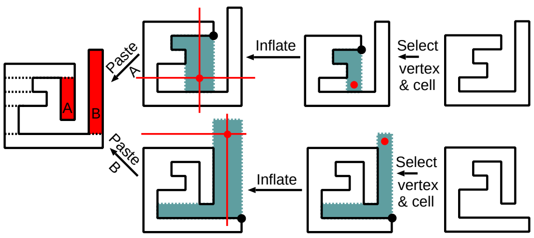


Fig. 6. The two rightmost grid 14-ogons are the unique ones that yield the 16-ogon on the left, by Inflate-Paste. It is also depicted $\mathrm{FSN}(v_i)$ for the two cases.

For the proof, we need to introduce some definitions and results. Given a simple orthogonal polygon P without holes, $\Pi_{\mathrm{H}}(P)$ represents the H-decomposition of P into rectangles obtained by extending all H-edges incident to reflex vertices towards the interior of P until they hit its boundary. Each *chord* (i.e., edge extension) separates exactly two adjacent pieces (faces), since it makes an *H-cut* (see e.g. [7]). The dual graph of $\Pi_{\mathrm{H}}(P)$ captures the adjacency relation between pieces of $\Pi_{\mathrm{H}}(P)$. Its nodes are the pieces of $\Pi_{\mathrm{H}}(P)$ and its non-oriented edges connect adjacent pieces. Surely, the V-decomposition $\Pi_{\mathrm{V}}(P)$ has identical properties.

Lemma 1. *The dual graph of $\Pi_{\mathrm{H}}(P)$ is a tree for all simple orthogonal polygons P without holes.*

Proof. This result follows from the well-known Jordan Curve theorem. Suppose the graph contains a simple cycle $F_0, F_1, \ldots, F_d, F_0$, with $d \geq 2$. Let $\gamma = (\gamma_{0,1}\gamma_{1,2}\ldots\gamma_{d,0})$ be a simple closed curve in the interior of P that links the centroids of the faces $F_0, F_1, \ldots, F_d$. Denote by v the reflex vertex that defines the chord $\overline{v\,s_v}$, which separates F_0 from F_1. Here, s_v is the point where this edge's extension intersects the boundary of P. Either v or s_v would be in the interior of γ, because γ needs to cross the H-line supporting $\overline{v\,s_v}$ at least twice and only $\gamma_{0,1}$ crosses $\overline{v\,s_v}$. But the interior of γ is contained in the interior of P, and there exist points in the exterior of P which are in the neighbourhood of v and of s_v, and so we achieve a contradiction. □

We may now prove that Inflate-Paste is complete.

Proposition 1. *For each grid $(n+2)$-ogon, with $n \geq 4$, there is a grid n-ogon that yields it by* Inflate-Paste.

Proof. Given a grid $(n+2)$-ogon P, we use Lemma 1 to conclude that the dual graph of $\Pi_{\mathrm{H}}(P)$ is a tree. Each leaf of this tree corresponds to a rectangle that could have been glued by Paste to yield P. Indeed, suppose that $\overline{v\,s_v}$ is the chord that separates a leaf F from the rest of P. Because grid ogons are in general position, s_v is not a vertex of P. It belongs to the relative interior of an edge of P. The vertex of F that is not adjacent to s_v would be $\tilde{c}$ in Inflate-Paste. If we cut F, we would obtain an inflated n-ogon, that we may deflate to get a grid n-ogon that yields P. The two grid lines $y = y_{\tilde{c}}$ and $x = x_{\tilde{c}}$ are free. Clearly s_v is the vertex we called v_i in the description of Inflate-Paste (more accurately, s_v is $\tilde{v}_i$) and $(p, q) \equiv (x_{\tilde{c}} - 1, y_{\tilde{c}} - 1) \in \mathrm{FSN}(v_i)$. □

