The Geometry of Carpentry and Joinery ⋆

Pat Morin and Jason Morrison

School of Computer Science, Carleton University, 1125 Colonel By Drive Ottawa, Ontario, CANADA K1S 5B6

Abstract

In this paper we propose to model a simplified wood shop. Following the work of Demaine, Demaine and Kaplan in [1] we limit the cutting tools of our carpenter to a circular saw. We extend that previous work to include a model of basic rules of carpentry and joinery. This model is then applied to the problem of building a polygon P by joining together strips of wood and cutting them with a circular saw. We describe a linear time algorithm to decide if a blueprint can be constructed in such a workshop.

1 Introduction

Demaine, Demaine and Kaplan [1] study the problem of cutting a polygon P from a convex polygon Q that contains P using a *circular saw*. In their model, a circular saw is represented by a line segment of positive length r, called the *radius* of the saw. A *cut* is a line segment s, disjoint from the interior of P such that $s \setminus Q$ contains a line segment whose length is at least r. When one or more cuts disconnects Q, the component(s) not containing P are removed to obtain a new polygon Q', on which further cuts can be made.

Figure 1 illustrates how all this is analogous to cutting a shape from a piece of plywood with a circular saw by making successive cuts and removing the parts of the plywood that become disconnected from the main form. This model is a reasonable mathematical abstraction of several types of hand-held and tabletop saws whose blade is circular and must be spinning before it enters the material to be cut.

⋆ This research was funded by the Natural Sciences and Engineering Research Council of Canada, NCE-GEOIDE and the Government of Ontario.

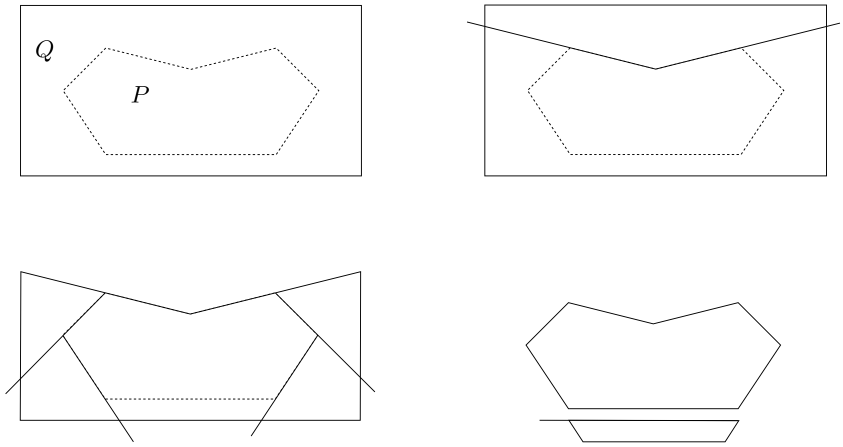


Fig. 1. Cutting a form with a circular saw.

We say that a polygon P is *cuttable with a circular saw of radius* r, if for any convex polygon Q containing P, there exists a finite sequence of cuts $c_1, \ldots, c_k$ resulting in a sequence of polygons $Q = Q_0, Q_1, \ldots, Q_k = P$ where Q_i is obtained from Q_{i-1} via cut c_i. More simply, we say that P is *cuttable by a circular saw* if there exists an $r > 0$ such that P is cuttable with a circular saw of radius r.

A *reflex vertex* of P is any vertex whose internal angle is greater than π. The main result of Demaine, Demaine and Kaplan is

Theorem 1 (Demaine, Demaine and Kaplan) *A polygon P is cuttable by a circular saw if and only if P does not have two consecutive reflex vertices on its boundary.*

In this paper we study what happens to Theorem 1 when the model is extended using some basic knowledge of carpentry. The basics we speak of encompass two fundamental areas, aesthetics and robust design. The aesthetic qualities state that in making something from wood it must be made to look as good as possible. Robust design implies that a design should eliminate as many possible sources of error as is feasible.

Aesthetics criteria imply that only quality wood can be used and thus plywood and particle board are out. Thus the large convex sheet Q, mentioned by Demaine, Demaine and Kaplan, must be made by joining smaller pieces. Also, wood pieces joined together whose grains have different orientations become a single piece which cannot be sanded. Thus the wood grain must have a specific orientation and desired polygons cannot be joined together with arbitrary pieces.

So that our assembly process is robust, we require that all of the pieces that form an edge of the polygon P must be joined together before that edge is cut. The rationale for this is to suppose that the pieces are cut before joining,

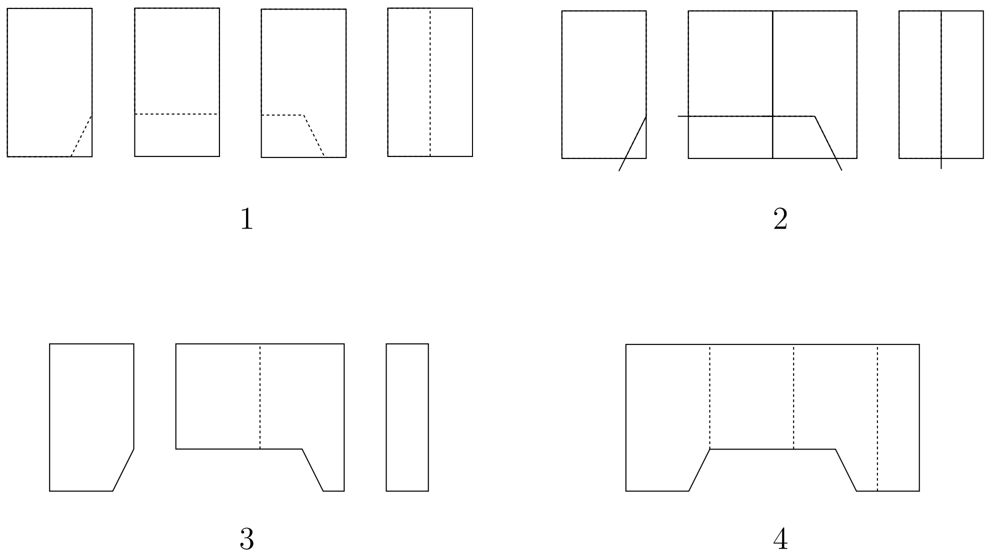


Fig. 2. Building a polygon by cutting and joining strips of wood.

then any portion of the cutting or joining process can cause the pieces not to be flush in the final product. This is in direct contradiction to the definition of robust design and therefore shows the need for the restriction.

With these ideas in mind it is clear that we study the case in which P is created by joining together regular strips (rectangles) at their edges and cutting the joined pieces with a circular saw. In this process, there are two rules that must be obeyed. The first is that once two strips are joined together they cannot be unjoined. The second is that, before cutting an edge e of P, all strips incident on e must be joined together. This model appears to be a reasonable facsimile of the process that creates many tabletops, desktops and wood floors.

Any polygon that can be cut with a circular saw can be fabricated by cutting and joining. This is because all of the wood strips can be joined into one large sheet and then cut. However, the converse is not true. Many polygons exist that cannot be cut from a large sheet using a circular saw that can be built by cutting and joining. An example is given in Figure 2. This is due to the fact that we can cut parts of the polygon individually and then join them together.

A *blueprint* $B = (P, C)$ is a polygon P, with n edges, and a set of m vertical line segments C, each of which is contained in P and has both endpoints on the boundary of P. The elements of C are called *chords* of P. In computational terms, a blueprint is represented as a subdivision of P induced by the chords in C. The chords in C partition and the edges of P partition the interior of P into maximally connected regions called *faces*.

A *join* is the process of removing a chord c from C, thereby merging the two faces incident on c. A join models the joining of two pieces of wood to form another, larger, piece of wood.