For this paper to be self-contained, we now give a proof of the completeness of Inflate-Cut, already sketched in [12]. It was inspired by work on convexification of simple polygons [3, 10, 13], in particular, by a recent paper of O. Aichholzer et al. [1]. It also shares ideas of a proof of Meisters' *Two-Ears theorem* [6] by O'Rourke. Fig. 7 shows the main ideas. A *pocket* of a nonconvex polygon P is a maximal sequence of edges of P disjoint from its convex hull except at the endpoints. The line segment joining the endpoints of a pocket is its *lid.* Any nonconvex polygon P has at least one pocket. Each pocket of an n-ogon, together with its lid, defines a simple polygon without holes, that is almost orthogonal except for an edge (lid). It is possible to slightly transform it to obtain an orthogonal polygon, as illustrated in Fig. 7. We shall refer to this polygon as an *orthogonalized pocket.*

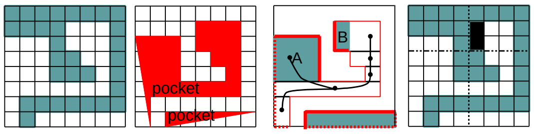
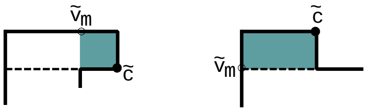


Fig. 7. The two leftmost grids show a grid 18-ogon and its pockets. The shaded rectangles A and B are either leaves or contained in leaves of the tree associated to the H-partitioning of the largest pocket. The rightmost polygon is an inflated grid 16-ogon that yields the represented grid 18-ogon, if CUT removes rectangle B.

Proposition 2. *For each grid $(n+2)$-ogon, there is a grid n-ogon that yields it by* INFLATE-CUT.

Proof. Given a grid $(n+2)$-ogon P, let Q be an orthogonalized pocket of P. Necessarily, Q is in general position. By Lemma 1 the dual graph of $\Pi_{\mathrm{H}}(Q)$ is a tree. We claim that at least one of its leaves contains or is itself a rectangle that might have been removed by CUT to yield P. Indeed, the leaves are of the two following forms, the shaded rectangles being the ones that might have been cut.



We have also represented the points that would be $\tilde{v}_m$ and $\tilde{c}$ in INFLATE-CUT. Here, we must be careful about the leaves that the lid intersects, to be able to conclude that $\tilde{c}$ is a vertex of P and that P resulted from an n-ogon in general position. Actually, an artificial H-edge, say h_Q, was introduced to render Q orthogonal, as well as an artificial V-edge. Each leaf that does not contain h_Q contains (or is itself) a rectangle that might have been removed by CUT. Every non-degenerated tree has at least two leaves. At most one leaf contains h_Q. Moreover, if the tree is degenerated (c.f. the smallest pocket in Fig. 7), then it is a leaf that could be filled. □

The notion of *mouth* [13] was crucial to reach the current formulation of CUT. Actually, INFLATE-CUT is somehow doing the reverse of an algorithm given by Toussaint that finds the convex hull of a polygon globbing-up mouths to successively remove its concavities [13]. For orthogonal polygons, we would rather define rectangular mouths. A reflex vertex v_i of an ogon P is *a rectangular mouth* of P iff the interior of the rectangle defined by v_{i-1} and v_{i+1} is in the exterior of P and neither this rectangle nor its interior contains vertices of P, except v_{i-1}, v_i and v_{i+1}. When we apply CUT to obtain a grid $(n+2)$-ogon, the vertex $\tilde{c}$ (that was the center the inflated grid cell C) is always a rectangular mouth of the resulting $(n+2)$-ogon. Thus, the proof of Proposition 2 presented above justifies Corollary 1, which rephrases the *One-Mouth theorem* by Toussaint.