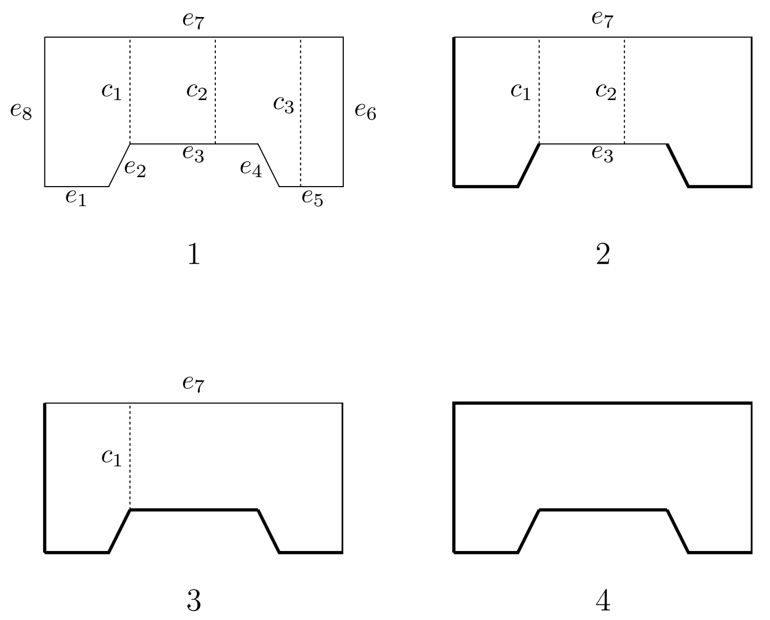


Fig. 3. Illustrating the construction $c_3, e_1, e_2, e_4, e_5, e_6, e_8, c_2, e_3, c_1, e_7$ of the desktop from Figure 2.

A *cut* is simply an edge e of P. Since a cut is intended to model the process of cutting an edge e of P, it must satisfy the following two rules:

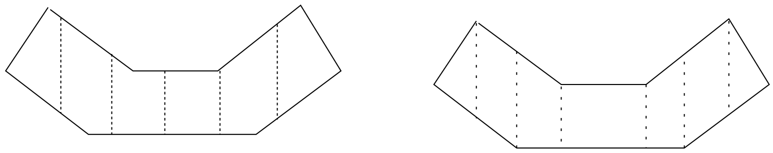
Rule 1 *The edge* e *must be on the boundary of only one face* f.

Rule 2 *At least one endpoint of* e *must be a non-reflex vertex in* f.

Rule 1 models the constraint that all strips incident on e must be merged together through a sequence of joins before cutting e. Rule 2 comes from Theorem 1 and the assumption that our cutting tool is a circular saw.

A *construction* $\mathcal{C} = v_1, \ldots, v_{n+m}$ of B is a sequence of joins and cuts in which each edge of P and each chord of C appears exactly once. We say that B is *feasible* if it has a construction. Figure 3 shows a construction of the desktop from Figure 2. Note that a construction of B only describes the order in which chords are joined and edges are cut. It does not actually provide a plan for cutting the non-reflex edges of a face preceding a join. It is possible to compute such a plan, in linear time [1].

In this paper we give a linear time algorithm to determine whether a blueprint is feasible. Section 2 describes our algorithm for determining whether a blueprint is feasible. Section 3 considers the problem of designing blueprints for a given polygon. Section 4 summarizes and concludes with directions for future research.



a) Chords C b) Steiner Chords

Fig. 4. Two different chord sets in a polygon.

2 Testing Feasibility

In this section, we study the problem of determining whether a blueprint $B = (P, C)$ is feasible and give an algorithm for finding a construction of B when it is feasible. Because of Rule 2, it is intuitively clear that consecutive reflex vertices will be the primary obstacle in finding a construction of B. Therefore, we call an edge e of P a *reflex edge* if both endpoints of e are reflex vertices.

Our algorithm is divided into two steps which are discussed in the next two subsections. In the first step we attempt to determine, for each reflex edge e, the direction the circular saw will travel when e is cut. In the second step, we find an ordering of the joining and cutting operations that gives us a construction of B.

We first observe that chords in C that have both endpoints strictly in the interior of edges in P are redundant, since nothing is lost by removing those chords immediately, and they must be removed (joined) before either of their incident edges are cut. Therefore, we assume C does not contain any chords with both endpoints in the interior of edges of P.

We begin by adding *Steiner* chords to our blueprint so that each face of the blueprint becomes a trapezoid. These Steiner chords are obtained by shooting vertical rays up and down from every vertex v in polygon P (see Figure 4). We denote by C' the set of all Steiner chords and observe that, by the assumption of the previous paragraph, $C \subseteq C'$. For clarity we say a chord is a *real chord* if it is in both C and C' and all other chords in C' are *false chords*.

We will show how to find a construction of $B' = (P, C')$ with the additional restriction that each false chord in C' must appear before the each of the edges incident on it. Once this is done, we can easily obtain the construction of B from the construction of B'.

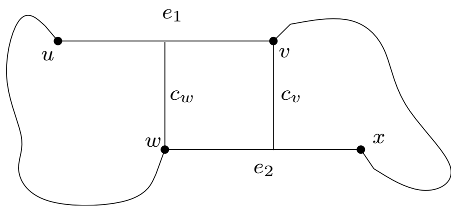


Fig. 5. An overlap.

2.1 Directing Reflex Edges

Let $e = (u, v)$ be an edge of P. Then, during a construction $\mathcal{C}$ we say that e is cut in direction $\overleftarrow{uv}$ (equivalently, $\overrightarrow{vu}$) if the joining of a chord incident on u is performed before cutting e. The following lemma states that we can join the chords of at most one endpoint of a reflex edge before cutting that edge.

Lemma 1 *There is no construction of B' in which a reflex edge $e = (u, v)$ is cut in direction $\overrightarrow{uv}$ and in direction $\overleftarrow{uv}$.*

Proof. Saying that e is cut in both directions is equivalent to saying that the chords incident on both endpoints of e are joined before e is cut. However, once these two chords are joined e is a reflex edge on the face containing e and, by Rule 2, cannot be cut. □

An *overlap* consists of two edges $e_1 = (u, v)$ and $e_2 = (w, x)$ and two chords c_v and c_w such that c_v is incident on v and on the interior of e_2 and c_w is incident on w and on the interior of e_1 (see Figure 5). The following lemma shows that reflex edges which overlap have constraints on the directions in which they can be cut.

Lemma 2 *Let $e_1 = (u, v)$, $e_2 = (w, x)$, c_v and c_w form an overlap. Then, in any construction of B' in which e_1 is cut in direction $\overleftarrow{uv}$, e_2 must be cut in direction $\overleftarrow{wx}$.*

Proof. Suppose that this were not the case, and that there is a construction $\mathcal{C}$ of B' in which e_1 is cut in direction $\overleftarrow{uv}$ and e_2 is cut in direction $\overrightarrow{wx}$. Then, by Rule 1, c_w must be joined before e_1 is cut. By Rule 2, e_2 must be cut before c_w is joined. By Rule 1, c_v must be joined before e_2 is cut. Finally, by Rule 2, e_1 must be cut before c_v is joined. If we use the notation $a \prec b$ to denote that a must occur before b in the construction then we have

$$e_1 \prec c_v \prec e_2 \prec c_w \prec e_1 \ ,$$

an impossibility. □

Lemma 2 provides a method for assigning directions to the reflex edges of P. We assign directions to edges of P using the following algorithm.

(1) If $e = (u, v)$ is a reflex edge with both endpoints on false chords then, by Lemma 1, there is no construction of B'.
(2) If $e = (u, v)$ is a reflex edge having only the endpoint u on a real chord then, by Lemma 1, any construction of B' cuts e in direction $\overrightarrow{uv}$.
(3) Finally, we iterate the following procedure until no more directions are assigned: For every reflex edge $e_2 = (w, x)$. If e_2 overlaps an edge $e_1 = (u, v)$ which has already been assigned the direction $\overleftarrow{uv}$ and which satisfies the conditions of Lemma 2 then we set the direction of e_2 to be $\overleftarrow{wx}$. If at any time this procedure attempts to reassign a different direction to an edge whose direction has already been assigned then we can terminate since, by repeated applications of Lemmas 1 and 2, B' is not feasible.
(4) Once Step 3 is complete, any reflex edge $e = (u, v)$ that has not yet had a direction assigned to it is assigned the direction $\overrightarrow{uv}$ if the x-coordinate of u is less than the x-coordinate of v and $\overleftarrow{uv}$ otherwise, so that all such edges are directed "left-to-right."

If the above algorithm succeeds in assigning directions to all edges of P then we say that the assignment of directions is *consistent*. The ability to consistently assign directions of reflex edges is a necessary condition for B' to be feasible, since the only points at which the above algorithm fails to assign consistent directions (Steps 1 and 3) provide a proof that B' is not feasible. However, we have not yet proven that it is a sufficient condition because we must also show that there exists a consistent ordering among the cut and join operations. This is the topic of the next section.

2.2 Ordering Joins and Cuts

Define the directed graph $G(B') = (V, E)$ as follows:

(1) The vertex set V consists of each edge of P and each chord of C'.
(2) The edge (c, e) is present in E if chord $c \in C'$ is false and has an endpoint on $e \in P$.
(3) The edge (c, e) is present in E if chord $c \in C'$ has an endpoint strictly in the interior of $e \in P$.
(4) The edge (e, c) is present in E if $e = (u, v)$, c has an endpoint on u and the direction of e is $\overleftarrow{uv}$.

An example of a blueprint B' for a polygon with one reflex edge along with the corresponding graph $G(B')$ is shown in Figure 6.

The following is the main result of this section.