Corollary 1. *Each grid n-ogon has at least one rectangular mouth, for $n \geq 6$.*

4 Quadratic-Time and Linear-Space Complexity

Our pseudocode for the two functions that yield a random grid n-ogon using INFLATE-CUT or INFLATE-PASTE is as follows, where REPLACE($\tilde{v}, \gamma, \tilde{P}$) means replace $\tilde{v}$ by chain γ in $\tilde{P}$.

```
RANDOM-INFLATE-CUT(n)
  r := n/2 − 2
  P := {(1,1), (1,2), (2,2), (2,1)}  /* (the unit square) */
  while r > 0 do
    repeat
      Select one cell C in the interior of P (at random)
      c := the center of C
      S := {points of P first shot by H-rays and V-rays emanating from c}
      A := {v_m | vertex v_m of P satisfies the CUT-condition for C}
    until A ≠ { }
    (p, q) := the northwest corner of C
    Select v_m from A (at random)  /* v_m is (x_m, y_m) */
    e_H(v_m) := the H-edge of P that contains v_m
    Apply INFLATE using (p, q) to obtain P~
    if e_H(v_m) = overline(v_{m−1} v_m) then
      P := REPLACE(v~_m, [(p+1, y~_m), (p+1, q+1), (x~_m, q+1)], P~)
    else P := REPLACE(v~_m, [(x~_m, q+1), (p+1, q+1), (p+1, y~_m)], P~)
    r := r − 1
  return P
```

```
RANDOM-INFLATE-PASTE(n)
  r := n/2 − 2
  P := {(1,1), (1,2), (2,2), (2,1)}
  A := P  /* (convex vertices) */
  while r > 0 do
    Select v_i from A (at random)  /* v_i is (x_i, y_i) */
    e_H(v_i) := the H-edge of P that contains v_i
    Compute FSN(v_i)
    Select cell C from FSN(v_i) (at random)
    (p, q) := the northwest corner of C
    Apply INFLATE using (p, q) to obtain P~, A~ and e~_H(v_i)
    if e_H(v_i) = overline(v_i v_{i+1}) then
      P := REPLACE(v~_i, [(x~_i, q+1), (p+1, q+1), (p+1, y~_i)], P~)
    else P := REPLACE(v~_i, [(p+1, y~_i), (p+1, q+1), (x~_i, q+1)], P~)
    A := (A~ \ {v~_i}) ∪ {(x~_i, q+1), (p+1, q+1)}
    if (p+1, y~_i) is not inside e~_H(v_i) then A := A ∪ {(p+1, y~_i)}
    r := r − 1
  return P
```

In Random-Inflate-Cut(n), “vertex v_m of P satisfies the Cut-condition” iff v_m is an extreme point of an edge of P that contains s, for some $s \in S$, and the rectangle defined by c and v_m does not contain other vertices of P except v_m.

Our implementation of Random-Inflate-Cut(n) uses linear space in n and runs in quadratic time in average. It yields a random grid 1000-ogon in 1.6 seconds in average (AMD Athlon Processor at 900 MHz). To achieve this, it keeps the vertices of P in a circular doubly linked list and keeps the total number of grid cells in the interior P per horizontal grid line (also in a linked list), but keeps no explicit representation of the grid. In addition, it keeps the current area of P (i.e., the number of cells), so that to select cell C it chooses only a positive integer less than or equal to the area. Cells in the interior of P are enumerated by rows from top to bottom. To locate C (i.e., its northwest corner (p, q)) the program uses the counters of number of cells per row to find row q and then left-to-right and top-to-bottom sweeping techniques to find the column p and the four delimiting edges. It is important to note that the V-edges (H-edges) of P that intersect each given horizontal (vertical) line occur always in couples, as shown in Fig. 8. This feature is used by the program to improve efficiency. To

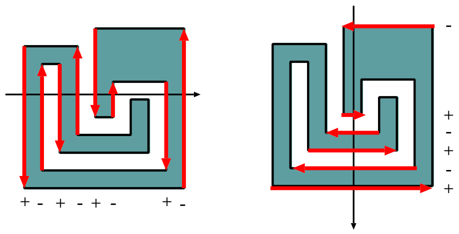


Fig. 8. Orientation of edges of P intersecting an H- or V-line.

check whether a rectangle may be cut, the program performs a rotational sweep of the vertices of P. After each Inflate or Cut transformation the counters and the area of the resulting polygon are updated. Inflate first creates a counter for the new H-line, with the same value as the counter of the previous row. Then, it analyses the sequence of H-edges that would intersect the new (imaginary) V-line, to increase counters accordingly. When a rectangle is removed, the row counters are updated by substracting the width of the rectangle removed from all counters associated with the involved rows.