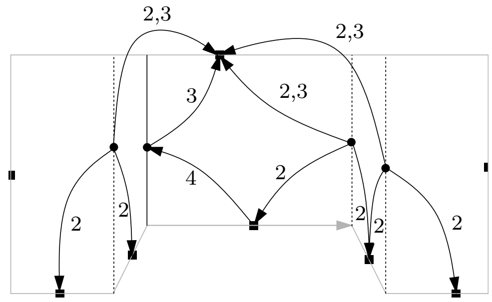


Fig. 6. A blueprint B' and the corresponding graph $G(B')$. Real chords are drawn as solid lines and false chords are drawn as dotted lines. Each edge is labelled with rule(s) (1–4) that generated it.

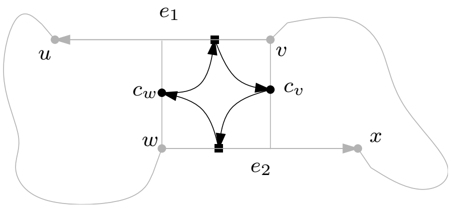


Fig. 7. A simple cycle in $G(B')$ corresponds to an overlap that violates the conditions of Lemma 2.

Theorem 2 *B' is feasible if and only if the reflex edges of P can be assigned consistent directions and $G(B')$ has no cycle.*

Proof. We have already shown that if we cannot assign consistent directions to the reflex edges of P then B' is not feasible. Therefore, suppose we can assign consistent directions to the edges of P but that $G(B')$ has a cycle. First note that, since P is a polygon and C' forms a trapezoidation of P, any cycle of length greater than 4 must have repeated vertices. Therefore, $G(B')$ contains a cycle of length exactly 4. This cycle includes 2 edges e_1 and e_2 of P and two chords c_v and c_w in C'. The edges e_1 and e_2 and chords c_v and c_w form an overlap, thereby contradicting Lemma 2 (see Figure 7).

We claim that the directions of e_1 and e_2 were not assigned during Step 4 of the algorithm for assigning directions. Clearly, one of them, say e_1, was not assigned during Step 4 otherwise they would both be directed "left-to-right" and would not violate Lemma 2. But then e_2 would also have had its direction assigned in Step 3. Therefore, in any construction of B, e_1 and e_2 must be assigned those directions. But then, by Lemma 2, there can be no construction of B'.

It remains to show the converse, i.e., if directions can be consistently assigned to reflex edges of P and $G(B')$ is acyclic then there exists a construction of B'. Suppose therefore that there is a consistent assignment of directions and $G(B')$ does not contain a cycle. Then we topologically sort $G(B')$ to obtain a total ordering $v_1, \ldots, v_m$ on V, i.e., the edges of P and the chords in C'. We claim that there exists a construction of B in which the cuts and joins occur in the order in which they appear in $v_1, \ldots, v_m$.

We prove this by showing that, by the time an edge e of P appears in the sequence $v_1, \ldots, v_{O(n)}$, e it is entirely contained on the boundary of a single face f and is not a reflex edge on f. Therefore, the construction satisfies Rules 1 and 2.

Because of the edges added during Step 1 in the construction of $G(B)$, all chords incident on e must appear before e in the total order. Therefore, Rule 1 is satisfied.

Next we need to show that the edge e is not reflex in f. If e is not a reflex edge in P then e is still not a reflex edge in f. If e is a reflex edge in P then it has been assigned a direction, and during Step 4 in the construction of $G(B')$ an edge was added that guarantees e appears in the total order before one of the chords, say c_v, incident on one of the endpoints, say v, of e. Therefore, on the face f, v is not a reflex vertex and e is not a reflex edge. Thus, Rule 2 is also satisfied. □

If P is a polygon with n edges then C' contains $O(n)$ chords and there is an algorithmic version of Theorem 2 that runs in $O(n)$ time provided that the blueprint B' is given in a topological data structure (e.g., a doubly-connected edge list [2]). If the input is not given in this form, then such a data structure can be computed in $O(n \log n)$ time using plane-sweep [3].

Step 1, 2, and 4 of the algorithm for directing reflex edges are easily implemented in $O(n)$ time. Step 3 can be implemented as a limited breadth-first traversal of the graph having reflex edges of P as vertices and an edge between two vertices if the corresponding edges of P are part of an overlap. This graph has size $O(n)$ since each reflex edge overlaps at most 2 other reflex edges and hence this step of the algorithm can be completed in $O(n)$ time.