Although we did not implement Random-Inflate-Paste(n) yet, it is not difficult to see that FSN(v_i) may be found in linear time. As we mentioned in Sect. 2.3, one possibility is to follow a sweep approach, adapting an algorithm described in [9]. We assume that the H-edges and V-edges are kept sorted by y-coordinate and x-coordinate, respectively, in doubled linked lists, to simplify insertion, updating and ray shooting. To compute FSN(v_i), we determine the point u shot by the V-ray emanating from v_i to the exterior of P. This point is either on an H-edge of P or on one of the two H-lines that are free in the extended grid. Then, we move a sweep V-line from v_i to the other vertex of $e_{\mathrm{H}}(v_i)$ (possibly passing it), shrinking the *visibility interval* if some event (vertex or V-edge) obstructs visibility, until the interval becomes a singleton (i.e., $[y_i, y_i]$). The initial interval corresponds to the V-segment defined by v_i and u. Using the V-decomposition of $\mathrm{FSN}(v_i)$ and its area, we may select and locate C also in linear time.

5 Conclusions

We prove that every orthogonal polygon in general position may be constructed by applying either a sequence of Inflate-Cut or Inflate-Paste transformations, using linear space. Each transformation may be performed in linear time using horizontal and vertical sweep, so that the construction requires quadratic-time in average for Inflate-Cut and in the worst case for Inflate-Paste. These methods, in particular Inflate-Paste, helped us prove some interesting properties of these kind of polygons [2] and may be easily adapted to generate simple orthogonal polygons with holes. Indeed, each hole is an orthogonal polygon without holes. We are studying whether the methods may be simplified.

References

1. Aichholzer, O., Cortés, C., Demaine, E.D., Dujmovic, V., Erickson, J., Meijer, H., Overmars, M., Palop, B., Ramaswawi, S., Toussaint, G.T.: Flipturning polygons. Discrete Comput. Geom. **28** (2002) 231–253.
2. Bajuelos, A.L., Tomás, A.P., Marques, F.: Partitioning orthogonal polygons by extension of all edges incident to reflex vertices: lower and upper bounds on the number of pieces. In Proc. of ICCSA 2004. LNCS, Springer-Verlag (this volume).
3. Erdös, P.: Problem number 3763. American Mathematical Monthly **42** (1935) 627.
4. Joe, B., Simpson, R.B: Corrections to Lee's visibility polygon algorithm. BIT **27** (1987) 458-473.
5. Lee, D.T.: Visibility of a simple polygon. Computer Vision, Graphics, and Image Processing **22** (1983) 207–221.
6. Meisters, G.H.: Polygons have ears. Am. Math. Mon. **82** (1975) 648–651.
7. O'Rourke, J.: An alternate proof of the rectilinear art gallery theorem. J. Geometry **21** (1983) 118–130.
8. O'Rourke, J., Pashchenko, I., Tewari, G.: Partitioning orthogonal polygons into fat rectangles. In Proc. 13th Canadian Conference on Computational Geometry (CCCG'01) (2001) 133–136.
9. Overmars, M., Wood, D.: On rectangular visibility. J. Algorithms **9** (1988) 372–390.
10. Sz.-Nagy, B.: Solution of problem 3763. Am. Math. Mon. **46** (1939) 176–177.
11. Tomás, A.P., Bajuelos, A.L., Marques, F.: Approximation algorithms to minimum vertex cover problems on polygons and terrains. In P.M.A Sloot et al. (eds): Proc. of ICCS 2003, LNCS 2657, Springer-Verlag, (2003) 869-878.
12. Tomás, A.P., Bajuelos, A.L.: Generating Random Orthogonal Polygons. In Post-conference Proc. of CAEPIA-TTIA'2003, LNAI, Springer-Verlag (to appear).
13. Toussaint, G.T.: Polygons are anthropomorphic. Am. Math. Mon. **122** (1991) 31–35.