Once the reflex edges of P have been assigned directions, the graph $G(B')$ can easily be constructed in time linear in the size of $G(B')$. Since $G(B')$ has $O(n)$ vertices and is planar, the construction of $G(B')$ can be completed in $O(n)$ time. Topologically sorting the vertices of $G(B')$ again takes $O(n)$ time (c.f., [4]).

2.3 Summary Notes

The previous sections provide an algorithm for testing feasibility of the blueprint $B' = (P, C')$ in $O(n)$ time. To obtain a construction for the original blueprint $B = (P, C)$, we observe that we can use the construction of B' and simply ignore the false chords in the construction. This works because Step 1 of the algorithm for computing G' guarantees that in the construction of B', all false chords appear before any of their incident edges.

3 Designing Blueprints

The algorithm in Section 2 provides a means of testing whether a given blueprint is feasible. An obvious question that arises is that of finding a blueprint for a given polygon P. If we do not place any constraints on our wood strips then designing a feasible blueprint is trivial. By adding all of the Steiner chords to the set C we obtain a feasible blueprint. To see this, observe that every reflex edge of P is incident on two chords in C and hence has its direction assigned in Step 4 of the algorithm for assigning directions to reflex edges. Then all reflex edges are directed "left-to-right," so it is clear that G does not contain any cycles, hence (P, C) is a feasible blueprint.

A more interesting problem arises when we require that the strips of wood be of a fixed width. This is a setting that models the construction of a piece using standard-sized (store-bought) pieces of wood. In order to express the constraint of fixed width strips we assume that the polygon P has been scaled relative to the size of the strips and that we must design a blueprint in which all chords have integer x-coordinates. In this way, we can compactly represent the blueprint $B = (P, C)$ by a translation and rotation of P. This representation avoids redundant, possibly large, space use.

If only translations of P are allowed (e.g., the grain of the wood must run in a certain direction) and all chords must lie on integer x coordinates then there is an optimal algorithm for designing and testing a blueprint. This algorithm runs in time $O(n)$ where n is the number of edges in the polygon P. Note that this is independent of the number of chords in the final blueprint (which may be much larger than n).

The first problem we encounter is that of computing a blueprint for P given a particular translation of P. To do this, we first partition P into trapezoids by finding all Steiner chords. This can be done in $O(n)$ time [5] and gives us a *trapezoidal decomposition* of P. Each Steiner chord can then be classified as false or real in constant time, so we obtain the blueprint in $O(n)$ time and test it for feasibility in an additional $O(n)$ time.

To find a feasible blueprint for P, we proceed as follows. If P has no reflex edges, then P is cuttable by a circular saw, so any blueprint is valid. If P has at least one reflex edge e then by Rule 2 in any feasible blueprint e must have at least one endpoint on a real chord. Due to the regular spacing of chords we can select either endpoint of the edge e to lie on a chord and the remainder of the chords are specified. Thus there are only two blueprints that need to be tested. Each blueprint can be created and tested for feasibility in $O(n)$ time, yielding an overall running time of $O(n)$.

4 Conclusions

We have studied the problems related to cutting strips of wood and joining them together to form a polygon P with n vertices. We show that given a blueprint for a polygon with n vertices using m strips of wood, we can test if the blueprint is feasible and, if so, give a construction in optimal $O(n)$ time. We have also shown that, if the orientation of the polygon is given, it is possible to decide if a blueprint exists in which all chords are on integer x-coordinates in $O(n)$ time.

An open problem that remains is that of determining if a blueprint exists when the orientation of the polygon is not fixed. In the preliminary version of this paper, we considered this problem but were not able to obtain algorithms whose running time was bounded by a function of n [6].

References

[1] E. D. Demaine, M. L. Demaine, C. S. Kaplan, Polygons cuttable by a circular saw, Computational Geometry: Theory & Applications 20 (1–2) (2002) 69–84, special Issue of Selected Papers from the 12th Annual Canadian Conference on Computational Geometry.

[2] F. P. Preparata, M. I. Shamos, Computational Geometry, Springer-Verlag, New York, 1985.

[3] J. L. Bentley, T. A. Ottman, Algorithms for reporting and counting geometric intersections, IEEE Transactions on Computing C-28 (1979) 643–647.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to Algorithms, MIT Press, Cambridge, 1990.

[5] B. Chazelle, Triangulating a simple polygon in linear time, Discrete and Computational Geometry 6 (1991) 485–524.

[6] P. Morin, J. Morrison, The geometry of carpentry and joinery, in: Proceedings of Fun with Algorithms 2 (FUN 2001), 2001, pp. 262–268.